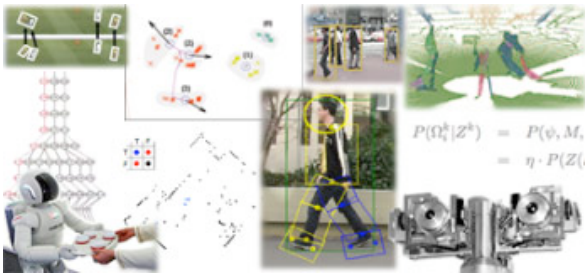
ICRA2009

2009 IEEE International Conference
on Robotics and Automation
Kobe, Japan, May 12-17, 2009

Proceedings of the IEEE ICRA 2009 Workshop on

People Detection and Tracking



Editors:

Kai O. Arras

Oscar Martinez Mozos

Tuesday, May 12, 2009, 9am-5pm

Room 405, Session TW-F5

Goal and Organization

As robots enter more domains in which they socially interact and cooperate closely with humans, the ability of machines to detect and track humans is becoming a key technology for many areas in robotics. This workshop brings together key researchers in the domain of people detection and tracking with an emphasis to unite people from the vision community and the community that has mostly worked with range finders. The goal is to provide a representative survey of the state-of-the-art and to transfer knowledge within and across the communities.

Workshop Chairs

- **Kai O. Arras**, Social Robotics Lab, Univ. of Freiburg, Germany
- **Oscar Martinez Mozos**, Robotics and Real-Time Group, Univ. of Zaragoza, Spain

Program Committee

- **Wael Abd-Almageed**, Inst. for Advanced Computer Studies, Univ. of Maryland, USA
- **Wolfram Burgard**, Autonomous Intelligent Systems Lab, Univ. of Freiburg, Germany
- **Henrik Christensen**, College of Computing, Georgia Inst. of Technology, USA
- **James L. Crowley**, INRIA Grenoble Research Center, France
- **Tsutomu Hasegawa**, Intelligent Robots and Vision Systems Lab, Kyushu Univ., Japan
- **Patric Jensfelt**, CAS, KTH Stockholm, Sweden
- **Ryo Kurazume**, Dept. of Intelligent Systems, Kyushu Univ., Japan
- **Bastian Leibe**, UMIC Research Centre, RWTH Aachen Univ., Germany
- **Ales Leonardis**, Visual Cognitive Systems Laboratory, Univ. of Ljubljana, Slovenia
- **Larry Matthies**, Jet Propulsion Laboratory, NASA, USA
- **Bernt Schiele**, Multimodal Interactive Systems Group, Univ. of Darmstadt, Germany
- **Roland Siegwart**, Autonomous Systems Lab, ETH Zurich, Switzerland
- **Luciano Spinello**, Autonomous Systems Lab, ETH Zurich, Switzerland
- **Josephine Sullivan**, Learning, Recognition, Visualisation Group, KTH Stockholm, Sweden

Homepage

http://srl.informatik.uni-freiburg.de/conferences/icra09ws

Table of Contents

Schedule

Time	Title, *Speaker*
9.00	**Welcome and Introduction to the Workshop** Kai O. Arras, Oscar Martinez Mozos
SESSION 1: People Detection/Tracking using Vision	
9.15	**Visual People Detection: Different Models, Comparison and Discussion** *Bernt Schiele*, Mykhaylo Andriluka, Nikodem Majer, Stefan Roth, Christian Wojek
9.40	**Multi-target Tracking on a Large Scale: Experiences from Football Player Tracking** *Josephine Sullivan*, Peter Nillius, Stefan Carlsson
10.05	**Results from a Real-time Stereo-based Pedestrian Detection System on a Moving Vehicle** *Max Bajracharya*, Baback Moghaddam, Andrew Howard, Shane Brennan, Larry H. Matthies
Coffee Break (10.30 - 10.50 am)	
10.50	**Poster Spotlight 1**, Kai O. Arras
10.55	**Improved Multi-Person Tracking with Active Occlusion Handling** *Andreas Ess*, Konrad Schindler, Bastian Leibe, Luc Van Gool
11.20	**Visual Person Tracking Using a Cognitive Observation Model** *Simone Frintrop*, Achim Königs, Frank Hoeller, Dirk Schulz
Lunch Break (11.45 am - 1.10 pm)	
SESSION 2: People Detection/Tracking using Laser	
1.10	**Poster Spotlight 2**, Oscar Martinez Mozos
1.15	**Spatially Grounded Multi-hypothesis Tracking of People** *Matthias Luber*, Gian Diego Tipaldi, Kai O. Arras
1.40	**Multi-Layer People Detection using 2D Range Data** *Oscar Martinez Mozos*, Ryo Kurazume, Tsutomu Hasegawa
2.05	**Multi-model Hypothesis Group Tracking and Group Size Estimation** *Boris Lau*, Kai O. Arras, Wolfram Burgard
SESSION 3: Multiple Sensors and/or Applications	
2.30	**Situation Models: A Tool for Observing and Understanding Activity** *James L. Crowley*, Patrick Reignier, Remi Barranquand
2.55	**A Trained System for Multimodal Perception in Urban Environments** *Luciano Spinello*, Rudolph Triebel, Roland Siegwart
Coffee Break (3.20 - 3.50 pm)	
3.50	**Poster Spotlight 3**, Kai O. Arras
3.55	**Motion Planning for People Tracking in Uncertain and Dynamic Environments** *Tirthankar Bandyopadhyay*, Nan Rong, Marcelo Ang, David Hsu, Wee Sun Lee
SESSION 4: Poster Session (4.20 - 5.00 pm)	

Invited Talks

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Situation Models: A Tool for Observing and Understanding Activity

James L. Crowley, *Member, IEEE*, Patrick Reignier and Remi Barranquand

***Abstract*— In this paper we describe the use of situation models for observing and understanding activity. Observing activity in natural environments can be an extremely complex perceptual problem. Situation models provide a means to both focus attention in such systems and to provide default reasoning to accommodate missing and erroneous observations. We briefly review the use of situations models in Cognitive Science and then describe how such models can be used to provide services based on observation of human activity. We present a layered component-oriented software architecture in which components for perception and action maintain a situation model for use in providing human services. We describe how this model can be used to observe activity.**

I. Introduction

Human activity is extremely rich. Real world scenes can contain an overwhelming number of possible agents and objects to detect and observe. As are result, systems and services based on observation of activity must, either implicitly or explicitly, be able to choose where to look next and what to look for. Designers of system for observing activity are increasingly confronted with the problem of control of attention.

Attention is not the only problem confronting designers of systems for observing activity. Activity in the real world often occurs in less than ideal viewing conditions. Poor lighting, background clutter, object texture, and occlusions can degrade the reliability of even the most well designed systems. Thus systems and services must be able to detect and discard uncertain and unreliable observations, and if appropriate, substitute default information. In addition, many services require real time information from perception. In such systems it may be preferable to provide an immediate response with default information and to use background processes to verify that the response was correct.

Current systems for observing activities tend to be constructed in an ad-hoc manner with control structures that are hard-wired into the system design. Such systems are generally restricted to detecting a very small set of activities observed within a highly controlled environment. Adapting such systems to different operating environments or modifying such systems to observe different forms of activity can involve extensive reprogramming.

In this paper we propose an approach for constructing systems for observing activity based on a model from Cognitive Science. We propose the use of situation models to organize, control, and interpret perception of activity. We will first provide some background from Cognitive Science concerning the use of situation models as a model of human cognition. We then describe how to use such a model to build software systems that provide services. We propose a layered, component-oriented software architecture for building situation aware services, and examine how situation models can be used to structure perceptual components and to provide default information for understanding activity. We conclude with a discussion of the problems of automatically acquiring situation models through developmental learning.

II. Situation Models as Models for Cognition

Situation models have been proposed by Johnson-Laird [1], as a cognitive theory for human mental models. Over the last 25 years, theories about situation models have been adopted and developed by a large community of cognitive psychologists. Key publications include [2], [3] as well as [4].

Situations are defined as a set of relations between entities, where a relation is a predicate function and an entity is anything that can be observed. According to Radansky [2], a situation model is a mental representation of a described or experienced situation in a real or imaginary world. Situation models are commonly composed of four primary types of information:

1) A spatial-temporal framework (spatial locations, time frames)
2) Entities (people, objects, ideas, etc.)
3) Properties of entities (color, emotions, goals, shape, etc.)
4) Relational information (spatial, temporal, causal, ownership, kinship, social, etc.)

Situation models can be structured along dimensions of space, time, causality, actors and objects. Extensions of situations models have been proposed to represent intentions of actors. It is commonly assumed that both general world knowledge (knowledge about concept types, e.g., scripts, schemas, categories, etc) and referent specific knowledge (knowledge about specific entities, independent of the situation) are used in constructing situation models.

Manuscript received March 9, 2009. This work was supported in part by project ANR CASPER, as well as the European IST projects FAME (IST 2000-28323), CAVIAR (IST 2001- 37540) and CHIL (IST 506909)

James L. Crowley is Professor at Grenoble National Polytechnique Institute (INPG) and directs the PRIMA Research group at INRIA Grenoble Centre de Recherche, 655 Ave de l'Europe, 38334 St. Ismier, France.

Patrick Reignier is a Junior Professor (MdC) at the University Joseph Fourier in Grenoble, and member of the PRIMA Research group at INRIA Grenoble Centre de Recherche, 655 Ave de l'Europe, 38334 St. Ismier, France.

Remi Barraquand is a doctoral student at Grenoble National Polytechnique Institute (INPG) under the direction of James L. Crowley and member of the PRIMA Research group at INRIA Grenoble Centre de Recherche.

Situation models are used for representations of:

1) Information about events.
2) Information about sequences of events.
3) Information about collections of episodes

We have adapted the concept of situation model to construct systems and services based on monitoring and observing human activity [5], [6], [7]. Although most of our implementations have been constructed using smart environments, such services can also be designed using robotic systems. Indeed, our approach to smart environments is to see the environment as a form of "inside out" robot, observing and interacting with occupants. Thus we maintain that models for understanding activity in smart environments may also be adapted for construction of autonomous robots.

III. Situation Models for Observing Activity

Situation models can be used to addresses the twin problems of focus of attention, and operation with unreliable, erroneous or missing data. They can also be used to decouple services from the time constraints normally imposed by real-time (or near real-time) vision systems. We present our technique in the context of a service-oriented architecture constructed using a layered, component-based, software model. For the robotics and vision communities, these concepts may require some explanation.

The term "service" is used here in its most general form. Generally, it will refer to assistance that informatics systems provide to people. User services can be designed as software agents that interact and assist people. Over the last few years, we have constructed a variety of services that observe and model human activity in order to provide assistance that is dependent on human context. Such systems are generally said to be "context aware". Exampels include services for lecture recording [8], meeting services [9], monitoring of the health and well-being of elderly, and availability monitoring [7]. As sensor and actuator technology mature, we can expect to see the emergence of an increasing variety of such systems for domestic services (cleaning, logistics, cooking), commercial services (shopping, queue management, customer assistance), health monitoring and assisted living, security monitoring, and a variety of other application domains. All of these examples require observing and understanding the actions of humans. We believe that situation models will provide an important component for such systems.

We note that the term "service oriented" also has a more technical meaning for the software engineering community. In software engineering, a "service oriented" system is one in which software components interact according to a well-defined contract. For example, a location service integrates information from a variety of sources to estimate the current location of a user. Although the two uses of the term "service" are not incompatible, they can cause some confusion. Thus we will use the explicit term "software services" for services that are primarily designed to interact with software components. We will interchangeably use "user services" or simply "services" for systems that interact with and assist people.

Modern software systems are generally designed using a layered architecture. A layered architecture organizes the system into a hierarchy of interchangeable components, with well-defined interfaces. The design and operation at a particular layer may proceed independently of the underlying components. Components that make up a particular layer may be reused or shared by a variety of services. Components that are temporarily inoperative may be replaced with alternative components. A common example of this approach is provided by the current generation of location aware services on mobile devices that can interchangeably use location information from GPS, cell phone repeaters, or WIFI repeater identity. Components for providing location from WIFI, GPS or cell-phone repeaters are a form of "perceptual component" that operate in parallel using competing methods to make available a key piece of information: current location. We propose a similar approach to building components for observing activity. Perceptual components can be constructed to observe a scene with competing methods to provide information that may then be shared between different services.

A situation model falls naturally at the interface between user services and perceptual components. For user services, the situation model provides a default reasoning system that can complete or repair partial or missing information from sensing. For the perceptual components, the situation model can be used to focus attention on the objects and events that are relevant to a service, allowing irrelevant objects or events to be ignored. The situation model can be used to predict possible events, both to focus attention, and to prepare a reaction before the event occurs.

In the following, we describe a layered architecture for context aware user services based on observation of activity. We then describe the elements of the situation model, and describe how such a model can be used to configure and control perceptual components, to focus attention, predict events, and to provide default reasoning for observation of activity.

A. Services, Sensors, and Components

We are interested in services that provide assistance through the observation of human activity. A service determines requirements for perception and action, without specifying how these requirements are to be met. Hard-wiring the interconnection between sensor signals and actuators is possible, and can provide simplistic services that are hardware dependent and have limited utility. Separating services from their underlying hardware makes it possible to build systems that operate in a larger range of environments, for a larger variety of functions. However such separation requires that the sensor-actuator layer provide logical interfaces, or standard API's, that are function centered and device independent. Hardware independence and generality require abstractions for perception and action.

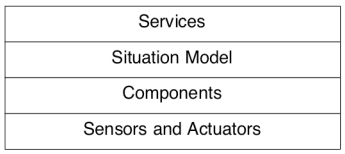


Fig. 1. A layered model for systems that observe human activity.

A layered architecture of user services is shown in figure 1. At the lowest layer, the service's view of the world is provided by a collection of physical sensors and actuators. This corresponds to the *sensor-actuator layer*. This layer depends on the technology and encapsulates the diversity of sensors and actuators by which the system interacts with the world. Information at this layer is expressed in terms of sensor signals and device commands.

Service abilities for perception and action are provided by components for perception and action. Components make observations about the environment, interact with users, and take actions to impart changes to the environment.

In our systems, services maintain information about users and the environment in a situation model. The situation model has the form of a network of situations. Each situation has three facets: Observation, Reaction and Prediction. The observation facet specifies the entities, properties and relations needed to define the situation. This can act as a specification that serves to activate and configure a set of perception components capable of providing observations about the required entities and their relations. The reaction facet specifies how the service should behave in each situation, including both the desired state of the environment, and a specification communications that the service should make with the user. The Prediction facet indicates possible changes to the current situation, by pointing to adjacent situations and describing the events that can indicate the change.

Sensors are devices that make measurements, ranging simple devices that measure temperature or humidity, to devices that capture motion (infrared motion detectors), acoustic energy (microphones) and images (cameras) or 3D structure (range sensors, stereo vision systems). Actuators impart change on the environment. Such devices can range from information displays, control of lighting and sound systems, motorized controls for doors, windows and window blinds, as well as mobile robotic devices for logistics, cleaning or entertainment.

Components for perception and action operate at a higher level of abstraction than sensors and actuators. While sensors and actuators operate on device-specific signals, perception and action operate in terms of environmental state. Perception interprets sensor signals by detecting, recognizing and observing people, things and events. Action components alter the environment to being it to a desired state. Tightly coupling perception and action can offer many advantages. Controlling action with perception allows a service to adapt action in accordance with the effect on the environment. Action can also be used to reconfigure the environment to improve perception, or even to probe the environment as part of perception.

B. Components for Perception and Action

Perception and action components are autonomous assemblies of modules executed in a cyclic manner by a component supervisor. Components communicate via synchronous data streams and asynchronous events in order to provide software services for action or perception. We propose a data-flow process architecture for software components for perception and action [10], [11], [12]. Component based architectures, as described in Shaw and Garlan [13], are composed of auto-descriptive functional components joined by connectors. Such an architecture is well adapted to interoperability of components, and thus provides a framework in which components can employ competing methods to accommodate sensor modes that are unreliable or available in only limited conditions.

Components are controlled by a supervisory module. The component supervisor interprets commands and parameters, supervises the execution of the transformation, and responds to queries with a description of the current state and capabilities of the component. The auto-critical report from modules allows a component supervisor to monitor the execution time and to adapt the schedule of modules for the next cycle so as to maintain a specified quality of service, such as execution time or number of targets tracked. Such monitoring can be used, for example, to reduce the resolution of processing an image by selecting 1 pixel of N [14] or to selectively delete targets judged to be uninteresting or erroneous [15].

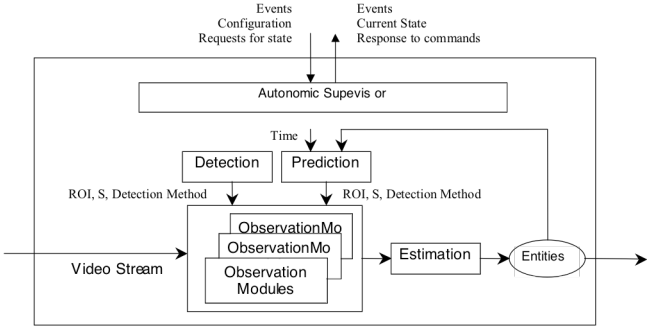


Figure 2. An example of perceptual component based on visual tracking

In addition to recognition, the supervisory component provides execution scheduling, self-monitoring, parameter regulation, and communications. The supervisor acts as a scheduler, invoking execution of modules in a synchronous manner. For self-monitoring, a component applies a model of its own behavior to estimate both quality of service and confidence for its outputs. Monitoring allows a process to detect and adapt to degradations in performance due to changing operating conditions by reconfiguring its component modules and operating parameters. Monitoring also enables a process to provide a symbolic description of its capabilities and state.

Homeostasis or "autonomic regulation of internal state" is a fundamental property for robust operation in an uncontrolled environment. A component is auto-regulated when processing is monitored and controlled so as to maintain a certain quality of service. The process supervisor maintains homeostasis by adapting module parameters to maximize estimated quality of service. For example, processing time and precision are two important state variables for a tracking process. Quality of service measures such as cycle-time, number of targets, or precision can be maintained by dropping targets based on a priority assignment or by changing resolution for processing of some targets.

During the communication phase, the supervisor may respond to requests from other components. These requests may ask for descriptions of process state, process capabilities, or may provide specification of new recognition methods. The supervisor acts as a programmable interpreter, receiving snippets of code script that determine the composition and nature of the process execution cycle and the manner in which the process reacts to events. Recognition procedures are small procedures interpreted by a lightweight language interpreter [16]. In our implementation, such procedures may be preprogrammed or they may be downloaded to the component during configuration as snippets of code using a lisp-like language.

For most human activities, there are a potentially infinite number of entities that could be observed and an infinite number of possible relations for any set of entities. The appropriate entities and relations must be determined with respect to the service to be provided. This is the role of the situation model. The situation model allows the system to focus computing resources, to provide missing information, and to determine appropriate or inappropriate system actions for the current state of the activity.

Perceptual components communicate using Streams, Events, and Queries. Streams are synchronous communication channels for communicating continual data such as image frames or acoustic signals. An important role for perceptual components is to process streams in order to observe entities and their properties. Events are asynchronous messages generated by components in response to changes in entities or their properties. Events may be sent to other components or to the situation model. Queries are communication transactions in which a service, the situation model, or another component exchange messages with the component supervisor in order to interrogate a component about its entities and their properties.

C. *Assembling Components to Provide Services*

We have constructed a middle-ware environment [17] that allows us to dynamically launch and connect components on different machines. This environment, called O3MiSCID, provides an XML based interface that allows components to declare input command messages, output data structures, as well as current operational state. In this environment, a user service may be created by assembling a collection of perceptual components.

Available components are discovered by interrogating an component data-base. An open research challenge is to provide an ontological system for indexing components based on function in a manner that is sufficiently general to capture future functionalities as they emerge. In addition the component data-base provides information about message formats and data types for communication of streams, events and queries.

Figure 3 shows a simple example of a service provided by an assembly of perceptual components. This service integrates information from multiple cameras to provide 3-D target tracking. A set of tracked entities is provided by a Bayesian 3D tracking process that tracks targets in 3D scene coordinates. This process specifies the predicted 2-D Region of Interest (ROI) and detection method for a set of pixel-level detection components. These components use color, motion or background difference subtraction to detect and track blobs in an image stream from a camera. The O3MICID middle-ware makes it possible to dynamically add or drop cameras to the process during tracking.

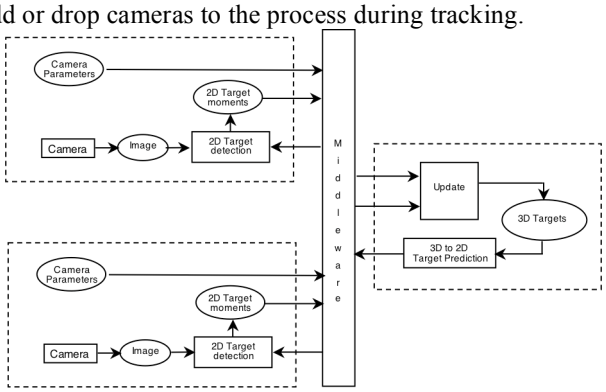


Fig. 3. An example of an assembly of perceptual components. The 3D Bayesian blob tracker provides a ROI and detection method for a number of 2D entity detection components. The result is used to update a list of 3D blobs.

D. *Entities and Relations*

Situations are defined as relations between entities. An "entity" is anything that can be observed. This solipsistic viewpoint admits that the system can only see what it knows how to see. At the same time, it sidesteps existential dilemmas related to how to define notions of "object" and "class".

Formally, entities are correlated sets of observations. Entities are grounded in the software components for observation of activity, typically through some form of tracking process that correlates observations over time. Entities can be decorated with properties that make possible the determination of relations between entities.

A relation is a predicate or binary function computed on the properties of one or more entities. Relations have an arity, that specifies the number of properties that serve as arguments. An arity-1 relation is true when a property is observed to be within some range of values, or is otherwise signaled as true by a sensor. Examples can include (standing person) or (running person). Relations of Arity-2 include

many of the classical spatial and temporal relations as well as more abstract functions describing social-behaviour or emotion. Spatial relations can be 2D or 3D and relative or absolute, depending on the requirements of the service. Examples can include absolute position of actors (at podium person), (seated-at table person), relative position (facing person1 person2), or even refer to the posture of persons (standing person). Observing human interaction can require perceptual components that detect more abstract social behaviour, such as (talking-to person1 person2) or (smiling-at person1 person2).

As mentioned above, the number of potential relations that might be observed is an unbounded set. The situation model for a service specifies the relations between that are required, the entities (agents and objects) that must be observed, the properties that are needed to determine relations. The task of the system designer is to provide perceptual components that can detect and track the required entities, measure the required properties, and detect when the required relations are true.

Human attention is an important relation in social situations. In our approach, we have adopted the attention model developed by Maisonnasse [18]. In this work, attention is defined as a cognitive process of selectively concentrating on one aspect of the environment while ignoring other things. We include attention of agents as one of the fundamental relations for describing social situations.

E. Generalizing with Roles

In most situations, the exact identity of the entity is not important. Thus we have generalized situation models by the introducing of the concept of "role" [5]. A role is a form of abstract model for an entity. In applying a situation model to describe a scene, a system will select from available entities to determine which entity can "fill" each role.

Operationally, a role is an abstract generalization for a class of entities. Role classes are typically defined based on the set of actions that entities in the class can take (actors), or the set of actions that the entities can enable (props). Formally, role is a function that selects an entity from the set of observed entities.

A “role” is NOT an intrinsic property of an entity, but rather, is an interpretation applied to an entity by the system. Entities are assigned to roles by a role assignment process. Role assignment generally occurs by applying a set of tests to available entities. The role assignment process acts as a form of "filter" [19] that sorts entities based on the suitability of their properties. The most suitable entity wins the role assignment.

In our experiments for automatic learning of situation models [6], we have discovered that roles provide generalization, making it possible to greatly accelerate learning. Reactions learned for a situation composed of one set of entities can be used to understand a different set of entities.

F. Situations as Scripts for Understanding Activity

The situation model acts as a non-linear script for interpreting activity and predicting the corresponding appropriate and inappropriate actions for services. This framework organizes the observation of interaction using a hierarchy of concepts: scenario, situation, role, entity and relations. A situation is defined as a configuration of relations over a set of entities playing roles. Thus a situation is a form of state, expressed as a logical expression (a conjunction of predicates). This logical expression is composed of predicates whose arguments are roles. This concept generalizes and extends the common practice of defining situations based on the relative position of actors and objects.

Relations test the properties of entities that have been assigned to roles. As mentioned above, situations also predict possible future situations. This is captured by the connectivity of a situation network. Changes in the logical expression of relations or in the selection of entities playing roles are represented as changes in situation. Such changes can trigger system actions.

A situation is a form of state, expressed as a logical expression (a conjunction of predicates). Situations are organized into networks, with transition probabilities, so that possible next situations may be predicted from the current situation. In our systems, the situation model drives focus of attention by specifying the entities and relations that should be attended. When a service is initiated, a list of relevant entities and relations are provided, along with the relevant configuration information. This list is used to initiate and configure the relevant perceptual and action components needed to maintain the situation model.

Each situation contains a list of expected relations, as well as expected observed entities and their expected properties. Transitions between situations can be triggered by events, and do not require verification for the entire set of relations, entities and properties. Thus it is possible for a situation to provide default values for relations, and properties that have not been verified. When interrogated by a service, a situation model may respond with the default values, whether or not these values can be currently verified. Such a response can be provided without waiting for an actual verification to occur. However, this verification can be used as an integrity check for the situation model

When a system responds with a default value, it is good practice for the system to query the relevant perceptual components to verify that the default value is correct. In some cases, this may indicate a divergence between the situation model and the environment. Such a divergence can be used to trigger a diagnostic process to recover from the current error, by adapting perception to changes in the environment or by developing the situation model by adding new situations or behaviours.

IV. 5. Learning Situation Networks

We distinguish the concepts of adaptation from development [20]. *Adaptation* allows a system to maintain consistent behaviour across variations in operating environments. The environment denotes the physical world (e.g., in the street, lighting conditions), the user (identification, location, goals and activities), social settings, and computational, communicational and interactive resources. *Development* refers to the acquisition of abilities, in this case encoded as situation models composed of the entities, roles and relations with which situation is described and service actions are performed.

Systems for providing services based on observing activity must both adapt and develop. Adaptation is necessary to maintain consistent behaviour while accommodating changes in the operating environment, task, user population, preferences or some other factors. At the same time, human activity is too complex to be fully captured in a pre-programmed situation model. An activity model must develop through observation and interaction with users. A fundamental challenge is to provide both automatic adaptation and automatic development without disruption.

Current learning technologies, such as hidden Markov models and neural networks, require large sets of training data – something that is difficult to obtain for an uncontrolled environment. Development of context models requires new ways of looking at learning, and suggests the need for a new class of minimally supervised learning algorithms. This requires that learning be studied as part of a semi-autonomous system. It requires that systems have properties of self-description, self-evaluation and auto-regulation, and may well lead to new classes of learning algorithms specifically suitable to developing and evolving context models in a non-disruptive manner.

We are currently experimenting with techniques for adapting activity models based on pre-defined stereotypical situations [21]. We are exploring different approaches to learning for development of activity models starting from a predefined stereotypical model using feedback about the system actions. Because the different components of the model (entities, roles, relations, and situations) depend on each other, these cannot be developed simultaneously. Thus we have focused on the development of the situation networks and the associated system actions.

Bayesian models (in particular Hidden Markov Models [22] as well as algorithms based on first-order logic [23] can be used to represent and adapt the situation network. However, these approaches do not have desirable properties concerning the extension of the number of situations. Bayesian models require a large amount of example data to extend the number of states. First-order logic algorithms cannot create new predicates (problem of higher order logic), which is necessary for the extension of situations. Thus we propose an approach for changes in the structure of the situation network, as shown in figure 4.

The input to the algorithm is a predefined situation network along with feedback from prior use mediated by a supervisor. The supervisor corrects, deletes or preserves the actions executed by the system while observing a user in the environment. Each correction, deletion, or preservation generates a training example for the learning algorithm containing current situation, roles and configuration of relations, and the (correct) (re)action. The differences between the actions given in the training examples and the actions provided in the predefined situation network will drive the different steps of the algorithm.

Initially, our approach has been to directly modify system actions using the existing situation network. If action A is associated with situation S, and all training examples indicate that action B must be executed instead of A, then B is associated to S and the association between A and S is deleted.

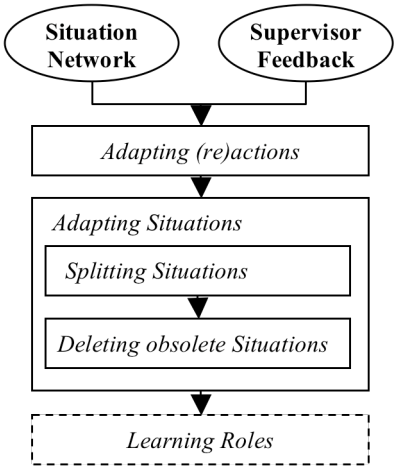


Fig 4: Overview of the algorithm for adapting system actions

V. Conclusions

Activity models for context aware services can be expressed as a network of situations concerning a set of roles, entities and relations. Roles are abstract classes for entities. Entities may be interpreted as playing a role, based on their current properties. Relations between entities playing roles define situations. This conceptual framework provides default reasoning, focus of attention, and real time response for services that require observation of human activity. This model can also provide a basis for adaptation and development of non-disruptive software services for aiding human-to-human interaction.

Socially aware observation of activity and interaction is a key requirement for development of non-disruptive context aware user services. For this to become reality, we need methods for robust observation of activity, as well as methods to automatically learn about activity without imposing disruptions. The framework and techniques described in this paper are intended as a foundation for such observation.

REFERENCES

[1] P. N. Johnson-Laird, Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness, Harvard Univ. Press, Cambridge, MA, 1983.

[2] Radvansky, G. A., & Zacks, R. T. (1997). The retrieval of situation-specific information. In M. A. Conway (Ed.) Cognitive Models of Memory, pp. 173-213. Cambridge, MA: MIT Press.

[3] Zwaan, R. A. Radvansky, G. A., "Situation Models in Language Comprehension and Memory, PSYCHOLOGICAL BULLETIN, VOL 123; NUMBER 2, pages 162-185, 1998.

[4] P.N. Johnson-Laird, Mental models, MIT Press Cambridge, MA, USA, 1989.

[5] J. L. Crowley, "Context Driven Observation of Human Activity", European Symposium on Ambient Intelligence, Amsterdam, 3-5 November 2003

[6] J. L. Crowley, O. Brdiczka, and P. Reignier. Learning Situation Models for Understanding Activity In The 5th International Conference on Development and Learning 2006 (ICDL06), Bloomington, Il., USA, June 2006

[7] O. Brdiczka, J. L. Crowley, P. Reignier, Learning situation models for providing context-aware services, in "IEEE Transactions on Man, Systems and Cybernetics, Part B", Volume 38, Number 1, January 2008.

[8] F. Metze, P. Gieselmann, H. Holzapfel, T. Kluge, I. Rogina, A. Waibel, and M. Wolfel, J. Crowley, P. Reignier and D. Vaufreydaz, F. Bérard, B. Cohen, J. Coutaz, V. Arranz, M. Bertran and H. Rodriguez, "The FAME Interactive Space", 2nd Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms, MLMI, Edinburgh, July 2005.

[9] M. Danninger, T. Kluge, R. Stiefelhagen, "MyConnector: analysis of context cues to predict human availability for communication", International Conference on Multimodal Interaction, ICMI 2006: pp12-19, Trento, 2006.

[10] Software Process Modeling and Technology, edited by A. Finkelstein, J. Kramer and B. Nuseibeh, Research Studies Press, John Wiley and Sons Inc, 1994.

[11] J. Rasure and S. Kubica, “The Khoros application development environment “, in Experimental Environments for computer vision and image processing, H. Christensen and J. L. Crowley, Eds, World Scientific Press, pp 1-32, 1994.

[12] J. L. Crowley, "Integration and Control of Reactive Visual Processes", Robotics and Autonomous Systems, Vol 15, No. 1, decembre 1995

[13] M. Shaw and D. Garlan, Software Architecture: Perspectives on an Emerging Disciplines, Prentice Hall, 1996.

[14] J. Piater and J. Crowley, "Event-based Activity Analysis in Live Video using a Generic Object Tracker", Performance Evaluation for Tracking and Surveillance, PETS-2002, Copenhagen, June 2002.

[15] D. Hall, R. Emonet, and J. L. Crowley, "An automatic approach for parameter selection in self-adaptive tracking." In International Conference on Computer Vision Theory and Applications (VISAPP), Setubal, Portugal, Feb. 2006.

[16] A. Lux, "The Imalab Method for Vision Systems", International Conference on Vision Systems, ICVS-03, Graz, april 2003.

[17] R. Emonet, D. Vaufreydaz, P. Reignier, J. Letessier, "O3MiSCID: an Object Oriented Opensource Middleware for Service Connection, Introspection an Discover", 1st IEEE International Workshop on Services Integration in Pervasive Environments - June 2006.

[18] J. Maisonnasse, N. Gourier, O. Brdiczka and P. Reignier, Attentional Model for Perceiving Social Context in Intelligent Environments, Artificial Intelligence Apllications and Innovations 2006.

[19] O. Brdiczka, J. Maisonnasse, P. Reignier, Automatic Detection of Interaction Groups, 2005 International Conference on Multimodal interaction, ICMI '05, Trento It., october 2005

[20] J Coutaz, J. L. Crowley, S. Dobson, and D. Garlan, "Context is Key", Communications of the ACM, Special issue on the Disappearing Computer, Vol 48, No 3, pp 49-53 March 2005.

[21] R. Barraquand and J. L. Crowley, "Learning Polite Behavior with Situation Models", Third International Conference on Human Robot Interaction (HRI 2008), 12-15 March 2008, Amsterdam, The Netherlands

[22] L. R. Rabiner, A Tutorial on Hidden Markov Models and selected Applications in Speech Recognition. Readings in speech recognition. p. 267-296, 1990.

[23] J. R. Quinlan, Learning Logical Definitions from Relations. Machine Learning. 5(3), p. 239-266, 1990.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Visual People Detection – Different Models, Comparison and Discussion

Bernt Schiele, Mykhaylo Andriluka, Nikodem Majer, Stefan Roth and Christian Wojek
Department of Computer Science, TU Darmstadt

Abstract

Over the last few years, visual people detection has made impressive progress. The paper gives an overview of some of the most successful techniques for people detection and also summarizes a recent quantitative comparison of several state-of-the-art methods. As a proof-of-concept we show that the combination of visual and laser-based people detection can result in a significant increase in performance. We also briefly discuss future research directions for visual people detection.

1. Introduction

People detection is one of the most challenging problems in computer vision due to large variations caused by articulation, viewpoint and appearance. At the same time detecting and tracking people has a wide range of applications including robotics, image and video indexing, surveillance and automotive safety. Consequently visual people detection has been researched intensively with a rapid rate of innovation. Recently, several researchers have reported impressive results [23, 33, 6, 18, 1, 36] for this task.

The aim of this paper is threefold. First, we provide an overview of some of the most successful methods for visual people detection. Second, we summarize a comparative study of sliding-window techniques [35]. And third, we show the potential of combining visual people detection with other modalities such as laser.

Broadly speaking there are two major types of approaches for visual people detection. Sliding-window methods exhaustively scan the input images over positions and scales independently classifying each sliding window (e.g. [23, 33, 6]). Other methods generate hypotheses by evidence aggregation often using part-based human body models (e.g. [12, 9, 21, 18, 37, 28, 1]). After discussing some of the most successful sliding-window approaches in section 2 we summarize a comparative study of such methods in section 3. Section 4 briefly describes a part-based model that has shown to outperform sliding-window techniques in the presence of partial occlusion. Section 5 then describes an experiment to complement visual people detection with a laser-range finder thereby significantly reducing the number of false positives of the visual people detector. The last section 6 discusses promising research directions to improve the performance of today's visual people detection methods.

2. Sliding-window techniques

Sliding window detection systems scan the image at all relevant positions and scales to detect a person. Consequently there are two major components: the *feature* component encodes the visual appearance of the person, whereas the *classifier* determines for each sliding window independently whether it contains the person or not. As typically many positions and scales are scanned these techniques are inherently computationally expensive. Fortunately, due to recent advances in GPUs, real-time people detection is possible as e.g. demonstrated by [34]. In [35] we conducted a quantitative comparison that we briefly summarize in section 3.

As a complete review on people detection is beyond the scope of this work, we focus on most related work. An early approach [23] used Haar wavelets and a polynomial SVM while [33] used Haar-like wavelets and a cascade of AdaBoost classifiers. Gavrila [13] employs a hierarchical Chamfer matching strategy to detect people. Recent work often employs statistics on image gradients for people detection. [30] uses edge orientation histograms in conjunction with SVMs while [6] uses an object description based on overlapping histograms of gradients. [27] employs locally learned features in an AdaBoost framework and Tuzel [32] presents a system that exploits covariance statistics on gradients in a boosting classification setting. Interestingly, most approaches use discriminant classifiers such as AdaBoost or SVMs while the underlying object descriptors use a diverse set of features. Therefore the following section briefly describe some of these features in more detail.

Haar wavelets have first been proposed by Papageorgiou and Poggio [23]. They introduce a dense overcomplete representation using wavelets at the scale of 16 and 32 pixel with an overlap of 75%. Three different types are used, which allow to encode low frequency changes in contrast: vertical, horizontal and diagonal. Thus, the overall length of

the feature vector for a 64×128 pixel detection window is 1326 dimensions. In order to cope with lighting differences, for each color channel only the maximum response is kept and normalization is performed according to the window's mean response for each direction. Additionally, the original authors report that for the class of people the wavelet coefficient's sign is not carrying information due to the variety in clothing. Hence, only the absolute values for each coefficient is kept. During our experiments we found that an additional L_2 length normalization with regularization of the feature vector improves performance.

Histograms of oriented gradients have been proposed by Dalal and Triggs [6]. Image derivatives are computed by centered differences in x- and y direction. The gradient magnitude is then inserted into cell histograms (8×8 pixels), interpolating in x, y and orientation. Blocks are groups of 2×2 cells with an overlap of one cell in each direction. Blocks are L_2 length normalized with an additional hysteresis step to avoid one gradient entry to dominate the feature vector. The final vector is constituted of all normalized block histograms with a total dimension of 3780 for a 64×128 detection window.

Shape Context has originally been proposed as a feature point descriptor [4] and has shown excellent results for people detection in the generative ISM framework [18, 28]. The descriptor is based on edges which are extracted with a Canny detector. Those are stored in a log-polar histogram with location being quantized in nine bins. For the radius 9, 16 and 23 pixels are used, while orientation is quantized into four bins. For sliding window search we densly sampled on a regular lattice with a support of 32 pixels (other scales in the range from 16 to 48 pixels performed worse). For our implementation we used the version of Mikolajczyk [20] which additionally applies PCA to reduce the feature dimensionality to 36 dimensions. The overall length of all descriptors concatenated for one test window is 3024.

Classifiers. The second major component for sliding-window approaches is the deployed classifier. For the classification of single windows two popular choices are SVMs and decision tree stumps in conjunction with the AdaBoost framework. SVMs optimize a hyperplane to separate positive and negative training samples based on the *global* feature vector. Different kernels map the classification problem to a higher dimensional feature space. For our experiments we used the implementation *SVM Light* [16]. In contrast, boosting is picking *single entries* of the feature vector with the highest discriminative power in order to minimize the classification error in each round.

3. Comparison of sliding-window techniques

In [35] we conducted a systematic evaluation of different feature/classifier combinations. For this we reimplemented the respective features and classifiers. Comparisons with published binaries (whenever available) verified that our reimplementations perform at least as good as the originally proposed feature/classifier combinations. In the following we report on some of the results that illustrate the state-of-the-art in sliding window based detection techniques.

To evaluate the performance for the introduced features and their combination with different classifiers we use the established INRIA Person dataset [1]. This data set contains images of humans taken from several viewpoints under varying lighting conditions in indoor and outdoor scenes. Unlike the original authors [6] we test the trained detectors on the full images. We do so, in order not only to evaluate the detector in terms of false positive detections per window (FPPW) but with respect to their frequency and spatial distribution. This gives a more realistic assessment on how well a detector performs for real image statistics. For further details see [35]

Due to space constraints we cannot report all the quantitative results from [35]. However, we still report the major results and figure 1 contains the results for four different settings.

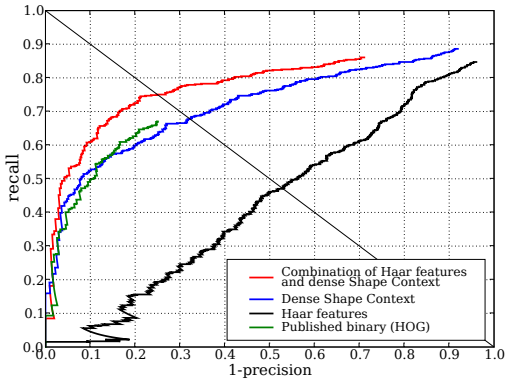


Figure 1. Recall-Precision detector performances for different features (Haar, HOG, Dense Shape Context, combination of Dense Shape Context and Haar) and linear SVM-classifier

Single feature detection. We start by summarizing the evaluation of using all features individually in combination with the three classifiers AdaBoost, linear SVM and RBF kernel SVM. First of all, the HOG descriptor and the similar Shape Context descriptor consistently outperform the other features (e.g. Haar-like features) independent of the learning algorithm. Overall, RBF kernel SVMs together with the gradient-based features HOG and Shape Context show the best results. All features except shapelets show better performance with the RBF kernel SVM compared to the linear SVM. AdaBoost achieves a similarly good performance in comparison with RBF kernel SVMs in particular for the Haar-like wavelet, the HOG feature and for shapelets. It does slightly worse for the dense Shape Context descriptor.

[1] http://pascal.inrialpes.fr/data/human

Multi-cue detection. A closer look on the single detectors' complementarity reveals that different features in combination with different classifiers have a varying performance on the individual instances. This can be explained by the fact, that the features encode different information. While gradients encode high frequency changes in the images, Haar wavelets as they are proposed by [23] also encode much lower frequencies. Figure 1 shows the combination of dense Shape Context features with Haar wavelets. In particular figure 1 shows, that in fact both features on their own cannot reach the performance that is reached with their combination. Compared to the state-of-the-art HOG object detector we improve recall considerably about 10% at 80% precision. Figure 2 shows sample detections of this multi-cue detector.

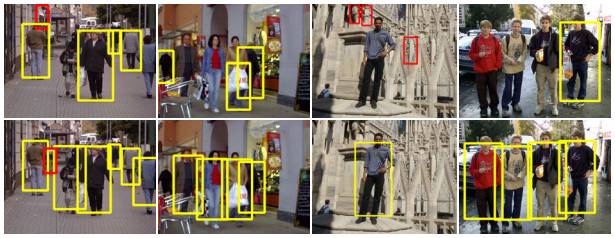


Figure 2. Sample detections at a precision of 80%. Red bounding boxes denote false detections, while yellow bounding boxes denote true positives. First row shows detection by the publically available HOG detector[6]; second row depicts sample detections for our combination of dense Shape Context with Haar wavelets in a linear SVM

Failure analysis. To get a feeling about the achievable performance of sliding-window based techniques we complete our brief summary with a failure case analysis. In particular, we analyzed the missing recall and the false positive detections at equal error rate (149 missing detections / 149 false positives) for the feature combination of Shape Context and Haar wavelets in combination with a linear SVM. Missing recall mainly occurred due to unusual articulations (37 cases), difficult background or contrast (44 cases), occlusion or carried bags (43 cases), under- or overexposure (18 cases) and due to detection at too large or too small scales (7). There were also 3 cases which were detected with the correct height but could not be matched to the annotation according to the PASCAL criterion due to the very narrow annotation.

False positive detections can be categorized as follows: Vertical structures like poles or street signs (54 cases), cluttered background (31 cases), too large scale detections with people in lower part (24 cases), too low scale on body parts (28 cases). There were also a couple of "false" detections (12 cases) on people which were not annotated in the database (mostly due to occlusion or at small scales). Some samples of missed people and false positives are shown in figure 3.

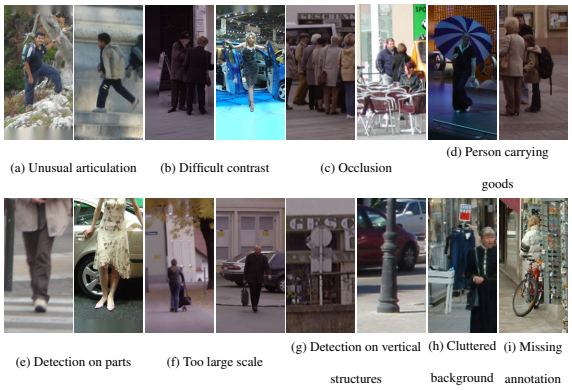


Figure 3. Missed recall (upper row) and false positive detections (lower row) at equal error rate

4. Part-based models for people detection

Part-based models have a long history in computer vision for object detection in general and for people detection in particular (e.g. [12, 9, 21, 18, 37, 28, 1]). There are two major components of these models. The first uses low-level features or classifiers to model individual parts or limbs of a person. The second component models the topology of the human body to enable the accumulation of part evidence.

A wide range of models have been proposed e.g. for upright people detection in traffic scenes [18], to estimate the pose of highly articulated people (e.g. in sports scenes [25]), or for upper body detection and pose estimation [11], e.g. for movie indexing. In this section we briefly summarize one of our own models [1] that builds upon and extends a number of previous approaches. The model is inspired by the pictorial structures model proposed by [10, 15], but uses more powerful part representations and detections, and as we will show outperforms recent pedestrian detectors [6, 28].

A part-based person model [1]. Following the general pictorial structures idea, a person is represented as a joint configuration of her body parts. In such a model the problem of locating a person in a test image is formulated as search for the modes of the posterior probability distribution $p(L|E)$ of the body part configuration L given the image evidence E and (implicit) class-dependent model parameters θ. In our model, the configuration is described as $L = \{\mathbf{x}^o, \mathbf{x}^1, \ldots, \mathbf{x}^N\}$, where $\mathbf{x}^o$ is the position of the body center and its scale, and $\mathbf{x}^i$ is the position and scale of body part i. The image evidence, which here is defined as a set of local features observed in the test image, will be denoted as $E = \{\mathbf{e}_k^{app}, \mathbf{e}_k^{pos} | k = 1, \ldots, K\}$, where $\mathbf{e}_k^{app}$ is an appearance descriptor, and $\mathbf{e}_k^{pos}$ is the position and scale of the local image feature with index k.

An important component of the pictorial structures model is an implicit model of a-priori knowledge about possible body configurations, which must be expressive enough

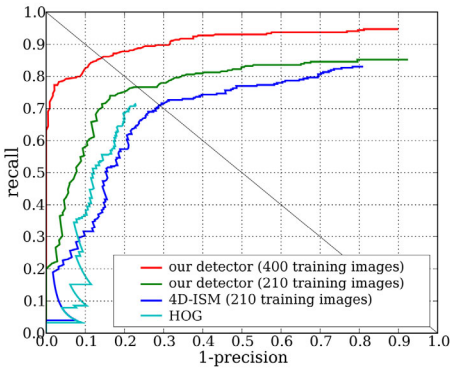


Figure 4. Comparison of our pedestrian detector with 4D-ISM detector [29] and HOG [6].

to capture all important dependencies between parts. For particular object categories, such as walking people, we can introduce auxiliary state variables that represent the *articulation state* or an *aspect* of the object, such as different phases in the walking cycle of a person [17], and make the parts conditionally independent. As we are not interested in knowing the articulation state, but only the object and limb positions, the articulation state a can be marginalized out: $p(L|E) = \sum_a p(L|a, E)p(a)$.

From decomposing $p(L|a, E) \propto p(E|L, a)p(L|a)$, assuming that the configuration likelihood can be approximated with product of individual part likelihoods [10] $p(E|L, a) \approx \prod_i p(E|\mathbf{x}^i, a)$, and assuming uniform $p(\mathbf{x}^i|a)$, it follows that

$$p(L|a, E) \approx p(\mathbf{x}^o) \prod_i p(\mathbf{x}^i|a, E)p(\mathbf{x}^i|\mathbf{x}^o, a). \quad (1)$$

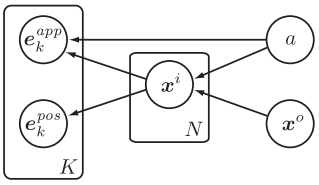


Figure 5. Graphical model structure describing the relation between articulation, parts, and features.

Please refer to [1] for the details concerning model training and inference. In the experiment (as presented in detail in [1]) we use shape context feature descriptors [3] and the Hessian-Laplace interest point operator [19] as detector. Ee compare the above detector on a challenging dataset of street scenes containing 311 side-view pedestrians with significant variation in clothing and articulation[2]. Fig. 4 shows the comparison of our detector with two state-of-the-art detectors. Using the same training set as [28] our detector outperforms the 4D-ISM approach [28] as well as the HOG-detector [6]. Increasing the size of the training set further improves performance significantly.

[2] Available at www.mis.informatik.tu-darmstadt.de.

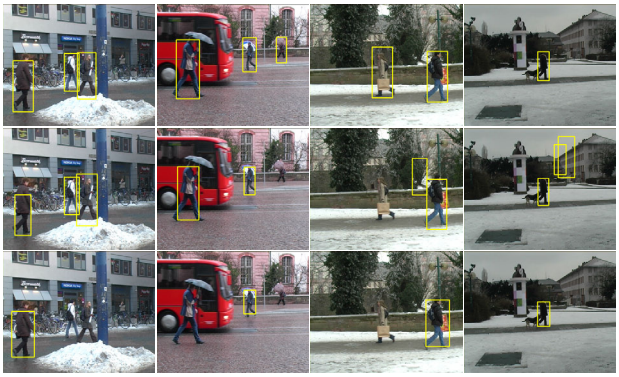


Figure 6. Example detections at equal error rate of our detector (top), 4D-ISM (middle) and HOG (bottom) on the "TUD-Pedestrians" dataset.

Fig. 6 shows sample detections of the 3 methods on test images. The 4D-ISM detector is specifically designed to detect people in cluttered scenes with partial occlusions. Its drawback is that it tends to produce hypotheses even when little image evidence is available (image 3 and 4), which results in increased number of false positives. The HOG detector seems to have difficulties with the high variety in articulations and appearance present in out dataset. However, we should note that it is a multi-view detector designed to solve a more general problem than we consider here.

Summary. From these experiment we can conclude that part-based people model can outperform sliding-window based methods (such as HOG) in the presence of partial occlusion and significant articulations. It should be noted however, that part-based models tend to require a higher resolution of the person in the image than most sliding-window based approaches.

5. Combining vision and laser to improve people detection

Cameras are not the only sensor that can be used for people detection. In robotics laser range scanners are widely used for tasks like localization and position estimation but have been also used for people detection [2] and place classification [22]. Recent approaches to fuse visual and laser information for classification and object detection tasks show promising results [26, 24, 38, 31]. This section explores a simple yet effective technique to combine vision and laser information for improved people detection. As visual people detection is never perfect laser range information is used to constrain the search space of plausible hypotheses.

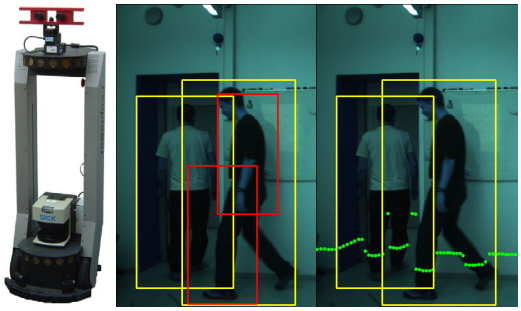
Setting. The platform used for data acquisition is a PeopleBot that runs a distributed component architecture devel-

oped during the CoSy project. The robot is equipped with a SICK LMS (180° fov, 1° angular resolution) mounted approximately 30 cm above the floor and a color camera stereo head located 97cm above the LMS. Only one camera is used for visual people detection. The camera is calibrated using the CALIB toolbox [5] while the transformation parameters between the camera and the LMS coordinate system are set by measuring the robots geometry.

Approach. In this section we use a sliding-window approach for people detection where we choose the HOG descriptor as feature and a linear SVM as classifier (see section 2). To achieve good generalization performance in various environments we decided to train the classifier on the INRIA people data set (see section 3). As expected the visual people detector already achieves good results. Figure 7 shows sample detections as well as typical false positive detections e.g. on partial people or vertical edge structures.

Many false positive detections do not fulfill simple constraints assuming that people usually walk on the floor and therefore the object scale is proportional to distance. This assumption can be formulated with the following two constraints to prune the space of valid hypotheses obtained from the HOG detection stage; (1): laser range measurements projected onto the image plane should hit the lower third (legs) of the detection window h_i. We denote the set of the associated range values that meet this condition with R_i. And (2): detection scale s_i of h_i is bounded by a factor proportional to the largest/smallest distance measurement found in R_i : $s^*/min(R_i) + c > s_i > s^*/max(R_i) - c$ where s^* is a scale estimate at 1m distance and c is a small constant accounting for errors in scale estimations. Both parameters are in pixel units and dependent on camera parameters. Since all detection hypotheses have the same aspect ratio we set s_i to the detection window width.We initially set s^* to 550 and c to 25. These values are estimated from a subset of the recorded data.

If a visual person hypothesis does not meet these constraints it is rejected. Figure 7(c) shows the effect of rejecting hypotheses that do not match these two constraints.



(a) (b) (c)

Figure 7. The PeopleBot Robot (a). Typical false positives from visual people detection (b). Rejection by simple range based constraints (c).

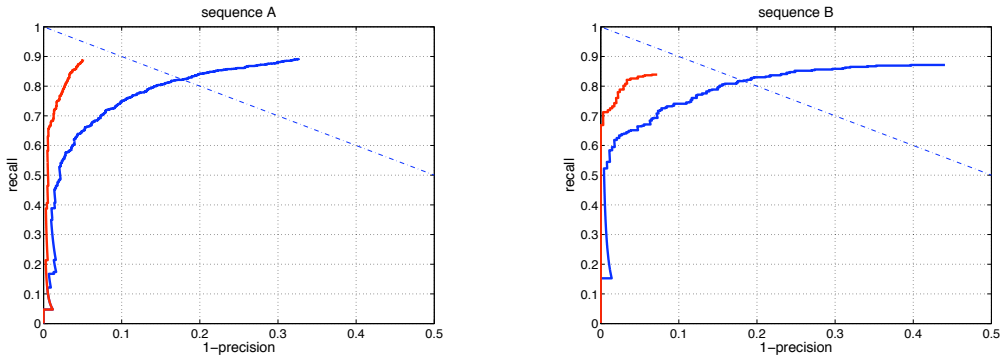
Evaluation. We evaluate this simple procedure on two sequences. Sequence (A) (samples shown in figures 7(b) and 9(c)) is recorded in an office sized room and sequence (B) (samples shown in figures 9(a) and 10) in a large foyer of a university building. Ground truth is annotated manually in form of bounding boxes and is quite complete in the sense that occluded people or people entering the visual field are also annotated if approximately half of the person is visible. As a consequence we cannnot expect to reach full recall. For both sequences the robot was placed to have a good visual view of the scene. Due to the nature of the concurrent distributed component architecture the sampling process for each sensor is asynchronous and tends to vary slightly with the overall system load. We aligned the laser and the vision sensor in a semi-automatic fashion such that each image frame is associated to the laser scan with the smallest temporal difference.

Sequence A. This sequence consists of 1023 image frames sampled at 2.5 Hz on average while laser recordings reach 20.8 Hz. The environment is an office sized room with people entering and leaving the room through two doorways. People might occlude each other and be occluded by the wall. Figure 8(a) shows the detection performance for this sequence.The HOG detector reaches a maximal recall of 89.2% with a precision of 67.3% The equal error rate (EER) is 82.2%. The use of laser range information clearly improves precision to 95% with a loss smaller than 0.25% in maximally achievable recall.

Sequence B. The second sequence consists of 124 images sampled at 0.7 Hz on average while laser recordings reach 37.5 Hz. In this sequence more people appear also at large scales so that they are not fully visible. The HOG detector reaches a maximal recall of 87.1% with a precision of 56% The EER is 81.7%. Laser range information improves precision to 92.8% at 83.9% recall (i.e. 3.2% loss in recall).

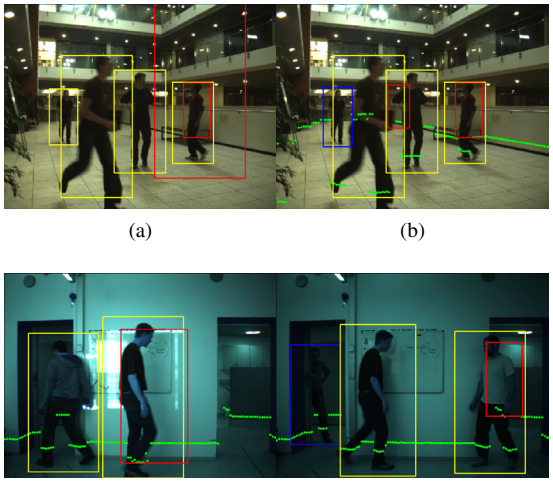
Loss of recall is a sign that the posed constrains are not necessarily true for all ground truth instances. This happens in cases where a true positive hypothesis occludes the laser which leads to rejection of a true positive detection at a smaller scale due to the missing laser readings. Not achieving full precision means that cases occur where false positive detections fulfill the laser constraints. This happens if multiple hypotheses at similar scales are found as true positive hypotheses or if a false positive hypothesis and laser range readings fulfill the constraints by chance. See Figure 9 for failure cases and figure 10 for sample detections.

Conclusion. Overall, in terms of EER, the proposed combination of camera and laser information improves precision/recall by 12.8% / 7 % on sequence A and 11.1% / 2.2 % on sequence B. This improvement is clearly significant and highly encouraging given the simplicity of the de-



(a) (b)

Figure 8. Detection performance for test sequences A and B. HOG detection in blue. Laser constrained HOG in red.



(a) (b)

(c) (d)

Figure 9. Visual detection only (a) & (c). Laser constrained detections (b) & (d). The laser range based constraints correct one false positive but also reject one true positive detection. True positive detections are marked yellow, missed objects blue and false detections are red.

scribed algorithm. We believe that this clearly demonstrates the potential to combine camera and laser information and that more elaborate algorithms should enable to improve performance further.

6. Conclusion and discussion

The primary aim of this paper was to give an overview of promising techniques for visual people detection (sections 2–4). In recent years the field has been moving rapidly thereby continuously improving detection performance. Given today's state-of-the-art in visual people detection it is clear however that the currently achievable performance is often neither sufficient nor satisfactory for many applications. In this last section we briefly discuss several research directions that have the potential to improve overall performance.

Motion cues. It is clear that human motion is an important cue for people detection. Quite surprisingly however, motion is seldom used for people detection. Notable exceptions are the work by Viola et. al [33], Dalal and Triggs [7] and Wojek et al. [36]. All three papers clearly demonstrate the potential gain when using motion information for visual people detection. However, we strongly believe that the current approaches still leave room for further improvement.

Integration of detection and tracking. Both detection and tracking people are challenging problems. People detectors have been shown to be able to locate pedestrians even in complex scenes, but false positives have remained frequent. Tracking methods are able to find a particular individual in image sequences, but are severely challenged by real-world scenarios such as crowded scenes. Therefore it is a promising research direction to combine the advantages

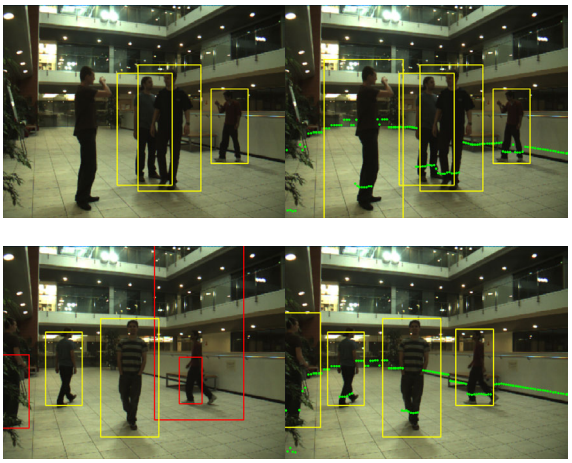


Figure 10. HOG hypotheses thresholded at EER on the left. Laser constrained hypotheses on the right.

of both detection and tracking in a single framework. In [1] we have proposed such an integrated framework that allows to detect and track multiple people in cluttered scenes with reoccurring occlusions. While this research direction is again largely under-explored we strongly believe that this is a highly promising route to pursue.

System integration. It seems clear that the integration of all of the above mentioned information into a single overall system has the potential to obtain an improved overall performance. Due to the complexity of this task however relatively view such systems exist. Probably the best known examples are the system by Gavrila and colleagues [14] and more recently the work by Ess and colleagues [8]. In these systems different components are integrated such as stereo and depth estimation, structure from motion, texture based classifiers and part-based people detectors.

Combination with other sensor modalities. Section 5 already demonstrated the potential of combining vision and laser information to improve overall detection performance. While this research direction has gained attention recently [26, 24, 38, 31] it is again under-explored and has the potential to enable robust people detection e.g. for robotics and automative applications.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. *CVPR 2008*.

[2] K. O. Arras, O. M. Mozos, and W. Burgard. Using boosted features for the detection of people in 2D range data. *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3402–3407, 2007.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. *NIPS*2000*.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *PAMI*, 24(4):509–522, 2002.

[5] J.-Y. Bouguet. Camera calibration toolbox, 2008.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR 2005*.

[7] N. Dalal., B. Triggs., and C. Schmid. Human detection using oriented hist. of flow and appearance. *ECCV*, 2006.

[8] A. Ess, K. Schindler, B. Leibe, and L. van Gool. Robust mulit-person tracking from a moving platform. *CVPR*, 2008.

[9] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. *CVPR*, 2000.

[10] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61:55–79, 2007.

[11] V. Ferrari, M. Marin, and A. Zisserman. Progressive search space reduction for human pose estimation. *CVPR 2008*.

[12] D. Forsyth and M. Fleck. Body plans. *CVPR*, 1997.

[13] D. Gavrila. Multi-feature hierarchical template matching using distance transforms. *Proceedings of the International Conference on Pattern Recognition*, v. 1, pp. 439–444, 1998.

[14] D. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *IJCV*, 73(1):41–59, 2007.

[15] S. Ioffe and D. Forsyth. Human tracking with mixtures of trees. *ICCV 2001*.

[16] T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pp. 169–184, Cambridge, MA, 1999. MIT Press.

[17] X. Lan and D. P. Huttenlocher. Beyond trees: Common-factor models for 2d human pose recovery. *ICCV 2005*.

[18] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. *CVPR*, pp. 878–885, 2005.

[19] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60:63–86, 2004.

[20] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10):1615–1630, 2005.

[21] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. *ECCV*, pp. 69–81, 2004.

[22] O. M. Mozos, C. Stachniss, and W. Burgard. Supervised learning of places from range data using AdaBoost. *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1742–1747, Barcelona, Spain, 2005.

[23] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38(1):15–33, 2000.

[24] A. Pronobis, O. Martínez Mozos, and B. Caputo. SVM-based discriminative accumulation scheme for place recognition. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA08)*, Pasadena, CA, USA, 2008.

[25] D. Ramanan. Learning to parse images of articulated objects. *NIPS*2006*.

[26] A. Rottmann, O. M. Mozos, C. Stachniss, and W. Burgard. Semantic place classification of indoor environments with mobile robots using boosting. *Proceedings of the National Conference on Artificial Intelligence*, pp. 1306–1311, Pittsburgh, PA, USA, 2005.

[27] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. *CVPR*, 2007.

[28] E. Seemann, B. Leibe, and B. Schiele. Multi-aspect detection of articulated objects. *CVPR*, pp. 1582–1588, 2006.

[29] E. Seemann and B. Schiele. Cross-articulation learning for robust detection of pedestrians. *DAGM*, 2006.

[30] A. Shashua, Y. Gdalyahu, and G. Hayun. Pedestrian detection for driving assistance systems: Single-frame classification and system level performance. *International Symposium on Intelligent Vehicles*, pp. 1–6, 2004.

[31] L. Spinello, R. Triebel, and R. Siegwart. Multimodal people detection and tracking in crowded scenes. *Proc. of The AAAI Conference on Artificial Intelligence (Physically Grounded AI Track)*, 2008.

[32] O. Tuzel, F. Porikli, and P. Meer. Human detection via classification on Riemannian manifolds. *CVPR*, 2007.

[33] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *CVPR*, 2003.

[34] C. Wojek, G. Dorko, A. Schulz, and B. Schiele. Sliding-windows for rapid object class localization: a parallel technique. *Pattern Recognition (DAGM) 2008*.

[35] C. Wojek and B. Schiele. A performance evaluation of single and multi-feature people detection. *Pattern Recognition (DAGM) 2008*.

[36] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. *CVPR 2009*.

[37] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. *ICCV*, 2005.

[38] Z. Zivkovic and B. Kröse. Part based people detection using 2d range data and images. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

A Trained System for Multimodal Perception in Urban Environments

Luciano Spinello, Rudolph Triebel and Roland Siegwart

Autonomous Systems Lab, ETH Zurich, Switzerland
email: {luciano.spinello, rudolph.triebel, roland.siegwart}@mavt.ethz.ch

***Abstract*— This paper presents a novel approach to detect and track multiple classes of objects based on the combined information retrieved from camera and laser rangescanner. Laser data points are classified using Conditional Random Fields (CRF) that use a set of multiclass Adaboost classified features. The image detection system is based on Implicit Shape Model (ISM) that learns an appearance codebook of local descriptors from a set of hand-labeled images of pedestrians and uses them in a voting scheme to vote for centers of detected people. We propose several extensions in the training phase in order to automatically create subparts and probabilistic shape templates, and in the testing phase in order to use these extended information to select and discriminate between hypothesis of different classes. Finally the two information are combined during tracking that is based on kalman filters with multiple motion models. Experiments conducted in real-world urban scenarios demonstrate the usefulness of our approach.**

I. Introduction

Urban environments are complex scenes where often multiple objects interact and move. In order to navigate and understand such environment a robot should be able to detect and track multiple classes of objects: most important pedestrians and cars. The ability to reliably detect these objects in real-world environments is crucial for a wide variety of applications including video surveillance and intelligent driver assistance systems. Pedestrians are particularly difficult to detect because of their high variability in appearance due to clothing, illumination and the fact that the shape characteristics depend on the view point. In addition, occlusions caused by carried items such as backpacks or briefcases, as well as clutter in crowded scenes can render this task even more complex, because they dramatically change the shape of a pedestrian. Cars are large objects that dramatically change their shape with respect to the viewpoint: for example a side view of a car is totally different from its back view. Shape symmetries can easily create false detections and shadows can drive off detection systems.

Our goal in this paper is to detect pedestrians and cars and localize them in 3D at any point in time. In particular, we want to provide a position and a motion estimate that can be used in a mobile robotic application. The real-time constraint makes this task particularly difficult and requires faster detection and tracking algorithms than the existing approaches. Our work makes a contribution into this direction. The approach we propose is multimodal in the sense that we use laser range data and images from a camera cooperatively. This has the advantage that both *geometrical structure* and *visual appearance* information are available for a more robust detection.

Managing detection of multiple classes in laser range data is a complex task due the problem of data segmentation. Often range data is grouped in consistent clusters and then classified, using heuristic rules and therefore creating a strong prior in the algorithm. In this paper, we propose an elegant solution to train and classify range data using Conditional Random Fields (CRF) through the use of a boosted set of features. Moreover each scan point will be labeled with a probability of owning to a certain class. In order to manage occlusions in complex visual scenarios a new extension of the Implicit Shape Model (ISM) for camera data classification has been developed. Finally, each detected object is tracked using a greedy data association method and multiple Extended Kalman Filters that use different motion models. This way, the filter can cope with a variety of different motion patterns for several persons simultaneously. In particular, the major contributions of this work are:

- An improved version of the image-based object detector by Leibe *et al.* [14]. It consists in several extensions to the Implicit Shape Model (ISM) in the training step, in the detection step and in the capability of coping with multiple classes. We introduce an automatic subpart extraction that is used to build an improved hypotheses selection, the concept of *superfeatures* that define a favorable feature selection that maintaining information richness. Moreover we introduce an automatically generated probability template map to ease the multiclass hypothesis selection.
- The combined use of Conditional Random Fields and camera detection to track objects in the scene.

This paper is organized as follows. The next section describes previous work that is relevant for our approach. Then, we give a brief overview of our overall object detection and tracking system. The following section presents in detail our detection method based on conditional random fields for 2D laser range data. Then, we introduce the implicit shape model (ISM) and present our extensions. Subsequently, we explain our EKF-based tracking algorithm. Finally, we present experiments and conclude the paper.

II. Previous Work

Several approaches can be found in the literature to identify a person in 2D laser data including analysis of local

minima [20], [24], geometric rules [26], or a maximum-likelihood estimation to detect dynamic objects [10], or learning AdaBoost classifiers from a set of geometrical features extracted from segments [2] or from Delaunay neighborhoods [21]. Most similar to our work is the work of [5] that makes use of a Conditional Random Field in order to label points to extract objects from a collection of laser scans.

In the area of image-based people detection, there mainly exist two kinds of approaches (see [9] for a survey). One uses the analysis of a *detection window* or *templates* [8], [25], the other performs a *parts-based* detection [6], [11]. Leibe *et al.* [14] presented an image-based people detector using *Implicit Shape Models* (ISM) with excellent detection results in crowded scenes. An extension of this method that proposes a feature selection enhancement and a nearest neighbor search optimization has been already shown in [22][23].

Existing people detection methods based on camera *and* laser rangefinder data either use hard constrained approaches or hand tuned thresholding. Zivkovic and Kröse [27] use a learned leg detector and boosted Haar features extracted from the camera images to merge this information into a parts-based method. However, both the proposed approach to cluster the laser data using Canny edge detection and the extraction of Haar features to detect body parts is hardly suited for outdoor scenarios due to the highly cluttered data and the larger variation of illumination encountered there. Therefore, we use an improved clustering method for the laser scans and SIFT features for the image-based detector. Schulz [19] uses probabilistic exemplar models learned from training data of both sensors and applies a Rao-Blackwellized particle filter (RBPF) in order to track the person's appearance in the data. However, in outdoor scenarios lighting conditions change frequently and occlusions are very likely, which is why contour matching is not appropriate. Moreover, the RBPF is computationally demanding, especially in crowded environments. The work of Douillard [5] also uses image features in order to enhance object detection but it doesn't explicitly handle occlusions and separate image detection hypotheses.

III. Overview of the method

Our system is composed of three main components: an appearance based detector that uses the information from camera images, a 2D-laser based detector providing structural information, and a tracking module that uses the combined information from both sensor modalities and provides an estimate of the motion vector for each tracked object. The laser based detection applies a Conditional Random Field (CRF) on a boosted set of geometrical and statistical features of 2D scan points. The image based detection system extends the multiclass version of the Implicit Shape Model (ISM)[13] and uses Shape Context descriptors [3] computed at Harris-Laplace and Hessian interest points. It also uses the laser based detection result projected into the image to constrain the position and scale of the detected objects. Then, the tracking module applies an Extended Kalman Filter (EKF), to the combined detection results where two different motion models are implemented to account for a high variety of possible object motions. In the following, we describe the particular components in detail.

IV. Appearance Based Detection

Our image-based people detector is mostly inspired by the work of Leibe *et al.* [14] on scale-invariant Implicit Shape Models (ISM). In summary, an ISM consists in a set of local region descriptors, called the *codebook*, and a set of displacements and scale factors, usually named *votes*, for each descriptor. The idea of the votes is that each descriptor can be found at different positions inside an object and at different scales, and thus a vote points from the position of the descriptor to the center of the object as it was found in the training data set. To obtain an ISM from labeled training data, all descriptors are first clustered, usually using agglomerative clustering, and then the votes are computed by adding the scale and the displacement of the objects' center to the descriptors in the codebook. For the detection, new descriptors are computed on a given test image and matched against the descriptors in the codebook. The votes that are cast by each matched descriptor are collected in a 3D *voting space*, and a maximum density estimator is used to find the most likely position and scale of an object.

A. Extensions to ISM

In the past, we presented already several improvements of the standard ISM approach (see [23], [22]). Here, we show some more extensions of ISM to further improve the classification results. These extensions concern both the learning and the detection phase and are described in the following.

1) ISM Extensions in the Learning Phase:

a) Learning of Subparts: The aim of this procedure is to enrich the information that is obtained from the voters by distinguishing between different object subparts from which the vote was cast. We achieve this by learning a circular histogram of interest points from the training data set for a given object class. The number of bins of this histogram is determined automatically by using K-means clustering. The number K of clusters is obtained using the Bayesian Information Criterion (BIC). Note that this subpart extraction does not guarantee a semantical subdivision (i.e.: legs, arms in the case of pedestrians) of the object but it is interesting to see that it nevertheless resembles this automatically without manual interaction by the user (see Fig. 1, left).

b) Applying a Template Mask: The idea here is to extract a common segmentation mask from the training data for each object by averaging over all masks from the particular object instances. This mask is later used to discard outlier voters by overlaying the mask at the hypothetical center of the object. Chamfer matching has been widely used in literature [4] to compute such a mask. However, it heavily depends on a robust detection of the contour edges and is strongly affected by noise. A more robust method is to build a

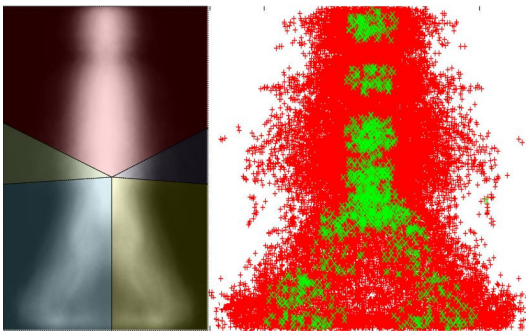


Fig. 1. **Left**: Probabilistic template and overlayed subparts are both automatically computed from the training set (in this case on the class 'pedestrian'). It is important to notice that even though the subparts are computed without a semantic subdivision, their segmentation shows legs, arms and upper body.**Right**: Superfeatures are stable features in image and descriptor space. This figure depicts Shape Context descriptors with Hessian Interest point (in red) in the case of pedestrian class. In green are depicted the selected superfeatures.

probabilistic template map from the individual segmentation masks in the training set. All the segmentation masks are collected, centered with respect to their center of gravity and averaged. Strong responses (common areas of the same objects) have high probability, whereas various details are softened in the average but still kept.

2) Learning Superfeatures: The original ISM does not perform feature selection but it maintains the complete probability distribution generated by extracted features of the training set. This has the disadvantage to potentially generate false positive due to inevitable feature mismatches. We here propose a method to drive the detection while still maintaining information richness. The idea is to find good features in the image space (namely $\langle x, y, scale \rangle$) and descriptor space ($n$-d space) that could vote for the object center with more weight to ease the hypothesis selection. The procedure can be sketched in three steps.

1) Interest points of the entire training dataset are collected.
2) Dense areas of interest points reflect a high informative content. We employ mean shift mode seeking with a uniform kernel in order to locate such areas.
3) On each convergence point descriptors are collected in pools. These pools are clustered using unsupervised clustering with average linkage in order to group closely similar features. We use the best 50% of the resulting groups (ranked by quantity) and collect them as *superfeatures*.

Noticeably, the resulting superfeatures inherently reflect the skeleton of the objects and constitute key points in the shape of the objects (see Fig. 1, right).

B. ISMe: extensions in testing phase

In this subsection we explain how we combine the richer learning information in order to obtain a better detection.

1) Using superfeatures: Superfeatures and features vote for object centers in the same voting space: the votes generated by the first are bigger than the latter. The resulting hypothesis score is enriched by their support. In visually simple scenes it is possible to apply just superfeature codebooks in order to obtain a very fast detection.

2) Using subparts and prob. template in the cost function: Each hypothesis is now defined by an angular histogram in which the bins are defined by the subparts. Moreover, the probabilistic template is used to prune feature matches that lie far outside the probabilistic shape (that is scaled according to the hypothesis). In order to determine which of the hypotheses better represents an object of a given class, we use a maximum likelihood estimation method. In particular, we solve:

$$\mathcal{H}_s = \operatorname{argmax} p(\mathcal{H}|\Theta), \tag{1}$$

where $\mathcal{H}$ represents the set of hypotheses and Θ is the feature assignment. In order to achieve the solution we consider pairwise comparisons. Given each pair of hypotheses h_a and h_b, their relative histograms $\mathcal{W}_a = \{w_1^a, ..., w_q^a\}$, and $\mathcal{W}_b$ we compute:

$$V = \sum_i^q v_i \tag{2}$$

where

$$v_i = \begin{cases} 1 & \text{if} \quad w_i^a > w_i^b \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

Then a simple sign condition is used to check which of the two hypotheses is the best. If we perform this simple and fast comparison on the set $\mathcal{H}$, we obtain $h_{max} = \operatorname{argmax} p(\mathcal{H}|\Theta)$ and put it in the selected hypothesis set $\mathcal{H}_c$.

3) Discriminate between object classes: In the previous subsection we explained how we selected the best object hypothesis for each class. Here we explain how we discriminate among hypothesis of different classes. In order to not bias the multiclass detection towards a class that has more features or codebook occurrences we used a common measure to do hypothesis selection. This comes from the probabilistic template area ratio. Each assigned feature for a certain hypothesis occupies a scaled square area in the probabilistic template. The ratio of the occupied area on the total object area is the score of each class hypothesis. For each object class hypothesis a score s_i is computed taking into account the overlapping area (if present) between hypotheses of different classes:

$$s_i = r_i - \frac{\Delta o}{\#o} \tag{4}$$

where r_i is the area ratio and Δo is the overlap ratio of the areas, and $\#o$ is the number of overlaps. The best score s_i defines the current winning object hypothesis. The features involved in the voting of this hypothesis are then removed from the voting space and the selection process (subparts voting and object class selection) continues until a detection with a minimum strength d_t is available.

This two step process is necessary to handle occlusions and multiple classes in a computationally feasible time: each hypothesis competes with the rest of its class to become the best hypothesis of its class. Then it is evaluated against all

the other candidates of the other class and then, if it is the case, selected.

V. Structure Based Detection

For the detection of objects in 2D laser range scans, several approaches have been presented in the past. Most of theses approaches have the disadvantage that they disregard the conditional dependence between data points in a close neighborhood: the fact that the label y_i of a given scan point $\mathbf{z}_i$ is more likely to be y_j if we know that y_j is the label of $\mathbf{z}_i$'s neighbor $\mathbf{z}_j$ is not reflected. One way to model this conditional independence is to use Conditional Random Fields (CRFs) [12], as has been shown by Douillard *et al.*[5]. CRFs represent the conditional probability $p(\mathbf{y} \mid \mathbf{z})$ using an undirected cyclic graph, in which each node is associated with a hidden random variable y_i and an observation $\mathbf{z}_i$. In our case, the y_i is a discrete label that ranges over 2 different classes (pedestrian and car) and the observations $\mathbf{z}_i$ are 2D points in the laser scan. Assuming a maximal clique size of 2 for the graph, we can compute the conditional probability of the labels $\mathbf{y}$ given the observations $\mathbf{z}$ as:

$$p(\mathbf{y} \mid \mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_{i=1}^{N} \varphi(\mathbf{z}_i, y_i) \prod_{(i,j)\in\mathcal{E}} \psi(\mathbf{z}_i, \mathbf{z}_j, y_i, y_j), \quad (5)$$

where $Z(\mathbf{z}) = \sum_{\mathbf{y}'} \prod_{i=1}^{N} \varphi(\mathbf{z}_i, y_i') \prod_{(ij)\in\mathcal{E}} \psi(\mathbf{z}_i, y_i', y_j')$ is usually called the *partition function* and $\mathcal{E}$ is the set of edges in the graph. To determine the node and edge potentials φ and ψ we use the log-linear model:

$$\varphi(\mathbf{z}_i, y_i) = e^{\mathbf{w}_n \cdot \mathbf{f}_n(\mathbf{z}_i, y_i)},\ \psi(\mathbf{z}_i, \mathbf{z}_j, y_i, y_j) = e^{\mathbf{w}_e \cdot \mathbf{f}_e(\mathbf{z}_i, \mathbf{z}_j, y_i, y_j)}$$

where $\mathbf{f}_n$ and $\mathbf{f}_e$ are feature functions for the nodes and the edges in the graph, and $\mathbf{w}_n$ and $\mathbf{w}_e$ are the feature weights that are determined in the training phase. The computation of the partition function Z is intractable due to the exponential number of possible labelings $\mathbf{y}'$. Instead, we compute the *pseudo-likelihood*, which approximates $p(\mathbf{y} \mid \mathbf{z})$ and is defined by the product of all likelihoods computed on the *markov blanket* (direct neighbors) of node i.

$$pl(\mathbf{y} \mid \mathbf{z}) = \prod_{i=1}^{N} \frac{\varphi(\mathbf{z}_i, y_i) \prod_{\mathbf{z}_j \in \mathcal{N}(\mathbf{z}_i)} \psi(\mathbf{z}_j, \mathbf{z}_i, y_j, y_i)}{\sum_{\mathbf{y}'} \left(\varphi(\mathbf{z}_i, y_i') \prod_{\mathbf{z}_j \in \mathcal{N}(\mathbf{z}_i)} \psi(\mathbf{z}_j, \mathbf{z}_i, y_i', y_j') \right)}$$

Here, $\mathcal{N}(\mathbf{z}_i)$ denotes the set of direct neighbors of node i. In the training phase, we compute the weights $\mathbf{w}_n$ and $\mathbf{w}_e$ that minimize the negative log pseudo-likelihood together with a Gaussian shrinkage prior as in [18]:

$$L(\mathbf{w}) = -\log pl(\mathbf{y} \mid \mathbf{z}) + \frac{(\mathbf{w} - \hat{\mathbf{w}})^T (\mathbf{w} - \hat{\mathbf{w}})}{2\sigma^2} \quad (6)$$

For the minimization of L, we use the L-BFGS gradient descent method [15]. Once the weights are obtained, they are used in the inference phase to find the labels $\mathbf{y}$ that maximize Eq. (5). Here, we do not need to compute the partition function Z, as it is not dependent on $\mathbf{y}$. We use max-product loopy belief propagation to find the distributions of each label y_i. The final labels are then obtained as those that are most likely for each node.

A. Node and Edge Features

As node features $\mathbf{f}_n$ we use a set of statistical and geometrical features such as height, width, circularity, standard deviation, kurtosis, etc. (for a full list see [21]). We compute these features in a local neighborhood around each point, which we determine by jump distance clustering. We can then use these features as an input to the CRF classification algorithm. However as stated in [18], and also from our own observation, the CRF is not able to handle non-linear relations between the observations and the labels, which is a consequence of the log-linear model described above. To overcome this problem, we apply AdaBoost [7] to the node features and use the outcome of AdaBoost as features for the CRF. For our particular classification problem with multiple classes, we train one binary AdaBoost classifier for each class against the others. As a result, we obtain a set of weak classifiers h_i (decision stumps) and corresponding weight coefficients α_i so that the sum

$$g_k(\mathbf{z}) = \sum_{i=1}^{M} \alpha_i h_i(\mathbf{f}(\mathbf{z})) \quad (7)$$

is positive for observations that are assigned with the class label k and negative otherwise. To obtain values between 0 and 1 we apply the inverse logit function $l(x) = (1 + \exp(-x))^{-1}$, which has a sigmoid shape and ranges between 0 and 1, to each value g_j. We do this for two reasons: First we obtain values that can be interpreted as likelihoods of corresponding to class k. Second, by applying the same technique also for the edge features, the resulting potentials are better comparable. The resulting node features are then computed as

$$\mathbf{f}_n(\mathbf{z}_i, y_i) = l(g_{y_i}(\mathbf{z}_i)), \quad (8)$$

i.e. the scalar component of the vector $l(\mathbf{g})$ that corresponds to the class with label y_i. For the edge features, we don't apply AdaBoost, but instead compute two values, namely the Euclidean distance d_{ij} between the points $\mathbf{z}_i$ and $\mathbf{z}_j$ and a value g_{ij} defined as

$$g_{ij}(\mathbf{z}_i, \mathbf{z}_j) = \text{sign}(g_i(\mathbf{z}_i) g_j(\mathbf{z}_j))(|g_i(\mathbf{z}_i)| + |g_j(\mathbf{z}_j)|) \quad (9)$$

This feature has a high value if both $\mathbf{z}_i$ and $\mathbf{z}_j$ are classified equally (its sign is positive) and low otherwise. Its absolute value is the sum of distances from the decision boundary of AdaBoost, which is given by $g(\mathbf{z}) = 0$. We define the edge features then as follows:

$$\mathbf{f}_e(\mathbf{z}_i, \mathbf{z}_j, y_i, y_j) = \begin{cases} (l(d_{y_i,y_j}) \quad l(g_{y_i,y_j})^T & \text{if } y_i = y_j \\ (0 \quad 0)^T & \text{otherwise} \end{cases} \quad (10)$$

Here, we omitted the arguments $\mathbf{z}_i$ and $\mathbf{z}_j$ of the functions d_{ij} and g_{ij} for brevity. The intuition behind Eq. (10) is that

edges that connect points with equal labels have a non-zero feature value and thus yield a higher potential. The latter is sometimes referred to as the generalized Potts model (see [1], [17]).

a) Connectivity: Nowadays many laser scanners have multilayer scanning capabilities. The CRF connectivity is defined by a separate Delaunay triangulation for each layer. Between layers connectivity is assured by connecting points located in the same vertical. This assures a good layer connection for the flow of BP and lessen the arc count with respect to a full triangulation.

VI. Tracking objects for sensor fusion

In order to fuse the information coming from both sensors (camera and laser) and to simultaneously keep track of the object we use an EKF based tracking system, first introduced in [23]. Here, each object is tracked with several motion models (in this case: brownian motion and linear velocity) in order to cope with pedestrian and car movements. We perform tracking in the laser data, therefore camera detections are projected and assigned to segments in the laser data. In order to reliably track wide objects, like cars, tracking single segments are not enough. Single segments tend to be spatially very unstable due to the noise present in outdoor environments and the scatter resulting from the distance with respect to the observer. We therefore group segments with the same class label using Delaunay triangulation and a trim distance rule. The resulting cluster will have a more stable position and a probability of being a class that is the average of its members. Each Kalman filter state ($\langle x, y, (v_x, v_y)\rangle$) is augmented with N states where N is the number of classes present in the detector. Indeed, the observation vector z fed to the tracking system consists of the position of the cluster and the class label probability. The matrix H that models the observations to mapping in the Kalman Filter $x = Hz$ is defined by $H = [H_{lsr}; H_{cam}]$ in order to manage multiple inputs from different sensors.

VII. Experimental Results

A car equipped with several active and passive sensors is used to acquire the datasets. In particular, we use a monocular camera in combination with a 2D laser range finder in front of the car. An accurate camera-laser synchronization and calibration has been developed for this work.

A. Image training datasets

The scope of this paper is to detect pedestrians and cars, we therefore used a pedestrian dataset and three different datasets for cars: front view, side view, back view. The class car itself consists in multiple classes because of its different visual appearance with respect to the viewpoint. The pedestrian dataset consists of 400 images of persons with a height of 200 pixels at different positions and dressed with different clothing and accessories such as backpacks and hand bags in a typical urban environment. Each car dataset consists in a set of 100 pictures taken in several urban scenes with occlusions due to people or traffic signs.

B. Laser training datasets

The laser detector has been trained using 203 annotated laser scans containing clutter, pedestrians and cars. There is not distinction between car views in the laser detector due to a not dramatic viewpoint change in the range data. The range data is organized in 4 layers with a relative orientation of $0.8°$. Each layer has a resolution of $0.25\circ$ and maximum range of $30m$.

C. Qualitative and quantitative multiclass results

In order to determine the performance of our detector we created two datasets consisting of cars and pedestrians. The image based detection uses Shape context descriptors [3] from Hessian-Laplace and Harris-Laplace [16] interest point. The quantitative results of the performance of pedestrian based image detection are shown in the precision-recall graph of Fig. 2-left. In the graph is shown a comparison with respect to a naive ISM implementation that does not uses hypothesis selection, Adaboost based Haar detector and our previous version of the image detector (labeled as ISMe1.0). The performance increase of our approach is mainly related to the introduction of the new hypothesis selection system. The performance of image based detection for cars is shown in Fig. 2-middle where a comparison with ISM and ISMe1.0 is shown. For clarity the results are averaged between the three different views of the class car. In general we can notice from the results that pedestrian classification is harder than car classification due to shape complexity and flexibility.

In order to justify our approach for laser range data detection we evaluated CRF against Boost classifier that uses the same set of features, the resulting precision-recall graph is shown in Fig. 2-right for pedestrian and in fig. 3-left for cars. Then we evaluated the current performance of combining the information togheter. A very informative way of showing the potential of our method is shown in the two graph of Fig. 3-middle and Fig. 3-right in which we show that combining the two information increases the hit rate and decreases the false positives. We show some qualitative results extracted from the testing datasets in Fig. 4-right in which cars are correctly detected from both sensors but the pedestrian is detected only with the laser and not with the camera due to its pose configuration and its visual neighborhood. Another result is shown in Fig. 4-left in which the camera classifier detects a false positive located on vertical structures of the trolleybus and detects the person on the scooter as a pedestrian due its visual similarity. Thanks to the structure information obtained from the laser the system can discriminate the false positive. Moreover, we show qualitative tracking results in Fig. 5, Fig. 7, Fig. 6 where passing cars and a crossing pedestrian are correctly tracked using multiple sensor information.

VIII. Conclusions

In this paper we presented a method to reliably detect and track multiple classes (cars and pedestrian) in outdoor scenarios using 2D laser range data and camera images. We showed that the overall performance of the system is

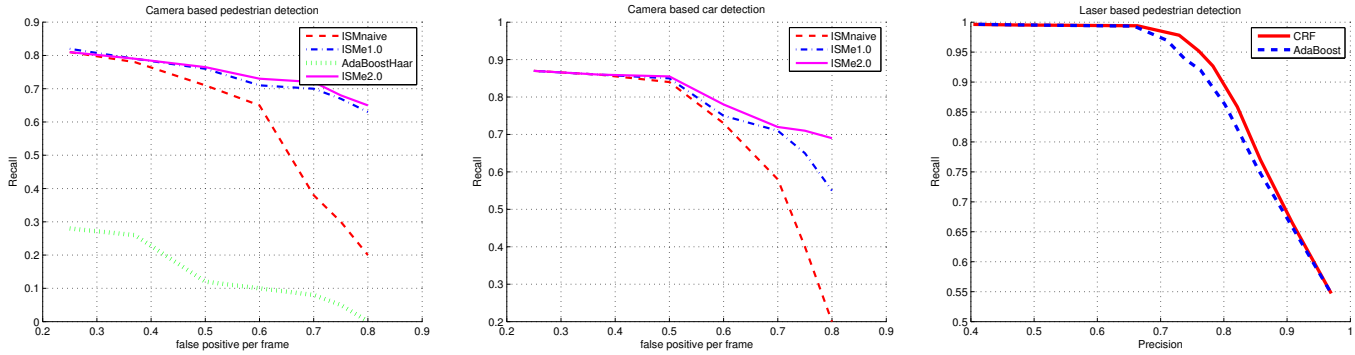


Fig. 2. **Left**:Precision-recall graph for image based pedestrian detection. Performance comparison is shown between ISM, our previous extension ISMe1.0 and Adaboost Haar based detector. **Middle**: Precision-recall graph for image based car detection. Performance comparison is shown between ISM, our previous extension ISMe1.0 and ISM **Right**: Precision-recall graph for laser range data based pedestrian detection. Performance comparison is shown between CRF and a Boost based approach

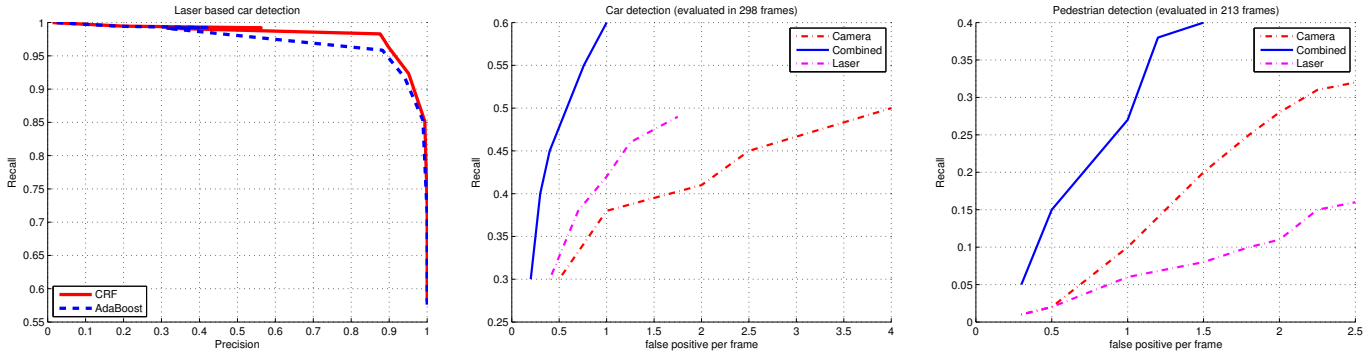


Fig. 3. **Left**: Precision-recall graph for laser range data based car detection. Performance comparison is shown between CRF and a Boost based approach. **Middle**: Recall-false positive per frame for camera-laser information fusion for car detection. A comparison of camera and laser is shown in figure. **Right** Recall-false positive per frame for camera-laser information fusion for pedestrian detection. A comparison of camera and laser is shown in figure.

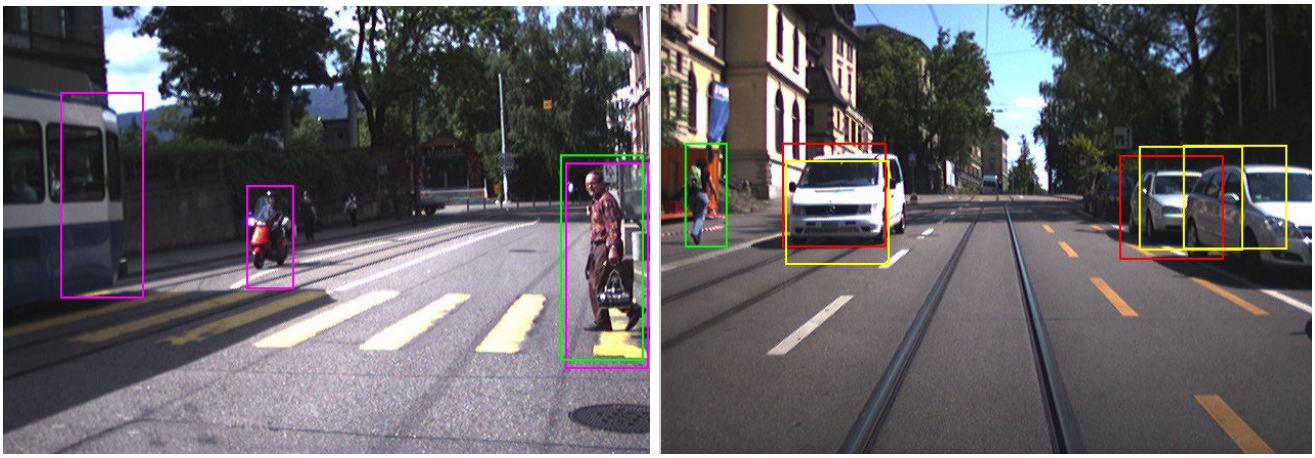


Fig. 4. Detections from multiple sensors. Green: laser based pedestrian detections; Yellow: laser based car detections; Magenta: camera based pedestrian detection; Red: camera based car detection

Fig. 5. Tracking cars in an intersection. A bounding box surrounds the tracked object with annotated distance and a colored marker that refers to the track in the laser plane.

Fig. 6. Tracking a pedestrian that crosses the road. A bounding box surrounds the tracked object with annotated distance and a colored marker that refers to the track in the laser plane. In the laser plane it is visible a false track associated with one steady detection of a cylinder concrete by the laser based detector. For clarity, the laser tracked cluster is plotted into the image (green points).

Fig. 7. Tracking cars in an intersection. A bounding box surrounds the tracked object with annotated distance and a colored marker that refers to the track in the laser plane. It is important to notice that also in case of the extreme closeup of the truck the track is still maintained

improved using a multiple sensor system. We presented several novel extensions to the ISM-based image detection in order to cope with multiple classes. We showed that a system based on CRF has better performance than a simpler Adaboost based classifier and presented tracking results on combined data. Finally, we presented experimental results on real-world data that point out the usefulness of our approach.

REFERENCES

[1] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *IEEE Conf. on Comp. Vis. and Pat. Recog. (CVPR)*, pages 169–176, 2005.

[2] K. O. Arras, Ó. M. Mozos, and W. Burgard. Using boosted features for the detection of people in 2d range data. In *IEEE Int. Conf. on Rob. & Autom. (ICRA)*, 2007.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. volume 24, pages 509–522, 2002.

[4] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. on Pattern Analysis & Machine Intelligence*, 10.

[5] B. Douillard, D. Fox, and F. Ramos. Laser and vision based outdoor object mapping. In *Robotics: Science and Systems (RSS)*, Zurich, Switzerland, June 2008.

[6] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *IEEE Conf. on Comp. Vis. and Pat. Recog. (CVPR)*, pages 66–73, 2000.

[7] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[8] D. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *IEEE Int. Conf. on Computer Vision (ICCV)*, 1999.

[9] D. M. Gavrila. The visual analysis of human movement: A survey. *Comp. Vis. and Image Und. (CVIU)*, 73(1):82–98, 1999.

[10] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *IEEE Int. Conf. on Rob. & Autom. (ICRA)*, 2003.

[11] S. Ioffe and D. A. Forsyth. Probabilistic methods for finding people. *Int. Journ. of Comp. Vis.*, 43(1):45–68, 2001.

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmentation and labeling sequence data. In *Proc. of the INt. Conf. on Machine Learning (ICML)*, 2001.

[13] B. Leibe, N. Cornelis, K. Cornelis, and L. V. Gool. Dynamic 3d scene analysis from a moving vehicle. In *IEEE Conf. on Comp. Vis. and Pat. Recog. (CVPR)*, 2007.

[14] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *IEEE Conf. on Comp. Vis. and Pat. Recog. (CVPR)*, pages 878–885, Washington, DC, USA, 2005. IEEE Computer Society.

[15] D. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Math. Programming*, 45(3, (Ser. B)), 1989.

[16] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[17] R. B. Potts. Some generalized order-disorder transformations. *Proc. Cambridge Phil Soc.*, 48, 1952.

[18] F. Ramos, D. Fox, and H. Durrant-Whyte. Crf-matching: Conditional random fields for feature-based scan matching. In *RSS*, 2007.

[19] D. Schulz. A probabilistic exemplar approach to combine laser and vision for person tracking. In *Robotics: Science and Systems (RSS)*, Philadelphia, USA, August 2006.

[20] D. Schulz, W. Burgard, D. Fox, and A. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *Int. Journ. of Robotics Research (IJRR)*, 22(2):99–116, 2003.

[21] L. Spinello and R. Siegwart. Human detection using multimodal and multidimensional features. In *IEEE Int. Conf. on Rob. & Autom. (ICRA)*, 2008.

[22] L. Spinello, R. Triebel, and R. Siegwart. Multimodal detection and tracking of pedestrians in urban environments with explicit ground plane extraction. In *IEEE Int. Conf. on Intell. Rob. and Sys. (IROS)*, 2008.

[23] L. Spinello, R. Triebel, and R. Siegwart. Multimodal people detection and tracking in crowded scenes. In *Proc. of The AAAI Conference on Artificial Intelligence*, July 2008.

[24] E. A. Topp and H. I. Christensen. Tracking for following and passing persons. In *IEEE Int. Conf. on Intell. Rob. and Sys. (IROS)*, 2005.

[25] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *IEEE Int. Conf. on Computer Vision (ICCV)*, page 734, Washington, DC, USA, 2003. IEEE Computer Society.

[26] J. Xavier, M. Pacheco, D. Castro, A. Ruano, and U. Nunes. Fast line, arc/circle and leg detection from laser scan data in a player driver. In *IEEE Int. Conf. on Rob. & Autom. (ICRA)*, pages 3930–3935, 2005.

[27] Z. Zivkovic and B. Kröse. Part based people detection using 2d range data and images. In *IEEE Int. Conf. on Intell. Rob. and Sys. (IROS)*, San Diego, USA, November 2007.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Multi-target Tracking on a Large Scale: Experiences from Football Player Tracking

J. Sullivan, P. Nillius and Stefan Carlsson,
Royal Institute of Technology,
Stockholm, Sweden.

Abstract

Multi-target tracking requires locating *the targets and* labeling *their identities. The latter is a challenge when many targets, with indistinct appearances, frequently occlude one another, as in football and surveillance tracking. We present an approach to solving this labeling problem.*

When isolated, a target can be tracked and its identity maintained. While, if targets interact this is not always the case. We build a track graph which denotes when targets are isolated and describes how they interact. Measures of similarity between isolated tracks are defined. The goal is to associate the identities of the isolated tracks, by exploiting the graph constraints and similarity measures.

We formulate this as a Bayesian network inference problem, allowing us to use standard message propagation to find the most probable set of paths in an efficient way. The high complexity inevitable in large problems is gracefully reduced by removing dependency links between tracks. We apply the method to a 10 min sequence of an international football game and compare results to ground truth.

1. Introduction

A multi-target tracking system capable of analyzing hours of footage reliably and robustly could potentially help automate many useful applications. There are numerous situations involving people/objects moving and interacting in a particular domain where the tracks of the targets over time provide a rich source of information for analysis of behavior. Such domains include - traffic-pedestrian junctions, travelers at airports, insect/animal tracking and team games.

However, automatic visual multi-target tracking in such domains with frequent interactions is a challenging problem (even when only considering instances with favorable viewing conditions). Given long enough sequences, situations will arise where it is not possible to reliably maintain a target's identity, when it occludes and/or is occluded by other targets, using continuity of appearance or motion alone. Some form of identity re-initialization is required when the interacting targets separate. This re-initialization can take the form of linking tracks before and after the interaction based on matching certain properties of the tracks involved. This in essence is the approach taken in this paper.

We see multi-target tracking as a two-stage process, when there are no real-time constraints. Initially targets are detected and tracked using background subtraction and continuity of motion constraints. When two or more targets meet and cannot be disambiguated a new track is formed and follows this target group. The process is repeated for all targets throughout the sequence. The result is a *track graph* with the different tracks as nodes and edges denoting how the tracks split and merge into new tracks.

In the second stage we try to find each target's path through the graph. This is achieved by exploiting the constraints imposed by the graph structure and by the feature vectors extracted to describe the appearance (e.g. image intensity, gait patterns) of each track. We view this as an inference problem where we want to find the most likely set of paths for the targets given the appearance of the tracks. This can be solved efficiently using Bayesian network inference.

For long sequences, with many targets, finding the global optimum of the resulting posterior becomes intractable due to the combinatorial explosion that occurs with the numerous split and merge situations. We solve this by reducing the dependencies between the tracks. In effect it means that similarities between tracks are only used for tracks within a certain time window. The size of this time window can be set dynamically to meet set criteria for complexity and memory use.

Over the last couple of years, many algorithms and results have been presented [7, 4] with regard to the problem of multiple object tracking. Prevalent are algorithms based on kalman filtering [12, 6], advanced techniques of particle filtering [11, 10, 9, 3] and multiple-hypothesis trackers [4]. The quality of the results presented though improving have yet to be shown working robustly on long sequences (>30 secs). Therefore, one of our major contributions is that we evaluate the performance of our method on a continuous 10

minute clip of an international football match and demonstrate its viability in solving large scale problems. The results obtained are promising.

1.1. Paper Overview

The paper is organized as follows. Section 2.1 provides a more detailed review of the *track graph*, mentioned in the introduction, the assumed starting point of our target linking algorithm. Section 2.2 describes the problem we wish to solve, of linking the identities of the nodes in our track graph. In Section 2.3 the solution space, imposed by the track graph, is defined and parameterized. Section 2.4 states the problem as an inference problem and shows how Bayesian network inference can be used to find the solution. For large problems containing thousands of nodes it is necessary to find an approximate solution. Section 2.4.1 discusses how this can be done by assuming independence between nodes distant in time. Section 3 reports on applying our method to football tracking. The experimental setup is described, as well as a brief review of how the track graph is constructed. The results of the path finding are then presented. To finish conclusions are made focusing on the quality of the results obtained and upon the scalability and generic nature of the solution put forward in the paper. Also discussed are possible improvements and future avenues of research involving combining unsupervised clustering and our path finding algorithm to provide a complete solution to the labeling problem.

2. Linking Identities in the Track Graph

2.1. Preliminaries

The theory in this paper assumes we have access to a track graph summarizing the interactions that occur between the targets in the sequence being analyzed. Therefore, before proceeding further we must introduce more formally the concept of the *track graph*. Each node in the graph represents a track. A track is a temporal sequence of image regions, one per frame (see figure 1). Each region corresponds to the spatial extent of one or more targets. During a track neither the number of targets it represents changes nor do the identities of these targets. The edges in the graph indicate when

- the targets from separate tracks merge (due to partial occlusion) to begin a new track or
- the targets in a track separate/split to begin several new tracks, each with fewer targets than the parent one.

Figure 2 displays a small example of such a *track graph*. The white nodes indicate tracks of a single target and grey those representing multiple targets.

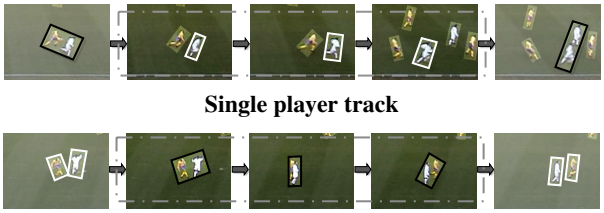


Figure 1. **Single and multiple target tracks** from a football game. The top row shows a single target track, shown in white. The bottom row is a multiple target track, shown in black. Tracks are sandwiched between interactions with other tracks. During a track the number of targets involved and their identities remain fixed.

There are, of course, numerous possible ways to obtain this graph [1]. For now though this issue is set aside and assumed to have been solved, however, we revisit it in section 3 while reviewing the methods put forward in [1].

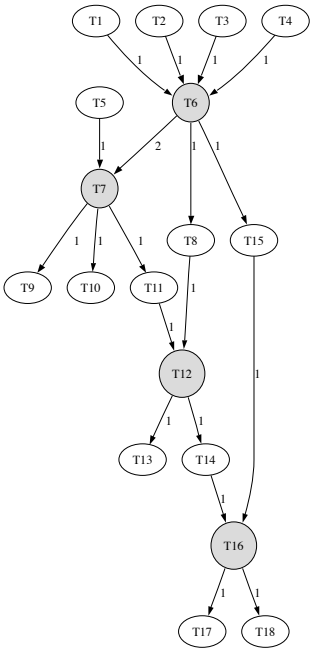


Figure 2. **An example of a simple track graph.** Each node corresponds to a track: white - an individual target, grey - multiple targets. The edges of the graph are directed corresponding to the temporal constraints and indicate when tracks merge or separate.

2.2. General Approach

On top of the track graph it is assumed that there are feature vectors measured from each single target track. These feature vectors can consist of elements such as color, shape, position and velocity. Using the feature vectors to compare tracks and the constraints imposed by the track graph we find the most likely configuration of paths. To do this we parameterize the solution space imposed by the track graph

so that we have a state vector that can represent all possible configuration of paths through the track graph. The paths are then found by inferring the state given the features of the tracks.

2.3. The solution space

Each targets path through the graph is known when it is known exactly how the incoming targets are distributed into the outgoing tracks when a track is split up. Therefore, we will represent the solution space by viewing the splits as track switches. Each split/switch has a state variable representing how the targets are distributed into the outgoing tracks.

When defining the state variables of the split nodes care must be taken so that the state space becomes as compact as possible. Each set of values of the state variables should correspond to one unique solution. This can be done in the following way.

Let N be the number of incoming targets for a particular split node. It doesn't matter how many incoming tracks the node has, so we can assume it has one single incoming track, as in Figure 3. Moreover, let the node have m outgoing tracks, each having $n_j, j = 1, ..., m$ targets also summing up to a total of N targets.

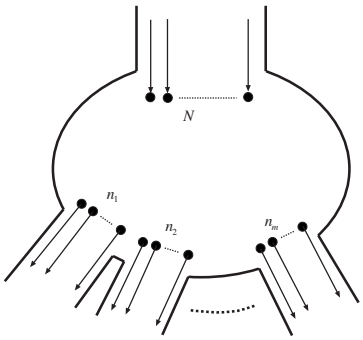


Figure 3. The size of the state variable of a split node is determined by the number of outgoing tracks and how many targets, n_i are in each track i.

The number of ways to distribute the targets into the outgoing tracks can be found through a process of iteratively selecting the targets to go into a track. Each track i selects n_i targets from the targets not yet selected. In this way each track can select its targets in $\binom{N-\sum_{j=1}^{i-1} n_j}{n_i}$ different ways. Hence, the total number of states of the split node is

$$\prod_{i=1}^{m} \binom{N - \sum_{j=1}^{i-1} n_j}{n_i}. \tag{1}$$

Note that the incoming targets is considered an ordered set while the selection in the process above is unordered. To define the ordering in the outgoing tracks we let them keep their relative ordering within each outgoing track, as illustrated with the example in Figure 4. This avoids getting redundant states, which would happen when two targets in the same track are switched in one split node and then switched back when entering a subsequent split node, which is equivalent to the targets not getting switched in either node.

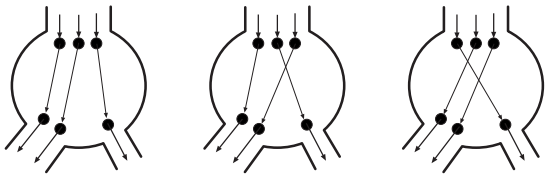


Figure 4. All three states of a node splitting three incoming targets into one double and one single target track.

The variable representing the split state of the node is a product of the "selection states" of the outgoing tracks of the node. Getting the selection states from the node state is a matter of using integer divisions and the modulo operator.

Let each split node, T_i, have a discrete state variable S_i which represents exactly how the targets are split into the outgoing tracks. The number of values S_i can take is given by (1). Moreover, let

$$\mathcal{S} = \{S_i; T_i \text{ is a split node}\} \tag{2}$$

be the set of state variables for all the split nodes. Then $\mathcal{S}$ can represent all possible solutions of paths given the track graph. There is also a one-to-one mapping between the values of the state variables and the solution space.

2.3.1 Computing the number of targets in the tracks

This section described how the number of targets in each link is computed. Let l_{ij} be the link count, i.e. the number of targets in the link between tracks T_i and T_j.

1. Let all link counts be undefined, $l_{ij} = 0$.
2. Set link counts to all single tracks to one,
   $l_{ij} = 1, \quad T_i$ connected to T_j and (T_i or T_j are single track)
3. For nodes with all links but one defined, set the undefined link so that number of targets in equals number of targets out.
4. . Repeat 3 until no more links are updated.

The above procedure will propagate the number of targets through the track graph. In practice there will be inconsistencies and some links will be left undefined. These parts of the graph are left unresolved at this point, but there are several possibilities how they could be handled in the future, e.g. by better modeling or by merging nodes.

2.4. The Inference Problem

Let each single track i have feature vector A_i and let

$$\mathcal{A} = \{A_i; T_i \text{ is a single target track}\} \tag{3}$$

be the set of all feature vectors.

We would like to infer the paths given the measurements using the max posterior estimate,

$$\hat{\mathcal{S}} = \underset{\mathcal{S}}{\operatorname{argmax}}\, P(\mathcal{S}|\mathcal{A}). \tag{4}$$

As usual, Bayes formula can be used to instead maximize the product of the prior and the likelihood function.

$$P(\mathcal{S}|\mathcal{A}) \propto P(\mathcal{A}|\mathcal{S})P(\mathcal{S}) \tag{5}$$

The split node state variables $\mathcal{S}$ are local and causally independent. The measurements on the other hand, depend on the state variables in the sense that the values of the state variables define the targets' paths. Tracks on the same path contain the same target, hence their measurements are dependent. Measurements from different targets are assumed to be independent.

We note that every path ends at a tail node, i.e. a node with no outgoing links. The tail nodes are used as representatives for the paths. Let

$$\mathcal{A}_{tails} = \{A_i; T_i \text{ is a tail node}\} \tag{6}$$

be the set of tail node features. Further, let

$$\text{path}(A_i, s) = \{A_j; T_j \text{ are on the same path as } T_i \text{ given } \mathcal{S} = s\} \tag{7}$$

be the feature vectors of all tracks on the path defined by the state s and leading to the track T_i. Then the likelihood function can be factorized as

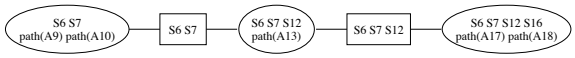
$$P(\mathcal{A}|\mathcal{S}) = \prod_{A_i \in \mathcal{A}_{tails}} P(\text{path}(A_i, s)|\mathcal{S} = s). \tag{8}$$

The dependencies between state variables and feature vectors can be viewed in a Bayesian network showing the causal dependencies between the nodes. The track graph in Figure 2 has the Bayesian network in Figure 5.

Inference on a Bayesian network can be done efficiently using message propagation. We use the junction tree algorithm, [2, 5]. This algorithm creates a secondary structure, the junction tree, consisting of cliques and sepsets. The cliques are the smallest sets of variables on which the inference can be solved using local computations and message propagation. The sepsets show the common variables between neighboring cliques which are the margins that is computed when performing the message propagation.

The most probable configuration of a Bayes net can be found by using max-marginalization in the message propagation. We have used Kevin Murphy's implementation in the Bayes Net Toolbox for Matlab [8]. All we have to provide are the likelihoods for the cliques and the priors.

(a)



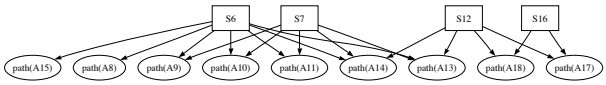
(b)

Figure 5. Bayesian network (a) and junction tree (b) for the track graph in Figure 2

2.4.1 Reducing complexity

Message propagation will solve the inference problem efficiently and it will give a globally optimal solution (under the assumptions). For large problems though, there will be a combinatorial explosion.

To apply this approach to large scale problems it is necessary to reduce the complexity. We do this by dropping the dependencies between feature vectors and split nodes that are more than a certain number of links away. The effect is that we optimize shorter but overlapping paths in the graph. A Bayes net for our track graph can look like the Bayes net in Figure 6. As can be seen, paths to all single target tracks are taken into account, but the levels of dependencies have been reduced. In this case the paths to T_{17} and T_{18} have dropped the dependencies to the split nodes T_6 and T_7. In effect this means that the tracks T_{17} and T_{18} will not be compared with tracks above T_6 and T_7. It also mean that the complexity have been reduced and for larger problems this is crucial.



(a)

(b)

Figure 6. To reduce the complexity we remove dependencies between tracks that are distant in time. The result is that the algorithm optimizes shorter local paths that are overlapping. The Bayesian network (a) now has paths to all single track nodes with ancestors and A_{17} and A_{18} does not depend on S_6 and S_7 anymore. (b) shows the resulting junction tree.

2.4.2 Building the Bayes Net

The Bayes net is built through the following procedure.

1. For each split node T_i in the track graph, add the state variable S_i.
2. For each single track node T_i with ancestors:

(a) Add an observed node representing all paths leading to A_i.

(b) In a breadth-first fashion collect split nodes that are ancestors to T_i until the product of the split nodes' state sizes (the clique size) have reached a set limit.

(c) Connect the collected split nodes' state variables to the new observed node.

2.4.3 Computing the Conditional Probability Tables

For the inference algorithm we need to provide the probability distributions for each clique in the junction tree. The probability distribution is the product of the prior and the likelihood. In this paper we use a flat prior.

The conditional probability tables are computed as described below.

1. For each observed node in the BN representing paths leading to A_i:

   (a) Get the parents $\mathcal{S}_{pa}$

   (b) For each combined state s_{pa} of the variables in $\mathcal{S}_{pa}$ (the number of combined states is the product of the size of each state variable $S \in \mathcal{S}_{pa}$):

      i. Compute the likelihood $P(\text{path}(A_i, s_{pa})|\mathcal{S}_{pa} = s_{pa})$

Basically, at this step we go through all paths in the local graph around the split nodes associated with parents, $\mathcal{S}_{pa}$, of the observed node. In our example with the Bayes net in Figure 6, when computing the likelihoods of for paths leading to T_{17} we go through all possible paths from T_{11}, T_8 and T_{15} to T_{17}.

2.4.4 Computing the Likelihoods

The likelihoods in this case are the probability density for the measurements given that they all are from the same model. This model is considered unknown, but if there is a set of models, $\mathcal{M}$, that will make the measurements independent we can compute the likelihood in the following way.

$$\begin{aligned} &P(\text{path}(A_i, s_{pa})|\mathcal{S}_{pa} = s_{pa}) \\ &\quad = \int_{M \in \mathcal{M}} P(\text{path}(A_i, s_{pa})|\mathcal{S}_{pa} = s_{pa}, M)P(M) \\ &\quad = \int_{M \in \mathcal{M}} \prod_{A_j \in \text{path}(A_i, s_{pa})} P(A_j|M)P(M) \end{aligned} \tag{9}$$

Sometimes it is easier to find a pairwise measure how likely two tracks contain the same target. These can be used in such a way that all pairwise similarity measure are used exactly once.

$$\begin{aligned} &P(\text{path}(A_i, s_{pa})|\mathcal{S}_{pa} = s_{pa}) \\ &\quad \approx \prod_{A_j \in \text{path}(A_i, s_{pa}) \setminus A_i} P(A_i, A_j) \end{aligned} \tag{10}$$

Since we have overlapping paths, the similarity measure between the other members in the path will be used in the cliques for paths originating from the those members.

3. Football Tracking

At this stage we focus on applying our method to the problem of tracking football players in a competitive professional game. Football occurs in a structured closed environment where it is relatively easy to perform reliable effective image processing, but on the other hand provides many complicated and challenging motions and interactions between players. It is a happy compromise between analyzing generic video sequences and those engineered in the lab.

Here we review an approach to constructing the track graph and to defining a measure of similarity between single player tracks. This takes us to the assumed starting point of our Bayesian inference problem.

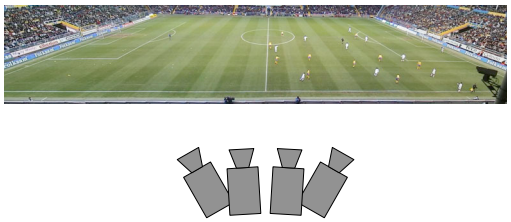
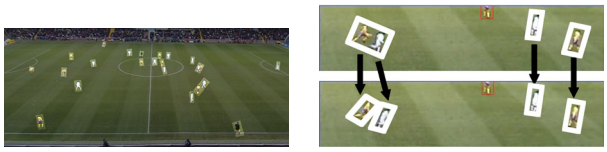


Figure 7. Multi-camera system used to capture a stationary, high-resolution video covering a large area.

3.1. Extracting the Track Graph

Figure 7 displays the multi-camera system used to provide a high resolution, wide-field of view video of the football game. The resulting video allows all the players to be seen at all times. As the cameras are stationary it is possible to perform reliable and accurate background subtraction to highlight the positions of the targets in each image (see figure 8 (a)). Temporal analysis of the foreground regions found at each frame, matching the regions in one frame to those in the next (figure 8 (b)), allows the identification of the single and multiple player tracks and the interactions between them. There are two teams wearing two distinctly colored uniforms, as well as the officials. It is possible to assign each single track to one of these three categories based on simple matching of exemplar rgb histograms. Figure 12 displays a portion of the track graph obtained from examining our football clip in this manner. For the interested



(a) foreground regions (b) matched regions

Figure 8. **(a)** The foreground regions found by background subtraction. **(b)** An example of matching the found regions between one frame and the next. Note that one of the examples is a split, marking the end of one track and the beginning of two more.

reader, the complete graph is included as part of the supplementary material. We manually obtained the ground truth for the identity of the team A single target tracks. The temporal extent of each player's single tracks are displayed in figure 9. We would like to obtain this figure automatically.

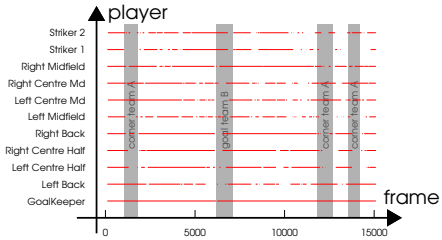


Figure 9. The temporal extent of the team A single player tracks for the ten minute clip examined. Each line corresponds to a single player track. The shaded areas display when the major congestion events occur.

3.2. Similarity Measure Between Player Tracks

We now define a measure of similarity between every pair of single player tracks from the same team. In football a player's identity is frequently revealed by his position relative to his teammates. Most obviously the goal-keeper is always behind all his teammates. We exploit this simple idea. For each single player track we build a histogram summarizing the player's position relative to his teammates for the duration of the track. Each bin of the histogram corresponds to a particular configuration of teammates to the left, right, behind and in front of the player. There a fixed number of such configurations as there are eleven players on a team, see [1] for details. Let I_{s}^{ij} denote the similarity score between two tracks based on comparing their relative spatial position histograms. This measure is particularly effective for matching tracks of long duration. Such tracks, generally, occur when the team is in typical formations and the players are in set positions within these formations. However, many of the shorter tracks occur when the team is in transition between typical team formations rendering the relative spatial position information less effective. To compensate for this deficiency we define temporally local measures.

Two tracks T_i and T_j are temporally close if the end of T_i occurs before and within t frames of the start of T_j. If t is small enough and T_i and T_j represent the same player, it is reasonable to assume continuity of appearance and motion. On this basis we construct appearance and motion based measures between temporally close track pairs. The appearance measure relies on cross-correlating the appropriate spatio-temporal volumes at the ends involved. This measure is denoted by I_{i}^{ij}. The velocity of the targets at the ends of these tracks is also estimated. Given these velocities and the final position of T_i, an estimate of the start position of T_j is obtained. The difference between this estimate and actual value is then used as our motion measure - I_{d}^{ij}. After appropriate rescaling of the different I's a combined similarity matrix is produced:

$$I^{ij} = \begin{cases} (1-2\alpha)I_{\mathrm{s}}^{ij} + \alpha I_{\mathrm{i}}^{ij} + \alpha I_{\mathrm{d}}^{ij} & \text{if } T_i, T_j \text{ temporally close.} \\ I_{\mathrm{s}}^{ij} & \text{otherwise} \end{cases} \tag{11}$$

with $0 \leq \alpha \leq 1$, see figure 10. The similarity scores are then converted into the appropriate form, (eqn 10), by setting $P(A_i, A_j) = \exp(-\lambda I^{ij})$, with $\lambda > 0$.

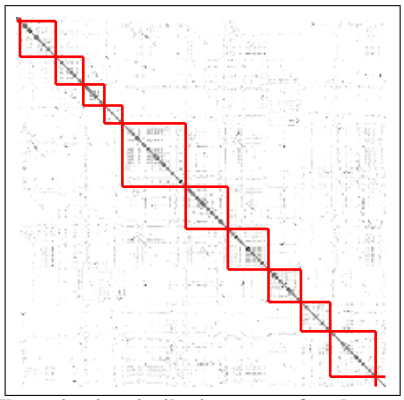


Figure 10. **The pairwise similarity scores for the team A tracks**. Black indicates high similarity and white low similarity. The rows of the matrix have been re-ordered to group tracks of the same identity together and to reveal the structure within the matrix. The red lines denote the sub-blocks of constant identity.

3.3. Results

We are now almost ready to present the identity linking results. Before running the inference procedure, the number of targets in each link is computed. In many parts of our football clip graph, it is not possible to determine the number of targets in each link and sometimes there are inconsistencies (the number of input and output targets at an interaction are unequal). Presently our theory cannot handle such situations, thus these parts of the graph are left unsolved. Accordingly, the Bayes net is divided into parts, which are analyzed separately. For our football graph 14 separate parts are isolated. In figure 12(b) we can see an explicit demonstration of solutions found for some split nodes.

For a clearer picture of the quality of the results obtained the found paths (for team A players) within largest parts are shown in figure 11. As can be seen, the majority of the tracks are correctly linked.

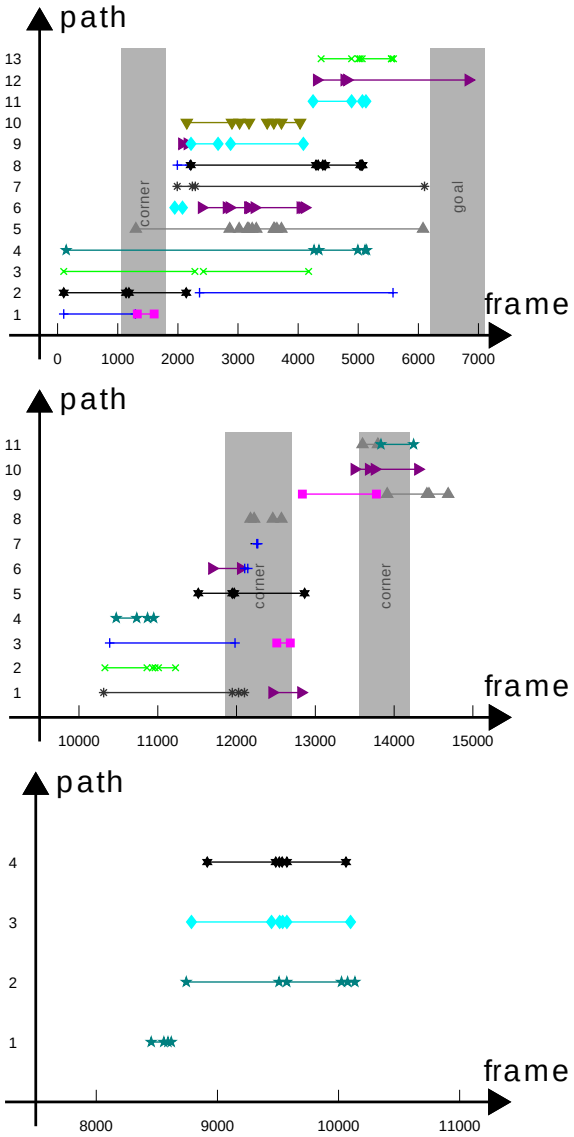


Figure 11. **Estimated paths for the three largest consistent parts of the graph.** Each line represents a single player track and row an estimated path. The color and symbol denote the true identity of the track. Ideally there should only be one color and symbol per row. On several occasions a target's trajectory is split into several paths. This is caused by links in a part having an undetermined number of targets. The rest of the parts contain a similar number of tracks and quality of results to the bottom graph.

To summarize the overall results - 85% (out of 73) of the connections considered are correctly resolved. However, not all decisions made at each split node have the same degree of confidence associated with them. Fortunately, as we are working with probabilities and within a Bayesian framework we can compute the absolute probability for each possible resolution of a particular split node. Comparing the relative value of the most probable to the next most probable resolution provides a confidence level of our estimate. By only including estimates that are certain, we can eliminate some of the connection errors. Figure 13 shows the percentage of correct connections as we remove the less certain split estimates. When 25% of the connections remain we have 100% correct connections.

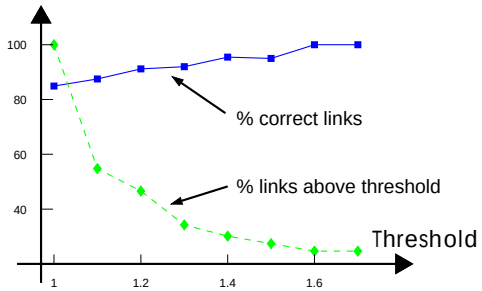
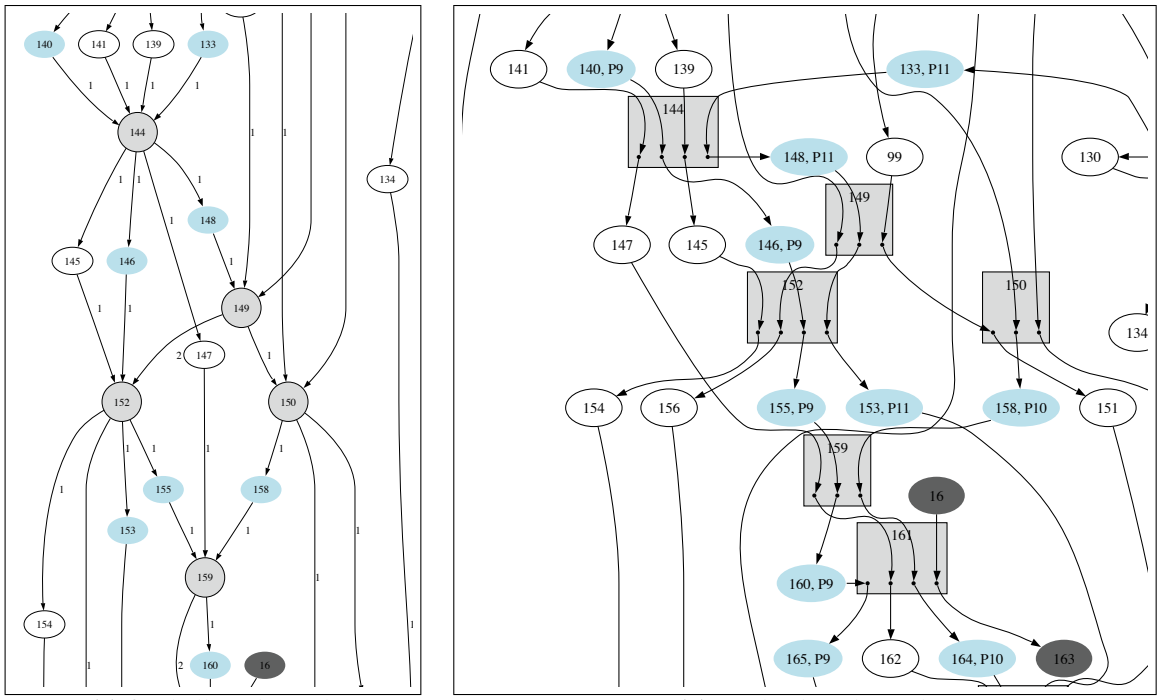


Figure 13. **The percentage of correct connections.** Using the marginal probabilities we can remove uncertain estimates. We threshold on the ratio between the most and second most probable state of a split node. This increases the percentage of correct links, but reduces the number of connections made. At a threshold of 1.6 we make no wrong connections, but only 25% of the connections are left (out of 73).

4. Conclusions and Future Research

When tracking multiple targets over a long period, it is inevitable that inter-target occlusions will occur where it is not possible to immediately link the identities of the targets entering and those exiting the interaction. It is therefore necessary to compare targets over extended periods of time in the attempt to link their identities. In this paper we achieve this by considering a two-stage solution. The first stage involves the construction of a track graph describing the interactions between targets. The second stage, the focus of this paper, exploits the track graph and similarity measurements between the tracks to infer the most likely configuration of paths for all targets. This is achieved by parameterizing the solution space imposed by the track graph and inferring the parameters given the measurements. To make large scale problems computationally feasible we only consider tracks within a certain window of interactions to be explicitly dependent.

Promising results on a challenging (and relatively lengthy) data set are presented. The main limitation to improving the results, presently, are the assumptions concerning the track graph. For problems with many targets interacting frequently it is not realistic to expect that the number of targets in each node can be counted explicitly. Relaxing



(a) football track graph **(b)** resolved track graph

Figure 12. **(a)** This is a small part of the football clip track graph. The node colors correspond to team A (light blue oval), team B (white), referees (dark grey) and multi-target nodes (black). **(b)** The corresponding resolved track graph. The square nodes display how the split nodes have been resolved. Ground truth player numbers can be seen for the team A players.

this assumption, with more sophisticated modeling, would increase the size of the parts of real-world track graphs we could examine. Of course, as in most tracking algorithms, there is room for improvement in the modeling of the appearance of the targets.

In a complementary approach, the identities of single player tracks can be linked by un-supervised clustering using the similarity matrices shown in figure 10. Clustering can be performed without reference to the *track graph*, thus by-passing the computational bottlenecks and inconsistencies in the graph. Reliable results have been obtained when long tracks are included in the clustering process [1]. A fruitful avenue of future research would be to investigate how to optimally combine clustering and the path finding algorithm presented here to obtain a more complete labeling of the player identities.

References

[1] Author. Tracking and Labelling of Interacting Multiple Targets. ECCV06 submission ID 1027. Supplied as additional material eccv06.pdf. 2, 5, 6, 8

[2] R. Cowell, A. Dawid, S. Lauritzen, and D. Spiegelhalter. *Probabilistic Networks and Expert Syst.* Springer, 1999. 4

[3] P. Figueroa, N. Leite, R. Barros, I. Cohen, and G. Medioni. Tracking soccer players using the graph representation. In *ICPR*, pages 787–790, 2004. 1

[4] M. Gelgon, P. Bouthemy, and J. Le Cadre. Recovery of the trajectories of multiple moving objects in an image sequence with a pmht approach. *J. Image & Vision Computing*, 23(1):19–31, 2005. 1

[5] C. Huang and A. Darwiche. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, 15(3):225–263, 1996. 4

[6] S. Iwase and H. Saito. Parallel tracking of all soccer players by integrating detected positions in multiple view images. In *ICPR*, pages 751–754, 2004. 1

[7] Z. Khan, T. Balch, and F. Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *European Conference on Computer Vision*, 2004. 1

[8] K. Murphy. The bayes net toolbox for matlab. In *Computing Science and Statistics*, volume 33, 2001. 4

[9] C. Needham and R. Boyle. Tracking multiple sports players through occlusion, congestion and scale. In *BMVC*, 2001. 1

[10] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, 2004. 1

[11] J. Vermaak, A. Doucet, and P. Perez. Maintaining multimodality through mixture tracking. In *International Conference on Computer Vision*, 2003. 1

[12] M. Xu, J. Orwell, and G. Jones. Tracking football players with multiple cameras. In *IEEE International Conference on Image Processing*, 2004. 1

Talks

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Results from a Real-time Stereo-based Pedestrian Detection System on a Moving Vehicle

Max Bajracharya, Baback Moghaddam, Andrew Howard, Shane Brennan, Larry H. Matthies

***Abstract*— This paper describes performance results from a real-time system for detecting, localizing, and tracking pedestrians from a moving vehicle. The end-to-end system runs at 5Hz on 1024x768 imagery using standard hardware, and has been integrated and tested on multiple ground vehicles and environments. We show performance on a diverse set of ground-truthed datasets in outdoor environments with varying degrees of pedestrian density and clutter. The system can reliably detect upright pedestrians to a range of 40m in lightly cluttered urban environments. In highly cluttered urban environments, the detection rates are on par with state-of-the-art non-real-time systems [1].**

I. INTRODUCTION

The ability for autonomous vehicles to detect and predict the motion of pedestrians or personnel in their vicinity is critical to ensure that the vehicles operate safely around people. Vehicles must be able to detect people in urban and cross-country environments, including flat, uneven and multi-level terrain, with widely varying degrees of clutter, occlusion, and illumination (and ultimately for operating day or night, in all weather, and in the presence of atmospheric obscurants). To support high-speed driving, detection must be reliable to a range of 100m. The ability to detect pedestrians from a moving vehicle in a cluttered, dynamic urban environments is also applicable to automatic driver-assistance systems or smaller autonomous robots navigating in environments such as a sidewalk or marketplace.

This paper describes results from a fully integrated real-time system capable of reliably detecting, localizing, and tracking upright (stationary, walking, or running) human adults at a range out to 40m from a moving platform. Our approach uses imagery and dense range data from stereo cameras for the detection, tracking, and velocity estimation of pedestrians. The end-to-end system runs at 5Hz on 1024x768 imagery on a standard 2.4GHz Intel Core 2 Quad processor. The ability to process this high resolution imagery enables the system to achieve better performance at long range compared to other state-of-the-art implementations. Because the system segments and classifies people based on stereo range data, it is largely invariant to the variability of pedestrians' appearance (due to different types and styles of clothing) and scale. The system also handles different viewpoints (frontal vs. side views) and poses (including articulations and walking) of pedestrians, and is robust to objects being carried or worn by them. Furthermore, the system makes no assumption of a ground-plane to detect or track people, and similarly makes no assumption about the predictability of a person's motion other than a maximum velocity.

The research described in this publication was carried out at the Jet Propulsion Laboratory, California Institute of Technology, with funding from the Army Research Lab (ARL) under the Robotics Collaborative Technology Alliance (RCTA) through an agreement with NASA

All authors are with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

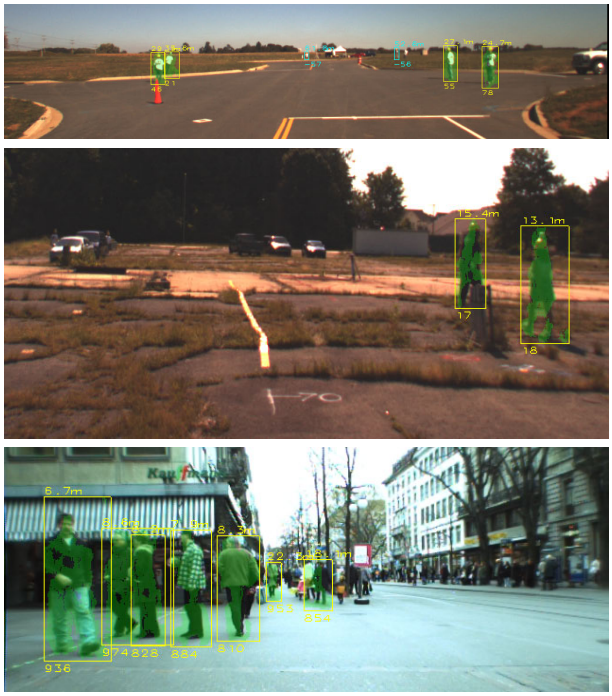


Fig. 1. Examples of test scenarios and the output of our pedestrian detection system (yellow boxes are detections with range and track ID text and a green overlay of the segmented person; the cyan boxes are missed detections).

The performance of the system is demonstrated on a variety of ground-truthed datasets in various outdoor environments, with different degrees of person density and clutter. An example of these scenes is shown in Figure 1. The majority of datasets used to evaluate the system consist of scenarios simulating the operation of an unmanned ground vehicle (UGV) traveling at moderate speed in semi-urban terrain (paved roads with light clutter and people walking along or into the road). In these scenarios, the system is capable of initial detections of pedestrians up to 60m, and reliable detection and tracking of pedestrians up to 40m, which correspond respectively to 30 pixel and 45 pixel tall pedestrians for our cameras. We also present performance results of our system on recently published datasets of crowded street scenes. Although not specifically designed for

highly cluttered urban environments, we show that results of our real-time system are comparable to the state-of-the-art systems that are designed to operate in these environments.

II. RELATED WORK

There has been extensive research on pedestrian detection from manned and unmanned ground vehicles using scanning laser rangefinders (LIDAR) and monocular and stereo vision in visible, near infrared, and thermal infrared wavelengths. Most such work assumes the scene contains a dominant ground plane that supports all of the pedestrians in upright postures. Maximum detection ranges tend to be 30m or less. Rates of missed detections and false alarms are not good enough to be satisfactory in deployed systems. Most prior work on pedestrian detection has been done for applications to smart automobiles, robotic vehicles, or surveillance. This literature is very large, so we only cover recent highlights and main trends here.

Research on pedestrian detection for smart automobiles has employed monocular vision [2], [3], [4] stereo vision [5], [6], [7], [8], [9] and LIDAR [10]. Vision-based methods have used visible [2], [3], near infrared [4], and thermal imagery [8]. Most work in this area has been strongly motivated by the requirement to be very low cost in eventual production. The approaches generally follow the architecture of detecting regions of interst (ROIs), classifying these regions, and tracking them.

Work on pedestrian detection for robotic vehicles in outdoor applications [11], [12], [13], [14], [15] includes methods that do range sensing with 2D LIDAR, 3D LIDAR, stereo vision, and/or structure from motion and do image sensing with visible and/or thermal infrared cameras. At a high level, algorithm architectures are analogous to the systems for the automotive domain, involving ROI detection, classification, and tracking, though the order and details of these steps differ. As a group, there is more emphasis in this domain on classification based on the 3D shape of the objects as perceived by LIDAR or stereo vision than there is in the automotive domain. The feature extraction and classification algorithms tend to be simpler than those used in either the automotive or video surveillance domains. Several of these approaches have been tested as part of third party field experiments, with results discussed by Bodt [16].

Finally, work on pedestrian detection in the surveillance arena largely divides into work with image sequences from stationary cameras, where background subtraction and/or image differencing is used to detect moving objects [17], [18], and work that applies trained pattern classifiers to individual images [19], [20], [21], [22], [23]. The former group is less relevant here, because background subtraction and temporal image differencing are more difficult to use from moving cameras. The latter group uses a variety of feature extraction and classification methods to achieve better detection and false alarm rates than single-frame results reported in the automotive pedestrian detection literature; however, the results are not directly comparable because computational requirements are generally higher, the testing protocol often uses image databases where positive examples are already centered in image chips or does exhaustive search over position and scale of ROIs in test imagery, and because only individual frames are considered, the systems do not include any tracking.

III. SYSTEM DESCRIPTION

Our system is fully described in earlier work [14], but we briefly summarize our approach here. We focus on two differences from our prior system: a slightly reduced feature set, and an improved tracker. Our system consists of the following steps:

- **Stereo vision** takes synchronized images from a pair of cameras and computes a dense range image.
- **Region-of-interest (ROI) detection** projects stereo data into a polar-perspective map and then segments the map to produce clusters of pixels corresponding to upright objects.
- **Classification** computes geometric features of the 3D point cloud of each ROI and classifies the object, resulting in a probability of being human.
- **Tracking** associates ROIs in sequential frames, accounting for vehicle motion, and estimates the velocity of the detected objects.

The system architecture allows the possibility of using appearance and motion features to improve the classification of people, but we currently do not make use of these features.

A. Stereo Vision

The first step in our system is to compute dense range data from stereo images. We use a multi-processor version of the real-time algorithm described by Goldberg [24] previously used on the NASA Mars Exploration Rovers and in the DARPA PerceptOR program. On a 2.4GHz Intel Core 2 Quad processor, the algorithm can process 1024x768 imagery at 10 frames/sec.

B. Region-of-Interest Detection

Detecting region-of-interest (ROI) areas from the stereo data serves as a focus-of-attention mechanism to reduce the runtime of subsequent classifiers and segments foreground pixels from background pixels in a region. This allows a shape-based classifier to be run on the 3D points that make up a specific object, rather than sliding a window over the image and explicitly performing foreground/background segmentation in each window.

The stereo range data is transformed into a gravity-leveled frame, accounting for the roll and pitch of the vehicle, and then projected into a two-dimensional polar-perspective grid map (PPM). The map is then segmented based on map cell statistics. Unlike a traditional Cartesian map, which is divided into cells of fixed size in Cartesian (x,y) space, the PPM is divided into cells with a fixed angular resolution but variable range resolution in polar (r, θ) space in order to preserve the coherency of the stereo range data. The PPM accumulates the number of range points projected into each cell. The map is then smoothed with an averaging filter with

an adaptive bandwidth in polar space corresponding to a fixed bandwidth in Cartesian space. For computational efficiency the filter is implemented using an integral image of the map. After smoothing, the map gradient is used to find all of the peaks in the map, which are then grown to the inflection points in the gradients, resulting in a segmentation of the map. Because the minimum expected size of the objects being detecting is known, segmented blobs whose peaks fall within half of this size are then merged together. Figure 2 provides an example of a filtered PPM with ROI detections.

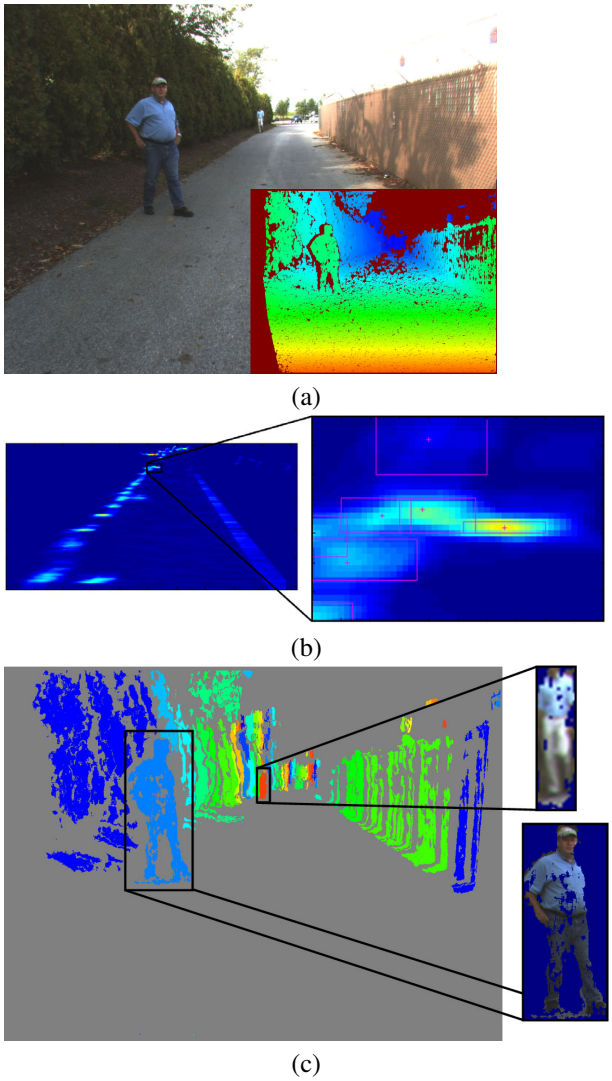


Fig. 2. An example of the stereo-based segmentation for region-of-interest detection. (a) shows the left image of a stereo pair with the resulting depth map (inset); (b) shows the polar-perspective map of point counts smoothed with an averaging filter with a close up of the map with segmented regions overlaid; and (c) shows the segmented regions in different colors, with examples of the foreground/background separation.

C. Classification

Geometric features of each segmented 3D point cloud are used to classify them as human or not human based on shape. After segmentation, a scene may contain hundreds of regions. To reduce the number of regions that must be classified, we first prefilter the regions with a fixed threshold on the width, height, and depth variance of each segmented region. This threshold is simply selected as the 3σ values obtained from the training data. After prefiltering, the features used for classification are computed for each region's point cloud.

We then compute geometry-based features for the remaining regions, including the fixed-frame shape moments (variances of point clouds in a fixed frame), rotationally invariant shape moments (the eigenvalues of the point cloud's scatter matrix), and "soft-counts" of various width, height, depth, and volume constraints. The logarithmic and empirical logit transforms of these moments and counts are used to improve the normality of the feature distribution.

To compute the features, we start by centering the point cloud about the x-axis by its mean value and setting the minimum depth z and height y to zero. The first feature is defined by the logarithm of the 2^{nd} order moment of the height:

$$f_1 = -\log(\sigma_y^2) \tag{1}$$

The "soft-count" features are defined by the number of points that fall inside certain preset coordinate bounds (or volumes). Such count-based features ignore "true shape" and focus instead on the object's size or extent. Unlike moment-based features, count-based features are more tolerant of outlier noise and some artifacts of stereo processing. Naturally there are strong correlations between these two different sets of features. However, this correlation or redundancy can be quite helpful for modeling purposes. For the total number of points n in a blob point cloud, we define $n_x = \#(|x| < 1)$ as the number (subset) of 3D points whose x value is less than 1m (in absolute value), $n_{y_0} = \#(y < 2)$ and $n_{y_1} = \#(y > 1)$ as the number of points whose height value is less than 2m and greater than 1m, and $n_{z_0} = \#(z < 4)$ and $n_{z_1} = \#(z < 3.5)$ as the number of points with a depth value less than 4m and 3.5m respectively. We also define n_v to be the number of 3D points that satisfy all three width, height, and depth constraints simultaneously (i.e., the number of points that fall within the prescribed rectangular volume of size 1m x 2m x 4m). Although these constraints were selected empirically, the process could easily be automated. In order to normalize the data as well as account for uncertainty due to the sample size (n), we use a logit transform with an empirical prior count c:

$$f_2 = \log \frac{n_x + c_x}{n - n_x + c_x} \qquad f_3 = \log \frac{n_{y_0} + c_{y_0}}{n - n_{y_0} + c_{y_0}} \tag{2}$$

$$f_4 = \log \frac{n_{z_0} + c_{z_0}}{n - n_{z_0} + c_{z_0}} \qquad f_5 = \log \frac{n_v + c_v}{n - n_v + c_v} \tag{3}$$

$$f_6 = \log \frac{n_{y_1} + c_{y_1}}{n - n_{y_1} + c_{y_1}} \qquad f_7 = \log \frac{n_{z_1} + c_{z_1}}{n - n_{z_1} + c_{z_1}} \tag{4}$$

The rotationally-invariant features are the logarithms of the eigenvalues of the point cloud's covariance (inertia) matrix, where $(\lambda_x, \lambda_y, \lambda_z)$ correspond to the major, intermediate, and minor axes, respectively:

$$f_8 = -\log(\lambda_x) \quad f_9 = -\log(\lambda_y) \quad f_{10} = -\log(\lambda_z) \tag{5}$$

We note that f_8 would be redundant with f_1 if all the blobs were oriented correctly (upright and "facing" downrange). However, this is often not the case, due to artifacts in stereo processing, and especially at long ranges where blob point-clouds are often tilted and/or slanted.

Analysis of the shape features indicated that a linear classifier (with a linear decision boundary) was too simple to always work effectively. However, a more complex decision boundary can be achieved while still using a linear classifier (which is desirable for its computational efficiency and robustness) by expanding the feature set to use higher-order terms. Specifically, a quadratic decision boundary is modeled using the augmented feature set:

$$\mathbf{x} = [\ 1 \ \ \{f_i\} \ \ \{f_i f_j\}_{i<j} \ \ \{f_i^2\}\ ]^T \tag{6}$$

Using this feature vector, a Bayesian generalized linear model (GLM) classifier (for logistic regression) is then trained using standard iteratively reweighted least squares (IRLS) to obtain a Gaussian approximation to the posterior mode. Simple MAP estimates of predictive probability (of being human) are obtained using this Gaussian mode-based approximation.

D. Tracking

Tracking ROIs in the scene is used to both reduce incorrect detections and estimate the velocity of the detected objects. By associating ROIs across multiple frames, the single frame classifications can be aggregated to eliminate false positives. Similarly, using the positions of a tracked object from stereo and the motion of the vehicle, estimated by visual odometry [25] or provided by an inertial navigation system (INS), the velocity of the object can be computed and extrapolated to provide a predicted motion to a path planner. The tracking algorithm is designed to be extremely computationally efficient and makes very few assumptions about the motions of objects.

Tracking is implemented as the association of ROIs in sequential frames. The ROIs extracted in a new frame are matched to existing nearby tracks by computing a cost based on each ROI's segmented foreground appearance and then solving a one-to-one assignment problem. For computational efficiency and simplicity, the cost between an ROI and a track is computed by comparing the new ROI to the last ROI in the track. Only ROIs within a fixed distance are considered; the distance is computed by using an assumed maximum velocity of 2m/s in any direction for each object. The cost between ROIs is then computed as the Bhattacharyya distance of a color (RGB) histogram between each ROI. For computational efficiency, we solve the assignment problem with co-occurring minima. If an ROI does not match an existing track, a new track is started. Tracks that are not matched for a fixed number of frames are removed. To eliminate the incorrect detections that lead to false positives while still maintaining detections on true positives where the classification score dropped for a small number of frames, we temporally filter the scores with the median of three consecutive scores and require three consecutive frames of detection before making a classification decision. The velocity of tracks is estimated by fitting a linear motion model to the track. We estimate the position and velocity uncertainty by combining the expected stereo error with the model fit.

IV. EXPERIMENTAL RESULTS

The end-to-end system has been tested on datasets with hand-labeled ground-truth and integrated onboard a vehicle for live testing. The primary datasets were collected from the vehicle on which the system was integrated in semi-urban, lightly cluttered scenarios. The results on these datasets show that our system can achieve initial detections at a range of 60m, with detections reliable enough for autonomous navigation out to 40m. To demonstrate that the system's performance is competitive with state-of-the-art systems in highly cluttered, urban scenarios, we also make use of datasets published by Ess [1], [26]. We show that we can achieve performance similar to Ess on these datasets while running at real-time rates.

A. Semi-Urban Datasets

The primary datasets used to evaluate the system use input imagery from a 3 CCD color stereo camera pair with 1024x768 pixels, a 50 cm baseline, a field of view approximately 60 degrees wide, and with frame rates between 3.5Hz and 10Hz. The cameras were either mounted on the roof of an SUV at a height of approximately 2m above the ground, and pointed down by approximately 5 degrees, or on the pan-tilt head of an unmanned vehicle at a height of approximate 2m above the ground, and pointed down by 20 degrees. The scenarios include the vehicle driving down a road at speeds varying from 15 to 30 kph, with stationary mannequins and people standing, walking, and running along the side of and across the road in varying directions. The scene also contains stationary and moving cars, trucks, and trailers, along with stationary crates, cones, barrels, sticks, and other similar objects. In many cases, the pedestrians experience a period of partial to full occlusion by these objects or each other. Several variations of the scenario also include one or two people walking in front of the vehicle, weaving between each other and occasionally going out of the field of view.

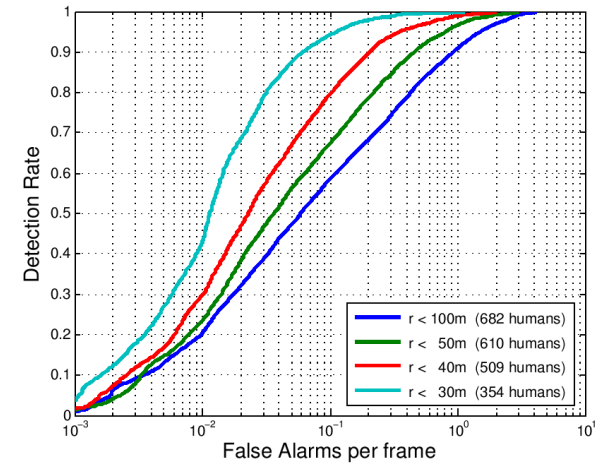
The imagery was manually ground-truthed by annotating a bounding box around each person in the left image of each frame, to a range of approximately 100m. In total, our corpus includes approximately 6,000 annotated frames with approximately 10,000 annotated people, although we restrict our analysis to specific datasets which are representative of operational scenarios. Although people are annotated regardless of their posture or degree of occlusion, we only consider people who are in an upright posture with less than 50% occlusion for our analysis. We use the measure of the area of the intersection over the area of the union of the annotated and detected bounding boxes to declare a correct detection. However, for these datasets, we found that relaxing the common evaluation criteria of 50% intersection-over-union to 25% produced more meaningful results. This

is because we are interested in detection at relatively long range where the segmentation error is dominated by the foreground fattening effect of stereo matching. Because the scenes are relatively uncluttered, using a looser matching criteria still remains representative of actual detections. In order to present results that are meaningful when developing a complete, autonomous system capable of safe navigation, we present our results as the probability of detection (Pd), defined as the number of detections divided by the true number of people in the scene, versus the false alarms per frame (FAPF), defined as the number of incorrect detections divided by the number of frames in the dataset. To illustrate the performance as a function of range, we restrict the detections and annotations to a maximum range.

To demonstrate the effectiveness of our feature set and classifier, we first present results on a cross-validation test over many of our datasets. Figure 3 (a) shows the performance of the system as an average of 1000 trials on a dataset combined from many different scenarios, totaling 4,396 frames with 3,409 annotated people. From these sequences, 21,824 ROIs were extracted and each curve was generated by randomly selecting 80% of these ROIs for training and using the remaining 20% for testing. The resulting number of effective frames in each test sequence is thus 879, and the average number of humans is shown in the plot for the respective range restriction. For this test, no temporal filtering was used to adjust the classification scores. Figure 3 (b) shows a sample of the images of the sequences used. The detections shown are indicative of the performance of the system (but are, in fact, based on a system trained without that sequence). Across our datasets, the system can achieve a 95% Pd at 0.1 FAPF for people less than 30m and 85% Pd at 0.1 FAPF for people less than 40m. For people out to 50m and 100m, the system achieves 95% and 90% Pd respectively at 1 FAPF.

Because the cross-validation results sample across all of the datasets being tested on, they do not necessarily provide compelling evidence that the system is effective in new, unseen scenarios. To demonstrate that our system is robust in new environments, we show the performance on individual sequences that have never been used for training. Although less statistically significant, they are perhaps more indicative of the performance to be expected of the fielded system. Figure 4 (a) and (b) show the results of the system without temporal filtering on two sequences held out from the training data. The same system was run on both datasets with no modification. As the plots show, the sequence shown in Figure 4 (a) is more difficult than (b), containing more clutter and occlusion. The system achieves well above 95% Pd at 0.1 FAPF for pedestrians less than 30m and 80% Pd for less than 40m. For a fielded system, we generally run at an operating point closer to 0.02 FAPF, which results in 90% Pd for $<$30m and 65% Pd for $<$40m, and maintain some degree of persistence of detected objects, propagating them with their predicted velocity for path planning.

The main source of false alarms of our system in these environments is due to the over segmentation of vehicles.

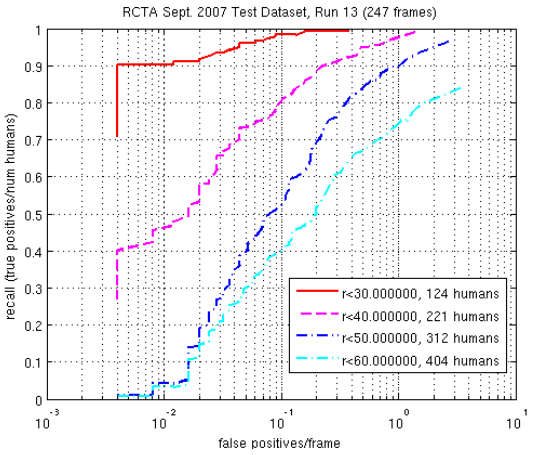


(a)

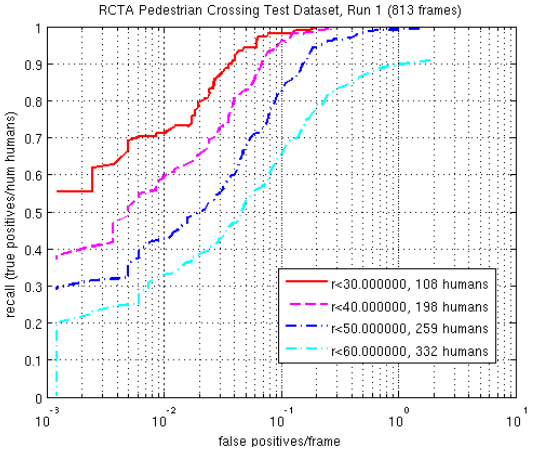
(b)

Fig. 3. (a) The performance resulting from 1000 trials of 80%/20% split cross-validation tests on 4,396 frames drawn from various scenarios. (b) Examples of images and detections from the various scenarios, with an example false alarm on the truck in the bottom image. The yellow boxes are detections, with a green overlay of the segmented person.

An example of a false alarm on the front of a pickup truck is shown in the lower image of Figure 3 (b). The individual distracter objects, such as barrels, tripods, and sign posts are only occasionally misclassified because they are normally segmented correctly. The main source of missed detections is due to variability of the stereo range data at long range, partial occlusion, and occasionally due to imprecise localization of the person due to under or over segmentation. Our system has some robustness to partial occlusion, but tends to break down after greater than 50% occlusion. The sequence shown in Figure 5 shows several examples of performance on occluding and overlapping people. The people in the near field are detected when they are unoccluded, or only slightly occluded. They are not detected when partially occluded either vertically (due to crossing the other person)



(a)



(b)

Fig. 4. The performance for two testing runs including people walking along and in the street, with moving cars and stationary distractor objects.

Fig. 5. A sequence of frames showing detections (yellow boxes, with green overlay the segmented person) and misses (cyan boxes) for people under occlusion. The number above the boxes indicates the range, and the number below indicates the track ID.

or horizontally (due to the posts). Notice, however, that the people are all tracked throughout the sequence (although with one incorrect association). The people in the far field are similarly not detected when they are partially occluded by the vehicles (or too far away), but are detected when they emerge into the open. The failure to detect partially occluded people is understandable because we only train a single classifier with data that does not contain many occluded people.

In addition to testing on ground-truthed datasets, the end-to-end system has been integrated into several systems for live testing. An earlier version of the system was fielded as part of the RCTA program "Safe Operations" test, as reported in [16]. The system described here has been integrated onboard the test vehicle for an upcoming test, for which results will be published in the future. The system has also been used to demonstrate autonomous navigation in a lightly cluttered dynamic environment on a small vehicle (with cameras at approximately 1m high and with a 12cm baseline) traveling at approximately 1m/s.

B. Urban Datasets

To illustrate that our system is competitive with other state-of-the-art stereo-based pedestrian detection systems, we also evaluated our system on datasets published by Ess [1], [26]. These datasets consist of 640x480 resolution color Bayer tiled imagery, taken at 15Hz, with a 40cm baseline camera pair pointed straight out at a height of approximately 1m. The scenarios are significantly more complex than the semi-urban data, with many people in a busy shopping district in Zürich, Switzerland, with significant occlusion, clutter, and motion. The annotations include all people whose torso is partially visible, and include children and partially upright postures, but not people sitting. To make a direct comparison to the results published by Ess, we use their detection criteria (50% intersection-over-union) and restrict the annotations used in the same way they do (with height greater than 80 pixels for sequence 2 of the 2008 data, and 60 pixels for all other data). We completely omit sequence 1 of the 2008 data because we were unable to generate acceptable stereo depth maps based

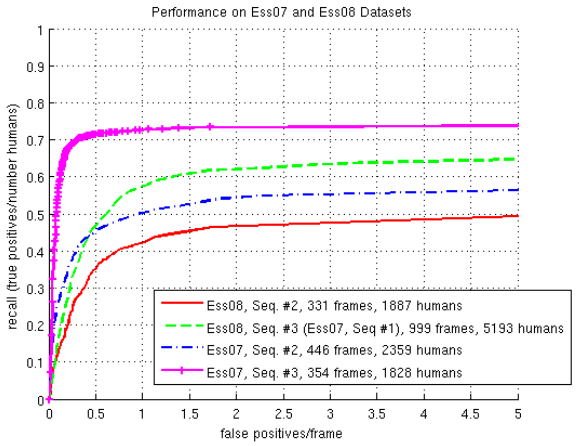
on the camera models provided. The depth data density on all other sequences is acceptable, but not as dense as it could be, and results in reduced performance as discussed later. For direct comparison, we also train on exactly the same data as well (sequence 0 of the 2007 data).

The performance of our end-to-end system with the Ess test sequences using exactly the same evaluation criteria are shown in Figure 6 (a). Although the performance does not appear very good (between 0.4 and 0.7 recall at 1 false positive per frame, and with maximum achievable recalls between 0.5 and 0.75), it is very similar to the results reported by Ess. In fact, the results are slightly better at 1 FAPF on all sequences except sequence 2 of the 2008 data (which is due to less stereo coverage). Examples of the scenes, along with stereo and the predicted velocity of certain pedestrians, are shown in Figures 7 and 8. Notice that people are detected when they are in various poses or stages of walking and while carrying bags or briefcases. The main cause of the missed detections is simply due to a lack of stereo depth data density on people who are either too close or occluded. To illustrate this point, we also show the performance for the sequences where annotated people must have at least 10% stereo coverage (of the pixels defined by the annotated bounding box) in Figure 6 (b). Because our system relies on stereo data for both detection and classification, it can never find these people, nor would it be able to localize them to plan around them in a fully autonomous mode.

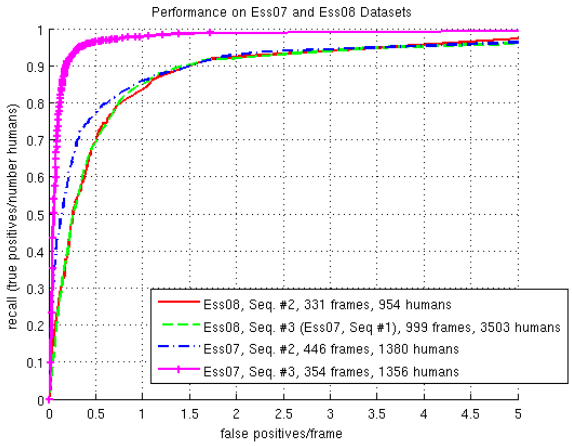
Our system misses detections and produces false positives in some understandable situations. For instance, it misses most children (left image of Figure 7), which were not included in any training data, and detects mannequins in shop windows or reflections of people in windows (right image of Figure 7). However, the majority of false detections is due to patchy stereo on flat surfaces such as buildings or cars, which results in the objects being over-segmented into a human sized objects (as seen on the car in the left image of Figure 8). Many times, this results in false positives high up on buildings (as seen in the center image of Figure 8), that could be removed by only considering people who might enter the street or be a danger. In other cases, explicitly detecting other objects such as cars would remove the false detections. Despite not designing for many of these situations, our system is capable of achieving competitive performance while running in real-time (10Hz on 640x480 imagery).

V. CONCLUSION

The results of our real-time, stereo-based pedestrian detection system show it to be effective at detecting people out to a range of 40m in semi-urban environments. It achieves results comparable with alternative approaches with other sensors, but offers the potential for long-term scalability to higher spatial resolution, smaller size, and lower cost than other sensors. It also performs similarly to state-of-the-art results from recent literature, while running at real-time rates.



(a)



(b)

Fig. 6. (a) The performance for sequences from [26] and [1] presented with the same evaluation criteria as their work. (b) the performance for the same sequences when all annotation that have less than 10% stereo coverage are eliminated, indicating that most of the misses in (a) are due to lack of stereo depth data on the people.

However, our system currently has some key limitations. Because the initial segmentation uses a projection into a 2D map, it cannot segment people or objects in close contact. To address this problem, we are investigating direct disparity-space and image-space segmentation techniques to provide regions of interest. Similarly, because we use a relatively small geometry-based feature set for classification, it is inherently limited. Any object with a similar shape to a person may be misclassified. To address this problem, we are investigating using appearance and motion features to improve classification. We are also using these extensions to handle the cases of pedestrians under partial occlusion and in non-upright postures.

REFERENCES

[1] A. Ess, B. Leibe, K. Schindler, , and L. van Gool, "A mobile vision system for robust multi-person tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[2] A. Shashua, Y. Gdalyahu, and G. Hayun, "Pedestrian detection for driving assistance systems: single frame classification and system level performance," in *IEEE Intelligent Vehicles Symposium*, 2004.

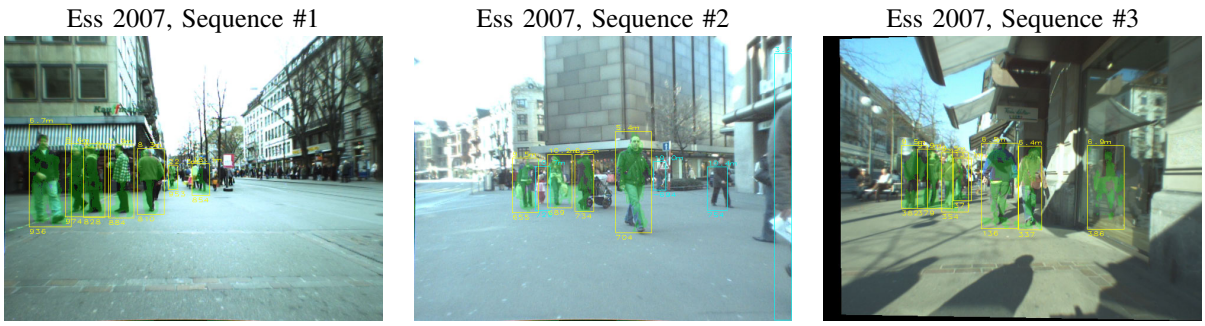


Fig. 7. Examples of detections (yellow boxes, with green overlay of segmented people) and misses (cyan boxes) for sequences from [26]. The false detection in the sequence 3 example is due to a reflection in the window.

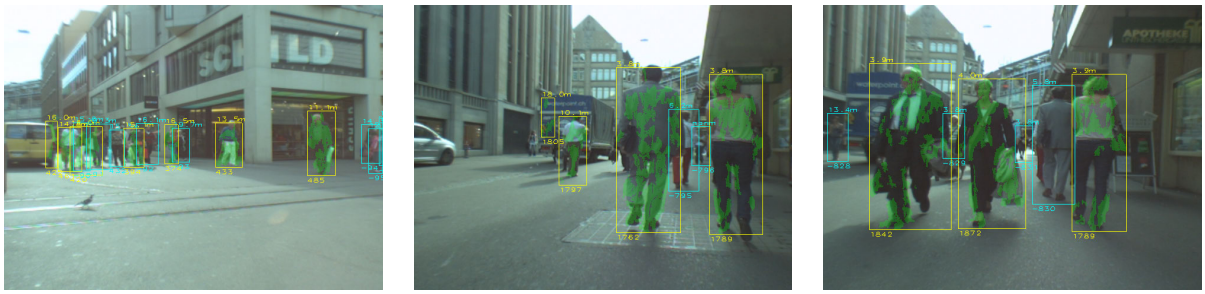


Fig. 8. Examples of detections (yellow boxes, with green overlay of segmented people) and misses (cyan boxes) for sequence 2 from [1]. There are false alarms on the car in the left image and the bus in the middle image. The misses are generally due to lack of stereo coverage or excessive clutter.

[3] G. Ma, S. Park, A. Ioffe, S. Muller-Schneiders, and A. Kummert, "A real time object detection approach applied to reliable pedestrian detection," in *IEEE Intelligent Vehicles Symposium*, 2007.

[4] R. Arndt, R. Schweiger, W. Ritter, D. Paulus, and O. Lohlein, "Detection and tracking of multiple pedestrians in automotive applications," in *IEEE Intelligent Vehicles Symposium*, 2007.

[5] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *International Journal of Computer Vision*, vol. 73, no. 1, pp. 41–59, 2007.

[6] B. Liebe, N. Cornelis, K. Cornelis, and L. V. Gool, "Dynamic 3d scene analysis from a moving vehicle," in *IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, 2007.

[7] M. Sotelo, I. Parra, D. Fernandez, and E. Naranjo, "Pedestrian detection using svm and multi-feature combination," in *IEEE Intelligent Transportation Systems Conference*, 2006.

[8] M. Bertozzi, A. Broggi, M. D. Rose, M. Felisa, A. Rakotomamonjy, and F. Suard, "A pedestrian detector using histograms of oriented gradients and a support vector machine classifier," in *IEEE Intelligent Transportation Systems Conference*, 2007.

[9] C. Tomiuc, S. Nedevschi, and M. M. Meinecke, "Pedestrian detection and classification based on 2d and 3d information for driving assistance systems," in *IEEE Intelligent Computer Communication and Processing Conference*, 2007.

[10] K. C. Fuerstenberg, K. Dietmayer, and V. Willhoeft, "Pedestrian recognition in urban traffic using a vehicle based multilayer laserscanner," in *IEEE Intelligent Vehicles Symposium*, 2002.

[11] S. M. Thornton, M. Hoffelder, and D. D. Morris, "Multi-sensor detection and tracking of humans for safe operations with unmanned ground vehicles," in *Workshop on Human Detection from Mobile Platforms, IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[12] L. E. Navarro-Serment, C. Mertz, and M. Hebert, "LADAR-based pedestrian detection and tracking," in *Workshop on Human Detection from Mobile Platforms, IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[13] A. Howard, L. Matthies, A. Huertas, M. Bajracharya, and A. Rankin, "Detecting pedestrians with stereo vision: safe operation of autonomous ground vehicles in dynamic environments," in *International Symposium of Robotics Research*, 2007.

[14] M. Bajracharya, B. Moghaddam, A. Howard, and L. Matthies, "Detecting personnel around UGVs using stereo vision," in *SPIE Unmanned Systems Technology X*, vol. 6962, March 2008.

[15] W. Abd-Almageed, M. Hussein, M. Abdelkader, and L. Davis, "Real-time human detection and tracking from mobile vehicles," in *IEEE Intelligent Transportation Systems Conference*, 2007.

[16] B. A. Bodt, "A formal experiment to assess pedestrian detection and tracking technology for unmanned ground systems," in *26th Army Science Conference*, December 2008.

[17] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *IEEE International Conference on Compuer Vision*, 2003.

[18] T. Zhao, R. Nevatia, and B. Wu, "Segmentation and tracking of multiple humans in crowded environments," in *IEEE Trans. Pattern Analysis and Machine Intelligence (to appear)*, 2008.

[19] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[20] P. Sabzmeydani and G. Mori, "Detecting pedestrians by learning shapelet features," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[21] B. Wu and R. Nevatia, "Simultaneous object detection and segmentation by boosting local shape feature based classifier," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[22] O. Tuzel, F. Porikli, and P. Meer, "Human detection via classification on Riemannian manifolds," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[23] E. Seeman, M. Fritz, and B. Schiele, "Towards robust pedestrian detection in crowded image sequences," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[24] S. Goldberg, M. Maimone, and L. Matthies, "Stereo vision and rover navigation software for planetary exploration," in *IEEE Aerospace Conference*, March 2002.

[25] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS)*, 2008.

[26] A. Ess, B. Leibe, and L. V. Gool, "Depth and appearance for mobile scene analysis," in *International Conference on Computer Vision (ICCV)*, October 2007.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Motion Planning for People Tracking in Uncertain and Dynamic Environments

Tirthankar Bandyopadhyay, Nan Rong, Marcelo Ang, David Hsu, Wee Sun Lee

***Abstract*—Target tracking is an important capability for autonomous robots. The goal of this work is to construct *motion strategies* for a robot so that it can handle visual and mobility obstruction due to obstacles and maneuver effectively to track a mobile target in a dynamic, uncertain environment. There are two broad approaches to address dynamic changes and uncertainties in the environment: to react fast or to plan ahead. The choice often depends on the amount of prior information available on the environment and the target behavior. This paper gives an overview of our work on target tracking using these two approaches.**

First, we present a greedy algorithm. It uses purely local geometric information from the robot's sensor to compute the robot's motion at each time step, and yet carefully balances the robot's ability to track the target in both the current and the future time. The algorithm uses only information from the robot's sensor and requires no prior information on the environment or the target behavior. This has been shown to work well on a real robot with a 2-D laser sensor in a crowded school cafetaria.

Second, we use partially observable Markov decision process (POMDP) to build a model of target behavior. As a result, the robot is capable of more sophisticated tracking behavior. For example, it may intentionally allow the target to get out of sight in order to minimize its own movement and save energy, but does not compromise long-term tracking performance. This is ongoing work and we show simulation results demonstrating the effectiveness of the approach.

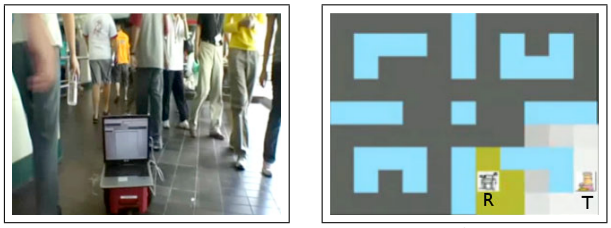
I. Introduction

Target tracking has many applications. In home care settings, a tracking robot can follow elderly people around and alert caregivers of emergencies. In security and surveillance systems, tracking strategies enable mobile sensors to monitor moving targets in cluttered environments. In this paper, we focus on developing motion strategies for a robot equipped with visual sensors so that it can effectively track and follow a moving target, despite obstruction by obstacles. Target identification, an important component of target tracking is assumed.

Just as in classic motion planning [13], we must consider *motion constraints* resulting from both obstacles in the environment and the robot's mechanical limitations. In particular, the robot must not collide with obstacles. Target following has the additional *visibility constraints* due to sensor limitations, *e.g.*, obstacles blocking the view of the robot's camera. Both motion constraints and visibility constraints play a significant role for target following in cluttered and dynamic environments.

The robot can address dynamic changes and uncertainties in the environment either by reacting fast to each changes or by modeling these uncertainties and planning ahead. The choice depends on the availability of prior information to the robot. When the environment or the target behavior is unknown, the robot has to plan its motion based on just the local information available and try to maximize the duration for which it can keep the target in view. On the other hand, when the environment is known and the target behavior can be modeled, the robot can incorporate this information to generate sophisticated motion strategies that maximizes the overall time that the target is in view.

This paper gives an overview of our work in following the target using these two approaches. In the rest of this section, we motivate and describe the approaches in light of two concrete examples (Figure 1).



(*a*) (*b*)

Fig. 1. Different scenarios: (a) Crowded canteen environment : Highly dynamic and unknown environment, suitable for local planning (b) Home care application : Uncertainty in target's position handled by POMDP tracker can generate sophisticated behaviors.

Let us take a specific scenario of an automated personal shopping assistant following an elderly person in a shopping mall, or keeping an eye on young kids while their parents shop. The shopping mall is a complex environment. People walking around add to the visual occlusions and motion obstructions, thereby creating a highly cluttered and dynamic environment (Figure 1*a*). While the layout of the environment might be available in some cases, exact maps for localizing the robot are hardly provided. On top of that, the target can be completely unpredictable in moving from one shop to another. In such situations where little is known about the target behavior or the environment, a local greedy strategy is more effective than complete planning.

Key to our algorithm is the definition of a risk function, which tries to capture the targets ability in escaping from the

Tirthankar Bandyopadhyay is with CENSAM IRG, SMART tirtha@smart.mit.edu

Nan Rong is a PhD. student at CMU nan.rong@gmail.com

Marcelo Ang is Associate Professor in Department of Mechanical Engineering, National University of Singapore, mpeangh@nus.edu.sg

David Hsu and Wee Sun Lee are Associate Professors in Department of Computer Science, National University of Singapore, dyhsu@comp.nus.edu.sg, leews@comp.nus.edu.sg

robot sensors visibility region in both short and long terms. To select actions effectively, the robot must balance between the short-term goal of preventing the immediate loss of the target and the long-term goal of keeping it visible for the maximum duration possible. Interestingly, a good compromise can be achieved, using only local information available to the robots sensors. By analyzing the local geometry, our algorithm computes a global risk function as a weighted sum of components, each associated with a single visibility constraint. It then chooses an action to minimize the risk locally in a greedy fashion.

As the algorithm uses only local geometric information available to the robots visual sensors, it does not require a global map and thus bypasses the difficulty of localization with respect to a global map. Furthermore, uncertainty in sensing and motion control does not accumulate. This improves the reliability of tracking. We have tested the algorithm in a crowded school cafeteria at lunch time. The crowd of students moving towards food stalls and then towards their seats create a truly dynamic and cluttered environment. Our implementation shows that the tracker is robust to temporary occlusions and in uneven terrain. The algorithm scales well with high clutter and obstructions and shows good performance for reasonable target behavior.

On the other hand, if enough information is available on the environment and the target behavior, the prior information can be used by the tracker to come up with 'smarter' strategies to improve the tracking performance. We formulate the tracking problem into a POMDP framework. POMDP trackers integrate global information on the target behavior and the environment for optimal decision making. Let us take a specific scenario from the homecare application. Imagine that an elderly person moves around at home and has a call button to call a robot over for help (Figure 1*b*). The call status stays on for some time and then goes off. If the robot arrives while the call status is on, it gets a reward; otherwise, it gets no reward. Clearly, the robot should stay close to the person in order to improve the chance of receiving rewards, but at the same time, the robot needs to minimize movement in order to reduce power consumption. Moreover, there might be regions where the robot might not be allowed to follow, e.g. bathroom etc. So the naive strategy of following right behind the person does not work well. The map of the environment is available to the robot but there are uncertainties in the location of the target and the robot itself w.r.t this map.

The problem of target tracking comprises of *target searching* and *target following*. By modeling target tracking as a partially observable Markov decision process (POMDP) [20], searching and following can be unified. The main idea is to represent the target position as a probability distribution, whether the target is visible to the robot sensors or not. So the target position is always "known" to the robot with some degree of uncertainty. The robot then chooses its actions according to a probabilistic model of target behaviors and a reward function that encourages the robot to keep the target visible.

The POMDP framework offers several other advantages. It provides a principled way to deal with uncertainties in robot control and sensing. It also easily incorporates additional requirements, *e.g.*, minimizing the robot's power consumption.

We formulate the tracking problem as a POMDP and use a sampling based algorithm SARSOP [10] to generate interesting tracking behaviors, *e.g.*, anticipatory moves that exploit target dynamics, information-gathering moves that reduce target position uncertainty, and energy-conserving actions that allow the target to get out of sight, but do not compromise tracking performance.

II. Prior Work

Target tracking has received tremendous amount to attention. One important part of target tracking is to detect and identify the target(s) from noisy, error prone and uncertain sensor data. Our mention of a few passing references below, is by no means representative of the work by the community. For single targets, kalman filter [4] and particle filters [11] have been used. For multiple targets, Joint Probability Data Association Filters (JPDAF) was proposed [8] which was implemented for people tracking among others by [19]. Multi-Hypothesis Tracking (MHT) was proposed by Reid [18]. An interesting and quite recent work on leg tracking has been described in [1].

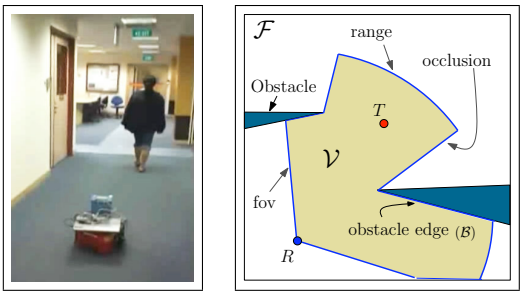
Motion strategies for target tracking depend on the amount of information available. If both the environment and the target trajectory are completely known, optimal target following strategies can be computed through dynamic programming [14], or by piecing together certain canonical curves [6], If only the environment is known, one can preprocess the environment by decomposing it into cells separated by critical curves. The decomposition helps to identify the best robot action as well as to decide the feasibility of tracking [15]. If the environment and the target trajectory are both unknown in advance, one approach is to move the robot so as to minimize an objective function that tries to capture the short- and long-term risk of losing the target [3], [9], [16]. With few exceptions [7], Most of these approaches do not handle uncertainties in robot control and sensing. Other probabilistic approaches to target tracking include, *e.g.*, [21].

Our POMDP tracking problem is related to the Tag problem described in [17]. However, the problems considered here involve a much larger number of states and more complex target behaviors. The SARSOP algorithm is also more efficient than the PBVI algorithm used in [17] and can handle more realistic target tracking tasks.

Another potential difficulty with the POMDP approach is the acquisition of a good probabilistic model of target behavior, but machine learning techniques can help [5].

III. Local Greedy Tracker

For an unknown environment and an unknown target behavior, the robot must execute an online reactive strategy that takes into account only local information. In this work, to identify and track a person, we use visibility based sensors, based on



(*a*) (*b*)

Fig. 2. Formulation of the target tracking problem into geometrical parameters extracted from local information

the standard straight line-of-sight visibility model (Figure 2*b*). In the free space $\mathcal{F}$, the *visibility set* $\mathcal{V}(x)$ is given by,

$$\mathcal{V}(x) = \{q \in \mathcal{F} \mid \overline{xq} \subset \mathcal{F} \text{ and } d(x,q) \leq D_{max} \text{ and } \theta_{min} \leq ang(x,q) \leq \theta_{max}\}$$

where $d(x,q)$ denotes the distance between x and q, while $ang(x,q)$ is the orientation of q w.r.t. x. Information about the local environment is encoded into the boundary ($\partial\mathcal{V}$), of the visibility polygon ($\mathcal{V}$).

Both the robot and the target's motion, are formulated with a simple discrete-time constant velocity model. As the target behavior is unknown, its velocity ($\mathbf{v}'$) is modeled by a gaussian around its current heading : $\mathbf{v}'(t+\Delta t) = \mathcal{N}(\mathbf{v}'(t), \sigma)$. The variance σ gives a measure of confidence in estimating the target velocity. Although we use a Gaussian distribution to model the uncertainty in the target behavior, the approach remains valid for any other velocity prediction method, even non-parametric ones.

A. Local Greedy Approach Overview

The objective of the robot is to keep the target inside the robot's visibility, $\mathcal{V}$, for as long as possible. The target can escape $\mathcal{V}$ through its boundaries that lie in free space. We term these boundaries as *escape edges* (Figure 2*b*). Since the robot has no control over the enviornment or the target's motion, it can only prevent the target's escape by manipulating the *escape edges*, $\{\mathcal{G}_i\}$ away from the target. The ability of the robot to manipulate $\mathcal{G}_i$ effectively is important in maintaining the target in view. Let us denote the manipulation ability of the robot for a single escape edge, $\mathcal{G}_i$, by the symbol, $\Delta\mathcal{G}_i$. $\Delta\mathcal{G}_i$ is a function of the robot position, $\mathbf{x}$, and its actions, $\mathbf{v}$: $\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})$. The risk of losing the target, on the other hand, depends on : (a) the target position ($\mathbf{x}'$), (b) the relative target velocity ($\mathbf{v}'$) w.r.t. to $\{\mathcal{G}_i\}$, and finally (c) the robot's ability to manipulate the edges, $\Delta\mathcal{G}_i$. We can then formulate a risk function (Φ) and choose the robot action, $\mathbf{v}^\star$, to minimize Φ:

$$\begin{aligned} Risk &= \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}) \\ \mathbf{v}^\star &= \textbf{arg min } \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}) \end{aligned} \quad (1)$$

While Φ is the risk of losing the target through any escape edge in the entire $\mathcal{V}$, we can assign a risk φ_i, of losing the target to each escape edge, $\mathcal{G}_i$. We approximate the total risk Φ, by the expected risk for all the gaps.

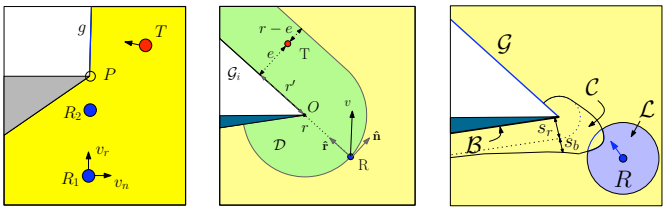
$$\Phi \approx E[\varphi_i] = \sum_i p_i \varphi_i(\mathbf{x}', \mathbf{v}', \mathcal{G}_i, \Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})), \mathbf{v}^\star \approx \sum_i p_i v_i^\star \quad (2)$$

where p_i is the probability of the target's escape through $\mathcal{G}_i$. p_i is computed based on the target's current velocity, $\mathbf{v}'$. The details can be found in [3].

However, in choosing $\mathbf{v}^\star$, the robot has to satisfy many constraints on the desired robot positions, *e.g. obstacle avoidance* considerations or on the robot's actions like *kinematic, dynamic constraints*. We define a *feasible region*, $\mathcal{L}$ (x), that satisfies all the constraints ($\mathcal{C}_i(x)$) in the position domain : $\mathcal{L}(x) = \bigcap_i \mathcal{C}_i(x)$. The local greedy optimization then chooses an action ($\mathbf{v}^\star$), that minimizes Φ while satisfying $\mathcal{L}$ (x) in the time step Δt,

$$\mathbf{v}^\star = \textbf{arg min } \Phi(\mathbf{x}', \mathbf{v}', \{\mathcal{G}_i\}, \{\Delta\mathcal{G}_i(\mathbf{x},\mathbf{v})\}), s.t \quad \mathbf{v}^\star\Delta t \in \mathcal{L}(\mathbf{x}) \quad (3)$$

B. Risk Formulation : φ



(*a*) (*b*) (*c*)

Fig. 3. (*a*) Relative position determines risk, (*b*) Local geometric parameters, (*c*) Obstacle Dilation

For successful tracking, the robot must balance the short-term goal of preventing the immediate loss of the target through these escape edges and the long-term goal of maximizing the duration of tracking in the future. Let us look at a simple 2-D example shown in Figure 3*a*.

For the robot positioned at R_1, the obstacle (the dark-colored triangle) creates an occlusion edge g with one endpoint at P. The robot has the short-term goal of preventing the target T's escape through g at the current instant. It achieves this by swinging g away from the target, using velocity v_n. The robot's longer-term goal is to move towards P using velocity v_r, because it can eliminate the occlusion edge g completely when it reaches P. Since the robot's maximum speed is bounded by V, there is a trade off in choosing the velocity components v_r and v_n. Clearly, this trade off depends on the relative positions and velocities of the target and the robot w.r.t P. For example, the robot at position R_2 can afford a higher v_r, as the shortest distance from the target to g is greater than that of the robot and there is no immediate risk of losing the target. Whereas at R_1, the target is closer to g than the robot, and the short-term goal of preventing the loss of target becomes much more important.

We formulate a risk function that incorporates this trade off between the current and future risk in terms of local geometrical parameters (Figure 3*b*).

In the previous work [3], a local greedy algorithm based on *relative vantage* was proposed. Relative vantage refers to the ability of the robot to eliminate $\mathcal{G}_i$ before the target can escape through it. We introduce a region around $\mathcal{G}_i$, called *vantage zone* as, $\mathcal{D} = \{q : q \in \mathcal{V}; dist(q, \mathcal{G}_i) \leq dist(\mathbf{x}, \mathcal{G}_i)\}$

The objective of the robot is to keep the targets away from $\mathcal{D}$ and accordingly, we can take the measure of time taken ($t_{r.v}$) to move the target outside $\mathcal{D}$, as the risk value. From Figure 3*b*,

$$\varphi_g = t_{r.v} \approx \frac{dist(\mathrm{t}, \mathcal{D})}{rel.vel(\mathrm{t}, \mathcal{D})} \approx \frac{r-e}{v_{\text{eff}}}, \qquad v_i^\star = \frac{\varphi_g}{v_{\text{eff}}} \left(\frac{r'}{r}\hat{\mathbf{n}} + \hat{\mathbf{r}} \right)$$

where $v_{\text{eff}} = v_r + v_n(r'/r) - v'_{\text{e}}$ is the effective velocity in the direction along the shortest path from the target to $\mathcal{G}_i$.

Similar considerations can be applied to the field of view (FoV) limits and the range limits. Derivation of these special cases are omitted due to space limitation. The reader is pointed to [2] for details.

C. Obstacle Avoidance

Although, purely low-level reactive obstacle avoidance techniques, can handle dynamic and unknown environment, they may sometimes move the robot contrary to the required tracking direction. On the other hand, planning in the configuration space may be too computationally expensive in a cluttered and dynamic environment. We propose a local obstacle avoidance method with a small look-ahead. The robot's velocity is used to enlarge the obstacle edges. These extended obstacles then constrain the planned robot motion.

First, we approximate the robot's size by the radius (s_r) of its bounding circle. Then, we compute the finite braking distance, s_b, using the maximum decceleration and the robot's current velocity. This braking distance, s_b and the robot's dimension, s_r, defines a collision region $\mathcal{C}$ (x), around the obstacle edge, $\mathcal{B}$, Figure 3*c*,

$$\mathcal{C}(x) = \{q \in \mathcal{V} : d(q, \mathcal{B}) \leq (s_r + s_b)\} \tag{4}$$

The robot can actively change the shape of $\mathcal{C}$ by changing its speed and heading. For safe navigation, the robot must avoid $\mathcal{C}$. If we denote the *reachable* region of the robot in Δt, as $\mathcal{R}$, the feasible region becomes $\mathcal{L} = \mathcal{R} - \mathcal{C}$. As an example, assuming omni-directional motion ability of the robot with no dynamics in Figure 3*c*, $\mathcal{R}$ is a disk of radius, $V\Delta t$, and the darker shaded region shown is $\mathcal{L}$. Appropriate motion models, non-holonomic constraints, motion dynamics *etc* change the shape of $\mathcal{R}$, but the basic approach remains the same.

We substitute the details of the escape edge risk and the obstacle avoidance constraints in expression

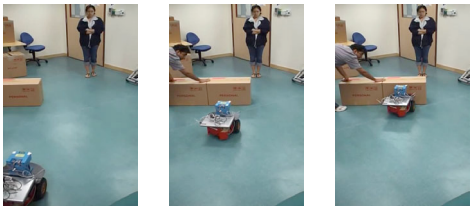
$$\mathbf{v}^\star = \sum_i p_i v_i^\star \quad s.t \quad \mathbf{v}^\star \Delta t \in \mathcal{L}(\mathbf{x}) \tag{5}$$

D. Experimental Results

The tracking algorithm is implemented on a Pioneer P3-DX differential drive robot. A SICK-lms200 range sensor is mounted on the robot. The laser returns 361 readings on a field of view of 180deg at the resolution of 0.5deg. The maximum range of the sensor is 8m. The control algorithm runs on a Pentium M Processor @1.5GHz laptop running Player server v-2.0.5 on linux. The algorithm runs at 10Hz. Implementation details are described in [2].

In the following we show a comparison of our algorithm with visual servo algorithm. Subsequently we showcase two of our experimental runs that was performed in the school cafeteria. The videos of these and more experiments performed on indoor, canteen and outdoor tracking are available online at http://guppy.mpe.nus.edu.sg/~tirtha/research/Hardware/hardImpl.html. More detailed analysis and comparison to existing algorithms is available in [2].

1) Comparison with Visual Servo (Figure 4 & Figure 5) : In Figure 5, a box is pushed between the target and the robot to occlude the target. The responses of a simple visual servo algorithm is compared to the vantage tracker. Since, the vantage tracker actively tries to avoid possible future occlusions, it is able to adapt to the changing environment (Figure 5*b-1*). A point to note is that the vantage tracker does not model the motion of the environment but just replans its motion at a high frequency, making the tracker independent of the dynamic nature of the environment. Later, when the box stops and the target starts to move (Figure 5*c*), the tracker is able to successfully follow the target (Figure 5*d*). In comparison, the simple visual servo tracker does not model the dynamic environment and loses the target from its view (Figure 4).



(*a*) (*b*) (*c*)

Fig. 4. Visual Servo : Since the robot does not take into account the environment information (the moving box), it moves straight ahead towards the target (b) and loses the target behind the occluding box (c).

2) Tracking in a Crowd (Figure 6) : This experiment was done during lunch hour to capture the dynamic environment of the canteen at peak rush time. The robot follows the target in grey t-shirt (1). As the target moves into the canteen area the crowd keeps increasing (2,3,4). Moreover in (3,4,5) the robot has to maneuver through a narrow pathway while avoiding incoming people and keeping the original target in view which makes following the target more difficult.

3) Visual Occlusions (Figure 7): A challenging aspect of following the target in a crowd is when someone walks in

Fig. 6. Target following lunch hour rush crowd at school cafeteria

Fig. 7. Fast online local greedy algorithm is robust to temporary occlusions

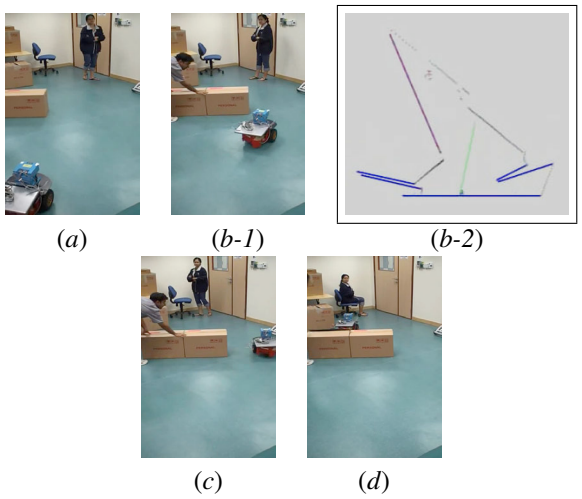


Fig. 5. Vantage tracker : (*b-2*) shows the robot's local perception of the environment. The target is marked by T, the blue lines are the occlusion edges, red line is the most critical occlusion and the green segment starting from R denotes the robot's motion decision. The robot sees the target too close to the occlusion and swings out.

between the robot and the target. In this set of snapshots, the robot is following the target in green t-shirt when it faces an temporary occlusion by a lady (in purple) walking across unexpectedly (2,3,4). The robot slows down to avoid collision (3,4) and returns to following the target when the occlusion has passed. Due to the fast online nature of the tracking algorithm, temporary occlusions in such a dynamic environment is handled well by the robot.

IV. POMDP Tracker

We start with a brief review of POMDPs. See [12] for a more complete introduction. We then describe how to model the target tracking problem as a POMDP.

A. Background on POMDPs

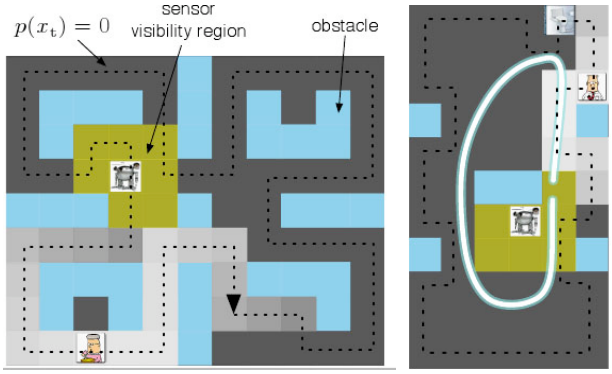
A POMDP models an agent taking a sequence of actions under uncertainty to maximize its total reward. Formally it is specified as a tuple $(S, A, \mathcal{B}, T, Z, R, \gamma)$, where S is a set of states, A is a set of actions, and $\mathcal{B}$ is a set of observations.

The agent always lies in some state $s \in S$. In each time step, it takes some action $a \in A$ and moves from a start state s to an end state s'. Due to the uncertainty in action, the end state s' is described as a conditional probability function $T(s, a, s') = p(s'|s, a)$, which gives the probability that the agent lies in s', after taking action a in state s. The agent then makes an observation on its current state. Due to the uncertainty in observation, the observation result $o \in \mathcal{B}$ is again described as a conditional probability function $Z(s, a, o) = p(o|s, a)$ for $s \in S$ and $a \in A$.

In each step, the agent receives a real-valued reward $R(s, a)$, if it lies in state s and takes action a. The goal of the agent is to maximize its expected total reward by choosing a suitable sequence of actions. In this work, we consider infinite-horizon POMDPs, in which the sequence of actions to be chosen has infinite length. We specify a discount factor $\gamma \in (0, 1)$ so that the total reward is finite and the problem is well defined. In this case, the expected total reward is $\mathrm{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)]$, where s_t and a_t denote the agent's state and action at time t.

The solution to a POMDP is an optimal *policy* that maximizes the expected total reward. Normally, a policy is a mapping from the agent's state to a prescribed action. However, in a POMDP, the agent's state is partially observable and not known exactly. So we rely on the concept of *belief state*, or *belief*, for short. A belief is a probability distribution over S. A POMDP policy $\pi: \mathcal{B} \rightarrow A$ maps a belief $b \in \mathcal{B}$ to the prescribed action $a \in A$.

A policy π induces a value function $V^{\pi}(b)$ that specifies the expected total reward of executing policy π starting from b. It is known that V^*, the value function associated the optimal policy π^*, can be approximated arbitrarily closely by a convex and piecewise-linear function $V(b) = \max_{\alpha \in \Gamma}(\alpha \cdot b)$, where b is a discrete vector representation of a belief and Γ is a finite set of vectors called α-vectors. Each α-vector is associated with an action, and the policy can be executed by selecting the action corresponding to the best α-vector at the current



(*a*) (*b*)

Fig. 8. Simulation experiments for target tracking.

belief b. So the policy can be represented by a set Γ of α-vectors. Policy computation, which, in this case, involves the construction of Γ, is usually performed offline.

Given an policy π, the control of the agent's actions is performed online in real time. It consists of two steps executed repeatedly. The first step is policy execution. If the agent's current belief is b, it then takes the action $a = \pi(b)$, according the given policy π. The second step is belief estimation. After the agent takes an action a and receives an observation o, its new belief state b' is given by

$$b'(s') = \tau(b, a, o) = \eta Z(s', a, o) \sum_{s \in S} T(s, a, s') b(s),$$

where η is a normalizing constant. The process then repeats.

B. Target Tracking as a POMDP

Our problem setting is motivated by homecare applications. Imagine that an elderly person moves around at home and has a call button to call a robot over for help. The call status stays on for some time and then goes off. If the robot arrives while the call status is on, it gets a reward; otherwise, it gets no reward. Clearly the robot should stay close the person in order to improve the chance of receiving rewards, but at the same time, the robot needs to minimize movement in order to reduce power consumption. So the naive strategy of following right behind the person does not work well.

When the environment information is known and the target behavior is known, we propose a POMDP tracker. To formulate the problem as a POMDP, we model the environment as a regular grid. See Figure 8 for examples. The robot and the target (in this case, the person with the call button) can occupy any of the grid cells that are free of obstacles. The state s of this POMDP is composed of the robot position x_r, the target position x_t, and the call status c: $s = (x_r, x_t, c)$. If the environment contains n free cells, then there are $n \cdot n \cdot 2 = 2n^2$ distinct states, resulting in a belief space of $2n^2$ dimensions.

In one time step, the target can stay where it is or move to a neighboring cell. The target motion is described by a given probability function T_t, conditioned on the target's current position: if the target is currently at x_t, it will be at x'_t in the next time step with probability $T_t(x_t, x'_t) = p(x'_t|x_t)$.

The person may turn on the call button in each step with probability p_1. If the call status is on, the person may turn it off with some probability p_2 in each time step, indicating that help is no longer needed. This model has two main implications. First, as the call duration follows the geometric distribution, the mean duration of a call is $1/p_2$, Second, most calls are short. The robot must arrive quickly in order to receive rewards, thus increasing the difficulty of tracking.

The robot motion resulting from an action is described similarly by another probability function T_r, conditioned on both the robot's current position and its action. The robot's actions consist of commands to stay where it is or to move to a neighboring cell. If the robot is currently at x_r and takes action a it will be at x'_r in the next time step with probability $T_r(x_r, a, x'_r) = p(x'_r|x_r, a)$. Note that the robot may not be able to execute the commands perfectly due to control uncertainty. This can be modeled with a suitable T_r.

We assume that the robot can see the target through its sensors if they lie in the same or neighboring cells. Uncertainty on the target position due to sensor noise can be modeled in the observation probability function Z.

The robot receives a reward, if it reaches the cell that the target occupies while the call button is on. In one step, if the robot does not move, it incurs no costs (*i.e.*, negative rewards). Otherwise, it incurs a cost proportional to the distance traveled. The robot's goal is to maximize the expected total discounted reward.

The POMDP formulation does not explicitly differentiate whether the target is visible or not. To execute a policy, the robot maintains a belief of the target position. When the target is visible to the sensors and the sensor data are good, the belief is sharpened. When the target is not visible or the sensor data are poor, the belief becomes more diffuse. In the extreme case, when the target remains invisible for a long time, the belief may eventually converge to a uniform distribution. This way, target searching and target following are unified in a natural way. Clearly, if the robot knows the target position well, it can choose better actions and receive higher rewards. Therefore, an optimal policy favors sharp beliefs, while also taking into account the cost of obtaining them.

C. Simulation Results

We used SARSOP [10] to compute tracking policies in several simulated environments. See Figure 8 for examples. The light blue areas in the figures indicate obstacles. The black dashed curve indicates the target's path. The target motion is non-deterministic: it follows this path, but in each time step, it may pause or proceed along the path with equal probabilities. The green area around the robot indicates the robot sensor' visibility region. The various shades of gray show the robot's belief of the target position. Lighter color indicates higher probability. To focus on target tracking behaviors, we assume in these experiments that there is no uncertainty in robot control and sensing. The robot can execute motion commands and observe its own position and call status perfectly. It can also observe the target position perfectly, if the target is visible.

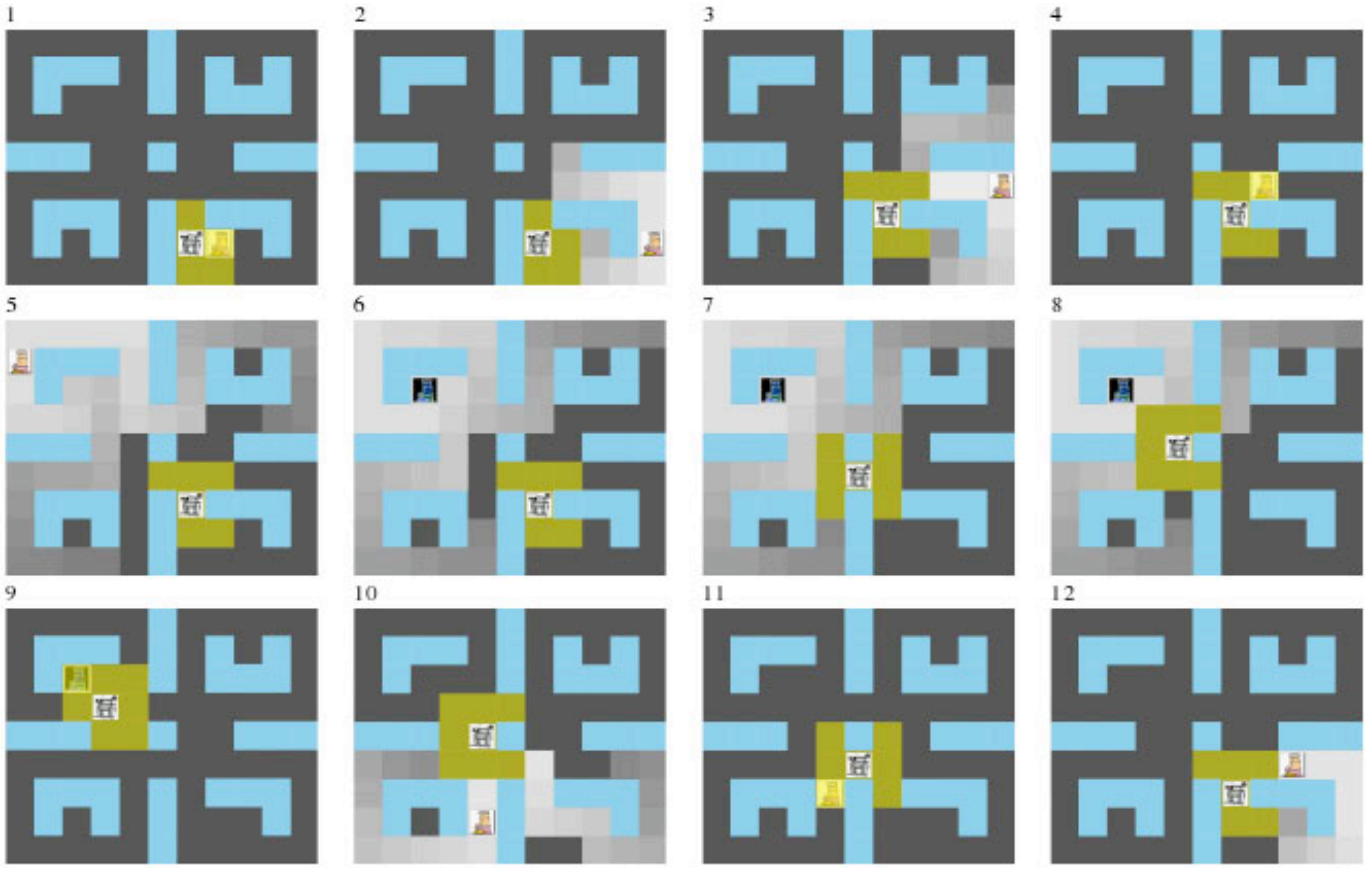


Fig. 9. Snapshots of a simulation run.

Uncertainties in control and sensing can be easily incorporated into the POMDP if needed. If the robot reaches the current target position while the call status is on, it receives a reward of 100. The robot receives a reward of -1 for a horizontal or vertical move, a reward of $-\sqrt{2}$ for a diagonal move, and a reward of 0 if it stays stationary. The discount factor is set to 0.95.

In the first experiment, we have a home-like environment (Figure 8*a*). The corresponding POMDP has $9,248$ states. SARSOP computed a policy in about 48 minutes. We performed several simulation runs to examine its performance and observed interesting robot tracking behaviors:

- anticipatory moves that exploit target dynamics,
- information-gathering moves that reduce target position uncertainty,
- approaching the target along a nearly optimal path when the robot is called,
- minimizing movement by allowing the target to get out of sight, but not compromising long-term tracking performance.

It is important to bear in mind that these behaviors are not manually specified, but automatically captured by the POMDP through policy computation.

Snapshots of a single simulation run are shown in Figure 9. Initially, the target lies within the robot sensor's visibility region, and the robot's belief on target position consists of a single peak (snapshot 1). As the target moves, the robot does not follow along and intentionally let the target get out of sight, in order to minimize movement and reduce energy consumption. Now, although the target is *not* visible, the robot still has the target reasonably well localized by maintaining a belief on the target position: the target is well within the high-probability region of the current belief (snapshot 2). Instead of following the target, the robot tries to anticipate the future position of the target by exploiting the target dynamics and makes a move towards this position (snapshot 3). As there is no call, the robot's move purely serves the purpose of gathering information on the target position. When the target passes by, the belief on target position is sharpened (snapshot 4). If the target is not observed for a while, the uncertainty may become large, but the robot is still able to maintain a belief that reflects the current target position well: the target is located within a high-probability region (snapshot 5). When there is a call (snapshot 6), it uses the current belief to find the region that contains the target with high probability. It then moves towards the region along the shortest path (snapshots 6–9). In general, the robot may need to search this region, but here it luckily finds the target right away and receives a high reward (snapshot 9). The robot then makes another anticipatory move to reduce target position uncertainty

(snapshots 10–12). Interestingly, the robot position in snapshot 12 is exactly the same as that in snapshot 3, despite that the target positions and beliefs are quite different. It is, of course, not coincidence. This particular position guards both of the two ways into the lower right corner of the environment. By occupying this position, the robot can intercept the target as it exits the entrances without following it. The tracking behavior here reveals that the computed policy captures well the interaction between the environment geometry and the target dynamics. In this simulation run, there are 3 calls in total, and all are answered in time. The target travels a total distance of 141, while the robot about 20.

In the second experiment, the environment contains a special cell corresponding to a bathroom lying on the target's path (Figure 8*b*). After entering the bathroom, the target stays there with probability 0.95 and leaves with probability 0.05 in each step. The corresponding POMDP has $7,200$ states. SARSOP computed a policy in about 16 minutes. Roughly, to execute this tracking policy, the robot moves on the inner loop (the thick white curve in Figure 8*b*) and follows the target that moves along the outer loop (the dashed black curve in Figure 8*b*). It approaches the target directly when called.

Videos of both experiments above as well as additional experiments are available at http://motion.comp.nus.edu.sg/projects/tracking/tracking.html. We are currently performing more experiments to evaluate tracking performance quantitatively.

V. Conclusion

In this paper, we gave a brief overview of two approaches that are adept at tackling the problem of target tracking in different scenarios depending on the information available for planning. When the environment and the target behavior is unknown, an online greedy algorithm that acts based only on local information is proposed. This has been shown to work on hardware in the school cafetaria on a crowded lunch hour. We also compare this work with visual servo in a controlled setup to show the inherent improvement of the approach. On the other hand, for known environment and target models, the POMDP tracker provides more sophisticated behaviors, where it could lose the target temporarily to minimize its energy consumption while not compromising the tracking effectiveness. Simplified assumption about the sensing and motion models have been made. Early simulation results show sophisticated robot behaviors.

In this work the target identification is assumed. The target identification can be seen as a complimentary problem to the motion planning aspect of target tracking. Improved techniques for target disambiguation and development of target's motion models help in the task of target following. On the other hand, motion planning for maintaining a good view of the target aids the sensors to continously sense the target improving the identification and modeling of the target reliably. Basic constraints of the sensors like the field of view and range limitations can be incorporated into the motion strategy such that the robot always keeps the sensor facing the target. Uncertainty in the target track can be included as an objective function for the robot to minimize by planning a suitable motion strategy.

References

[1] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard. Efcient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In *Proceedings of the Int. Conf. on Robotics and Automation.*, 2008.

[2] T. Bandyopadhyay, D. Hsu, and M. Ang Jr. Motion strategies for people tracking in cluttered dynamic environments. In *Proc. Int. Symp. on Experimental Robotics*, 2008.

[3] T. Bandyopadhyay, Y. Li, M. Ang Jr., and D. Hsu. A greedy strategy for tracking a locally predictable target among obstacles. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 2342–2347, 2006.

[4] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press Inc., Orlando, Florida, 1988.

[5] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun. Learning motion patterns of people for compliant robot motion. *Int. J. Robotics Research*, 24(1):3148, December 2005.

[6] A. Efrat, H. González-Baños, S. Kobourov, and L. Palaniappan. Optimal strategies to track and capture a predictable target. In *Proc. IEEE. Int. Conf. on Robotics & Automation*, pages 3789–3796, 2003.

[7] P. Fabiani, H. González-Baños, J. Latombe, and D. Lin. Tracking a partially predictable target with uncertainties and visibility constraints. *J. Robotics & Autonomous Systems*, 38(1):31–48, 2002.

[8] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, OE-8(3):173184, July 1983.

[9] H. González-Baños, C.-Y. Lee, and J.-C. Latombe. Real-time combinatorial tracking of a target moving unpredictably among obstacles. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 1683–1690, 2002.

[10] D. Hsu, W. Lee, and N. Rong. A point-based POMDP planner for target tracking. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 2644–2650, 2008.

[11] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29-1:5–28, 1998.

[12] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[13] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[14] S. LaValle, H. González-Baños, C. Becker, and J. Latombe. Motion strategies for maintaining visibility of a moving target. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 731–736, 1997.

[15] R. Murrieta, A. Sarmiento, and S. Hutchinson. A motion planning strategy to maintain visibility of a moving target at a fixed distance in a polygon. In *IEEE Int. Conf. on Robotics & Automation*, 2003.

[16] R. Murrieta-Cid, H. H. González-Baños, and B. Tovar. A reactive motion planner to maintain visibility of unpredictable targets. In *Proc. IEEE. Int. Conf. on Robotics & Automation*, pages 4242–4248, 2002.

[17] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *Proc. Int. Jnt. Conf. on Articial Intelligence*, page 477484, 2003.

[18] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24(6):843854, December 1979.

[19] D. Schulz, W. Burgard, D. Fox, and A. Cremens. People tracking with mobile robots using sample-based joint probabilistic data association filters. *International Journal of Robotics Research (IJRR),*, 22(2):99–116, 2003.

[20] R. Smallwood and E. Sondik. The optimal control of partially observable markov processes over a nite horizon. *Operations Research*, 21:1071–1088, 1973.

[21] R. Vidal, O. Shakernia, H. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Trans. on Robotics & Automation*, 18:662669, 2002.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Improved Multi-Person Tracking with Active Occlusion Handling

A. Ess[1], K. Schindler[1,2], B. Leibe[3], L. van Gool[1,4]

[1] Computer Vision Laboratory, ETH Zurich, Switzerland
[2] Computer Science Dept., TU Darmstadt, Germany
[3] UMIC Research Centre, RWTH Aachen, Germany
[4] ESAT/PSI-VISICS IBBT, KU Leuven, Belgium

{aess|leibe|schindler|vangool}@vision.ee.ethz.ch

***Abstract*— We address the problem of vision-based multi-person tracking in busy inner-city locations using a stereo rig mounted on a mobile platform. Specifically, we are interested in the application of such a system for autonomous navigation and path planning. In such a scenario, semantic information about the moving scene objects becomes important. In order to estimate this robustly, we combine classical geometric world mapping with multi-person detection and tracking. In this paper, we refine an approach presented in earlier work, which jointly estimates camera position, stereo depth, object detections, and trajectories based only on visual information. We analyze the influence of the trajectory generator, which forms part of any tracking-by-detection system, and propose a set of measures to improve its performance. The extensions are experimentally evaluated on challenging, realistic video sequences recorded at busy inner-city locations. The results show that the proposed extensions significantly improve overall system performance, making the resulting detecting and tracking capabilities an interesting component of future navigation system for highly dynamic scenes.**

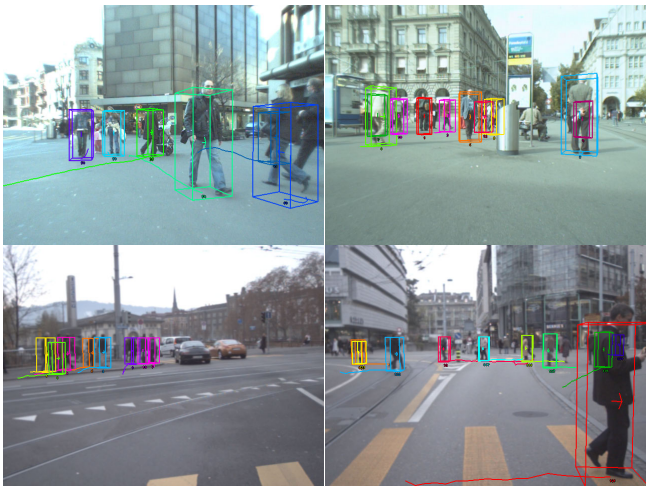


Fig. 1. Reliable tracking in busy urban scenarios requires careful design of trajectory (candidate) generation, accounting for partial occlusions, a multitude of scales, and measurement uncertainties.

I. Introduction

Reliable autonomous navigation of robots and cars requires appropriate models of their static and dynamic environment. While remarkable successes have been achieved in relatively clean highway traffic situations [3] and other largely pedestrian-free scenarios such as the DARPA Urban Challenge [7], scenes with many independently moving pedestrians, as in busy city centers, still pose significant challenges. What makes the task so much harder is the large number of independently moving actors that are frequently occluding each other. To represent such environments and make predictions for path planning, semantic information about the individual moving objects becomes a vital component.

Compared to range sensors such as LIDAR or SONAR, digital cameras offer the advantage that they deliver not only geometry, but also rich appearance information, which is more amenable to semantic interpretation. Recent work has shown that with modern computer vision tools, vision-based modeling of the environment for robot navigation is becoming possible [9], [27]. A key ingredient of these visual modeling approaches is that they partially rely on semantic *object category detection*—in the context of autonomous driving especially detection and tracking of cars and pedestrians.

For dynamic path planning, pedestrians need not only be detected, but should also be tracked over time in order to predict their future locations. However the two tasks are closely related: State-of-the-art approaches for people tracking in complex environments are based on the tracking-by-detection paradigm, in which the output of an (appearance based) object detector is linked between frames to recover pedestrian trajectories. In this work, we adopt such an approach for robust multi-person tracking and investigate some important design choices for improving its performance.

Our system is purely visual, using as input synchronized video streams from a forward-looking camera pair. Based on this data, the system continuously performs self-localization by visual odometry and obstacle detection using stereo depth and combines the resulting 3D measurements with tracking-by-detection, in order to follow pedestrians in the scene over time. Its results can be used directly as input for path planning algorithms which support dynamic obstacles. Key steps of our approach are the use of a state-of-the-art object detector for identifying an obstacle's category, as well as the reliance on a robust multi-hypothesis tracking framework to handle the complex data association problems that arise in crowded scenes. This allows our system to apply category-specific motion models for robust tracking and prediction. Our focus on vision alone does not preclude the use of other sensors such as LIDAR or GPS/INS—in any practical robotic system those sensors have their well-deserved place,

and their integration can be expected to further improve performance.

An important observation is that while each of the system components is affected by relatively strong noise, feedback between the components can remedy some of the resulting errors. Our system therefore has numerous feedback paths: we jointly estimate the ground surface and supporting object detections and let both steps benefit from each other; detections are transferred into world coordinates with the help of visual odometry and are grouped into 3D candidate trajectories by the tracker; selected tracks are then again fed back to stabilize visual odometry and depth computation through their predictions; finally, the results are combined in a dynamic occupancy map such as the one shown in Fig. 4(right), which allows free space computation for a later navigation module.

The main contribution of this paper is to investigate design options for the practical implementation of such a system and to evaluate their effects on overall performance. After reviewing related work (Sec. II) and the employed reconstruction and tracking system (Sec. III), we propose modifications to the trajectory generator (Sec. IV). The influence of these changes and of different stereo matching methods for depth computation are quantitatively evaluated in Sec. V.

II. Related Work

A main challenge in traffic scene understanding is to accurately detect moving objects in the scene. Such objects can be extracted independent of their category by modeling the shape of the road surface and treating everything that does not fit that model as an object (e.g. in [18], [24], [31]). However, such simple approaches break down in crowded situations where not enough of the ground may be visible. More accurate detections can be obtained by applying category-specific models, either directly on the camera images [6], [16], [23], [28], on the 3D depth information [1], or both in combination [10], [14], [25].

Tracking detected objects over time presents additional challenges due to the complexity of data association in crowded scenes. Targets are typically followed using classic tracking approaches such as Extended Kalman Filters (EKF), where data assignment is optimized using Multi-Hypothesis Tracking (MHT) [5], [20] or Joint Probabilistic Data Association Filters (JPDAF) [12]. Several robust approaches have been proposed based on those components either operating on depth measurements [21], [22], [26] or as tracking-by-detection approaches from purely visual input [13], [15], [17], [28], [30]. The approach employed in this paper is based on our own previous work [17]. It works online and simultaneously optimizes detection and trajectory estimation for multiple interacting objects and over long time windows by operating in a hypothesis selection framework.

III. System

Our vision system is designed for a mobile platform equipped with a pair of forward-looking cameras. From the synchronized videos, we estimate dense stereo depth, ground plane parameters, the platform's ego-motion, pedestrian tracks, and the locations of other (non-pedestrian) obstacles. Fig. 3(a) gives an overview of the proposed vision system. For each frame, the blocks are executed as follows. First, a depth map is calculated and the new frame's camera pose is predicted. Then objects are detected together with the supporting ground surface, taking advantage of appearance, depth, and previous trajectories. The output of this stage, along with predictions from the tracker, helps stabilize visual odometry, which updates the pose estimate for the platform and the detections, before running the tracker on these updated detections. As a final step, we use the estimated trajectories in order to predict future locations for dynamic objects and fuse this information with a static occupancy map. The whole system is held entirely causal, i.e. at any point in time it only uses information from the past and present.

Fig. 2. Mobile recording platforms used in our experiments. Note that in this paper we only employ image information from a stereo camera pair and do not make use of other sensors such as GPS or LIDAR.

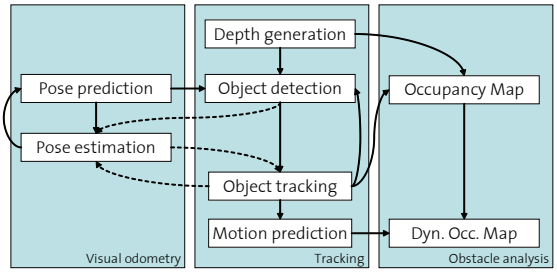
For the basic tracking-by-detection components, we rely on the framework described in [8], [9]. The main contribution of this paper is to propose a set of detailed improvements that considerably boosts tracking performance, both with respect to accuracy and speed, as explained in Section IV. The following subsections briefly review the overall system—see the above references for a full description.

A. Object Detection and Ground Plane Estimation

Instead of directly using the output of a pedestrian detector for the tracking stage, we introduce scene knowledge at an early stage to reduce false positives: a simple scene model is assumed where all objects of interest reside on a common ground plane. Instead of using a fixed ground plane, we allow a set of feasible planes to account for changes in terrain or tilted cameras due to e.g. braking. As a wrong estimate of this plane has far-reaching consequences for all later stages, we try to avoid making hard decisions here and instead model the coupling between object detections and the scene geometry probabilistically using a Bayesian network (see Fig. 3(b)). The network is constructed for each frame and models the dependencies between object hypotheses o_i, object depth d_i, and the ground plane $\boldsymbol{\pi}$ using evidence from the image $\mathcal{I}$, the depth map $\mathcal{D}$, a stereo self-occlusion map $\mathcal{O}$, and the ground plane evidence $\boldsymbol{\pi}_{\mathcal{D}}$ in the depth map. Following standard notation, the plate indicates repetition of the contained parts for the number of objects n.

An object's probability depends on its geometric world

(a)



(b)

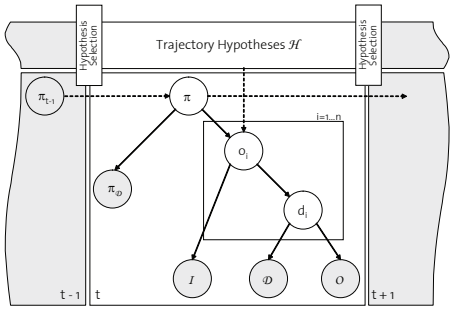


Fig. 3. Flow diagram for our vision system. (see text for details).

position and size, on its correspondence with the depth map, and on the object likelihood estimated by the object detector. The likelihood of each candidate ground plane is modeled by a robust estimator taking into account the uncertainty of the inlier depth points. The ground plane prior and the conditional probability tables are learned from training data.

In addition, we introduce temporal dependencies, indicated by the dashed arrows in Fig. 3(b). For the ground plane, we propagate the posterior from the previous frame, which stabilizes the per-frame information from the depth map. For the detections, we add a spatial prior for object locations that are supported by previously tracked candidate trajectories. As shown in Fig. 3(b), this dependency is not a first-order Markov chain, but reaches many frames into the past, as a consequence of the tracking framework explained in Section III-B.

The advantage of the Bayesian network formulation is that evidence is propagated in both directions: for a largely empty scene the ground plane can be reliably estimated from depth and significantly constrains object detection; in a crowded situation less of the ground is visible, but a large number of detected objects provide information about the ground plane.

B. Tracking and Prediction

Object detections from the previous step are placed into a common world coordinate system using camera positions estimated from visual odometry. The tracking system then uses detected object locations (projected onto the ground plane) as input for a multi-hypotheses tracker, similar to the one described in [17]: the set of object detections from the current and past frames is linked to an over-complete set of trajectory candidates with a holonomic constant-velocity model. Section IV deals with the careful design of the linking step—in this step the search space for the final set of pedestrian trajectories is generated, which obviously makes it important for system performance.

The set of candidate trajectories is then pruned to a minimal consistent explanation using model selection, while simultaneously resolving conflicts between overlapping candidates. In a nutshell, the pruning employs quadratic pseudo-boolean optimization to pick the subset of trajectories with maximal joint probability, given the observed evidence. This probability

- increases as the selected trajectories explain more detections and as they better fit the observed 3D locations and 2D appearance;
- decreases when trajectories would imply that two pedestrians occupy the same space at the same time;
- decreases with the number of required trajectories in order to balance the complexity of the model against its goodness-of-fit and to avoid over-fitting.

For the mathematical details, we refer to [17]. Important features of the method are automatic track initialization (usually, after ≈ 5 detections) and the ability to recover from temporary track loss and occlusion.

The selected trajectories are then used to provide a spatial prior for object detection in the next frame. This prediction has to take place in the world coordinate system, so tracking critically depends on an accurate ego-motion estimate.

C. Visual Odometry

To allow reasoning about object trajectories in world coordinates, the camera position for each frame is estimated using visual odometry. The employed approach builds upon previous work by [8], [19]. Please refer to those publications for details. Compared to standard visual odometry, our system includes scene knowledge obtained from the tracker to mask out image regions not showing the static background. Furthermore it explicitly detects failures by comparing the estimated position to a Kalman filter prediction. In the event of failure, the visual odometry is re-initialized to yield collision-free navigation (at the cost of possible global drift).

D. Static Obstacles

For static obstacles, we construct a stochastic occupancy map with the method from [2]: incoming depth maps are projected onto a polar grid on the ground and are fused with the integrated and transformed map from previous frames. Free space for driving is then computed with dynamic programming. In contrast to the original method, we filter out pedestrians found during tracking for two reasons: firstly, integrating non-static objects can result in smeared occupancy maps. Secondly, we are interested not so much in the *current* positions of the pedestrians as in their *future* locations. These can be predicted more accurately with a specific motion model inferred from the tracker.

IV. Trajectory Generation

Given space-time detections and a motion model, the obvious approach to generate putative trajectories is to continue the candidate trajectories from the previous frame

with an EKF. This method, which we refer to as *extension*, works quite robustly in cases without too much interaction between trajectories. To find newly appearing pedestrians and alternative explanations which contradict the previous candidates, one can additionally start an independent EKF backwards in time for each new detection, which we will call *parallel generation*. This basic approach was also used in our previous work [9], [17].

Here, we describe an ensemble of extensions to the hypothesis generation stage that (i) robustify data assignment, (ii) can actively handle occlusions from by both static as dynamic scene parts, and (iii) reduce the set of candidates and hence the runtime.

A. Clustering detections

When using detections from both cameras of a stereo pair, the same world object often generates one detection in each camera. Keeping two such detections separate increases the number of generated candidate trajectories, which increases the runtime, and can also affect the actual selection process. Hence, we propose to carry out a conservative clustering on detections from both cameras using world and appearance distance. This effectively replaces two measurements—originating from different views of the same object—by a single measurement for the physical 3D object. In our experiments, this reduces the number of candidates to ≈ 50–60% and the tracking time to $\approx 70\%$ of the original figures.

B. Greedy assignment

When generating/extending the candidate trajectories independently of each other, they cannot compete for measurements—the competition is left to the final selection algorithm. In difficult crowded cases, candidates will therefore include wrong measurements of other nearby objects. We have devised a simple strategy to remedy this behaviour: the *clustering* described above ensures that there is only one measurement per object. Hence, only the detection closest to the EKF's predicted location is used to update the state, rather than using all nearby detections weighted by the distance. In order to solve conflicts which arise when a measurement is the closest one for two or more candidate trajectories, the *extension* step is carried out simultaneously for all existing candidates, greedily assigning each detection to the trajectory candidate with the closest prediction. Candidates which do not manage to claim any detection during this process are merely extended through extrapolation. In the same way, only the best candidate at each time step is also chosen during *parallel generation*.

The effect of the competitive hard assignment of detections is twofold. Firstly, it avoids unwanted attraction between candidates and better separates closely interacting pedestrians. (When using soft assignment, the same measurement can influence several nearby trajectory candidates, pulling them closer together). Secondly, the set of candidates tends to be more compact, because each measurement can only support a single candidate in a crowded region, making weak candidates more prone to attrition.

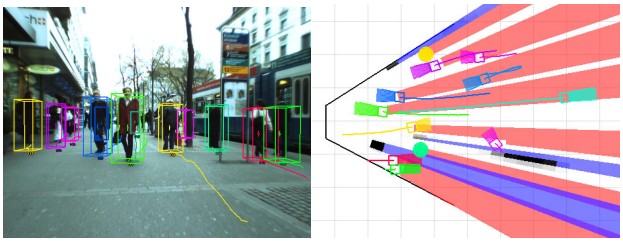


Fig. 4. From the image data (left) we infer occlusion regions (right) due to both static obstacles (black, casting blue umbra) and the previous frame's object predictions (red umbra). This information is used to correctly treat occluded candidate tracks.

C. Occlusion handling

Due to the camera placement on our vehicle, pedestrians frequently occlude each other, but are also often occluded by unmodeled scene objects. We therefore opt to explicitly model occlusion, rather than treat it as yet another case of missing detections. To this end, we generate an occlusion map on the ground plane, again discretized to a polar grid like the occupancy map in Section III-D. An example is shown in Fig. 4. The map contains the regions occluded by both pedestrians and static obstacles. To compute the map, pedestrian locations are estimated by extrapolating the previous tracker state to the current frame, whereas static obstacles are read out of the occupancy map.

As long as a candidate trajectory remains in an occluded region, it is kept alive and its state is extrapolated. Here the uncertainty modeling of the EKF becomes important: continued extrapolation without measurements leads to progressively larger location uncertainties and hence larger search regions for supporting detections. This increases the chances of finding the object once it becomes visible again. The *greedy assignment* described above meanwhile makes sure that such a candidate does not steal detections from less uncertain competitors. As a result, we obtain longer people tracks, which better support path planning [9].

V. Results

In order to evaluate our vision system, we applied it to two test sequences, showing strolls and drives through busy pedestrian zones. The sequences were acquired with the platforms seen in Fig. 2.[1] The first test sequence ("Seq. A"), recorded with platform (a) at considerably worse image contrast, contains 5'193 pedestrian annotations in 999 frames. The second test sequence ("Seq. B") consists of 800 frames and was recorded from a car passing through a crowded city center, where it had to stop a few times to let people pass. We annotated pedestrians in every fourth frame, resulting in 960 annotations for this sequence.

For a quantitative evaluation, we measure bounding box overlap in each frame and plot recall over false positives per image for three stages of our system. The results of this experiment are shown in Table I. We compare the raw

[1]Data and videos are available on `http://www.vision.ee.ethz.ch/~aess/icra2009/`.

detector output, the intermediate output of the Bayesian network, and the final tracking output. As can be seen, discarding detections that are not in accordance with the scene by the Bayesian network almost always increases recall at the same number of false positives. The tracking stage additionally improves the results and in most cases achieves a higher performance than the raw detector. It should be noted, though, that a single-frame comparison is not entirely fair here, since the tracker requires some detections to initialize (losing recall) and reports tracking results through occlusions (losing precision if the occluded persons are not annotated). However, the tracking stage provides the necessary temporal information that makes the entire motion prediction system at all possible. The line "Tracker (orig)" denotes the tracking performance of the system of [9] without the improvements described here. As can be seen, our new method consistently outperforms the original one. When only considering the immediate range up to 15m distance (which is suitable for a speed of 30 km/h in inner-city scenarios), performance is considerably better, as indicated by the second part of Table I.

Recall	Seq. A		Seq. B	
	FP 0.5	FP 1.0	FP 0.5	FP 1.0
Detector	0.57	0.65	0.61	0.67
Bayesian Net	0.65	0.67	0.63	0.66
Tracker (orig) [9]	0.60	0.74	0.52	0.60
Tracker (new)	0.64	0.73	0.55	0.65
Restricted to 15m				
Detector	0.51	0.62	0.76	0.78
Bayesian Net	0.66	0.66	0.74	0.74
Tracker (orig) [9]	0.72	0.74	0.70	0.70
Tracker (new)	0.73	0.77	0.80	0.80

Table I. Detection rates for two test sequences from different platforms. The Bayesian network consistently improves the detector. The tracker with the improvements proposed here also outperforms the original implementation [9]. Performance in the near range approaches a level where it becomes interesting for navigation.

We also compare the effect of using different methods for depth-map generation in Table II. This is of special interest, since nowadays a plethora of stereo algorithms of varying quality and runtime is available. Specifically, we compare the originally used belief-propagation-based stereo algorithm [11] (BP) with a fast GPU-based plane sweeping technique [4] (GPU), and a high-quality global-optimization approach [29]. Example depth maps are shown in Fig. 5. On the one hand, computationally intensive algorithms indeed yield an improvement in both scene analysis and tracking performance, but come at the cost of considerably higher runtime (20ms for GPU vs. 30s for the others). On the other hand, we are using robust statistics on the estimated depth values, hence top-of-the-line stereo matching does not yield noticeable improvements in system performance, despite producing visibly better depth maps.

	No depth		GPU		BP		Zach	
FP	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
BN	-	-	0.63	0.68	0.65	0.67	0.65	0.67
Tr.	0.19	0.29	0.60	0.70	0.64	0.73	0.64	0.73
Restricted to 15m								
BN	-	-	0.66	0.67	0.66	0.66	0.67	0.67
Tr.	0.32	0.47	0.66	0.74	0.73	0.77	0.73	0.78

Table II. Detection rates for Seq. A with different stereo matching methods. Better depth maps improve localization, and hence also tracking, in the near field. Fast GPU methods come at the expense of slightly worse performance. Since we use robust statistics on depth, elaborate stereo algorithms bring little improvement.

Fig. 6 shows results for Seq. A. The bounding boxes are color coded to show the tracked identities; darker boxes indicate objects in occlusion (due to the limited palette, some color labels repeat). Note that both adults and children are identified and tracked correctly even though they differ considerably in their appearance.

Fig. 7 demonstrates the system in an automotive application. Compared to the previous sequences, the viewpoint is quite different, and faster scene changes result in fewer data points for creating trajectories. Still, stable tracking performance can be obtained even for quite distant pedestrians.

VI. Conclusion

In this paper, we have presented a mobile vision system which combines classical geometric localization and mapping with tracking-by-detection of relevant object categories (in our case pedestrians). In this way, not only a geometric map of the world, but also tracks of dynamic objects of interest are available for subsequent path planning and decision making. Since object category detection inherently delivers the semantic information which type of object is tracked, customized motion models can be used for tracking and prediction.

The method relies on closely coupling the modules (detection, tracking, visual odometry, depth estimation). To resolve the complex interactions that occur between pedestrians in urban scenarios, a multi-hypothesis tracking approach is employed. The presented paper has focused on careful design of the hypothesis generation step, which turns out to be an important factor for improving system performance. The resulting system can handle very challenging scenes and delivers accurate predictions for many simultaneously tracked objects.

Acknowledgments. This project has been funded in parts by Toyota Motor Corporation/Toyota Motor Europe and the EU projects DIRAC (IST-027787) and EUROPA (ICT-2008-231888). We thank Nico Cornelis and Christopher Zach for providing GPU implementations of their stereo matching methods.

References

[1] K. O. Arras, O. M. Mozos, and W. Burgard. Using boosted features for the detection of people in 2d range data. In *ICRA*, 2007.

[2] H. Badino, U. Franke, and R. Mester. Free space computation using stochastic occupancy grids and dynamic programming. In *ICCV Workshop on Dynamical Vision (WDV)*, 2007.

[3] M. Betke, E. Haritaoglu, and L. S. Davis. Real-time multiple vehicle tracking from a moving vehicle. *MVA*, 12(2):69–83, 2000.

[4] N. Cornelis and L. van Gool. Real-time connectivity constrained depth map computation using programmable graphics hardware. In *CVPR (1)*, pages 1099–1104. IEEE Computer Society, 2005.

[5] I. J. Cox. A review of statistical data association techniques for motion correspondence. *IJCV*, 10(1):53–66, 1993.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[7] DARPA. DARPA urban challenge rulebook. In *Webpage*, 2008. `http://www.darpa.mil/GRANDCHALLENGE/docs/Urban_Challenge_Rules_102707.pdf`.

[8] A. Ess, B. Leibe, K. Schindler, and L. van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, 2008.

Fig. 5. Example stereo depth maps for a given image. From left to right: GPU-based [4] (20 ms), belief-propagation based [11] (20–30 s), global optimization [29] (30–40 s) algorithm. Parts that are believed to be inaccurate (by a left-right check) are painted black. More advanced algorithms give visually better results, but take more time and are often not necessary (see text).

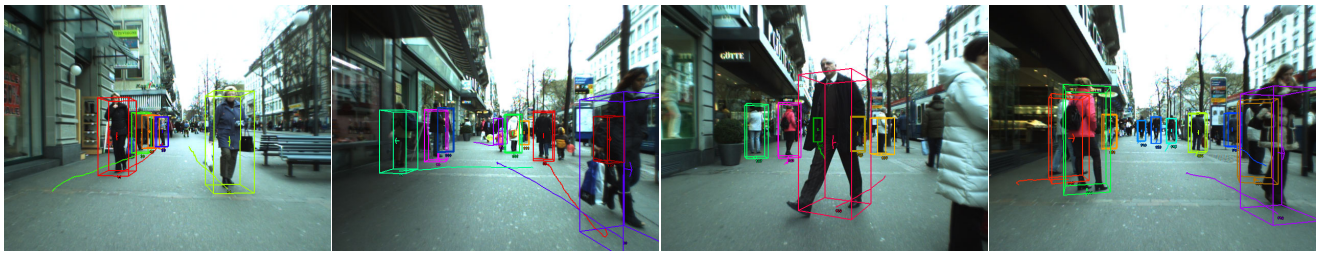


Fig. 6. Example tracking results for Seq. A, recorded from a child stroller.

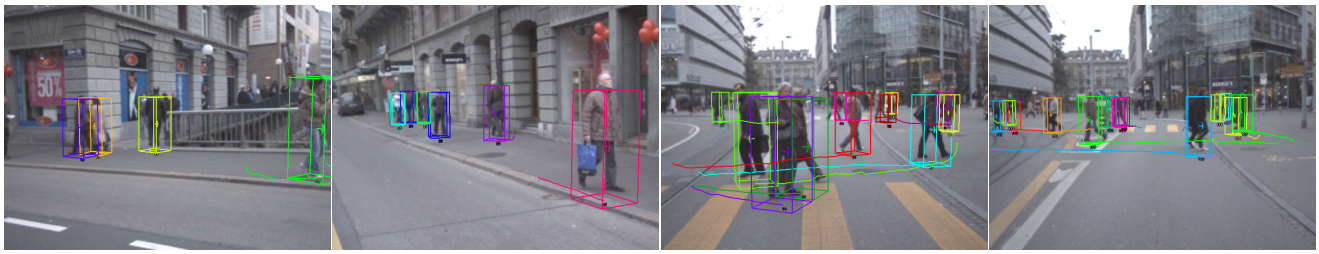


Fig. 7. Example tracking results for Seq. B, recorded from a moving car.

[9] A. Ess, B. Leibe, K. Schindler, and L. van Gool. Moving obstacle detection in highly dynamic scenes. In *ICRA*, 2009.

[10] A. Ess, B. Leibe, and L. van Gool. Depth and appearance for mobile scene analysis. In *ICCV*, 2007.

[11] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70:41–54, 2006. Available from `http://people.cs.uchicago.edu/~pff/bp/`.

[12] T. E. Fortmann, Y. Bar Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE J. Oceanic Engineering*, 8(3):173–184, 1983.

[13] D. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *IJCV*, 73:41–59, 2007.

[14] D. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *ICCV*, pages 87–93, 1999.

[15] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008.

[16] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 77(1-3):259–289, May 2008.

[17] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool. Coupled detection and tracking from static cameras and moving vehicles. *IEEE TPAMI*, 30(10):1683–1698, 2008.

[18] S. Nedevschi, R. Danescu, D. Frentiu, T. Graf, and R. Schmidt. High accuracy stereovision approach for obstacle detection on non-planar roads. In *Proc IEEE Intelligent Engineering Systems*, 2004.

[19] D. Nistér, O. Naroditsky, and J. R. Bergen. Visual odometry. In *CVPR*, 2004.

[20] D. B. Reid. An algorithm for tracking multiple targets. *IEEE T. Automatic Control*, 24(6):843–854, 1979.

[21] M. Scheutz, J. McRaven, and G. Cserey. Fast, reliable, adaptive, bimodal people tracking for indoor environments. In *IROS*, 2004.

[22] D. Schulz, W. Burgard, D. Fox, and A. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *IJRR*, 22(2):99–116, 2003.

[23] A. Shashua, Y. Gdalyahu, and G. Hayun. Pedestrian detection for driving assistance systems: Single-frame classification and system level performance. In *IVS*, 2004.

[24] M. Soga, T. Kato, M. Ohta, and Y. Ninomiya. Pedestrian detection with stereo vision. In *IEEE International Conf. on Data Engineering*, 2005.

[25] L. Spinello, R. Triebel, and R. Siegwart. Multimodal people detection and tracking in crowded scenes. In *Proc. of The AAAI Conference on Artificial Intelligence (Physically Grounded AI Track)*, July 2008.

[26] C.-C. Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *ICRA*, 2003.

[27] C. Wojek and B. Schiele. A dynamic crf model for joint labeling of object and scene classes. In *ECCV*, 2008.

[28] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet part detectors. *IJCV*, 75(2):247–266, 2007.

[29] C. Zach, J.-M. Frahm, and M. Niethammer. Continuous maximal flows and wulff shapes: Application to mrfs. In *CVPR*, 2009. accepted for.

[30] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.

[31] L. Zhao and C. Thorpe. Stereo- and neural network-based pedestrian detection. In *ITS*, 2000.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Visual Person Tracking Using a Cognitive Observation Model

Simone Frintrop[1] Achim Königs[2] Frank Hoeller[2] Dirk Schulz[2]

***Abstract*— In this article we present a cognitive approach to person tracking from a mobile platform. The core of the technique is a biologically inspired observation model that combines several feature channels in an object and background dependent way, in order to optimally separate the object from the background. This observation model can be learned quickly from a single training image and is easily adaptable to different objects. We show how this model can be integrated into a visual object tracker based on the well known Condensation algorithm. Several experiments carried out with a mobile robot in an office environment illustrate the advantage of the approach compared to the Camshift algorithm which relies on fixed features for tracking.**

I. Introduction

An important skill for mobile service robots is the ability to detect and keep track of individual humans in their surrounding. Especially robots that are designed to provide services to individual persons need to be able to distinguish their client from the surrounding environment. During the last decade, several algorithms have been developed for detecting and tracking people with mobile robots using laser range data, vision, or both [1], [2], [3], [4], [5], [6]. Most of these approaches have in common that they rely on a single pre-specified feature domain to compute cues that allow to discriminate the robot's client from other objects in the sensor data. For example, in vision-based approaches color histograms are often employed, or shape information is used. Laser-based approaches mainly rely on range-features extracted from the laser rage scans. However, relying on a single feature leads to the problem that depending on the actual environment conditions, the chosen feature might not be discriminative enough; well known problems for color-histogram based approaches are changing lighting conditions or a cluttered multi-colored background.

In this article we propose to employ a visual attention system for choosing the cues which best distinguish a person from the background depending on the situation the robot currently faces [7]. Based on a cognitive perception model [8], the attention system utilizes a larger set of different simple features to discriminate particular objects from the background. Depending on the environment and the appearance of the object to detect, it automatically determines the suitable cues by computing a weighting of the different features available, such that the resulting mixture discriminates the object from the background best. The attention system being used is able to compute such weight vectors from a single training image. A similarity measure based on these weight vectors is then usually applied for finding the object within images. Instead of searching for the object, we employ the similarity measure within a CONDENSATION-based person tracker [9]. For this purpose, the similarity measure is converted to a likelihood function that is used as the observation model within the particle filter. Using this approach, the robot is able to quickly learn the current appearance of the person it wants to track. This leads to an improved tracking performance, compared to tracking approaches based on single feature cues. In order to evaluate the technique, we implemented an application, where the robot follows a person on its way through our laboratory environment. The experiments show that the approach is able to track the person in varying lighting conditions and backgrounds, and that it is considerably less prone to track loss than, for example, the purely color-based Camshift algorithm.

The remainder of the article is organized as follows. After discussing related work in Section II we explain the cognitive tracking system in Section III. In Section IV we briefly explain how the approach is integrated into a prototypical person following application and we present experimental results. We finally conclude in Section V.

II. Related work

In mobile robotics, person tracking can be performed with different sensors. Several groups have investigated person tracking with laser range finders [1], [2], [3]. These approaches usually only keep track of the motion of people and do not try to distinguish individuals. One approach which distinguishes different motion states in laser data is presented in [4]. Combinations of laser and vision data are presented in [5] and [6]. Both detect the position of people in the laser scan and distinguish between persons based on vision data. Bennewitz et al. [5] base the vision part on color histograms whereas Schulz [6] learns silhouettes of individuals from training data. This however requires a time-consuming learning phase for each new person.

In machine vision, people tracking is a well-studied problem. Two main approaches can be distinguished: *model-based* and *feature-based* methods. In model-based tracking approaches, a model of the object is learned in advance, usually from a large set of training images which show the object from different viewpoints and in different poses [10]. Learning a model of a human is difficult because of the dimensionality of the human body and the variability in human motion. Current approaches include simplified human

[1]Simone Frintrop is with the Institute of Computer Science III, Rheinische Friedrich-Wilhems-Universität, 53117 Bonn, Germany. Contact: frintrop@iai.uni-bonn.de.

[2]All other authors are with the Research Institute for Communication, Information Processing and Ergonomics (FKIE), 53343 Wachtberg, Germany. Contact: {koenigs,hoeller,schulz}@fgan.de

body models, e.g. stick, ellipsoidal, cylindric or skeleton models [11], [12], [13], or shape-from-silhouettes models [14]. While these approaches have reached good performance in laboratory settings with static cameras, they are usually not applicable in real-world environments on a mobile system. They usually do not operate in real-time and often rely on a static, uniform background.

Feature-based tracking approaches on the other hand do not learn a model but track an object based on simple features such as color cues or edges. One approach for feature-based tracking is the Mean Shift algorithm [15], [16] which classifies objects according to a color distribution. Variations of this method are presented in [17], [18]. While most approaches are not especially designed for person tracking, they might be applied in this area as well. One limitation with the above methods is that they operate only on color and are therefore dependent on colored objects.

Visual attention systems are especially suited to automatically determine the features which are relevant for a certain object. These systems are motivated by mechanisms of the human visual system and based on psychological theories on visual attention [8], [19]. During the last decade, many computational attention systems have been built, e.g., [20], [7], and recently, some systems came up that are able to operate in real-time [21], [22], [23]. Important for our application is that the systems compute a feature vector that describes the appearance of a salient region [24], [7].

Applications of visual attention systems range from object recognition to robot localization. However, they have rarely been applied to visual tracking. Some approaches track static regions, such as visual landmarks, from a mobile platform for robot localization [25]. This task is easier than tracking a moving object since the environment of the target remains stable. Another approach aims to track moving objects such as fish in an aquarium [26]. In this case however, the camera is static. The here presented VOCUS tracker is partly based on [27]. We have also applied a simpler approach based on visual attention (but without particle filters) to object tracking [28] and to person tracking [29].

III. The Cognitive Tracking System

The tracking system we present is based on a particle filter approach with a cognitive observation model. It employs the standard Condensation algorithm [9] which maintains a set of weighted particles over time using a recursive procedure based on the following three steps: First, the system draws particles randomly from the particle set of the previous time step, where each particle is drawn with a probability proportional to the associated weight of the particle. Second, the particles are transformed (predicted) according to a motion model. In vision-based tracking this step usually consists of a drift component in combination with random noise. Third, all particles are assigned new weights according to an observation model.

In the following, we first introduce the notation (sec. III-A), second mention how the system is initialized (sec. III-B), and third describe the motion model (sec. III-C). Finally, we specify in detail the observation model as core of the system (sec. III-D).

A. Notation

At each point in time $t \in \{1, .., T\}$, the particle filter recursively computes an estimate of the probability density of the person's location within the image using a set of J particles $\mathbf{\Phi}_t = \{\phi_t^1, ...\phi_t^J\}$ with

$$\phi_t^j = (\mathbf{s}_t^j, \pi_t^j, \mathbf{w}_t^j), \quad j \in \{1, ..., J\}.$$

Here, $\mathbf{s}_t^j = (x, y, v_x, v_y, w, h)$ is the state vector that specifies the particle's region with center (x, y), width w and height h – in the following, the region is also denoted as $\mathbf{R}_t^j = (x, y, w, h)$. The v_x and v_y components specify the current velocity of the particle in the x and y directions. Each particle additionally has a weight π_t^j determining the relevance of the particle with respect to the target, and a feature vector $\mathbf{w}_t^j$ that describes the appearance of the particle's region.

B. Initialization

In order to start the tracking process, the initial target region $\mathbf{R}^* = (x^*, y^*, w^*, h^*)$ has to be specified in the first frame. This can either be carried out manually or automatically using a separate detection module. Based on the initial target region $\mathbf{R}^*$, a feature weight vector $\mathbf{w}^*$ is computed that describes the appearance of the person. The initial particle set

$$\mathbf{\Phi}_0 = \{(\mathbf{s}_0^j, \pi_0^j, \mathbf{w}_0^j) \,|\, j = 1, ..., J\}. \tag{1}$$

is generated by randomly distributing the initial target location around the region's center (x^*, y^*). The velocity components v_x and v_y are initially set to 0 and the region dimensions of each particle are initialized with the dimensions of $\mathbf{R}^*$. The particle weights π_0^j are set to $1/J$.

C. Motion model

Currently, the object's motion is modeled by a simple first order autoregressive process in which the state $\mathbf{s}_t^j$ of a particle depends only on the state of the particle in the previous frame:

$$\mathbf{s}_t^j = \mathbf{M} \cdot \mathbf{s}_{t-1}^j + \mathbf{Q}.$$

Here, $\mathbf{M}$ is a state transition matrix of a constant velocity model and $\mathbf{Q}$ is a random variable that denotes some white Gaussian noise. This enables a flexible adaption of position and size of the particle region as well as of its velocity. Thus the system is able to quickly react to velocity changes of the object.

D. Observation model

In visual tracking, the choice of the observation model is the most crucial step since it decides which particles will survive. It therefore has the strongest influence on the estimated position of the target. Here, we use a cognitive observation model which favors the most discriminative features in the current setting based on concepts of human visual perception. It determines the feature description for the target and for each particle, enabling the comparison and weighting of particles.

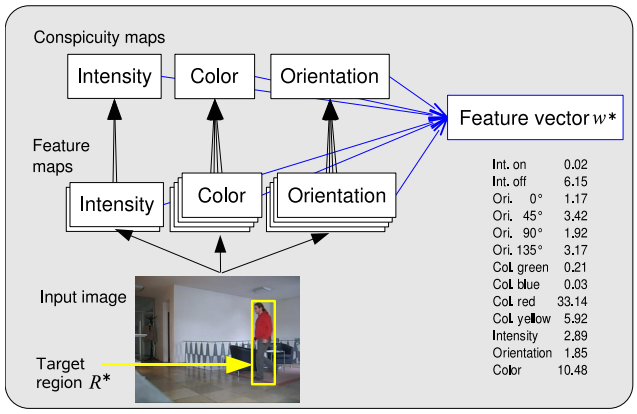


Fig. 1. Initialization: the attention system VOCUS learns the target appearance by computing feature and conspicuity maps for the image and determining a feature vector $\mathbf{w}^*$ for the manually provided search region $\mathbf{R}^*$ (yellow rectangle).

1) Computation of the feature vector: The feature vector is computed based on a cognitive perception model which computes the saliency of a region based on concepts of the human visual system (cf. Fig. 1). This *computational attention system* is called *VOCUS* and was originally built to simulate human eye movements [7]. It computes feature contrasts for different scales and feature types and assigns a saliency value to each image region. Additionally, a feature vector is computed for each salient region that determines the contribution of the different feature channels to the region.

In this paper, we use the system in a slightly different manner than the usual case: we do not determine the most salient regions in an image, but the feature saliency of predefined regions, the particle regions. However, the computation of the feature maps is the same.

The feature computations are performed on 3 different scales using image pyramids. The feature intensity is computed by *center-surround mechanisms* (similar to DoG filters); on-off (bright on dark) and off-on (dark on bright) contrasts are determined separately. After summing up the scales, this yields 2 intensity maps. Similarly, 4 color maps (green, blue, red, yellow) and 4 orientation maps $(0\,^\circ, 45\,^\circ, 90\,^\circ, 135\,^\circ)$ are computed. The color maps compute color contrasts based on the Lab color space (CIELAB), since this is known to approximate human perception well. To achieve real-time performance, the intensity and color maps are computed using integral images [30]. These provide an efficient way to determine the average value of a rectangular region of arbitrary size in constant time (4 operations per region), after once creating the integral image in linear time. For the orientation maps, Gabor filters highlight the gradients with a certain orientation (details in [7]).

Before the features are fused, they are weighted according to their *uniqueness*, i.e. a feature which occurs seldomly in a scene is assigned a higher saliency than a frequently occurring feature. This is a mechanism which enables humans to instantly detect outliers like a black sheep in a white herd. The uniqueness $\mathcal{W}$ of map X is computed as $\mathcal{W}(X) = X/\sqrt{m}$, where m is the number of local maxima that exceed a threshold. Here, '/' stands for the pixel-wise division of an image with a scalar. The weighted maps are summed up to 3 *conspicuity maps* for intensity, orientation, and color. In the following, we denote the 10 feature and 3 conspicuity maps for image I_t as $F_i(I_t), i \in \{1, .., 13\}$. In the original VOCUS system, the conspicuity maps are weighted again and fused into a saliency map. However, this map is not required in our approach.

For an arbitrary region in the image, a feature vector can be computed which describes the appearance of the region with respect to its surrounding. In the original system, feature vectors are computed for the most salient regions in a saliency map. Here, we compute a vector for each particle region. The feature vector $\mathbf{w} = (w_1, ..., w_{13})$ for a region $\mathbf{R}$ is computed as follows. For each map $F_i(I)$, the ratio of the mean saliency in the target region $\mathbf{R}$ and in the background $I \backslash R$ is determined as:

$$w_i = \frac{\text{mean}(\mathbf{R})}{\text{mean}(I \backslash \mathbf{R})}, \qquad i \in \{1, .., 13\}. \tag{2}$$

This computation does not only consider which features are the strongest in the target region, it also regards which features separate the region best from the rest of the image.

Since this computation involves computing the average value of a particle region of arbitrary size for a usually large collection of particles and for 13 feature maps, the process can be time consuming. To maintain real-time performance, the computations are also performed with integral images. This increased the average processing speed of VOCUS considerably from 10 Hz to 40 Hz. The result of the computations in this section is a feature vector $\mathbf{w}_t^j$ for each particle.

2) Weighting of the particles: The feature vector $\mathbf{w}_t^j$ of a particle ϕ_t^j is now used to determine the similarity of the particle region $\mathbf{R}_t^j$ with the initial target region $\mathbf{R}^*$. As similarity measure we use the Tanimoto-coefficient

$$T(\mathbf{w}^*, \mathbf{w}_t^j) = \frac{\mathbf{w}^* \cdot \mathbf{w}_t^j}{||\mathbf{w}^*||^2 + ||\mathbf{w}_t^j||^2 - \mathbf{w}^* \cdot \mathbf{w}_t^j}.$$

The Tanimoto coefficient produces values in the interval $[0, 1]$, the higher the value the higher the similarity. If the two vectors are identical, the coefficient is 1. Compared to Euclidean distance, it turned out that the Tanimoto coefficient is better suited to distinguish between true and false matches [28]. Based on the Tanimoto coefficient the weight of a particle is computed as

$$\pi_t^j = c \cdot e^{\lambda \cdot T(\mathbf{w}^*, \mathbf{w}_t^j)}.$$

This function prioritizes particles which are very similar to the target vector $\mathbf{w}^*$ by assigning an especially high weight. A value of $\lambda = 14$ has shown to be useful in our experiments. The parameter c is a normalization factor which is chosen so that $\sum_{j=1}^{J} \pi_t^j = 1$.

3) Determining the target state: From the weighted particle set, the current target state, including target position and size, can be estimated by

$$\mathbf{x}_t = \sum_{j=1}^{J} \pi_t^j \cdot \mathbf{s}_t^j.$$

IV. Experiments and Results

The experiments were carried out using a RWI B21 robot equipped with a simple USB web camera mounted on a pan-tilt unit (see Fig. 2, left). The camera captures 15 frames/sec, with a resolution of 320×240. The complete software runs on a 2GHz dual core PC onboard the robot. For the experiments, the tracking application was implemented within the software framework RoSe developed at FKIE [31]. This framework consists of roughly 30 modules which exchange information over a UDP-based communication infrastructure. The RoSe framework is specifically designed to allow for the easy assembly of multi-robot applications, which extensively use wireless ad-hoc communication. However, for the tracking experiments, we only required two modules on a single robot:

1) A visual tracking module, which captures the images and employs the tracking algorithm (VOCUS or Camshift) for tracking a single person within the image. Based on the pixel location of the person computed by the vision-based tracker, the module computes a heading direction relative to the robot, steers the pantilt unit in order to center the person within the image and commands the robot to follow the person. This is achieved by continuously instructing the reactive collision avoidance component of the robot to drive to goal locations behind the moving person.

2) The collision avoidance component of the robot. It is specifically designed for the task of following moving persons based on motion tracking information. It does so by applying an expansive spaces tree algorithm, which carries out a search for admissible paths in time and space, based on information about static obstacles provided by a laser range scanner, as well as motion information, i.e. position and velocity vectors of moving obstacles and the person being followed, provided by the external tracking component [32].

We performed two series of experiments with this system within the hallways of the FKIE building – an outline of the floorplan is shown in Figure 2, right. The first series of experiments illustrates the benefit of the VOCUS tracker for the actual people tracking task; the second series evaluates the robustness of the image-based tracker using the VOCUS system, compared to simpler feature-based techniques like Camshift.

Both series were performed during normal working hours with people walking around. The lighting conditions varied strongly during the experiments: some areas show natural daylight (see Fig. 2, right), others artificial light. In some parts, the light was switched off resulting in rather poorly illuminated areas. These conditions resulted in several images with very poor quality (cf. Fig. 5). Furthermore, after quick camera movements the camera was out of focus for some frames and capturing images was sometimes delayed resulting in large changes between consecutive frames.

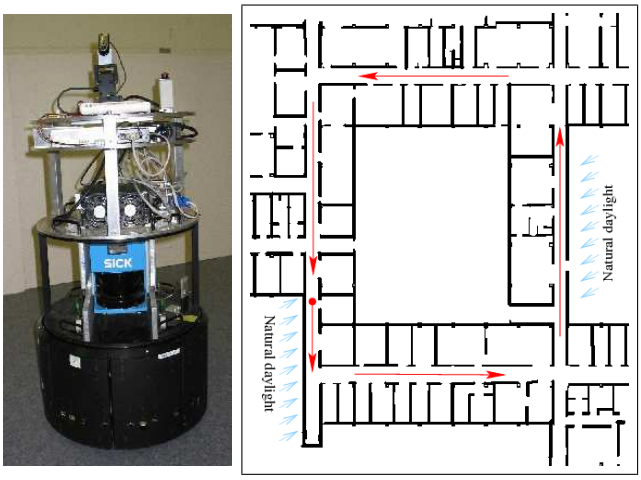


Fig. 2. Left: the RWI B21 robot *Blücher* used for the experiment. The images were taken using the small pantilt mounted webcam on top of the robot. Right: An outline of the environment used for the experiments. The robot tracked the person through the indicated round trip tour (red arrows) and encountered different lighting conditions on its path. The start and end location is marked with a small red circle.

A. Autonomous Person Tracking

In the first series of experiments, the robot followed a person autonomously through the hallways (red arrows in Fig. 2, right). We performed 4 runs with 2 different persons and 3 different kinds of clothing. Initialization of the target was done with user interaction by marking the person in the first frame. After that, the robot estimated the position of the person in each frame and drove autonomously into the direction of the estimated target state. The camera was controlled to center the target in the frame.

To evaluate the tracking, we counted the number of *detections* manually. A detection occurs if the center of the target state was on the person[1]. The results are shown in Tab. I. Images in which the target was not visible were not considered for the detection rate but are shown in Tab. I. In three of the runs, the detection rate was about 80%. In the 2nd run, the detection rate is considerably lower. The reason was that the center of gravity of the particle cloud was in many frames next to the target (cf. Fig. 5, right).

B. Comparison with Camshift

Most similar to the here presented VOCUS tracking are color-based trackers such as trackers based on the MeanShift algorithm [16]. One well-known modification is the Camshift algorithm [17] that is able to adapt dynamically to the target it is tracking[2]. It is a statistical method of finding the peak of a probability distribution, usually obtained with a color histogram. In the 2nd series of experiments, we used Camshift as benchmarking system for our approach.

[1]This is an approximation which is actually too optimistic since the region might include a part of the background and still have its center on the region. It is reasonable here anyway since the center is the point the robot uses as target direction.

[2]Camshift is publically available from the OpenCV library: http://opencvlibrary.sourceforge.net/

	# Frames	detections [%]	# frames without target
1	1918	81	1
2	1486	58	37
3	1202	87	8
4	559	80	79
Average	1291	77	31

TABLE I

VOCUS TRACKING IN ONLINE EXPERIMENTS

	# Frames	correct detections [%]			
		VOCUS	Cam (HSV)	Cam (RG)	Cam (Lab)
1	1477	79	51	88	39
2	1158	96	53	62	54
3	1596	65	5	28	50
4	1392	54	13	1	10
5	1519	71	46	47	46
	Average	73	33	45	40

TABLE II

COMPARISON OF VOCUS AND CAMSHIFT TRACKING. CAMSHIFT IS INVESTIGATED FOR DIFFERENT COLOR SPACES (HSV, RG, LAB). THE ROWS SHOW THE RESULTS FOR THE 5 PERSONS IN FIG. 3.

Although the Camshift algorithm has shown good results in other applications, it is only of limited use for a flexible online tracker. Usually, it is necessary to adapt the parameters of the algorithm for each object to obtain good results. While this may be acceptable for some applications like face tracking in which each face has a similar hue value, it is difficult for targets like persons which vary strongly in appearance due to different clothing. Since our VOCUS tracker is applicable to different objects without adapting parameters, we used the Camshift algorithm with the standard parameter set of the OpenCV implementation for all test sequences to make the approaches comparable. The Camshift usually uses the HSV color space. Additionally to this implementation, we used it with two other color spaces: RG chromaticity space and Lab space.

To be able to compare the approaches on the same data, several image sequences were acquired by teleoperating the robot and processed offline. We tested 5 different runs, each covering one circle in our environment (approx. 160 m per run). Each run was performed with a different person as target, with different clothing (cf. Fig. 3). The runs consisted of 1000–1600 frames each. Tab. III shows the initial feature vectors w^* that were learned from the frames in Fig. 3. The results are displayed in Tab. II. All approaches clearly have difficulties with the challenging conditions, mainly resulting from the strong changes in illumination. In most cases, the VOCUS tracker performed best, with an average detection rate of 73%. The Camshift approaches perform considerably worse (33, 45 and 40%). All approaches had most difficulties with person 4. This is partly due to the white shirt which is similar to the color of the walls. For all approaches it turned out that the clothing of the person made a strong difference in performance: the larger the contrast and difference to the background, the easier the tracking.

Feature	1)	2)	3)	4)	5)
intensity on-off	0.14	0.14	0.19	0.62	0.44
intensity off-on	2.48	4.06	4.36	1.95	4.30
orientation 0°	1.2	1.58	2.00	2.56	1.86
orientation 45°	1.66	2.35	1.25	1.75	1.69
orientation 90°	1.08	1.90	1.40	1.65	1.81
orientation 135°	1.27	1.59	1.21	1.52	2.07
color green	0.35	2.62	0.90	0.75	1.10
color blue	5.55	2.68	3.24	3.02	6.02
color red	1.53	31.40	3.41	1.67	6.88
color yellow	1.48	3.71	0.80	1.54	1.41
intensity	1.26	1.86	2.18	1.14	2.85
orientation	1.21	1.81	1.38	1.80	1.84
color	1.93	10.44	1.60	1.81	2.61

TABLE III

FEATURE VECTORS $\mathbf{w}^*$ THAT ARE LEARNED FOR THE TARGET PERSONS IN FIG. 3 (THE COLUMNS CORRESPOND TO THE IMAGES).

V. CONCLUSION

In this paper, we have presented a cognitive approach for person tracking from a mobile platform. The appearance of an object of interest is learned from an initially provided target region and the resulting target feature vector is used to search for the target in subsequent frames. Advantages of the system are that it uses several feature channels in parallel, that it considers not only the target appearance but also the appearance of the background, and that it is quickly adaptable to a new target without a time-consuming learning phase. Furthermore, it is capable to work on a mobile platform since it works in real-time, does not rely on a static background, and copes with varying illumination conditions.

We obtained promising first results in different settings. However, the task of person tracking in natural conditions is very challenging and we just scratched the surface of the problem. Although our image sequences are more difficult than most of the data used in research groups for similar tasks, they show by far not the most difficult settings. Persons with similar clothing to the background, bright sunlight, and crowded environments in which the person is temporarily occluded would make the problem worse. We will investigate such settings in future work.

There are several ways the current approach could be improved. Currently, we learn target appearance from a single frame. While this works reasonably well in many cases, it will fail if the environment changes strongly. Learning target appearance online from several frames and adapting the feature vector to new conditions is subject to future work. We also plan to integrate additional features, e.g. motion cues, into the tracking system.

REFERENCES

[1] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *Int'l Conference on Robotics and Automation (ICRA)*, 2002.

[2] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, "People tracking with mobile robots using sample-based joint probabilistic data association filters," International Journal of Robotics Research, 22(2), 2003.

Fig. 3. Initial frames and initial target regions $\mathbf{R}^*$ (yellow rectangles) used to learn the appearance of the 5 persons.

Fig. 4. Successful tracking. Green points: particles that matched to target; cyan points: particles that didn't match. Rectangles show estimated target state.

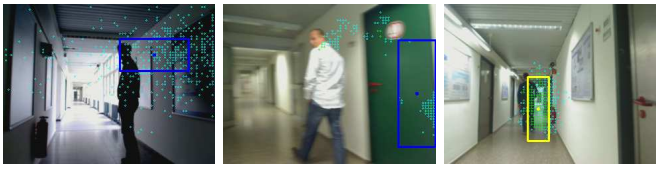


Fig. 5. Failed tracking. The points denote the particles (green points: particles that matched to the target, cyan points: particles that did not match). Rectangles denote the estimated target state (blue: less than 50% of particles match, otherwise yellow).

[3] K. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *Proc. of IEEE Int'l Conference on Robotics and Automation (ICRA'08)*, Pasadena, USA, 2008.

[4] G. Taylor and L. Kleeman, "A multiple hypothesis walking person tracker with switched dynamic model," in *Australasian Conference on Robotics and Automation (ACRA)*, 2004.

[5] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 31–48, 2005.

[6] D. Schulz, "A probabilistic exemplar approach to combine laser and vision for person tracking," In Proc. of the International Conference on Robotics Science and Systems (RSS 2006), 2006.

[7] S. Frintrop, "VOCUS: a visual attention system for object detection and goal-directed search," Ph.D. dissertation, University of Bonn, Germany, July 2005, published 2006 in Lecture Notes in Artificial Intelligence (LNAI), Vol. 3899, Springer Verlag.

[8] A. M. Treisman and G. Gelade, "A feature integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97–136, 1980.

[9] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *Int. J. of Computer Vision (IJCV)*, vol. 29, no. 1, pp. 5–28, 1998.

[10] K. Rohr, "Towards model-based recognition of human movements in image sequences," *CVGIP – Image Understanding*, vol. 59, no. 1, pp. 94–115, 1994.

[11] C. Breglera, J. Malik, and K. Pullen, "Twist based acquisition and tracking of animal and human kinematics," *International Journal of Computer Vision (IJCV)*, 2004.

[12] R. Urtasun, D. J. Fleet, and P. Fua, "Temporal motion models for monocular and multiview 3d human body tracking," *Computer Vision and Image Understanding (CVIU), special issue Modeling People*, 2006.

[13] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman, "Human body model acquisition and tracking using voxel data," *International Journal of Computer Vision*, vol. 53, no. 3, pp. 199–223, 2003.

[14] K. Cheung, S. Baker, and T. Kanade, "Shape-from-silhouette across time part II: Applications to human modeling and markerless motion," *International Journal of Computer Vision (IJCV)*, 2005.

[15] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 5, 2002.

[16] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," Proc. Conf. Computer Vision and Pattern Recognition (CVPR), vol. 2, 2000.

[17] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, 1998.

[18] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," Proceedings of the 7th European Conference on Computer Vision (ECCV) London, UK, Springer-Verlag, 2002.

[19] J. M. Wolfe, "Guided search 2.0: A revised model of visual search," *Psychonomic Bulletin and Review*, vol. 1, no. 2, pp. 202–238, 1994.

[20] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[21] S. Frintrop, M. Klodt, and E. Rome, "A real-time visual attention system using integral images," in *Proc. of the 5th Int'l Conf. on Computer Vision Systems (ICVS)*, Bielefeld, Germany, March 2007.

[22] S. May, M. Klodt, and E. Rome, "GPU-accelerated Affordance Cueing based on Visual Attention," in *Proc. of Int'l Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2007, pp. 3385–3390.

[23] M. Björkman and J.-O. Eklundh, "Vision in the real world: Finding, attending and recognizing objects," *Int'l Journal of Imaging Systems and Technology*, vol. 16, no. 2, pp. 189–208, 2007.

[24] V. Navalpakkam, J. Rebesco, and L. Itti, "Modeling the influence of task on attention," *Vision Research*, vol. 45, no. 2, pp. 205–231, 2005.

[25] S. Frintrop and P. Jensfelt, "Active gaze control for attentional visual SLAM," in *Proc. of the IEEE Int'l Conf. on Robotics and Automation (ICRA'08)*, 2008.

[26] M. Veyret and E. Maisel, "Attention-based target tracking for an augmented reality application," Int'l Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision, 2006.

[27] M. Kessel, "Aufmerksamkeitsbasiertes Objekt-Tracking," Master's thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 2008.

[28] S. Frintrop and M. Kessel, "Most salient region tracking," in *Proc. of the IEEE Int'l Conf. on Robotics and Automation (ICRA'09)*, Kobe, Japan, 2009.

[29] ——, "Cognitive data association for visual person tracking," in *Proc. of the 1st IEEE Workshop on Human Detection from Mobile Platforms (HDMP) at ICRA*, Pasadena, CA, May 2008.

[30] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision (IJCV)*, vol. 57, no. 2, pp. 137–154, May 2004.

[31] A. Tiderko, T. Bachran, F. Hoeller, D. Schulz, and S. F. E., "RoSe – a framework for multicast communication via unreliable networks in multi-robot systems," *Robotics and Autonomous Systems*, vol. 56, no. 12, pp. 1017–1026, 2008.

[32] F. Hoeller, D. Schulz, M. Moors, and F. E. Schneider, "Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps," in *Proc. of the International Conference on Robots and Systems (IROS)*. IEEE, 2007, pp. 1260–1265.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Multi-model Hypothesis Group Tracking and Group Size Estimation

Boris Lau Kai O. Arras Wolfram Burgard

***Abstract*— People in densely populated environments typically form groups that split and merge. In this paper we track groups of people so as to reflect this formation process and gain efficiency in situations where maintaining the state of individual people would be intractable. We pose the group tracking problem as a recursive multi-hypothesis model selection problem in which we hypothesize over both, the partitioning of tracks into groups (models) and the association of observations to tracks (assignments). Model hypotheses that include split, merge, and continuation events are first generated in a data-driven manner and then validated by means of the assignment probabilities conditioned on the respective model. Observations are found by clustering points from a laser range finder given a background model and associated to existing group tracks using the minimum average Hausdorff distance. We further propose a method to estimate the number of people in groups based on the number of human-sized clusters. Experiments with a stationary and a moving platform show that, in populated environments, tracking groups is clearly more efficient than tracking people separately. The results also show a high accuracy in the estimation of group sizes. Our system runs in real-time on a typical desktop computer.**

I. Introduction

The ability of robots to keep track of people in their surrounding is fundamental for a wide range of applications including personal and service robots, intelligent cars, or surveillance. People are social beings and as such they form groups, interact with each other, merge to larger groups or separate from groups. Tracking individual people during these formation processes can be hard due to the high chance of occlusion and the large extent of data association ambiguity. This causes the space of possible associations to become huge and the number of assignment histories to quickly become intractable. Further, for many applications, knowledge about groups can be sufficient as the task does not require to know the state of every person. In such situations, tracking groups that consist of multiple people is more efficient and furthermore contains semantic information about activities of the people.

This paper focuses on group tracking in populated environments with the goal to track a large number of people in real-time. The approach attempts to maintain the state of groups of people over time, considering possible splits and merges as illustrated in Fig. 1. For our experiments we use a mobile robot equipped with a laser range finder, but our method should be applicable to data from other sensors as well.

All authors are with the University of Freiburg, Germany, Department of Computer Science {lau,arras,burgard}@informatik.uni-freiburg.de.

This work was partly funded by the European Commusion under contract number FP6-IST-045388

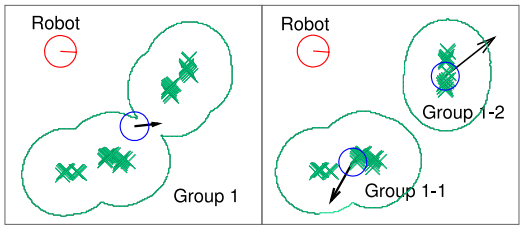


Fig. 1. Tracking groups of people with a mobile robot. Groups are shown by their position (blue), velocity (black), the associated laser points (green) and a contour for visualization. In the two frames, a group of four people splits up into two groups with two people each.

In most related work on laser-based people tracking, tracks correspond to individual people [1], [2], [3], [4], [5]. In Taylor *et al.* [6] and Arras *et al.* [7], tracks represent the state of legs which are fused to people tracks in a later stage. Khan *et al.* [8] proposed an MCMC-based tracker that is able to deal with non-unique assignments, i.e., measurements that originate from multiple tracks, and multiple measurements that originate from the same track. Actual tracking of groups using laser range data was, to our knowledge, first addressed by Mucientes *et al.* [9]. Most research in group tracking was carried out in the vision community [10], [11], [12]. Gennari *et al.* [11] and Bose *et al.* [12] both address the problem of target fragmentation (splits) and grouping (merges). They do not integrate data association decisions over time – a key property of the Multi-Hypothesis Tracking (MHT) approach, initially presented by Reid [13] and later extended by Cox *et al.* [14]. The approach belongs to the most general data association techniques as it produces joint compatible assignments, integrates them over time, and is able to deal with track creation, confirmation, occlusion, and deletion.

The works closest to this paper are Mucientes *et al.* [9] and Joo *et al.* [15]. Both address the problem of group tracking using an MHT approach. Mucientes *et al.* employ two separate MHTs, one for the regular association problem between observations and tracks and a second stage MHT that hypothesizes over group merges. However, people tracks are not replaced by group tracks, hence there is no gain in efficiency. The main benefit of that approach is the semantical extra information about formation of groups.

Joo *et al.* [15] present a visual group tracker using a single MHT to create hypotheses of group splits and merges and observation-to-track assignments. They develop an interesting variant of Murty's algorithm [16] that generates the k-best *non-unique* assignments which enables them to make multiple assignments between observations and tracks, thereby describing target splits and merges. However, the method only produces an approximation of the optimal k-

best solutions since the posterior hypothesis probabilities depend on the number of splits, which, at the time when the k-best assignments are being generated, is unknown. In our approach, the split, merge and continuation events are given by the model *before* computing the assignment probabilities, and therefore, our k-best solutions are optimal.

In this paper we propose a tracking system for groups of people using an extended Multi-Hypothesis Tracking (MHT) approach to hypothesize over both, the group formation process (models) and the association of observations to tracks (assignments). Each model, defined to be a particular partitioning of tracks into groups, creates a new tree branch with its own assignment problem. As a further contribution we propose a group representation that includes the shape of the group and we show how this representation is updated in each step of the tracking cycle. This extends previous approaches where groups are assumed to have Gaussian shapes only [11], [9]. We also present an estimation method to determine the number of people in groups which extends the approach presented by the same authors in [17]. Finally, we use the psychologically motivated *proxemics* theory introduced by Hall [18] for the definition of a group. The theory relates social relation and body spacing during social interaction.

It is structured as follows: the following section describes the extraction of groups of people from laser range data. Section III introduces the definition of groups. Section V briefly describes the cycle of our Kalman filter-based tracker. Section VI explains the data-driven generation of models and how their probabilities are computed. Whereas Section VII presents the multi-model MHT formulation and derives expressions for the hypothesis probabilities, Section VIII describes the experimental results.

II. Group Detection in Range Data

Detecting people in range data has been approached with motion and shape features [1], [2], [3], [4], [5], [9] as well as with a learned classifier using boosted features [19]. However, these recognition systems were designed (or trained) to extract single people. In the case of densely populated environments, groups of people typically produce large blobs in which individuals are hard to recognize. We therefore pursue the approach of background subtraction and clustering. Given a previously learned model (a map of the environment for mobile platforms), the background is subtracted from the scans and the remaining points are passed to the clustering algorithm. This approach is also able to detect standing people as opposed to [9] which relies on motion features.

Concretely, a laser scanner generates measurements $\mathbf{z}_i = (\phi_i, \rho_i)^T,\ \ i \in \{1, \dots N_z\}$, with ϕ_i being the bearing and ρ_i the range value. The measurements $\mathbf{z}_i$ are transformed into Cartesian coordinates and grouped using *single linkage clustering* [20] with a distance threshold d_P. The outcome is a set of clusters $\mathcal{Z}_i$ making up the current observation $Z(k) = \{\mathcal{Z}_i \,|\, i = 1, \dots, N_\mathcal{Z}\}$. Each cluster $\mathcal{Z}_i$ is a complete set of measurements $\mathbf{z}_i$ that fulfills the cluster condition, i.e., two clusters are joined if the distance between their closest points is smaller than d_P. A similar concept, using a connected components formulation, has been used by Gennari and Hager [11]. The clusters then contain range readings that can correspond to single legs, individual people, or groups of people, depending on the cluster distance d_P.

III. Group Definition

This section defines the concept of a group and derives probabilities of group-to-observation and group-to-group assignments.

What makes a collection of people a *group* is a highly complex question in general which involves difficult-to-measure social relations among subjects. A concept related to this question is the proxemics theory introduced by Hall [18] who found from a series of psychological experiments that social relations among people are reliably correlated with physical distance during interaction. This finding allows us to infer group affiliations by means of body spacing information available in the range data. The distance d_P thereby becomes a threshold with a meaning in the context of group formation.

A. Representation of Groups

Concretely, we represent a group as a tuple $G = \langle \mathbf{x}, C, \mathcal{P} \rangle$ with $\mathbf{x}$ as the track state, C the state covariance matrix and $\mathcal{P}$ the set of contour points that belong to G. The track state is composed of the position $(x,\, y)$ and the velocities $(\dot{x},\, \dot{y})$ to form the state vector $\mathbf{x} = (x,\, y,\, \dot{x},\, \dot{y})^T$ of the group.

The points $\mathbf{x}_{\mathcal{P}_i} \in \mathcal{P}$ are an approximation of the group's current shape or spatial extension. Shape information will be used for data association under the assumption of *instantaneous rigidity*. That is, a group is assumed to be a rigid object over the duration of a time step Δt, and consequently, all points in $\mathcal{P}$ move coherently with the estimated group state $\mathbf{x}$. The points $\mathbf{x}_{\mathcal{P}_i}$ are represented relative to the state $\mathbf{x}$.

B. Group-to-Observation Assignment Probability

For data association we need to calculate the probability that an observed cluster $\mathcal{Z}_i$ belongs to a predicted group $G_j = \langle\, \mathbf{x}_j(k+1|k),\, C_j(k+1|k),\, \mathcal{P}_j \,\rangle$. A distance function $d(\mathcal{Z}_i, G_j)$ is sought that, unlike the Mahalanobis distance used by Mucientes *et al.* [9], accounts for the shape of the observation cluster $\mathcal{Z}_i$ and the group's contour $\mathcal{P}_j$, rather than just for their centroids. To this end, we use a variant of the Hausdorff distance. As the regular Hausdorff distance is the *longest* distance between points on two contours, it tends to be sensitive to large variations in depth that can occur in range data. This motivates the use of the minimum average Hausdorff distance [21] that computes the minimum of the averaged distances between contour points,

$$d_{\text{HD}}(\mathcal{Z}_i, G_j) = \min\left\{d(\mathcal{Z}_i, \mathcal{P}_j), d(\mathcal{P}_j, \mathcal{Z}_i)\right\} \qquad (1)$$

where $d(\mathcal{Z}_i, \mathcal{P}_j)$ is the directed average Hausdorff distance. Since we deal with uncertain entities, $d(\mathcal{Z}_i, \mathcal{P}_j)$ is calculated using the squared Mahalanobis distance $d^2 = \nu^T\, S^{-1}\, \nu$,

$$d(\mathcal{Z}_i, \mathcal{P}_j) = \frac{1}{|\mathcal{Z}_i|} \sum_{\mathbf{z}_i \in\, \mathcal{Z}_i} \min_{\mathbf{x}_{\mathcal{P}_j} \in\, \mathcal{P}_j} \left\{d^2(\nu_{ij},\, S_{ij})\right\}, \qquad (2)$$

with ν_{ij}, S_{ij} being the innovation and innovation covariance between a point $\mathbf{z}_i \in \mathcal{Z}_i$ and contour point $\mathbf{x}_{\mathcal{P}_j}$ of the predicted set $\mathcal{P}_j$ transformed into the sensor frame,

$$\nu_{ij} = \mathbf{z}_i - (H\mathbf{x}_j(k+1|k) + \mathbf{x}_{\mathcal{P}_j}) \quad (3)$$

$$S_{ij} = H\,C_j(k+1|k)\,H^T + R_i \quad (4)$$

where $H = \left(\begin{smallmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{smallmatrix}\right)$ is the measurement Jacobian and R_j the 2×2 observation covariance whose entries reflect the noise in the measurement process of the range finder.

The probability that cluster $\mathcal{Z}_i$ originates from G_j is finally

$$\mathcal{N}_i := \mathcal{N}(d^2_{\text{HD}}(\mathcal{Z}_i, G_j), S_{ij}) \quad (5)$$

where $\mathcal{N}(\mu, \Sigma)$ denotes the normal distribution.

C. Group-to-Group Assignment Probability

To determine the probability that two groups G_i and G_j merge, we compute the distance between their closest contour points in a Mahalanobis sense. In doing so, we have to account for the clustering distance d_P that states identity of G_i, G_j as soon as their contours come closer than d_P. Let $\Delta\mathbf{x}_{\mathcal{P}_{ij}} = \mathbf{x}_{\mathcal{P}_i} - \mathbf{x}_{\mathcal{P}_j}$ be the vector difference of two contour points of G_i and G_j respectively, we then subtract d_P from $\Delta\mathbf{x}_{\mathcal{P}_{ij}}$ unless $\Delta\mathbf{x}_{\mathcal{P}_{ij}} \leq d_P$ for which $\Delta\mathbf{x}_{\mathcal{P}_{ij}} = 0$. Concretely, the modified difference becomes $\Delta\mathbf{x}'_{\mathcal{P}_{ij}} = \max(0,\ \Delta\mathbf{x}_{\mathcal{P}_{ij}} - d_P\,\mathbf{u}_{\mathcal{P}_{ij}})$ where $\mathbf{u}_{\mathcal{P}_{ij}} = \Delta\mathbf{x}_{\mathcal{P}_{ij}} / |\Delta\mathbf{x}_{\mathcal{P}_{ij}}|$.

In order to obtain a similarity measure that accounts for nearness of group contours *and* similar velocity, we augment $\Delta\mathbf{x}'_{\mathcal{P}_{ij}}$ by the difference in the velocity components, $\Delta\mathbf{x}^*_{\mathcal{P}_{ij}} = (\Delta\mathbf{x}'^T_{\mathcal{P}_{ij}},\ \dot{x}_i - \dot{x}_j\,,\ \dot{y}_i - \dot{y}_j)^T$. Statistical compatibility of two groups G_i and G_j can now be determined with the (four-dimensional) minimum Mahalanobis distance

$$d^2_{\text{min}}(G_i, G_j) = \min_{\mathbf{x}_{\mathcal{P}_i} \in \mathcal{P}_i,\, \mathbf{x}_{\mathcal{P}_j} \in \mathcal{P}_j} \left\{ d^2(\Delta\mathbf{x}^*_{\mathcal{P}_{ij}},\, C_i + C_j) \right\}.$$

The probability that two groups actually belong together, is finally given by $\mathcal{N}_{ij} := \mathcal{N}\left(d^2_{\text{min}}(G_i, G_j),\, C_i + C_j\right)$.

IV. Estimating the Number of People in Groups

As described above, our group tracking approach considers the joint state of groups rather than the states of the individuals that form the groups. However, knowing the number of people in a group is interesting information, e.g., for interaction, data association or motion planning. We therefore augment the state vector of group tracks by a fifth state variable, n_s, the group size. A group state is then the vector $\mathbf{x} = (x,\, y,\, \dot{x},\, \dot{y},\, n_s)^T$.

As an observation of the group size, we take the number of human-sized clusters in the set of contour points $\mathcal{P}$ of a group track G. Reapplying single-linkage clustering with a cluster distance of $d_P = 0.3\,m$ yields groups of points that are likely to correspond to human individuals.

For state prediction and in case of a track confirmation event, we assume, analogous to the constant velocity motion model, constant group size. Noise in the motion model accounts for people joining or leaving the group without being noticed. If two tracks are merged, the resulting size estimate is the sum of the sizes of the joining groups. The variances simply sum up, as we assume independent size estimates across groups. If two tracks are split, we split the group size in half and increase the variance to account for uneven splits.

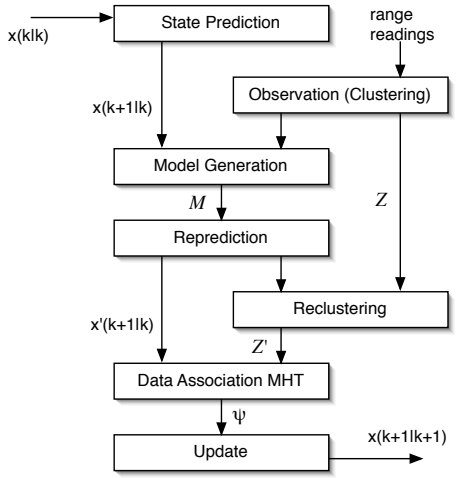


Fig. 2. Flow diagram of the tracking system. See explanations in section V.

V. Tracking Cycle

This section describes the steps in the cycle of our Kalman filter-based group tracker. An overview is given by the flow diagram in Fig. 2. The structure differs from a regular tracker in the additional steps *model generation*, *track reprediction* and *reclustering*.

- *State prediction:* The state prediction of a group track based on the previous posterior estimates $\mathbf{x}(k|k)$, $C(k|k)$ is given by $\mathbf{x}(k+1|k) = A\ \mathbf{x}(k|k)$ and $C(k+1|k) = A\,C(k|k)\,A^T + Q$, where A is the state transition matrix for a constant velocity motion model and Q the 4×4 process noise covariance matrix whose entries reflect the acceleration capabilities of a typical human. The set of contour points $\mathcal{P}$ is now relative to $\mathbf{x}(k+1|k)$.
- *Observation:* As described in section II, this step involves grouping the laser range data into clusters $\mathcal{Z}$.
- *Model Generation:* Models are generated based on the predicted group tracks and the clusters $\mathcal{Z}$, see section VI.
- *Reprediction:* Based on the model hypotheses that postulate a split, merge or continuation event for each track, groups are repredicted so as to reflect the respective model:

  If a model hypothesis contains a split of a group, two new groups are created by duplicating its predicted state. The same applies for the set $\mathcal{P}$.

  If a model hypothesis contains a merge of two groups G_i, G_j, the repredicted group state $\mathbf{x}_{ij}$, C_{ij} is computed as the multivariate weighted average (omitting $(k+1|k)$),

$$C_{ij}^{-1} = C_i^{-1} + C_j^{-1}$$

$$\mathbf{x}_{ij} = C_{ij}\,(C_i^{-1}\mathbf{x}_i + C_j^{-1}\mathbf{x}_j)\,. \quad (6)$$

  The set of contour points of the merged group is the union of the two former point sets, $\mathcal{P}_{ij} = \mathcal{P}_i \cup \mathcal{P}_j$.

- *Reclustering:* Reclustering an observed cluster $\mathcal{Z}_i$ is necessary when it has been produced by more than one group track, that is, it is in the gate of more than one track. If the model hypothesis postulates a merge for the involved tracks, nothing needs to be done. Otherwise, $\mathcal{Z}_i$ needs to be reclustered, which is done using a nearest-neighbor rule: those points $\mathbf{z}_i \in \mathcal{Z}_i$ that share the same nearest neighbor track are combined in a new cluster. This step follows from the uniqueness assumption – common in target tracking – which says that a target can only produce a single observation.
- *Data Association MHT:* This step involves the generation, probability calculation, and pruning of data association hypotheses that assign repredicted group tracks to reclustered observations. See section VII.
- *Update:* A group track G_j that has been assigned to a cluster $\mathcal{Z}_i$ is updated with a standard linear Kalman filter using the centroid position $\bar{\mathbf{z}}_{\mathcal{Z}_i}$ of $\mathcal{Z}_i$. The contour points in $\mathcal{P}_j$ are replaced by the points in $\mathcal{Z}_i$, transformed into the reference frame of the posterior state $\mathbf{x}(k+1|k+1)$. Thereby, $\mathcal{P}_j$ contains always the group's most actual shape approximation.

VI. Model Generation and Model Probability

A model is defined to be a partitioning of tracks into groups. It assumes a particular state of the group formation process. New models, whose generation is described in this section, hypothesize about the evolution of that state.

The space of possible model transitions is large since each group track can split into an unknown number of new tracks, or merge with an unknown number of other tracks. We therefore bound the possible number of model transitions by the assumption that merge and split are binary operators. We further impose the gating condition for observations and tracks using the minimum average Hausdorff distance, thereby implementing a data-driven aspect into the model generation step. Concretely, we assume:

- A track G_i can split at most into two tracks in one frame provided two compatible observations with G_i.
- At most two group tracks G_i, G_j can merge into one track at the same time but only if there is an observation which is statistically compatible with G_i and G_j.
- A group track can only split into tracks that are both matched in that very time step. Splits into occluded or obsolete tracks are not allowed.
- A group track can not be involved in a split and a merge action at the same time.

Gating and statistical compatibility are both determined on a significance level α. The limitation to binary operators is justified by the realistic assumption that we observe the world much faster than the rate with which it evolves. Even if, for instance, a group splits into three subgroups at once, the tracker requires only two cycles to reflect this change.

A new model now defines for each group track if it is continued, split or if it merges with another group track. The probability of a model is calculated using constant prior probabilities for continuations and splits, p_C and p_S respectively, and the probability for a merge between two tracks G_i and G_j as $p_G \cdot \mathcal{N}_{ij}$. The latter term consists of a constant prior probability p_G and the group-to-group assignment probability $\mathcal{N}_{ij}$ defined in section III-C. Let N_C and N_S be the number of continued tracks and the number of split tracks in model M respectively, then the probability of M conditioned on the parent hypothesis Ω^{k-1} is

$$P(M|\Omega^{k-1}) = p_C^{N_C} \cdot p_S^{N_S} \prod_{G_i,G_j \in \Omega^{k-1}} (p_G \cdot \mathcal{N}_{ij})^{\delta_{ij}} \quad (7)$$

with δ_{ij} being 1 if G_i, G_j merge and 0 otherwise.

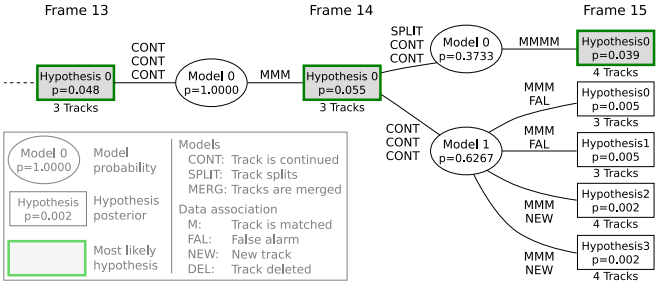


Fig. 3. The multi-model MHT. For each parent hypothesis, model hypotheses (ellipses) branch out and create their own assignment problems. In our application, models define which tracks of the parent hypothesis are continued, split or merge. The tree shows frames 13 to 15 of figure 4. The split of group 1 between frames 14 and 15 is the most probable hypothesis following model branch 0. See the legend for details.

VII. Multi-Model MHT

In this section we describe our extension of the original MHT by Reid [13] to a multi-model tracking approach that hypothesizes over both, data associations and models.

Let Ω_i^k be the i-th hypothesis at time k and $\Omega_{p(i)}^{k-1}$ its parent. Let further $\psi_i(k)$ denote a set of assignments which associates predicted tracks in $\Omega_{p(i)}^{k-1}$ to observations in $Z(k)$. As there are many possible assignment sets given $\Omega_{p(i)}^{k-1}$ and $Z(k)$, there are many children that can branch off a parent hypothesis, each with a different $\psi(k)$. This makes up an exponentially growing hypothesis tree.

The multi-model MHT introduces an intermediate tree level for each time step, on which models spring off from parent hypotheses (Fig. 3). In each model branch, the tracks of the parent hypothesis are first repredicted to implement that particular model and then assigned to the (reclustered) observations. Possible assignments for observations are *matches* with existing tracks, *false alarms* or *new tracks*. Using the generalized formulation of Arras *et al.* [7] to deal with more than two track interpretation labels, tracks are interpreted as *matched*, *obsolete* or *occluded*.

A. Probability Calculations

The probability of a hypothesis in the multi-model MHT is calculated as follows. According to the Markov assumption, the probability of a child hypothesis Ω_i^k given the observations from all time steps up to k, denoted by Z^k, is the joint probability of the assignment set $\psi_i(k)$, the model M

and the parent hypothesis $\Omega_{p(i)}^{k-1}$, conditioned on the current observation $Z(k)$. Using Bayes rule, this can be expressed as the product of the data likelihood with the joint probability of assignment set, model and parent hypothesis,

$$\begin{aligned} P(\Omega_i^k|Z^k) &= P(\psi, M, \Omega_{p(i)}^{k-1}|Z(k)) \quad (8) \\ &= \eta \cdot P(Z(k)|\psi, M, \Omega_{p(i)}^{k-1}) \cdot P(\psi, M, \Omega_{p(i)}^{k-1}). \end{aligned}$$

By using conditional probabilities, the third term on the right hand side can be factorized into the probabilities of the assignment set, the model and the parent hypothesis,

$$P(\psi, M, \Omega_{p(i)}^{k-1}) = P(\psi|M, \Omega_{p(i)}^{k-1}) \cdot P(M|\Omega_{p(i)}^{k-1}) \cdot P(\Omega_{p(i)}^{k-1}).$$

The last term is known from the previous iteration while the second term was derived in section VI.

The first term is the probability of the assignment set ψ. The set ψ contains the assignments of observed clusters $\mathcal{Z}_i$ and group tracks G_j either to each other or to one of their possible labels listed above. Assuming independence between observations and tracks, the probability of the assignment set is the product of the individual assignment probabilities. They are: p_M for matched tracks, p_F for false alarms, p_N for new tracks, p_O for tracks found to be occluded and p_T for obsolete tracks scheduled for termination. If the number of new tracks and false alarms follow a Poisson distribution (as assumed by Reid [13]), the probabilities p_F and p_N have a sound physical interpretation as $p_F = \lambda_F V$ and $p_N = \lambda_N V$ where λ_F and λ_N are the average rates of events per volume multiplied by the observation volume V (the sensor's field of view). The probability for an assignment ψ, given a model M and a parent hypothesis Ω^{k-1} is then computed by

$$P(\psi|M, \Omega^{k-1}) = p_M^{N_M}\ p_O^{N_O}\ p_T^{N_T}\ \lambda_F^{N_F} \lambda_N^{N_N}\ V^{N_F+N_N}, \quad (9)$$

where the Ns are the number of assignments in ψ to the respective labels.

Thanks to the independence assumption, also the data likelihood $P(Z(k)|\,\psi, M, \Omega_{p(i)}^{k-1})$ is computed by the product of the individual likelihoods of each observation cluster $\mathcal{Z}_i$ in $Z(k)$. If ψ assigns an observation $\mathcal{Z}_i$ to an existing track, we assume the likelihood of $\mathcal{Z}_i$ to follow a normal distribution, given by Eq. 5. Observations that are interpreted as false alarms and new tracks are assumed to be uniformly distributed over the observation volume V, yielding a likelihood of $1/V$. The data likelihood then becomes

$$P(Z(k)|\psi, M, \Omega^{k-1}) = \left(\tfrac{1}{V}\right)^{N_N+N_F} \prod_{i=1}^{N_{\mathcal{Z}}} \mathcal{N}_i^{\delta_i}, \quad (10)$$

where δ_i is 1 if $\mathcal{Z}_i$ has been assigned to an existing track, and 0 otherwise.

Substitution of Eqs. (7), (9), and (10) into Eq. (8) leads, like in the original MHT approach, to a compact expression, independent on the observation volume V.

Finally, normalization is performed yielding a true probability distribution over the child hypotheses of the current time step. This distribution is used to determine the current best hypothesis and to guide the pruning strategies.

TABLE I

SUMMARY OF THE DATA USED IN THE TWO EXPERIMENTS.

	Experiment 1	Experiment 2
Number of frames	578	991
Avg. / max people	6.25 / 13	8.99 / 20
Avg. / max groups	2.60 / 4	4.16 / 8
Number of splits / merges	5 / 10	48 / 44
Number of new tracks / deletions	19 / 15	34 / 39

B. Pruning

Pruning is essential in implementations of the MHT algorithm, as otherwise the number of hypotheses grows boundless. The following strategies are employed:

K-best branching: instead of creating all children of a parent hypothesis, the algorithm proposed by Murty [16] generates only the K most probably hypotheses in polynomial time. We use the multi-parent variant of Murty's algorithm, mentioned in [22], that generates the global K best hypotheses for all parents.

Ratio pruning: a lower limit on the ratio of the current and the best hypothesis is defined. Unlikely hypotheses with respect to the best one, being below this threshold, are deleted. Ratio pruning overrides K-best branching in the sense that if the lower limit is reached earlier, less than K hypotheses are generated.

N-scan back: the N-scan-back algorithm considers an ancestor hypothesis at time $k-N$ and looks ahead in time onto all children at the current time k (the leaf nodes). It keeps only the subtree at $k-N$ with the highest sum of leaf node probabilites, all other branches at $k-N$ are discarded.

VIII. EXPERIMENTS

To analyze the performance of our system, we collected two data sets in a large entrance hall of a university building. We used a Pioneer II robot equipped with a SICK laser scanner mounted at 30 cm above floor, scanning at 10 fps. In two unscripted experiments (experiment 1 with a stationary robot, experiment 2 with a moving robot), up to 20 people are in the sensor's field of view. They form a large variety of groups during social interaction, move around, stand together and jointly enter and leave the hall (see Tab. I).

To obtain ground truth information, we labeled each single range reading. Beams that belong to a person receive a person-specific label, other beams are labeled as non-person. These labels are kept consistent over the entire duration of the data sets. People that socially interact with each other (derived by observation) are said to belong into a group with a group-specific label. Summed over all frames, the ground truth contains 5629 labeled groups and 12524 labeled people.

The ground truth data is used for performance evaluation and to learn the parameter probabilities of our tracker. The values, determined by counting, are $p_M = 0.79$, $p_O = 0.19$, $p_T = 0.02$, $p_F = 0.06$, $p_N = 0.02$ for the data association probabilities, and $p_C = 0.63$, $p_S = 0.16$, $p_G = 0.21$ for the group formation probabilities. When evaluating the performance of the tracker, we separated the data into a training set and a validation set to avoid overfitting.

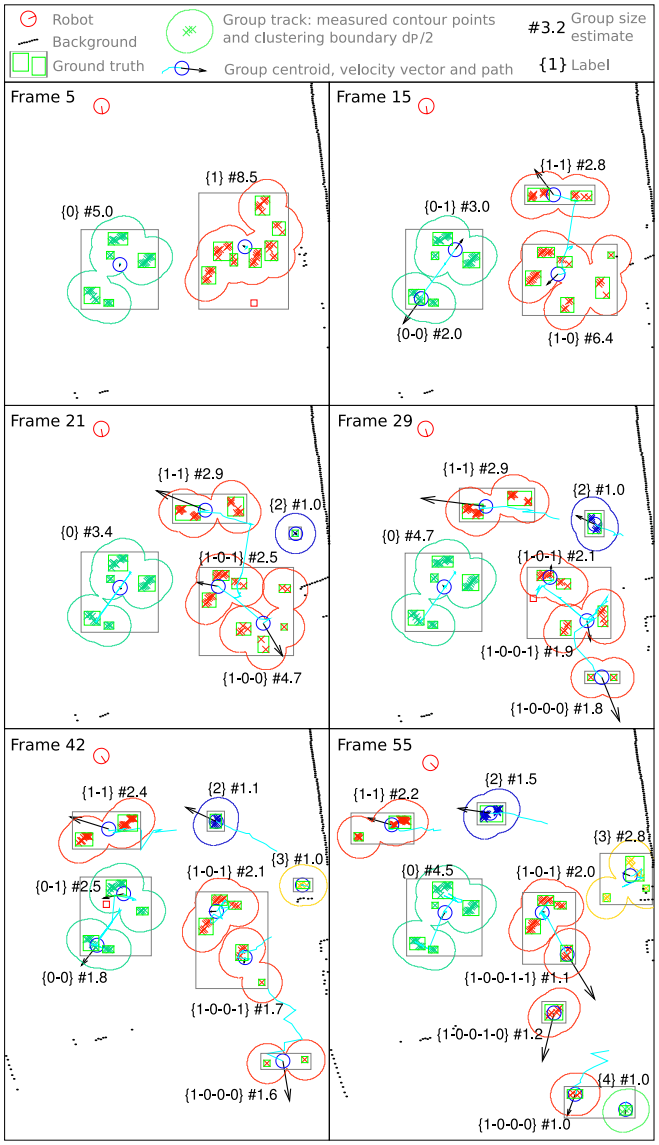


Fig. 4. Tracking results from experiment 2. In frame 5, two groups are present. In frame 15, the tracker has correctly split group 1 into 1-0 and 1-1 (see Fig. 3). Between frames 15 and 29, group 1-0 has split up into groups 1-0-0 and 1-0-1, and split up again. New groups, labeled 2 and 3, enter the field of view in frames 21 and 42 respectively.

Six frames of the current best hypothesis from experiment 2 are shown in Fig. 4, the corresponding hypothesis tree is shown in Fig. 3. The sequence exemplifies movement and formation of several groups.

A. Clustering Error

Given the ground truth information on a per-beam basis we can compute the clustering error of the tracker. This is done by counting how often a track's set of points $\mathcal{P}$ contains too many or wrong points (undersegmentation) and how often $\mathcal{P}$ is missing points (oversegmentation) compared to the ground truth. Two examples for oversegmentation errors can be seen in Fig. 4, where group 0 and group 1-0 are temporarily oversegmented. However, from the history of group splits and merges stored in the group labels, the correct group relations can be determined in such cases.

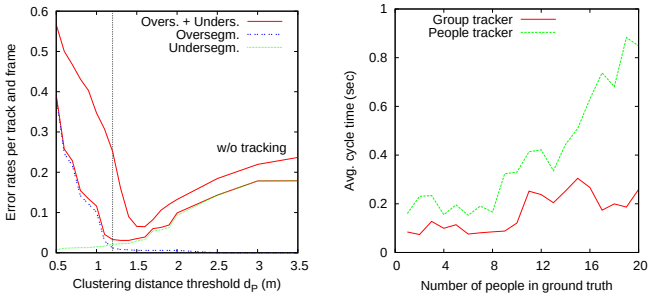


Fig. 5. Left: clustering error of the group tracker compared to a memory-less single linkage clustering (without tracking). The smallest error is achieved for a cluster distance of 1.3 m which is very close to the border of personal and social space according to the proxemics theory, marked at 1.2 m by the vertical line. Right: average cycle time for the group tracker versus a tracker for individual people plotted against the ground truth number of people.

For experiment 1, the resulting percentages of incorrectly clustered tracks for the cases undersegmentation, oversegmentation and the sum of both are shown in Fig. 5 (left), plotted against the clustering distance d_P. The figure also shows the error of a single-linkage clustering of the range data as described in section II. This implements a memory-less group clustering approach against which we compare the clustering performance of our group tracker.

The minimum clustering error of 3.1% is achieved by the tracker at $d_P = 1.3\,m$. The minimum error for the memory-less clustering is 7.0%, more than twice as high. In the more complex experiment 2, the minimum clustering error of the tracker rises to 9.6% while the error of the memory-less clustering reaches 20.2%. The result shows that the group tracking problem is a *recursive* clustering problem that requires integration of information over time. This occurs when two groups approach each other and pass from opposite directions. The memory-less approach would merge them immediately while the tracking approach, accounting for the velocity information, correctly keeps the groups apart.

In the light of the proxemics theory the result of a minimal clustering error at 1.3 m is noteworthy. The theory predicts that when people interact with friends, they maintain a range of distances between 45 to 120 cm called personal space. When engaged in interaction with strangers, this distance is larger. As our data contains students who tend to know each other well, the result appears consistent with Hall's findings.

B. Tracking Efficiency

When tracking groups of people rather than individuals, the assignment problems in the data association stage are of course smaller. On the other hand, the introduction of an additional tree level on which different models hypothesize over different group formation processes comes with additional computational costs. We therefore compare our system with a person-only tracker which is implemented by inhibiting all split and merge operations and reducing the cluster distance d_P to the very value that yields the lowest error for clustering single people given the ground truth. For

experiment 2, the resulting average cycle times versus the ground truth number of people is shown in Fig. 5 (right). The plots are averaged over different k from the range of 2 to 200 at a scan-back depth of $N = 30$.

With an increasing number of people, the cycle time for the people tracker grows much faster than the cycle time of the group tracker. Interestingly, even for small numbers of people the group tracker is faster than the people tracker. This is due to occasional oversegmentation of people into individual legs tracks. Also, as mutual occlusion of people in densely populated environments occurs often, the people tracker has a lot more occluded tracks to maintain than the group tracker, as occlusion of entire groups is rare. Also, the additional complexity of multiple models in the group tracker virtually disappears when the tracks are isolated due to the data-driven model generation.

This result clearly shows that the claim of higher efficiency holds for this group tracking approach. With an average cycle time of around 100 ms for up to 10 people on a Pentium IV at 3.2 GHz, the algorithm runs in real-time even with a non-optimized implementation.

C. Group Size Estimation

To evaluate the accuracy of our group size estimation approach, we define the error as the absolute difference between the estimated number of people in a group and the true value according to the labeled ground truth.

In experiment 1, we find that the average error is 0.23 people with a standard deviation of 0.30. In the more complex experiment 2, the average error is 0.33 people with a standard deviation of 0.49. If the estimated group sizes are rounded to integers, the tracker determined the correct value in 88.9% of all cases in experiment 1 and in 84.3% for experiment 2.

If only deviations of more than one person are considered an error, the system was correct in 99.5% of all cases in experiment 1 and 97.5% in experiment 2.

IX. Conclusion

In this paper, we presented a multi-model hypothesis tracking approach to track groups of people. We extended the original MHT approach to incorporate model hypotheses that describe track interaction events that go beyond what data association can express. In our application, models encode the formation of groups during split, merge, and continuation events. We further introduced a representation of groups that includes their shape, and employed the minimum average Hausdorff distance to account for the shape information when calculating association probabilities.

The proposed tracker has been implemented and tested using a mobile robot equipped with a laser range finder. It is able to robustly track groups of people as they undergo complex formation processes. Given ground truth data with over 12,000 labeled occurrences of people and groups, the experiments showed that the tracker could reproduce such processes with a low clustering error and very accurate estimates of the number of people in groups.

Further experiments carried out from a stationary and a moving platform in populated environments with up to 20 people demonstrated that tracking groups of people is clearly more efficient than tracking individual people. They also showed that our system performs significantly better than a memory-less single-frame clustering which underlines the recursive character of this model selection problem.

References

[1] B. Kluge, C. Köhler, and E. Prassler, "Fast and robust tracking of multiple moving objects with a laser range finder," in *Proceedings of the IEEE Int. Conf. on Robotics and Automation*, 2001.

[2] A. Fod, A. Howard, and M. J. Mataric, "Laser-based people tracking," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Washington DC, May 2002, pp. 3024–3029.

[3] D. Schulz, W. Burgard, D. Fox, and A. Cremers, "People tracking with a mobile robot using sample-based joint probabilistic data association filters," *Intl. J. of Robotics Research (IJRR)*, vol. 22, no. 2, 2003.

[4] J. Cui, H. Zha, H. Zhao, and R. Shibasaki, "Tracking multiple people using laser and vision," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, 2005.

[5] W. Zajdel, Z. Zivkovic, and B. Kröse, "Keeping track of humans: Have I seen this person before?" in *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[6] G. Taylor and L. Kleeman, "A multiple hypothesis walking person tracker with switched dynamic model," in *Proc. of the Australasian Conference on Robotics and Automation*, Canberra, Australia, 2004.

[7] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, CA, USA, May 2008.

[8] Z. Khan, T. Balch, and F. Dellaert, "MCMC data association and sparse factorization updating for real time multitarget tracking with merged and multiple measurements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, December 2006.

[9] M. Mucientes and W. Burgard, "Multiple hypothesis tracking of clusters of people," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2006, pp. 692–697.

[10] S. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler, "Tracking groups of people," *Computer Vision and Image Understanding*, vol. 80, no. 1, pp. 42–56, October 2000.

[11] G. Gennari and G. D. Hager, "Probabilistic data association medhods in visual tracking of groups," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[12] B. Bose, X. Wang, and E. Grimson, "Multi-class object tracking algorithm that handles fragmentation and grouping," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[13] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. on Automatic Control*, vol. AC-24, no. 6, pp. 843–854, 1979.

[14] I. Cox and S. Hingorani, "An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 138–150, February 1996.

[15] S.-W. Joo and R. Chellappa, "A multiple-hypothesis approach for multiobject visual tracking," *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2849–2854, November 2007.

[16] K. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, pp. 682–687, 1968.

[17] B. Lau, K. O. Arras, and W. Burgard, "Tracking groups of people with a multi-model hypothesis tracker," in *International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009.

[18] E. Hall, *Handbook of Proxemics Research*. Society for the Anthropology of Visual Communications, 1974.

[19] K. O. Arras, Óscar Martínez Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA'07)*, Rome, Italy, 2007.

[20] J. Hartigan, *Clustering Algorithms*. John Wiley & Sons, 1975.

[21] M. P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," in *Intl. Conference on Pattern Recognition*, vol. 1, Jerusalem, Israel, 1994, pp. A:566–568.

[22] I. Cox and M. Miller, "On finding ranked assignments with application to multi-target tracking and motion correspondence," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 31, no. 1, pp. 486–489, 1995.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Spatially Grounded Multi-Hypothesis Tracking of People

Matthias Luber Gian Diego Tipaldi Kai O. Arras

***Abstract*— People tracking is an important yet challenging task for mobile robots operating in populated environments and interacting with humans. What makes this problem difficult is that human behavior is complex and hard to predict. However, motion of people, the rate at which people appear and where they appear are not random but strongly place-dependent and follow patterns that are engendered by the environment. In this paper we make use of such information for the purpose of people tracking. Concretely, we learn a probabilistic representation, called *spatial affordance map*, to spatially ground activity events acquired by observing people in the environment. This representation is a non-homogeneous spatial Poisson process for which we derive expressions for life-long Bayesian learning. We show how the spatial affordance map can be used to compute refined probability distributions over hypotheses in a multi-hypothesis tracker and to make better, place-dependent predictions of human motion. In experiments with real data from a laser range finder, we demonstrate how both extensions lead to more accurate tracking behavior. The system runs in real-time on a typical desktop computer.**

I. Introduction

As robots enter more domains in which they interact and cooperate closely with humans, people tracking is becoming a key technology for several areas in robotics such as human-robot interaction, intelligent cars or human activity understanding.

In this paper we pursue the approach to learn and represent human spatial behavior for improved people tracking. Human activity is strongly place-dependent. By learning a spatial model that represents activity events in a global reference frame and on large time scales, the robot acquires place-dependent priors on human behavior. As we will demonstrate, such priors can be used to better hypothesize about the state of the world (that is, the state of people in the world), and to make place-dependent predictions of human motion that better reflect how people are using space. Concretely, we propose a non-homogeneous spatial Poisson process to represent the spatially varying distribution over relevant human activity events for people tracking. The representation, called *spatial affordance map*, holds space-dependent Poisson rates for the occurrence of track events such as creation, confirmation or false alarm. The map is then incorporated into a multi-hypothesis tracking framework using data from a laser range finder.

All authors are with the Social Robotics Lab, Department of Computer Science, University of Freiburg, Germany {luber,tipaldi,arras}@informatik.uni-freiburg.de.

In most related work on laser-based people tracking [1], [2], [3], [4], [5], [6], [7], a person is represented as a single state that encodes torso position and velocities. People are extracted from range data as single blobs or found by merging nearby point clusters that correspond to legs. The problem of people tracking has also been addressed as a leg tracking problem [8], [9], [10] where people are represented by the states of two legs, either in a single augmented state [9] or as a high-level track to which two low-level leg tracks are associated [8], [10].

Different tracking and data association approaches have been used for laser-based people tracking. The nearest neighbor filter and variations thereof are typically employed in earlier works [1], [2], [3]. A sample-based joint probabilistic data association filter (JPDAF) has been presented in Schulz *et al.* [4] and adopted by Topp *et al.* [5]. The Multi-hypothesis tracking (MHT) approach according to Reid [11] and Cox *et al.* [12] has been used in [8], [7], [10]. What makes the MHT an attractive choice is that it belongs to the most general data association techniques. The method generates joint compatible assignments, integrates them over time, and is able to deal with track creation, confirmation, occlusion, and deletion events in a probabilistically consistent way. Other multi-target data association techniques such as the global nearest neighbor filter, the track splitting filter or the JPDAF are suboptimal in nature as they simplify the problem in one or the other way [13], [14]. For this reasons, the MHT has become a widely accepted tool in the target tracking community [14].

The MHT framework assumes that new track and false alarm events are uniformly distributed in the sensor field of view with fixed Poisson rates. This assumption is justified in settings for which the approach has been originally developed (using, e.g., radar or underwater sonar). However, in the context of people tracking with vision or laser these models are overly simplified. Particularly since people do not use environments randomly but move, appear and disappear at specific locations that correspond, for instance, to doors, entrances, or convex corners. Further, false alarms are more likely to arise in areas with cluttered backgrounds rather than in open spaces. In this paper, we extend the MHT approach by incorporating learned distributions over track interpretation events that serve as domain knowledge to the system to better hypothesize about the state of the world.

For motion prediction of people, most researchers employ the Brownian motion model and the constant velocity motion model. The former makes no assumptions

about the target dynamics, the latter assumes linear target motion. Better motion models for people tracking have been proposed by Bruce and Gordon [15] and Liao *et al.* [16].

In [15], the robot learns goal locations in an environment from people trajectories obtained by a laser-based tracker. Goals are found as end points of clustered trajectories. Human motion is then predicted along paths that a planner generates from the location of people being tracked to the goal locations. The performance of the tracker was improved in comparison to a Brownian motion model. Liao *et al.* [16] extract a Voronoi graph from a map of the environment and represent the state of people being on edges of that graph. This allows them to predict motion of people along the edges that follow the topological shape of the environment.

With maneuvering targets, a single model can be insufficient to represent the target's motion. Multiple model based approaches in which different models run in parallel and describe different aspects of the target behavior are a widely accepted technique to deal with maneuvering targets, in particular the Interacting Multiple Model (IMM) algorithm [17]. Different target motion models are also studied by Kwok and Fox [18]. The approach is based on a Rao-Blackwellized particle filter to model the potential interactions between a target and its environment. The authors define a discrete set of different target motion models from which the filter draws samples. Then, conditioned on the model, the target is tracked using Kalman filters.

Our approach extends prior work in two aspects, learning and place-dependent motion prediction. Opposed to [16], [18] and IMM related methods, we do not rely on predefined motion models but apply learning for this task in order to acquire place-dependent models. In [16], the positions of people is projected onto a Voronoi graph which is a topologically correct but metrically poor model for human motion. While sufficient for the purpose of their work, there is no insight why people should move on a Voronoi set, particularly in open spaces whose topology is less well defined. Our approach, by contrast, tracks the actual position of people and predicts their motion according to metric, place-dependent models. Opposed to [15] where motion prediction is done along paths that a planner plans to a set of goal locations, our learning approach predicts motion along the trajectories that people are actually following.

The paper is structured as follows: the next section gives an overview of the people tracker that will later be extended. Section III introduces the theory of the spatial affordance map and expressions for learning its parameters. Section IV describes how the spatial affordance map can be used to compute refined probability distributions over hypotheses, while section V contains the theory for the place-dependent motion model. Section VI presents the experimental results followed by the conclusions in section VII.

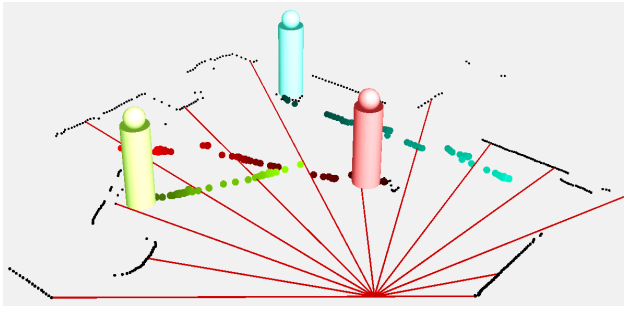


Fig. 1. An example scene from experiment 2 (frame 185) where three people are being tracked.

II. Multi-Hypothesis Tracking of People

For people tracking, we pursue a Multi-Hypothesis Tracking (MHT) approach described in Arras *et al.* [10] based on the original MHT by Reid [11] and Cox and Hingorani [12]. As we will use the tracker to learn the spatial affordance map described hereafter, we give a short outline. Sections IV and V, where the approach will be extended, contains the technical details.

Summarizing, the MHT algorithm hypothesizes about the state of the world by considering all statistically feasible assignments between measurements and tracks and all possible interpretations of measurements as false alarms or new track and tracks as matched, occluded or obsolete. A hypothesis Ω_i^t is one possible set of assignments and interpretations at time t.

For learning the spatial affordance map, the hypothesis with maximal probability Ω_{best}^t at time t is chosen to produce the track events that subsequently serve as observations for the map. In case of a sensor mounted on a mobile platform, we assume the existence of a metric map of the environment and the ability of the robot to self-localize. Observations are then transformed from local, robot-centric coordinates into the world reference frame of the map.

III. Spatial Affordance Map

The spatial affordance map is a non-homogeneous spatial Poisson process. This section describes the theory and how learning is implemented in this application of a Poisson process.

A Poisson distribution is a discrete distribution to compute the probability of a certain number of events given an expected average number of events over time or space. The parameter of the distribution is the positive real number λ, the rate at which events occur per time or volume units. As we are interesting in modeling events that occur randomly in time, the Poisson distribution is a natural choice.

Based on the assumption that events in time occur independently of one another, a *Poisson process* can deal with distributions of time intervals between events. Concretely, let $N(t)$ be a discrete random variable to represent the number of events occurring up to time t

with rate λ. Then we have that $N(t)$ follows a Poisson distribution with parameter λt

$$P(N(t) = k) = \frac{e^{-\lambda t}(\lambda t)^k}{k!} \qquad k = 0, 1, \ldots \tag{1}$$

In general, the rate parameter may change over time. In this case, the generalized rate function is given as $\lambda(t)$ and the expected number of events between time a and b is

$$\lambda_{a,b} = \int_a^b \lambda(t)\, dt. \tag{2}$$

A homogeneous Poisson process is a special case of a non-homogeneous process with constant rate $\lambda(t) = \lambda$.

The *spatial* Poisson process introduces a spatial dependency on the rate function given as $\lambda(\vec{x}, t)$ with $\vec{x} \in X$ where X is a vector space such as $\mathbb{R}^2$ or $\mathbb{R}^3$. For any subset $S \subset X$ of finite extent (e.g. a spatial region), the number of events occurring inside this region can be modeled as a Poisson process with associated rate function $\lambda_S(t)$ such that

$$\lambda_S(t) = \int_S \lambda(\vec{x}, t)\, d\vec{x}. \tag{3}$$

In the case that this generalized rate function is a separable function of time and space, we have:

$$\lambda(\vec{x}, t) = f(\vec{x})\lambda(t) \tag{4}$$

for some function $f(\vec{x})$ for which we can demand

$$\int_X f(\vec{x})\, d\vec{x} = 1 \tag{5}$$

without loss of generality. This particular decomposition allows us to decouple the occurrence of events between time and space. Given Eq. 5, $\lambda(t)$ defines the occurrence rate of events, while $f(\vec{x})$ can be interpreted as a probability distribution on where the event occurs in space.

Learning the spatio-temporal distribution of events in an environment is equivalent to learn the generalized rate function $\lambda(\vec{x}, t)$. However, learning the full continuous function is a highly expensive process. For this reason, we approximate the non-homogeneous spatial Poisson process with a piecewise homogeneous one. The approximation is performed by discretizing the environment into a bidimensional grid, where each cell represents a local homogeneous Poisson process with a fixed rate over time,

$$P_{ij}(k) = \frac{e^{-\lambda_{ij}}(\lambda_{ij})^k}{k!} \qquad k = 0, 1, \ldots \tag{6}$$

where λ_{ij} is assumed to be constant over time. Finally, the spatial affordance map is the generalized rate function $\lambda(\vec{x}, t)$ using a grid approximation,

$$\lambda(\vec{x}, t) \simeq \sum_{(i,j)\in X} \lambda_{ij} \mathbf{1}_{ij}(\vec{x}) \tag{7}$$

with $\mathbf{1}_{ij}(\vec{x})$ being the indicator function defined as

$$\mathbf{1}_{ij}(x) = \begin{cases} 1 & \text{if } x \in \text{cell}_{ij}, \\ 0 & \text{if } x \notin \text{cell}_{ij}. \end{cases} \tag{8}$$

The type of approximation is not imperative and goes without loss of generality. Other space tessellation techniques such as graphs, quadtrees or arbitrary regions of homogeneous Poisson rates can equally be used. Subdivision of space into regions of fixed Poisson rates has the property that the preferable decomposition in Eq. 4 holds.

Each type of human activity event can be used to learn its own probability distribution in the map. We can therefore think of the map as a representation with multiple layers, one for every type of event. For the purpose of this paper, the map has three layers, one for new tracks, for matched tracks and for false alarms. The first layer represents the distribution and rates of people appearing in the environment. The second layer can be considered a space usage probability and contains a walkable area map of the environment. The false alarm layer represents the place-dependent reliability of the detector.

A. Learning

In this section we show how to learn the parameter of a single cell in our grid from a sequence $K_{1..n}$ of n observations $k_i \in \{0, 1\}$. We use Bayesian inference for parameter learning, since the Bayesian approach can provide information on cells via a prior distribution. We model the parameter λ using a Gamma distribution, as it is the conjugate prior of the Poisson distribution. Let λ be distributed according to the Gamma density, $\lambda \sim \text{Gamma}(\alpha, \beta)$, parametrized by the two parameters α and β,

$$\text{Gamma}(\lambda; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\,\lambda} \qquad \text{for } \lambda > 0. \tag{9}$$

Then, learning the rate parameter λ consists in estimating the parameters of a Gamma distribution. At discrete time index i, the posterior probability of λ_i according to Bayes' rule is computed as

$$P(\lambda_i | K_{1..i}) \sim P(k_i | \lambda_{i-1})\, P(\lambda_{i-1}) \tag{10}$$

with $P(\lambda_{i-1}) = \text{Gamma}(\alpha_{i-1}, \beta_{i-1})$ being the prior and $P(k_i|\lambda_{i-1}) = P(k_i)$ from Eq. 6 the likelihood. Then by substitution, it can be shown that the update rules for the parameters are

$$\alpha_i = \alpha_{i-1} + k_i \qquad \beta_i = \beta_{i-1} + 1. \tag{11}$$

The posterior mean of the rate parameter in a single cell is finally obtained as the expected value of the Gamma,

$$\widehat{\lambda}_{\text{Bayesian}} = \mathbb{E}[\lambda] = \frac{\alpha}{\beta} = \frac{\#\text{positive events} + 1}{\#\text{observations} + 1}\,. \tag{12}$$

For $i = 0$ the quasi uniform Gamma prior for $\alpha = 1$, $\beta = 1$ is taken. The advantages of the Bayesian estimator are that it provides a variance estimate which is a measure of confidence of the mean and that it allows to properly initialize never observed cells.

Given the learned rates we can estimate the space distribution of the various events. This distribution is

obtained from the rate function of our spatial affordance map $\lambda(\vec{x}, t)$. While this estimation is hard in the general setting of a non-homogeneous spatial Poisson process, it becomes easy to compute if the separability property of Eq. 4 holds[1]. In this case, the pdf, $f(\vec{x})$, is obtained by

$$f(\vec{x}) = \frac{\lambda(\vec{x}, t)}{\lambda(t)} \tag{13}$$

where $\lambda(\vec{x}, t)$ is the spatial affordance map. The nominator, $\lambda(t)$, can be obtained from the map by substituting the expression for $f(\vec{x})$ into the constraint defined in Eq. 5. Hence,

$$\lambda(t) = \int_X \lambda(\vec{x}, t)\, d\vec{x}. \tag{14}$$

In our grid, those quantities are computed as

$$f(\vec{x}) \quad = \quad \frac{\sum_{(i,j)\in X} \lambda_{ij} \mathbf{1}_{ij}(\vec{x})}{\sum_{(i,j)\in X} \lambda_{ij}}. \tag{15}$$

In case of several layers in the map, each layer contains the distribution $f(\vec{x})$ of the respective type of events. Note that learning in the spatial affordance map is simply realized by counting in a grid. This makes life-long learning particularly straightforward as new information can be added at any time by one or multiple robots.

Figure 2 shows two layers of the spatial affordance map of our laboratory, learned during a first experiment. The picture on the left shows the space usage distribution of the environment. The modes in this distribution correspond to often used places and have the meaning of goal locations in that room (two desks and a sofa). On the right, the distribution of new tracks is depicted whose peaks denote locations where people appear (doors). The reason for the peaks at other locations than the doors is that when subjects use an object (sit on a chair, lie on the sofa), they cause a track loss. When they reenter space, they are detected again as new tracks.

IV. MHT With Spatial Information

The Multi-Hypothesis Tracking approach has its roots in the target tracking community and was designed for sensors such as radar or underwater sonar. When employed with data from a mobile platform with cameras or laser range finders, it is questionable if the same statistical assumptions hold. The MHT assumes a Poisson distribution for the occurrences of new tracks and false alarms over time and a uniform probability of these events over space within the sensor field of view V. While this is a valid assumption for a radar aimed upwards into the sky, this is unrealistic for people being tracked by a mobile robot. The arrival of people is well modeled by a Poisson distribution but is clearly non-uniform over space. People typically appear and disappear at specific locations that correspond, for instance, to doors, entrances, or convex corners.

[1]Note that for a non-separable rate function, the Poisson process can model places whose importance changes over time.

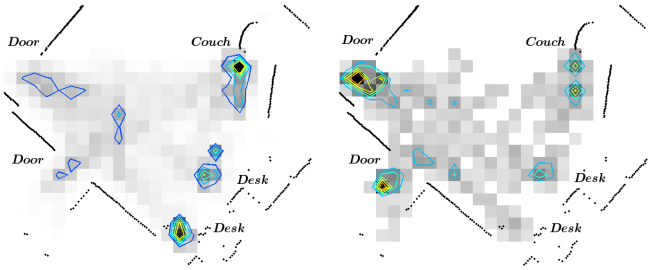


Fig. 2. Spatial affordance map of the laboratory in experiment 1. The probability distribution of matched track events is shown on the left, the distribution of new track events is shown on the right. The marked locations in each distribution (extracted with a peak finder and visualized by contours of equal probability) have different meanings. While on the left they correspond to places that are often used by people (two desks and a sofa), the maxima of the new track distribution (right) denote locations where people appear (two doors and a sofa).

It is exactly this information that the spatial affordance map holds. We can therefore seamlessly extend the MHT approach with the learned Poisson rates for the arrival events of people and learned location statistics for new tracks and false alarms.

At time t, each possible set of assignments and interpretations forms a hypothesis Ω_i^t. Let $Z(t) = \{z_i(t)\}_{i=1}^{m_t}$ be the set of m_t measurements which in our case is the set of detected people in the laser data. For detection, we use a learned classifier based on a collection of boosted features [19]. Let further $\psi_i(t)$ denote a set of assignments which associates predicted tracks to measurements in $Z(t)$ and let Z^t be the set of all measurements up to time t. Starting from a hypothesis of the previous time step, called a parent hypothesis $\Omega_{p(i)}^{t-1}$, and a new set $Z(t)$, there are many possible assignment sets $\psi(t)$, each giving birth to a child hypothesis that branches off the parent. This makes up an exponentially growing hypothesis tree. For a real-time implementation, the growing tree needs to be pruned. To guide the pruning, each hypothesis receives a probability, recursively calculated as the product of a normalizer η, a measurement likelihood, an assignment set probability and the parent hypothesis probability [11],

$$\begin{aligned} p(\Omega_l^t \mid Z^t) \quad &= \quad \eta \cdot p(Z(t) \mid \psi_i(t), \Omega_{p(i)}^{t-1}) \\ &\quad\; p(\psi_i(t) \mid \Omega_{p(i)}^{t-1}, Z^{t-1}) \cdot p(\Omega_{p(i)}^{t-1} \mid Z^{t-1}). \end{aligned} \tag{16}$$

While the last term is known from the previous iteration, the two expressions that will be affected by our extension are the measurement likelihood and the assignment set probability.

For the measurement likelihood, we assume that a measurement $z_i(t)$ associated to a track $\mathbf{x}_j$ has a Gaussian pdf centered on the measurement prediction $\hat{z}_j(t)$ with innovation covariance matrix $S_{ij}(t)$, $\mathcal{N}(z_i(t)) := \mathcal{N}(z_i(t); \hat{z}_j(t), S_{ij}(t))$. The regular MHT now assumes that the pdf of a measurement belonging to a new track or false alarm is uniform in V, the sensor field of view,

with probability V^{-1}. Thus

$$p(Z(t) \mid \psi_i(t), \Omega_{p(i)}^{t-1}) = V^{-(N_F+N_N)} \cdot \prod_{i=1}^{m_t} \mathcal{N}(z_i(t))^{\delta_i} \quad (17)$$

with N_F and N_N being the number of measurements labeled as false alarms and new tracks respectively. δ_i is an indicator variable being 1 if measurement i has been associated to a track, and 0 otherwise.

Given the spatial affordance map, the term changes as follows. The probability of new tracks V^{-1} can now be replaced by

$$p_N(\vec{x}) = \frac{\lambda_N(\vec{x}, t)}{\lambda_N(t)} = \frac{\lambda_N(\vec{x}, t)}{\int_V \lambda_N(\vec{x}, t)\, d\vec{x}} \quad (18)$$

where $\lambda_N(\vec{x}, t)$ is the learned Poisson rate of new tracks in the map and $\vec{x}$ the position of measurement $z_i'(t)$ transformed into global coordinates. The same derivation applies for false alarms. Given our grid, Eq. 18 becomes

$$p_N(\vec{x}) = \frac{\lambda_N(z_i'(t), t)}{\sum_{(i,j)\in V} \lambda_{ij,N}}. \quad (19)$$

The probability of false alarms $p_F(\vec{x})$ is calculated in the same way using the learned Poisson rate of false alarms $\lambda_F(\vec{x}, t)$ in the map.

The original expression for the assignment set probability can be shown to be [10]

$$\begin{aligned} p(\psi_i(t) \mid \Omega_{p(i)}^{t-1}, Z^{t-1}) &= \eta' \cdot p_M^{N_M} \cdot p_O^{N_O} \cdot p_D^{N_D} \quad (20) \\ &\quad \lambda_N^{N_N} \cdot \lambda_F^{N_F} \cdot V^{(N_F+N_N)} \end{aligned}$$

where N_M, N_O, and N_D are the number of matched, occluded and deleted tracks, respectively. The parameters p_M, p_O, and p_D denote the probability of matching, occlusion and deletion that are subject to $p_M + p_O + p_D = 1$. The regular MHT now assumes that the number of new tracks N_N and false alarms N_F both follow a fixed rate Poisson distribution with expected number of occurrences $\lambda_N V$ and $\lambda_F V$ in the observation volume V.

Given the spatial affordance map, they can be replaced by rates from the learned spatial Poisson process with rate functions $\lambda_N(t)$ and $\lambda_F(t)$ respectively.

Substituting the modified terms back into Eq. 16 makes, like in the original approach, that many terms cancel out leading to an easy-to-implement expression for a hypothesis probability

$$\begin{aligned} p(\Omega_l^t \mid Z^t) = \eta'' \cdot p_M^{N_M} \cdot p_O^{N_O} \cdot p_D^{N_D} \cdot \prod_{i=1}^{m_t} [\mathcal{N}(z_i(t))^{\delta_i} \quad (21) \\ \lambda_N(z_i'(t), t)^{\kappa_i} \cdot \lambda_F(z_i'(t), t)^{\phi_i}] \cdot p(\Omega_{p(i)}^{t-1} \mid Z^{t-1}) \end{aligned}$$

with δ_i and κ_i being indicator variables whether a track is matched to a measurement or new, respectively, and ϕ_i indicating if a measurement is declared to be a false alarm.

The insight of this extension of the MHT is that we replace fixed parameters by learned distributions. This kind of domain knowledge helps the tracker to better interpret measurements and tracks, leading to refined probability distributions over hypotheses at the same run-time costs.

V. Place-Dependent Motion Models

Tracking algorithms rely on the predict-update cycle, where a motion model predicts the future target position which is then validated by an observation in the update phase. Without validation, caused, for instance, by the target being hidden during an occlusion event, the state evolves blindly following only the prediction model. Good motion models are especially important for people tracking as people typically undergo lengthy occlusion events during interaction with each other or with the environment.

As motion of people is hard to predict, having a precise model is difficult. People can abruptly stop, turn back, left or right, make a step sideways or accelerate suddenly. However, motion of people is not random. In particular, it follows patterns that are strongly place-dependent. They, for instance, turn around convex corners, avoid static obstacles, stop in front of doors and do not go through walls. Clearly, the Brownian and the constant velocity motion model are unable to capture the complexity of these movements and even higher-order models would be a very approximate choice.

For this reason, we extend the constant velocity motion assumption with a place-dependent model derived from the learned space usage distribution in the spatial affordance map. Let $\mathbf{x}_t = (\ x_t \quad y_t \quad \dot{x}_t \quad \dot{y}_t\ )^T$ be the state of a track at time t and Σ_t its covariance estimate. The motion model $p(x_t|x_{t-1})$ is then defined as

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t; F\, x_{t-1}, F\, \Sigma_{t-1}\, F^T + Q) \quad (22)$$

with F being the state transition matrix. The entries in Q represent the acceleration capability of a human. We extend this model by considering how the distribution of the state at a generic time t is influenced by the previous state and the map. This distribution is approximated by the following factorization

$$p(x_t|x_{t-1}, m) \simeq p(x_t|x_{t-1}) \cdot p(x_t|m) \quad (23)$$

where m is the spatial affordance map and $p(x_t|m) = f(x)$ denotes the space usage probability of the portion of the environment occupied by x_t, as defined by Eq. 15.

A closed form estimation of this distribution does not exist since the map contains a general density, poorly described by a parametric distribution. We therefore follow a sampling approach and use a particle filter to address this estimation problem. The particle filter is a sequential Monte Carlo technique based on the importance sampling principle. In practice, it represents a target distribution in form of a set of weighted samples

$$p(x_t|x_{t-1}, m) \simeq \sum_i w^{(i)} \delta_{x_t^{(i)}}(x_t). \quad (24)$$

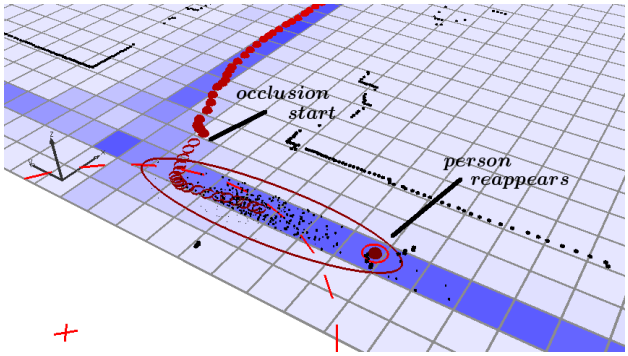


Fig. 3. Trajectory of a person in experiment 2 taking a left turn during an occlusion event. Predictions from a constant velocity motion model (dashed ellipse) and the new model (solid ellipse) are shown. The background grid (in blue) shows the learned space usage distribution of the spatial affordance map. The small black dots are the weighted samples of the place-dependent motion model. The model is able to predict the target "around the corner" yielding much better motion predictions in this type of situations.

where $\delta_{x_t^{(i)}}(x_t)$ is the impulse function centered in $x_t^{(i)}$. Sampling directly from that distribution is not possible so the algorithm first computes samples from a so called proposal distribution, π. The algorithm, then, computes the importance weight related to the i-th sample that takes into account the mismatch among the target distribution τ and the proposal distribution $w = \frac{\tau}{\pi}$. The weights are then normalized such that $\sum w = 1$.

In our case, we take the constant velocity model to derive the proposal π. The importance weights are then represented by the space usage probability

$$w^{(i)} = \frac{p(x_t|x_{t-1}, m)}{p(x_t^{(i)}|_{t-1})} = p(x_t^{(i)}|m). \tag{25}$$

The new motion model has now the form of a weighted sample set. Since we are using Kalman filters for tracking, the first two moments of this distribution is estimated by

$$\hat{\mu} = \sum_i w^{(i)} x_t^{(i)} \tag{26}$$

$$\hat{\Sigma} = \sum_i w^{(i)} (\hat{\mu} - x_t^{(i)})(\hat{\mu} - x_t^{(i)})^T. \tag{27}$$

The target is then predicted using $\hat{\mu}$ as the state prediction with associated covariance $\hat{\Sigma}$. Obviously, the last step is not needed when using particle filters for tracking.

An example situation that exemplifies how this motion model works is shown in Figure 3. A person that takes a left turn in a hallway is tracked over a lengthy occlusion event. The constant velocity motion model (dashed ellipse) predicts the target into a wall and outside the walkable area of the environment. The place-dependent model (solid ellipse) is able to follow the left turn with a state covariance in the shape of the hallway. In other words, the model predicts the target "around the corner". The tracker with the constant velocity motion loses track as the reappearing person is outside the validation gate (shown as 95% ellipses).

VI. Experiments

For the experiments we collected two data sets, one in a laboratory (experiment 1, Figure 4) and one in an office building (experiment 2, Figure 6). As sensors we used a fixed Sick laser scanner with an angular resolution of 0.5 degree.

The spatial affordance maps were trained based on the tracker described in [10], the grid cells were chosen to be 30 cm in size. The parameters of the tracker have been learned from a training data set with 28 tracks over 889 frames. All data associations including occlusions have been hand-labeled. This led to a matching probability $p_M = 0.515$, an occlusion probability $p_O = 0.472$, a deletion probability $p_D = 0.013$, a fixed Poisson rate for new tracks $\lambda_N = 0.033$ and a fixed Poisson rate for false alarms as $\lambda_F = 0.0011$. The rates have been estimated using the Bayesian approach in Eq. 12.

The implementation of our system runs in real-time on a 2.8 GHz quad-core CPU. The cycle time of a typical setting with $N_{Hyp} = 50$, 500 samples for the particle filter, and up to eight parallel tracks is around 12 Hz when sensor data are immediately available.

A. MHT with Spatial Information

The original MHT is compared to the approach using the spatial affordance map on the data set from the laboratory over 4588 frames and with a total number of 130 people entering and leaving the sensor field of view. The ground truth has been determined by manual inspection. For the comparison we count the total number of tracks that are created by the current best hypotheses of the two tracking methods. This value is indication of the tracking accuracy, especially of the ability to deal with track occlusion. We use a pruning strategy which limits the maximum number of hypotheses at every step to N_{Hyp} (the multi-parent variant of the pruning algorithm proposed by Murty [20]). In order to show the evolution of the error as a function of N_{Hyp}, the computational effort, N_{Hyp} is varied from 1 to 50. The results are shown in Figure 5.

The result shows a significant improvement of the extended MHT over the regular approach. The explanation is given by an example. As can be seen in Figure 2 right, few new track events have been observed in the center of the room. If at such a place a track occlusion occurs (e.g. from another person), hypotheses that interpret this as an obsolete track followed by a new track receive a much smaller probability through the spatial affordance map than hypotheses that assume this to be an occlusion. The fact that the green graph in Figure 5 is below the ground truth indicates that the modified approach favors track occlusions slightly too much over deletion/creation pairs. The result however demonstrates clearly that the spatial affordance map enables a tracker to better hypothesize about the state of tracks, leading to a more accurate tracking behavior.

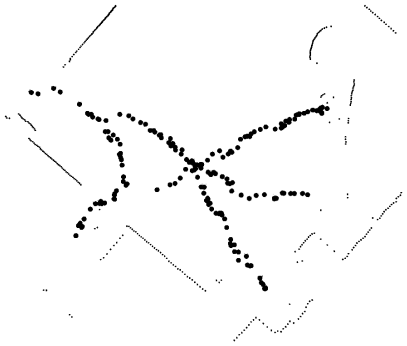


Fig. 4. Four (of 28) example tracks from experiment 1.

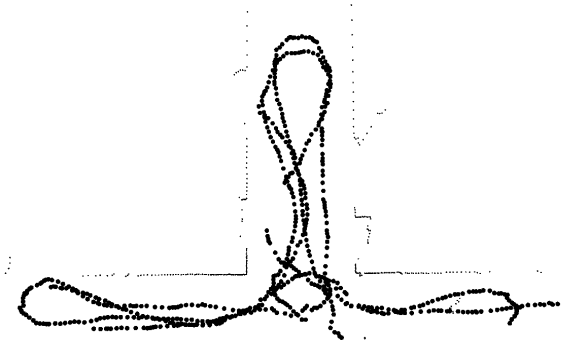


Fig. 6. Six (of 50) example tracks from experiment 2.

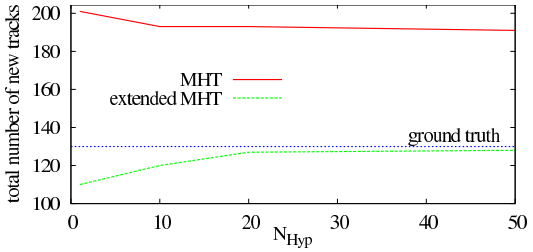


Fig. 5. The total number of tracks as a function of N_{Hyp}, the number of generated hypotheses. The tracking experiment had 4588 frames with a total of 130 people. The red line shows the MHT approach, the dotted green line the extended approach. The graph shows that replacing the fixed Poisson rates by the ones in the spatial affordance map improves the tracking accuracy significantly.

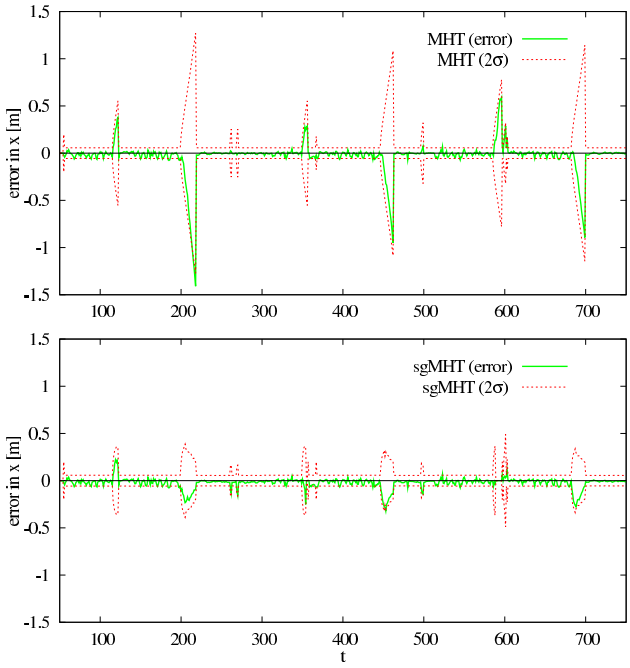


Fig. 7. Comparison between constant velocity motion model (top) and place-dependent motion model (bottom). Peaks correspond to occluded target maneuvers (turns around corners and U-turns). Fig. 3 shows the left turn of a person at step 217 of this experiment. While both approaches are largely consistent from a estimation point of view, the place-dependent model results in an overall smaller estimation error and smaller uncertainties. For eight manually inspected tracks, the constant velocity motion model lost a track three times while the new model had no track loss.

B. Place-Dependent Motion Model

In the second experiment, the constant velocity motion model is compared to our place-dependent motion model. A training set over 7443 frames with 50 person tracks in a office-like environment was recorded to learn the spatial affordance map (see Figure 6 and Figure 3). A test set with 1611 frames and eight people tracks was used to compare the two models. The data set was labeled by hand to determine both, the ground truth positions of people and the true data associations. In order to make the task more difficult, we defined areas in which target observations are ignored as if the person had been occluded by an object or another person. These areas were placed at hallway corners and U-turns where people typically maneuver. As the occlusion is simulated, the ground truth position of the targets is still available. As a measure of accuracy, the posterior position estimates of both approaches to the ground truth is calculated. The resulting estimation error in x is shown in Figure 7 (the error in y is similar).

The diagram shows much smaller estimation errors and 2σ bounds for the place-dependent motion model during target maneuvers. An important result is that the predicted covariances do not grow boundless during the occlusion events (peaks in the error plots). As illustrated in Figure 3, the shape of the covariance predictions follows the walkable area map at the very place of the target. Smaller covariances lead to lower levels of data association ambiguity, and thus, to decreased computational costs and more accurate probability distribution over pruned hypothesis trees.

For eight manually inspected tracks, the constant velocity motion model lost a track three times while the new model had no track loss. By tuning the entries of the process noise covariance Q, the constant velocity motion model can be made to avoid such losses, but this is clearly the wrong way to go as it brings along an even higher level of data association ambiguity.

VII. Conclusions

In this paper we presented an extended multi-hypothesis approach to laser-based people tracking that incorporates information on how people use space.

We proposed a non-homogeneous spatial Poisson process, called *spatial affordance map*, to represent the spatially varying distributions over track interpretation events of a MHT tracker and derive expressions for Bayesian learning of the map.

The spatial affordance map enabled us to relax and overcome the simplistic fixed Poisson rate assumption for new tracks and false alarms in the MHT approach. Using a learned spatio-temporal Poisson rate function, the system was able to compute refined probability distributions over hypotheses, resulting in a significantly more accurate tracking behavior in terms of steady track identities. The map further allowed us to derive a new, place-dependent model to predict target motion. The model showed superior performance in predicting maneuvering targets especially during lengthy occlusion events when compared to a constant velocity motion model.

In the future, we plan to extend the representation to a non-stationary Poisson process.

Acknowledgment

This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8.

References

[1] B. Kluge, C. Köhler, and E. Prassler, "Fast and robust tracking of multiple moving objects with a laser range finder," in *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, 2001.

[2] A. Fod, A. Howard, and M. Mataríc, "Laser-based people tracking," in *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, 2002.

[3] M. Kleinhagenbrock, S. Lang, J. Fritsch, F. Lömker, G. Fink, and G. Sagerer, "Person tracking with a mobile robot based on multi-modal anchoring," in *IEEE International Workshop on Robot and Human Interactive Communication (ROMAN)*, Berlin, Germany, 2002.

[4] D. Schulz, W. Burgard, D. Fox, and A. Cremers, "People tracking with a mobile robot using sample-based joint probabilistic data association filters," *International Journal of Robotics Research (IJRR)*, vol. 22, no. 2, pp. 99–116, 2003.

[5] E. Topp and H. Christensen, "Tracking for following and passing persons," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Alberta, Canada, 2005.

[6] J. Cui, H. Zha, H. Zhao, and R. Shibasaki, "Tracking multiple people using laser and vision," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Alberta, Canada, 2005.

[7] M. Mucientes and W. Burgard, "Multiple hypothesis tracking of clusters of people," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

[8] G. Taylor and L. Kleeman, "A multiple hypothesis walking person tracker with switched dynamic model," in *Proc. of the Australasian Conf. on Robotics and Automation*, Canberra, Australia, 2004.

[9] J. Cui, H. Zha, H. Zhao, and R. Shibasaki, "Laser-based interacting people tracking using multi-level observations," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China, 2006.

[10] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, 2008.

[11] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, 1979.

[12] I. J. Cox and S. L. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, vol. 18, no. 2, pp. 138–150, 1996.

[13] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, USA: YBS Publishing, 1995.

[14] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 19, no. 1, pp. 5–18, 2004.

[15] A. Bruce and G. Gordon, "Better motion prediction for people-tracking," in *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, Barcelona, Spain, 2004.

[16] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Schulz, "Voronoi tracking: Location estimation using sparse and noisy sensor data," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2003.

[17] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: a survey," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 1, pp. 103–123, Jan 1998.

[18] C. Kwok and D. Fox, "Map-based multiple model tracking of a moving object," in *RoboCup 2004: Robot Soccer World Cup VIII*, 2005, pp. 18–33.

[19] K. O. Arras, Óscar Martínez Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, Rome, Italy, 2007.

[20] K. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, 1968.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Multi-Layer People Detection using 2D Range Data

Oscar Martinez Mozos Ryo Kurazume Tsutomu Hasegawa

***Abstract*— This paper addresses the problem of detecting people using multiple layers of 2D range scans. Detecting persons is an important capacity for intelligent systems that have to interact with people. Our approach uses a supervised learning algorithm to train one classifier for each layer, which concentrates in a different body part. The classifiers are then combined in a probabilistic way to create a final robust detector. Experimental results with real data demonstrate the effectiveness of our approach to detect persons in cluttered environments, and its ability to deal with occlusions.**

I. Introduction

Detecting people is a key capacity for intelligent systems that have to interact in populated environments such as service robots [3], [23], [18], autonomous vehicles [17], [10], or ambient intelligence and surveillance systems [6], [16]. A robust detection of persons in the environment will improve the ability of these systems to communicate with people and to take decisions accordingly.

In this paper we address the problem of detecting people using 2D laser range finders. These kind of proximity sensors are often used in robotic applications since they provide a wide field of view and a high data rate. In addition, their measurements are invariant to illumination changes. Previous works have used 2D laser range finders to detect people in the environment. Typically the lasers are located at a height which permits the detection of legs [5], [8], [14], [4], [15], [18], [3], [2], [17]. Although good classifications rates have been obtained using machine learning techniques [2], [17], there is still the need to improve the robustness of the final detectors. One of the main problems is the little information that range scans provide about legs. An example is shown in the bottom right of Figure 1. Here, the legs of a person are represented by short segments composed of few points. In cluttered environments like homes or offices, these segments can be easily misclassified due to the different objects in the environment, such as tables, chairs or other furniture. Finally, occlusions often occur and make the detection of people quite difficult, or even impossible when the legs are hidden.

The key idea of this work is to improve the robustness of people detection systems by taking into account different body parts. Our approach uses 2D laser range scans situated at different heights. Each laser is responsible for detecting a different body part like the legs, the upper body or the head. The output of the different detectors is then combined in a probabilistic framework to obtain a robust final classifier. The complete system is shown in the left image of Figure 1. Our method is based on the classification of segments that represent each body part (right image of Figure 1). For each layer, a classifier is trained using a supervised learning approach based on boosting [2]. The training data for each classifier is composed of the segments that represent the body part of the corresponding layer. In the classification step, each new segment accumulates evidence for its final classification using a probabilistic voting approach [9]. In our method, the voting for a specific segment takes into account the classification of all segments in the scene.

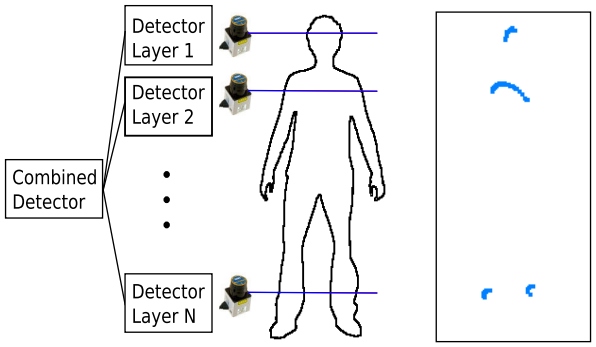


Fig. 1. The left image shows the configuration for the complete multi-layer system with 2D range scans situated at different layers. A classifier is learned for the body part found in each layer. These classifiers are then combined to create a final person detector. The right image depicts examples of segments representing body parts at three different layers: legs, upper body, and head (bird's eye view for each layer).

Experimental results shown in this paper illustrate that the resulting classification system can detect persons in cluttered environment with high recognition rates. Moreover, we present results illustrating that the multi-layer classifier improves the detection over single-layer ones. Finally, we show the robustness of the classifier under occlusions.

II. Related Work

In the past, several researchers focused on the problem of detecting/tracking people in range scans. One of the most popular approaches in this context is to extract legs by detecting moving blobs that appear as local minimum in the range image [5], [8], [4], [14], [15], [18], [4], [22]. Some of these works additionally extract some geometrical or moving features. However, these features are selected by hand. In comparison, our work learns automatically a classifier selecting the best features for the detection. In the

This work was supported by the Canon Foundation in Europe.

Oscar Martinez Mozos is with Dept. of Computer Science and System Engineering, University of Zaragoza, Spain.

Ryo Kurazume and Tsutomu Hasegawa are with Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan.
omozos@gmail.com, kurazume@is.kyushu-u.ac.jp
hasegawa@irvs.is.kyushu-u.ac.jp

work by Arras *et al.* [2], boosting is used to learn a classifier to detect legs segments. In this work we additionally learn classifiers for other body parts, and we introduce a method to combine the classifications.

The multi-part detection of people has been studied mainly in vision. Leibe *et al.* [9] use a voting approach to detect people in images with a previous learned codebook. The works from Ioffe and Forsyth [7] and Ronfard *et al.* [13] incrementally assemble body parts detected in a picture. Also Mikolajczyk *et al.* [11] use a probabilistic assembly of different body part detectors. Wu and Nevatia [21] apply a Bayesian combination of body parts detected using edgelet features. Finally, Zivkovic and Kröse [24] combine different body parts detected using Haar-like features in omnidirectional images.

Other works combine different sensors to detect people. Spinello *et al.* [17] use laser and vision sensors to detect people from a car. Also Zivkovic and Kröse [24] combine panoramic images with laser scans. In contrast to these works we use only laser range finders.

AdaBoost has been successfully used as a Boosting algorithm in different applications for object recognition. Viola and Jones [20] boost simple features based on grey level differences to create a fast face classifier using images. Treptow *et al.* [19] use the AdaBoost algorithm to track a ball without color information in the context of RoboCup. Further, Mozos *et al.* [12] apply AdaBoost to create a classifier able to recognize places in 2D maps. Our application of boosting is similar to [2], although we extended it to other body parts.

III. Single Layer Classification

This section describes the individual classifiers used in each layer. Each classifier is trained to detect a different body part of a person like the legs, the upper body or the head.

A. Boosting

To create the individual classifier $\mathcal{C}_n$ for layer n we follow the approach introduced in [2]. This method uses the supervised AdaBoost algorithm to create a final strong classifier by combining several weak classifiers. The requirement to each weak classifier is that its accuracy is better than a random guessing. In a series of rounds $t = 1, \ldots, T$, the AdaBoost algorithm selects the weak classifiers that have a small classification error in the weighted training examples. Each weak classifier h_j is based on a single-valued feature f_j and has the form

$$h_j(e) = \begin{cases} +1 & \text{if } p_j f_j(e) < p_j \theta_j \\ -1 & \text{otherwise,} \end{cases} \quad (1)$$

where θ_j is a threshold, and p_j is either $+1$ or -1 and thus represents the direction of the inequality. In each round t of the algorithm, the values for θ_j and p_j are learned so that the misclassification in the training data is minimized. The final strong classifier is a weighted combination of the best T weak classifiers. The output of the final binary classifier $\mathcal{C}_n$ has two values $\{+1, -1\}$ representing the positive and negative classification respectively. More details about this approach are given in [2].

B. Geometrical Features

In this section we describe the segmentation method and the features used in our system. Our system is equipped with several range sensors that deliver observations. The observation z from one laser sensor is composed of a set of beams $z = \{b_1, ..., b_L\}$. Each beam b_j corresponds to a tuple (ϕ_j, ρ_j), where ϕ_j is the angle of the beam relative to the sensor and ρ_j is the length of the beam. Following the approach in [2], each observation z is split into an ordered partition of segments $\mathcal{S} = \{s_1, s_2, ..., s_M\}$ using a jumping distance condition. The elements of each segment $s = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ are represented by Cartesian coordinates $\mathbf{x} = (x, y)$, where $x = \rho\cos(\phi)$ and $y = \rho\sin(\phi)$, and (ϕ, ρ) are the polar coordinates of the corresponding beam.

The set of training examples for the AdaBoost algorithm is then composed of the segments together with their label, and their pre-calculated single-valued features

$$X = \left\{(s_i, y_i, f_i) \mid l_i \in \{+1, -1\}, f_i \in \Re^d\right\},$$

where $y_i = +1$ indicates that the segment s_i is a positive example and $y_i = -1$ indicates that the segment s_i is a negative example. The set of positives examples is composed of segments that correspond to body parts of persons. The negatives examples are represented by segments that correspond to other objects in the environment. The dimension d of the feature vector f_i depends on the number of single features extracted from each segment. In our case we calculate eleven features selected from the list given in [2]: number of points, standard deviation, mean average deviation from median, width, linearity, circularity, radius, boundary length, boundary regularity, mean curvature, and mean angular difference.

IV. Multi-Layer Detection

After training the individual classifiers for each body part, our system is able to detect in each layer the segments corresponding to a person. In this section we explain how to combined the output of the different classifiers to obtain a more robust final people detector.

A. Shape Model

Based on [9], we learn a shape model of persons that specifies the geometrical relations among the different body parts. Figure 2 shows an example of a shape model for the segments corresponding to the three layers shown in the right image of Figure 1. To calculate the geometrical relations in our shape model, we first project the segments pertaining to a person into the 2D horizontal plane (bird's eye view). We then calculate the maximum distance of a segment corresponding to a concrete body part with respect to the segments corresponding to the other body parts as

$$\text{rel}(\mathcal{L}_i, \mathcal{L}_j) = \max_{\forall x \in X} \text{dist}(s_i^+, s_j^+) \mid s_i^+ \in \mathcal{L}_i, s_j^+ \in \mathcal{L}_j, \quad (2)$$

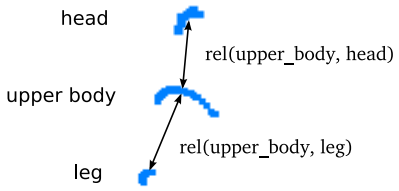


Fig. 2. This figure illustrates two examples of geometrical relations. In particular, the relations between an upper body segment with respect a head segment, and with respect a leg segment. Segments were projected to the 2D horizontal plane. The distance between the segments has been increased by hand for a better visualization.

where $\mathcal{L}_i$ indicates the layer corresponding to body part i (for example the head), and s_i^+ indicates a positive segment of that body part. Finally, $\text{dist}(,)$ is a function which calculates the Euclidean distance between the centers of two segments. These relations are learned from a set of positive training examples. The process for obtaining positive examples is explained in Section V.

Finally, for each relation we create a test function $\delta : S \times S \rightarrow \{0, 1\}$ which indicates whether two new segments s_j and s_j satisfy it

$$\delta(s_i, s_j) = \begin{cases} 1 & \text{if } \text{dist}(s_i, s_j) \leq \text{rel}(\mathcal{L}_i, \mathcal{L}_j) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

B. Probabilistic Voting

In the detection step, each range sensor delivers an observation z_j which corresponds to the scan taken at layer $\mathcal{L}_j$. This layer may correspond to the legs, upper body, head, or other body part (Figure 1). After segmenting the observations (Section III-B), each segment accumulates evidence of being a positive example of the body part corresponding to the layer it was located at.

Let s_i be a segment in the scene, and let l_i be the layer where s_i is located. Now let $c_i \in \{+1, -1\}$ be the classification of segment s_i. Following a similar approach to [9], we calculate the score for a positive classification $c_i = +1$ of segment s_i by marginalizing over all segments found in the scene

$$V(c_i^+) = \sum_j P(c_i^+, s_j) \quad (4)$$

$$= \sum_j P(c_i^+ \mid s_j) P(s_j). \quad (5)$$

Here c_i^+ is equivalent to $c_i = +1$. The first term in (5) represents the probability of a positive classification for segment s_i given all segments found in the scene. We further marginalize over the classification of all segments

$$P(c_i^+ \mid s_j) = \sum_{c_j} P(c_i^+, c_j \mid s_j) \quad (6)$$

$$= \sum_{c_j} P(c_i^+ \mid c_j, s_j) P(c_j \mid s_j). \quad (7)$$

In our system, the are two possible values for a segment classification $c_j \in \{+1, -1\}$. These values indicate whether the segment s_i corresponds to a person $c_j = +1$ or not $c_j = -1$. Instantiating the variable c_j in (7) we obtain

$$P(c_i^+ \mid s_j) = \quad (8)$$
$$P(c_i^+ \mid c_j^+, s_j) P(c_j^+ \mid s_j) + P(c_i^+ \mid c_j^-, s_j) P(c_j^- \mid s_j).$$

Here c_j^- is equivalent to $c_j = -1$. Substituting in (5), we get the final expression for the score of a positive classification $V(c_i^+)$ as

$$\sum_j \Big(\begin{array}{l} P(c_i^+ \mid c_j^+, s_j) P(c_j^+ \mid s_j) \\ + P(c_i^+ \mid c_j^-, s_j) P(c_j^- \mid s_j) \end{array} \Big) \cdot P(s_j). \quad (9)$$

It remains to explain how to calculate each term in (9). The term $P(c_j^+ \mid s_j)$ indicates the probability of a positive classification of segment s_j. This value can be obtained directly from the output of the classifier $\mathcal{C}_{l_j}$ at the layer l_j where s_j was found

$$P(c_j^+ \mid s_j) = \begin{cases} 1 & \text{if } \mathcal{C}_{l_j}(s_j) = +1 \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

Thus, the probability for a negative classification is obtained as

$$P(c_j^- \mid s_j) = 1 - P(c_j^+ \mid s_j). \quad (11)$$

The term $P(c_i^+ \mid c_j^+, s_j)$ indicates the probability of a positive classification for segment s_i given there is another segment s_j in the scene which corresponds to a person, i.e., $c_j = +1$. This value is obtained using the test function of the shape model (Section IV-A)

$$P(c_i^+ \mid c_j^+, s_j) = \delta(s_i, s_j). \quad (12)$$

Finally we need to obtain a value for expression $P(c_i^+ \mid c_j^-, s_j)$, which indicates the probability for a positive classification of segment s_i given there is another segment in the scene which corresponds to other object. We call this expression the *occlusion model*, since it indicates the relation of the people with other objects in the scene. In this work, we apply the following model

$$P(c_i^+ \mid c_j^-, s_j) = \begin{cases} \theta & \text{if } \delta(s_i, s_j) = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

This expression indicates that whenever we find a segment in the scene corresponding to an object other than a person, this object can not fulfill the shape model of a person.

C. Person Detection

After accumulating evidences for all segments found in all layers, we have a distribution of probabilistic votes among the different hypotheses c_i. To detect a person in the environment, we look for the hypothesis c_p^+ which maximum positive score

$$c_p^+ = \underset{c_i^+}{\text{argmax}}\, V(c_i^+). \quad (14)$$

The segment s_p corresponding to c_p^+ is then selected as the representative for the person in the scene. To detect several persons one can look for different local maximum in the hypotheses space. In our experiments we try to detect one person only, and for this reason we apply (14) for selecting the final hypothesis that represents the person.

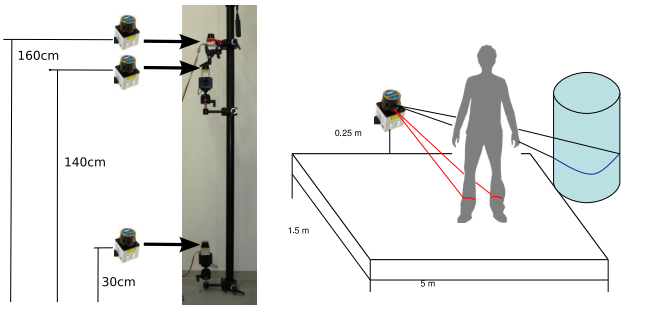


Fig. 3. The left image shows the 3-layer system used in the experiments. Each laser is located at a different height to detect a different body part: head ($160cm$), upper body ($140cm$), legs ($30cm$). The right image depicts the process for obtaining positive training data. A free space ($5m \times 1.5m$) is left in front of the lasers. A person walks inside this space and the corresponding segments are automatically labeled as positive examples. The segments falling outside the rectangle are automatically labeled as negative examples

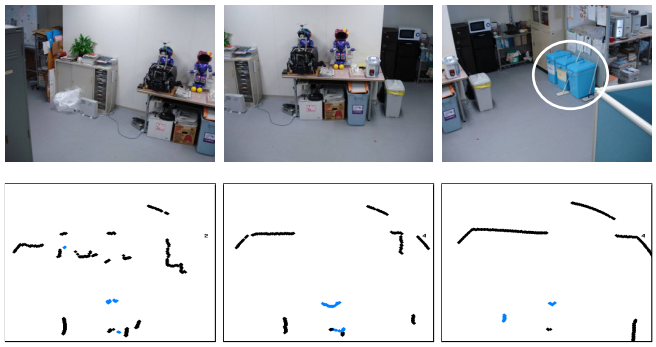


Fig. 4. First scenario for the experiments. The top pictures were taken from the position were the sensors were located. The blue rubbish in the right image (marked with a white circle) are used for the occlusion experiments. The bottom images show examples of scans taken at the different layers. The left image corresponds to the lower layer (legs), the middle image to the middle layer (upper body), and the right image to the top layer (head). Blue points indicate segments classified as positive (body parts) . Black points correspond to segments classified as negative (non body parts).

V. Experiments

The approach presented above was implemented using a three layer system as shown in Figure 1. At each layer, we located a URG-04LX laser range finder with a field of view of 240 degree. The resolution of the lasers was of 0.36 degree. Each laser is situated at a different height and detects a different body part. The upper laser is located $160cm$ above the floor. This laser is thought to detect heads. The middle one is located $140cm$ above the floor. This laser detects upper bodies. The final one is located $30cm$ above the floor, and its task is to detect legs. The complete system is shown in the left image of Figure 3. The experiments were carried out in the Laboratory for Intelligent Robots and Vision Systems at the University of Kyushu in Japan. The sensors were kept stationary during the experiments.

We first explain how to obtain a training set for the learned step. We then demonstrate how a multi-layer classifier can be learned in an indoor environment to detect people. In addition we show the robustness of this classifier under occlusions and in very cluttered environments. Finally, we show the improvements of the detection rates when using our multi-layer detector in comparison to a single-layer system.

One important parameter of the AdaBoost algorithm is the number of weak classifiers T used to form each final strong classifier. We performed several experiments with different values for T and we found that $T = 200$ weak classifiers provide the best trade-off between the error rate of the classifier and the computational cost of the algorithm. Another parameter that has to be set for the occlusion model is θ. In our experiments we found that a value of 0.05 gives good results under occlusion situations. Finally, we selected a jump distance of $15cm$ for segmenting the scans.

A. Training Data

The first step in the experiments was to train the classifiers for each layer. As explained in Section III, we used the supervised algorithm AdaBoost to create each classifier. The input to the algorithm is composed of positive and negative examples. The set of positive examples contains segments corresponding to the different body parts: legs, upper body, and head. The set of negative examples is composed of segments corresponding to other objects in the environment such as tables, chairs, walls, etc. We used the same training algorithm for the three layers, with the only difference being the training data used as input.

To obtain the positive and negative examples we left a free space of $5m \times 1.5m$ in front of the lasers. This space did not contain furniture or other objects. We then started recording laser scans while a person was walking randomly inside the rectangle. The obtained scans were segmented following the approach in Section III-B. The segments were then automatically labeled as positive examples of a body part if they were inside the rectangle, and as negative examples if they fell outside the rectangle. This process is shown in the right image in Figure 3. This is a straightforward method to obtain training data without the need of hand-labeling.

B. Multi-Layer Classification

In the the following experiments we tested our multi-layer approach in an indoor environment. We first obtained the training data following the procedure explained above. The data was obtained in a location of the laboratory shown in the top images of Figure 4. The training data was composed of 344 multi-layer observations containing 17286 segments. Examples of training scans are shown in the bottom images of Figure 4.

In a first experiment, the same person walked in front of the lasers following different trajectories from the training data. In this way we obtained a different test set. We then applied our multi-layer detector to this test. An example of observation with its corresponding detection is shown in Figure 5. The results of the detections are shown in the $Test$ row of Table I. The detection rate of 92% indicates that we can use our method to detect people with high accuracy in indoor environments.

In a second experiment we tested the performance of our method with partially occluded bodies. In this experiment,

TABLE I

MULTI-LAYER DETECTION RATES

	True detection	False detection	Total observations
Test	**92.0%** (149)	8.0% (13)	162
Occlusion	**85.8%** (272)	14.2% (45)	317
Hard	**75.2%** (161)	24.8 % (53)	214

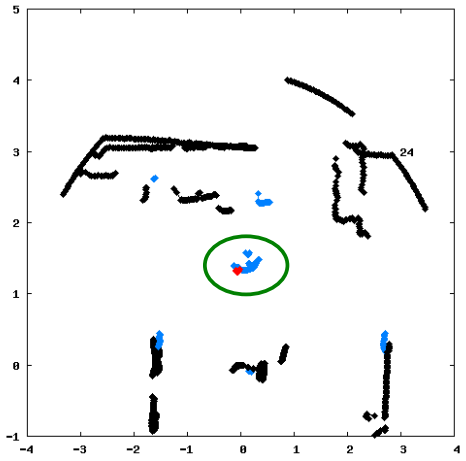


Fig. 5. The image shows an example of a detection for the experiment called $Test$ in Table I. Different colors indicate different classifications. Blue segments are classified as body parts, the red segment is the one with best evidence of been a person. Black segments are classified as other objects. The segments corresponding to the person (ground truth) are marked with a green ellipse. The lasers are located at $(0, 0)$.

a person walked in front of the lasers and, at same point in time, he took two rubbish bins and put them in front of the lasers. The bins are shown in the top right image of Figure 4. Following, the person walked around them, and finally put the bins back in their initial position. In this situation several occlusion problems appear. First, while the person was walking around the bins his legs remained occluded. Second, while the person was bending down to take/leave the bins his upper body and his head disappeared.

We applied our detector to this sequence of observations and obtained the results shown in the $Occlusion$ row in table Table I. The false positives often occurred when the person was in contact with the bins, taking them, moving them or leaving them. In these situations it was difficult to detect all body parts. However, a detection rate of 85.8% indicates that we still can use our approach to detect partially occluded persons. An example observation taken while the person was behind a bin is shown in Figure 6.

In a third experiment, we tested the performance of our learned multi-layer detector in a new and very cluttered environment. Figure 7 shows images of this third scenario. In this experiment a person walked around and the obtained observations where classified. Results of the detections are shown in the $Hard$ row of Table I. The detection rate decreased to 75.2, however we think this is still a good result for such an extremely challenging scenario. Figure 8 shows a snapshot of this experiment. Videos for the three experiments are available in [1].

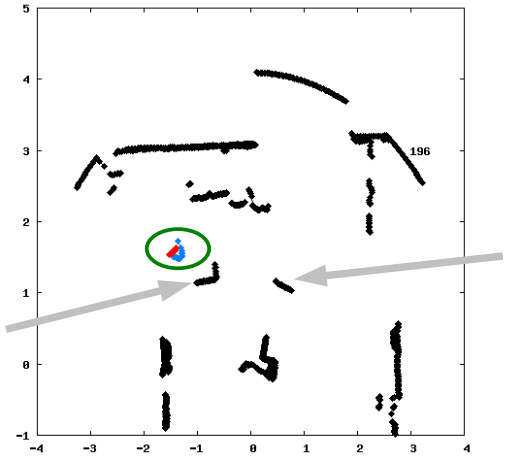


Fig. 6. The image shows an example of a detection for the experiment called $Occlusion$ in Table I. The meaning of the colors are the same as in Figure 5. The position of the bins are pointed with light grey arrows. The person is behind one of the bins with his legs occluded. The lasers are located at $(0, 0)$.

Fig. 7. These images correspond to part of the Laboratory for Intelligent Robots and Vision Systems which is used for experiments. As we can see the location is very cluttered. This scenario is called $Hard$ in Table I.

C. Comparison with Single-Layer Detection

In these experiments we analyze the improvement of our multi-layer system in comparison to a single-layer detector. To do this, we apply our probabilistic model (Section IV-B) in the layer corresponding to the legs. We repeat the detection in the three scenarios from the previous section: $Test$, $Occlusion$, and $Hard$. Results are shown in Table II. For the $Test$ experiment the results are quite similar, since there are no occlusions and the legs are correctly detected. However, we can see the improvement of our method in the experiment $Occlusion$, in which the multi-layer obtains a detection rate of 85.8% in comparison to 73.2% obtained with the single-layer. Finally, in the $Hard$ scenario the single-layer obtained a detection rate of 41.1%, while our multi-layer approach got a rate of 75.2%. This is a very important improvement.

D. Individual Classification Rates

In this last experiment we compare the classification rates for the different layers. In this experiment we used the test set from the $Test$ experiment, and analyzed the performance of each layer when classifying segments. Results are summarized in Table III. We can appreciate that the classification rate for the legs 94.3% is higher than the classification for the other levels. One reason for this is that the person has two legs, and thus we obtain double number of positive training examples. In the upper levels (upper body and head) the

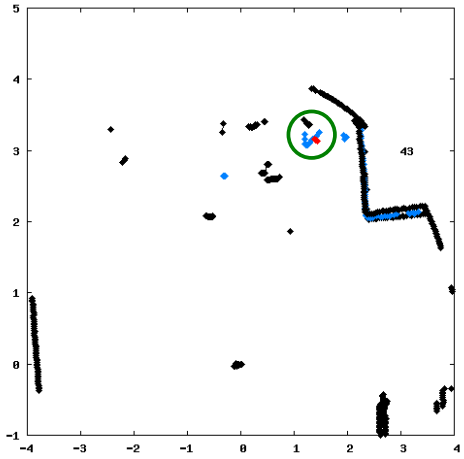


Fig. 8. The image shows an example of a detection for the experiment called $Hard$ in Table I. The meaning of the colors are the same as in Figure 5. The lasers are located at $(0, 0)$.

TABLE II

SINGLE-LAYER DETECTION RATES

	True detection	False detection	Total observations
Test	**92.6%** (150)	7.4% (12)	162
Occlusion	**73.2%** (232)	26.8% (85)	317
Hard	**41.1%** (88)	58.9% (126)	214

classifications decrease to 84%-86%. The classification rates for these body parts are a novelty in this paper.

TABLE III

CONFUSION MATRICES FOR SINGLE LAYERS

		Classification	
	True Label	Person	Not Person
Legs	Person	**94.3%**	5.7%
	No Person	7.8%	**92.2%**
Upper body	Person	**84.4%**	15.6%
	No Person	11.2 %	**88.8%**
Head	Person	**86.2%**	13.8% (26)
	No Person	12.5%	**87.5%**

VI. CONCLUSION

This paper presented a novel approach for people detection using multiple layers of 2D range scans. Each laser is responsible for detecting a different body part of a person like the legs, the upper body or the head. For each body part, we learned a classifier using Boosting. The output of the different classifiers was combined in a probabilistic framework to obtain a more robust final classifier. In practical experiments carried out in different environments we obtained encouraging detection rates even in very cluttered ones. Finally, the comparison of our multi-layer method with a single-layer procedure clearly demonstrated the improvement obtained when detecting people using different body parts simultaneously.

REFERENCES

[1] http://www.informatik.uni-freiburg.de/~omartine/publications/mozos2009iros.html.

[2] K.O. Arras, O.M. Mozos, and W. Burgard. Using boosted features for the detection of people in 2D range data. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3402–3407, 2007.

[3] M. Bennewitz, W. Burgard, and S. Thrun. Learning motion patterns of persons for mobile service robots. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.

[4] J. Cui, H. Zha, H. Zhao, and R. Shibasaki. Tracking multiple people using laser and vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, 2005.

[5] A. Fod, A. Howard, and M.J. Mataric. Laser-based people tracking. In *Proceedings of the IEEE International Conference on Robotics & Automation (ICRA)*, 2002.

[6] G.L. Foresti, L. Micheloni, C. Snidaro, and P Remagnino. *Ambient intelligence: a novel paradigm*, chapter Security and building intelligence: from people detection to action analysis, pages 199–212. Springer, 2005.

[7] S. Ioffe and D.A. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision*, 43(1):45–68, 2001.

[8] M. Kleinhagenbrock, S. Lang, J. Fritsch, F. Lömker, G.A. Fink, and G. Sagerer. Person tracking with a mobile robot based on multi-modal anchoring. In *IEEE International Workshop on Robot and Human Interactive Communication (ROMAN)*, Berlin, Germany, 2002.

[9] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1-3):259–289, 2008.

[10] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 30(10):1683–1698, 2008.

[11] K. Mikolajczyk, C. Schmid, and A. Zisserman. *Computer Vision - ECCV 2004*, chapter Human Detection Based on a Probabilistic Assembly of Robust Part Detectors, pages 69–82. Lecture Notes in Computer Science. Springer-Verlag, 2004.

[12] O.M. Mozos, C. Stachniss, and W. Burgard. Supervised learning of places from range data using AdaBoost. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1742–1747, Barcelona, Spain, April 2005.

[13] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *European Conference of computer Vision*, 2002.

[14] M. Scheutz, J. McRaven, and G. Cserey. Fast, reliable, adaptive, bimodal people tracking for indoor environments. In *IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[15] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *International Journal of Robotics Research (IJRR)*, 22(2):99–116, 2003.

[16] L. Snidaro, C. Micheloni, and C. Chiavedale. Video security for ambient intelligence. *IEEE Transactions on Systems, Man and Cybernetics*, 35(1):133–144, Jan. 2005.

[17] L. Spinello and R. Siegwart. Human detection using multimodal and multidimensional features. In *Proc. of The International Conference in Robotics and Automation (ICRA)*, 2008.

[18] E.A. Topp and H.I. Christensen. Tracking for following and passing persons. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005.

[19] André Treptow and Andreas Zell. Real-time object tracking for soccer-robots without color information. *Robotics and Autonomous Systems*, 48(1):41–48, 2004.

[20] P. Viola and M.J. Jones. Robust real-time object detection. In *Proc. of IEEE Workshop on Statistical and Theories of Computer Vision*, 2001.

[21] Bo Wu and Ram Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *Int. J. Comput. Vision*, 75(2):247–266, 2007.

[22] J. Xavier, M. Pacheco, D. Castro, and A. Ruano. Fast line, arc/circle and leg detection from laser scan data in a player driver. In *Proc. of the IEEE Int. Conference on Robotics & Automation (ICRA'05)*, 2005.

[23] H. Zender, O.M. Mozos, P. Jensfelt, G-J.M. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502, June 2008.

[24] Z. Zivkovic and B. Krose. Part based people detection using 2d range data and images. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 214–219, 2007.

Poster Presentations

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Visual Receding Horizon Estimation for human presence detection

Damien Brulin
Institut PRISME/MCDS UPRES EA 4229
Ecole Nationale Supérieure d'Ingénieurs de Bourges
88, bd. Lahitolle 18020 BOURGES Cedex, France
Email: damien.brulin@ensi-bourges.fr

Estelle Courtial and Guillaume Allibert
Institut PRISME/MCDS UPRES EA 4229
Université d'Orléans - Polytech'Orléans
8, rue Léonard de Vinci, 45072 Orléans Cedex 2, France
Email: estelle.courtial@univ-orleans.fr
guillaume.allibert@univ-orleans.fr

***Abstract*—This paper deals with human presence detection by using a receding horizon estimator based on computer vision results. The visual position estimation problem is formulated into a nonlinear constrained optimization problem in the image plane. A global model combining the behavior of the human motion and the camera model is used to estimate the evolution of the visual features on a past finite horizon. The main interest of this method is the capability to easily take into account constraints. Experimentations in two different configurations highlight the efficiency of the proposed approach, especially with an image occlusion treated as a visual constraint.**

***Index Terms*—Receding Horizon Estimation, Human detection, Computer Vision**

I. Introduction

The management of energy consumption (electricity, heating...), the improvement of the autonomy of the elderly or also the automation of lighting systems are all issues of our society against the rising of energy prices, the ageing of population and environmental concerns. Besides, both industries and individuals wish to better manage their energy consumption thanks to a maximum control of the house or building equipments, like lighting or heating. This is why Domotic science has been developed to offer solutions to supervise and organize a global system that ensures comfort and security. Among the applications stem from Domotic, human presence detection holds an important place.

Several systems, from the state of the art, already offer solution more or less reliable, depending on their use and the monitored phenomenon. Passive InfraRed Detector (PIR) or hyper-frequency sensor can be cited [1][2]. All these systems are called "presence" detectors but in the majority of cases, they are simply movement detectors. The goal of CAPTHOM project is to really detect the presence of a human in an indoor environment by using a multi-sensors system. The final objective is to develop an application which will be able to detect a situation of emergency. The first stage is then to estimate correctly the position on the ground of the human target.

Among the different kinds of sensors, vision seems to be well-adapted to give information about the human presence in a given environment [3]. As an image is a rich source of information, an algorithm should be used to extract the relevant data. In [4], the authors have developed an algorithm which can detect the presence of a human in a scene and can give information about its approximate position. The position estimation of a moving object by using visual information in real time has been largely investigated in the computer vision literature [5][6][9]. However, because visual measurements are usually affected by significant noise and disturbances, for example due to lens distortion, the estimation of the position and orientation could be a difficult task. To enhance the estimation, the extended Kalman Filter (EKF) is usually chosen because it offers many advantages, e.g., accuracy of estimation, prediction capability, temporal filtering [7][8]. The limits of EKF are the conditions that have to be satisfied in order to obtain good results. Adaptative EKF has been proposed in [10] for visual applications. However, difficulties like occlusion or obstacle avoidance, which can be considered as visual constraints, can not be taken into account with EKF based approaches.

The aim of this article is thus to propose a method based on Receding Horizon Estimation (RHE) to realize the estimation of the human target's position in the image. The position estimation problem is transformed into a nonlinear optimization problem. A global model combining the camera model and the human motion model is used to estimate the visual features over a past finite horizon. The optimization algorithm minimizes the error between the features measured thanks to the computer vision algorithm and the features estimated by the RHE. The estimation horizon moves one step forward at each sampling instant and the procedure is repeated. The main advantage of RHE is the capability to easily handle constraints contrary to EKF. When an occlusion appears, the computer vision algorithm gives a false position of the target. The proposed method can bypass this problem either by considering the occlusion as a visual constraint or by considering the largest admissible movement of the human as a state constraint. The comparison between the position given by computer vision and the position estimated by the visual receding horizon estimator is then used to detect an emergency situation.

The paper is organized as follows. In section 2, the issue of human presence detection is introduced. Difficulties due to presence detection or due to the computer vision algorithms are pointed out. In section 3, the principle of the receding horizon estimation is briefly recalled. Then, the proposed approach, called Visual Receding Horizon Estimation (VRHE), is detailed. Finally, section 4 presents experimental results in two different configurations: a first one illustrating the method without constraints and a second one showing the efficiency of the method in case of occlusion.

II. The issue of human presence detection

The current devices, for example PIR or ultrasound sensors, allow to detect the movement of a person in a room but not really its presence. If the target stops and does not move, it becomes invisible for the system and the latter answers that there is nobody in the room. The goal is to always be able to know if there is someone or not in the given environment. However, human presence detection sets out some difficulties.

A. Problems due to presence detection

The first problem which can appear is the detection of non-human targets. The system should be able to differentiate a detection brought out by the movement of an animal and the detection due to the presence of a human being. It exists two ways to solve this differentiation problem. The first approach is a technological one, simply by making a sensor positioning that does not detect movement of small entities. The second one is rather software. It consists of registering excluded areas of the scene or by defining threshold values.

Another problem of presence detection is the presence of several persons in the same room. Furthermore, a room can have more than one exit. So, one has to be able to manage the possibility that a person can enter in the room by one door and leave by a different door. Finally, because the presence of several people can happen, the system must differentiate two or more human targets and must adapt its behavior.

With computer vision algorithm, we can find solutions to deal with these problems. In [4], the authors have proposed a real time human detection based on visual information. Firstly, in order to reduce the search space of the classifier, they perform a background substraction to detect change. The program draws a box including the detected target. Then the algorithm tries to find if the detected target is a human or not. The classification between human and non-human being is done with machine learning tools. Furthermore, each box has its own identifying number. So we can also bypass the problem of multi-presence in a same environment and track each target independently.

In our application, we need to extract the coordinates of the box in order to estimate the human position in the image. However, even if the use of vision offers solutions faced with human presence detection difficulties, it possesses its own drawbacks.

B. Problems due to the computer vision algorithm

Although visual sensor gives a lot of information, many difficulties appear during its use. We will not do an exhaustive list of these difficulties but we will raise the main problems encountered during the development of our application.

In order to detect a change in images, computer vision programs store in memory a model which serves as background. Each image is then compared with the background model to detect a difference. This background is regularly updated. However, if a lightning change happens in the environment, the program will be disrupted by this sudden noise and it will not return good results.

Limitations concerning the camera placement also exist. To use a camera, we need to calibrate it in order to obtain the transformation matrix which allows to calculate the coordinates in the image reference of a point, knowing its coordinates in the environment reference. Once the computation of the matrix is done, the camera does not have to move because the coefficients of the matrix are linked to the camera position.

Another problem, that can have an impact on our position estimation, is the occlusion. If the person walks behind an obstacle, the person is partially masked and so the computer vision algorithm could encounter some difficulties to decide if this target is a human or not. In all environment, there are several obstacles like table, chair or just a box that can disturb the visual acquisition. A last problem, concerning the difficulty of human recognition, could happen if the person falls. The majority of computer vision algorithms use a database composed of images with humans standing up. So, if a person lays down or falls, it will not be recognized as a human being.

Our approach will try to propose a solution, faced with these problems, by combining a receding horizon estimation approach with visual information in order to estimate the position of a human in the image.

III. Visual Receding Horizon Estimation

A. Receding Horizon Estimation

The estimation of the position and orientation of a moving object has been largely investigated in the literature for the past few years. The estimation of the pose of the target is often required in position-based visual servoing approaches. Kalman filtering, especially the Extended Kalman Filter (EKF), offers a satisfactory rejection of disturbance or noise and an accurate estimation. In [10], the authors proposed an adaptative version of EKF for visual applications. However, the EKF may encounter difficulties for practical implementations, when state constraints have to be handled and when the process is highly nonlinear [11]. In order to

overcome these problems, a receding horizon estimation (RHE) can be used. The strategy of the RHE is to formulate the constrained state estimation into an online nonlinear optimization problem. The constraints can easily be added to the optimization problem [12].

We propose to extend the receding horizon estimation to visual estimation.

B. Visual Receding Horizon Estimation (VRHE)

The estimation problem of the human position is formulated into a nonlinear optimization problem in the image plane over a past receding horizon N_e. The difference between the measured features in the image plane denoted y_{imag} and the estimated features denoted y_{mod} defines the cost function J. The estimated features are obtained by using a global model combining the human motion model and the camera model. The cost function is to be minimized with respect to the human position $\underline{p} = (x_h, y_h)$ at time $k - N_e$. The position estimation at the current time k is computed thanks to the human motion model and $\underline{p}_{k-N_e}$.
At each sampling time, the past finite estimation horizon moves one step forward and the procedure is then repeated to ensure the robustness of the approach in regard to disturbances and model mismatches.

The cost function can be written in discrete-time as:

$$J(p) = \sum_{j=k-N_e}^{k} [y_{imag}(j) - y_{mod}(j)]^T Q\ [y_{imag}(j) - y_{mod}(j)] \tag{1}$$

Q is a symmetric definite positive matrix. The mathematical formulation of the Visual RHE is then given by:

$$\min_{\underline{p}_{k-N_e}} \quad J(p) \tag{2}$$

subject to the nonlinear global model describing the dynamics :

$$\begin{cases} p(k+1) = f(p(k), \Delta u(k)) \\ y_{mod}(k) = h(p(k)) \end{cases} \tag{3}$$

The Figure 1 shows the scheme of the VRHE.
One of the main advantages of VRHE is the capability to explicitly take into account constraints, contrary to EKF. Numerous constrained optimization routines are available in software libraries. A drawback of the RHE strategy is the computational time required for the resolution of the nonlinear constrained optimization problem. However this computational burden is not a strong limitation for real time application due to the increase of PC power.

C. Global overview of our method

The receding horizon estimation algorithm represents the keystone of our approach as we can see in the Figure 2.

The first step consists in positioning and calibrating the camera so as to compute the model of the camera. Moreover, the transformation matrix is required for the change from image to environment reference. In [13], the authors use a planar homography matrix to compute the position of a flame front on the ground in the world reference from its position in the image.

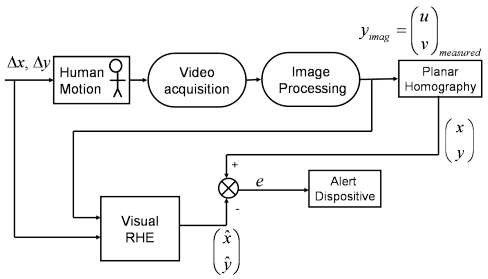


Figure 1. Scheme of the VRHE

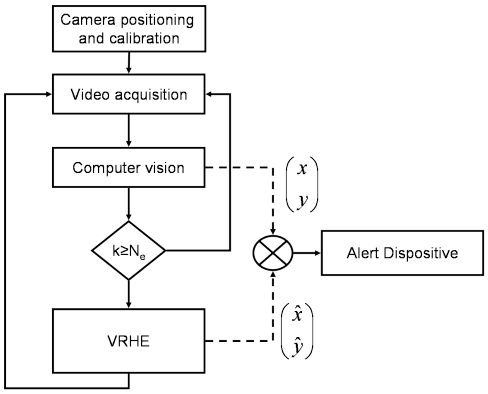


Figure 2. Overview of the VRHE

The computer vision algorithm gives the measure of the coordinates (u and v) of the middle point of the box's bottom side. Thanks to the homography matrix, we calculate the coordinates of feet on the ground (x and y) and also, for each step, the distance with the previous step (Δx and Δy). Once we have enough measures, it means when we have reached the size of the estimation horizon N_e, we can run the VRHE procedure.

IV. Experimental Results

The feasibility and the performance of the proposed visual position estimation algorithm have been experimentally tested using a single camera.

A. The camera and the computer vision algorithm

The camera has been calibrated by using a least square method. The resolution of the camera is 640 x 480 pixels but in the display result of the computer vision algorithm, the image size is reduced to 320 x 240 pixels. The sample time used is the minimum time allowed by the camera frame rate, T_e=0.07s. To conclude with camera's characteristics, it was placed at a height of 1.98m. The scene viewed by the camera is illustrated in Figure 3.

To compute the homography matrix, we used a reference, which can be seen in Figure 3, measuring 0.92 x 0.59 m. This

Figure 3. Camera's view of the scene and image reference

matrix permits to compute the position on the ground from the position in the image. We need a third matrix of transformation because the environment reference used for the homography and the environment reference used for the calibration is not the same. We have computed the transition matrix between these two references. In brief, the three different transition matrices are :

$$M_{intr} = \begin{pmatrix} 328.17 & 0 & 170.88 \\ 0 & -327.89 & 103.51 \\ 0 & 0 & 1 \end{pmatrix}$$

$$M_{hom} = \begin{pmatrix} 0.0065 & 0.0035 & -1.4892 \\ -0.0003 & 0.0175 & -3.7361 \\ 0.0003 & 0.0033 & -0.4043 \end{pmatrix}$$

$$M_{trans} = \begin{pmatrix} 1 & 0 & 0 & -1.02 \\ 0 & 0 & 1 & 1.98 \\ 0 & -1 & 0 & 5.11 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where M_{intr}, M_{hom} and M_{trans} are respectively the intrinsic parameter matrix, the homography matrix and the transition matrix between environment reference of homography and environment reference of calibration. The Figure 4 shows the two different environment references, one used for the homography matrix and the other one used for calibration.

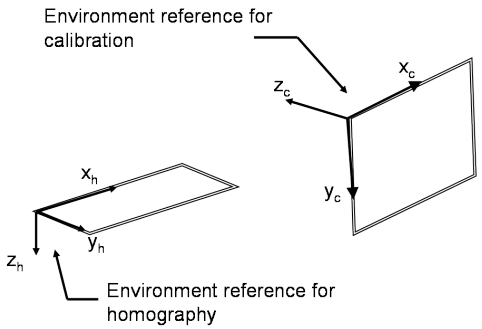


Figure 4. Environment references

Due to the knowledge of the three transition matrices, the model camera can be written as :

$$y_{mod} = \begin{pmatrix} u_{mod} \\ v_{mod} \end{pmatrix} = \begin{pmatrix} \alpha_u \frac{X_c}{Z_c} + u_0 \\ \alpha_v \frac{Y_c}{Z_c} + v_0 \end{pmatrix} \quad (4)$$

$$and \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = M_{trans} \begin{pmatrix} x_h \\ y_h \\ z_h \\ 1 \end{pmatrix} \quad (5)$$

We suppose that the person walks on a flat environment (z_h is constant).

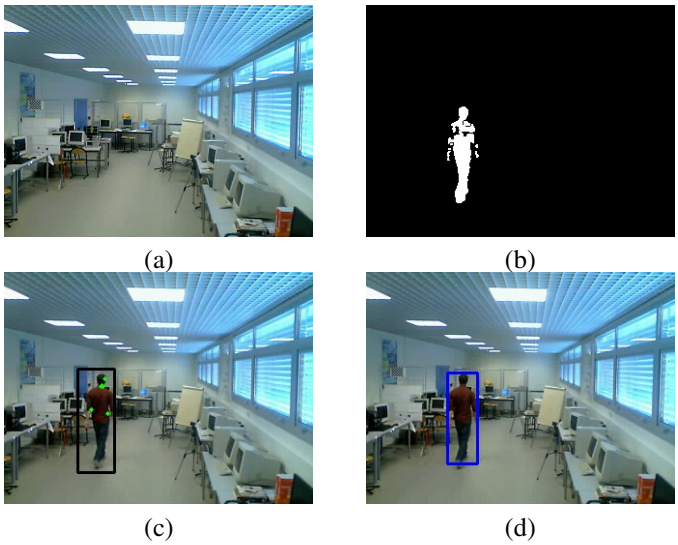


Figure 5. Representation of the different steps of the computer vision algorithm

The principle of the computer vision algorithm, used in our application is depicted in the Figure 5. At the beginning, the program needs to take some pictures of a clear scene in order to have a fixed background (a). Then, each picture is compared with the background.

In [4], the authors first detect changes by computing the Mahalanobis' distance between pixels of the current image and the background model(b), composed of the mean of the three RGB components and of the co-occurrence matrix. This first step is done in order to reduce the search space of the classifier. Then, the preceding detected objects are observed by using a tracking of point of interest(c). The final step consists in determining the nature of the tracked object(d). Authors built a cascade of boosted classifiers based on Haar-like filters and on a boosting method to discriminate humans and non humans entities. We can then define the measured features :

$$y_{imag} = \begin{pmatrix} u_{mes} \\ v_{mes} \end{pmatrix} \quad (6)$$

coordinates of the middle of the box's bottom side.

Two different cases are considered. The first aims at illustrating the feasibility of our method with videos where there is no occlusion of the human. The second aims at studying the capability of the proposed method to deal with large path variations due to occlusion. For each case, several videos have been tested to vary movements in the observed scene. No occlusion model is used. The occlusion is treated as a visual constraint in the estimation procedure.

B. The model of the human movement

Human motion model is necessary to estimate the position of a human. However, it is difficult to model and describe precisely the movement of a man. Indeed, one can not predict where the target will be at the next step because it does not follow precise rules. The motion of a human can be described by nonholonomic model [14]. To prove the feasibility of our method, we have just chosen a simple model of the human motion. By observing the human displacements on video, we have observed that the movement can be modeled by a single integrator.

With a first order discretization, the model of human motion can be written as :

$$\begin{cases} x_h(k+1) = x_h(k) + \Delta x \\ y_h(k+1) = y_h(k) + \Delta y \end{cases} \quad (7)$$

where $\Delta x = x_h(k) - x_h(k-1), \Delta y = y_h(k) - y_h(k-1)$ are the displacements respectively in x and y.

The global model is then composed of the camera model (4), the transition matrix (5) and the human motion model (7).

C. Simulations without any constraints

For all experimentations, the size of the estimation horizon is fixed to 5 ($N_e = 5$), the matrix Q is the identity matrix. The VRHE algorithm has been implemented in Matlab software and the computer vision algorithm in Visual C++.

In this case study, scenarios have been realized in a room without any occlusion possibilities. The aim was to verify the feasibility of our method with a simple case. The human just goes to the far end of the room, stays in position during a short time and goes back. The Figures 6 and 7 illustrate respectively the estimation of the position with VRHE according to u and v in the image reference. For both figures, the dashed line represents the estimation, result of VRHE, and the solid line indicates the measures obtained by the computer vision algorithm.

The first five points are at zero because the VRHE begins to run as soon as the program has reached the estimation horizon N_e and has sufficient information. As we can see, the estimates are closed to the measures in both directions.

Same results have been obtained with several videos and proved the feasibility of the proposed approach.

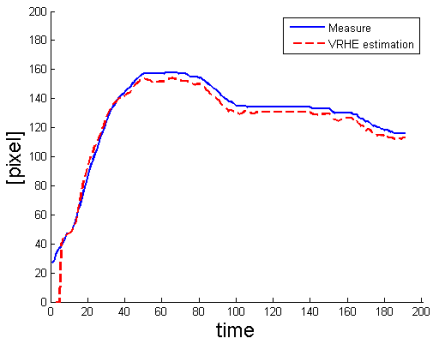


Figure 6. Time history of the measured trajectory and the estimated trajectory according to u

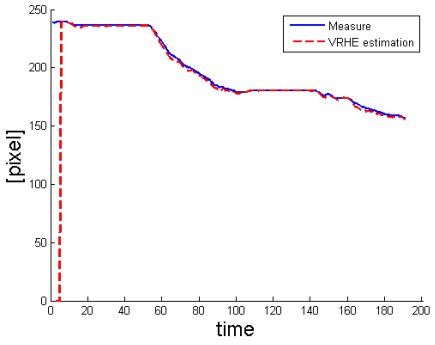


Figure 7. Time history of the measured trajectory and the estimated trajectory according to v

D. Simulations with occlusion

If the person walks behind an obstacle, a problem of occlusion will appear and the size of the box determined by the computer vision algorithm will suddenly change, leading to deviant measures. The Figure 8 illustrates the problem of occlusion and shows how the including box dimension changes when the target is behind an obstacle. In (a), we can see a representation of the scene viewed by the camera. Before the person walks behind the obstacle, the box correctly includes the person (b). When the person is partially masked by the obstacle (c), we clearly see that the box is two times smaller than the previous one. Once the person is no longer hidden by the obstacle, the box recovers its original size (d). We can remark that the human reflect on the window has also been detected by the computer vision algorithm. It is one of the practical difficulties.

The Figures 9 and 10 represent respectively the position estimation according to u and v axis. We clearly see, in the Figure 10, the two moments when the person has been hidden by the obstacle. During these two events, the position estimation according to v axis does not follow the measure. A constraint on the admissible displacement of the human has been taken into consideration. The displacement computed from computer vision is aberrant. So, based on the past movements, a constrained admissible displacement has been

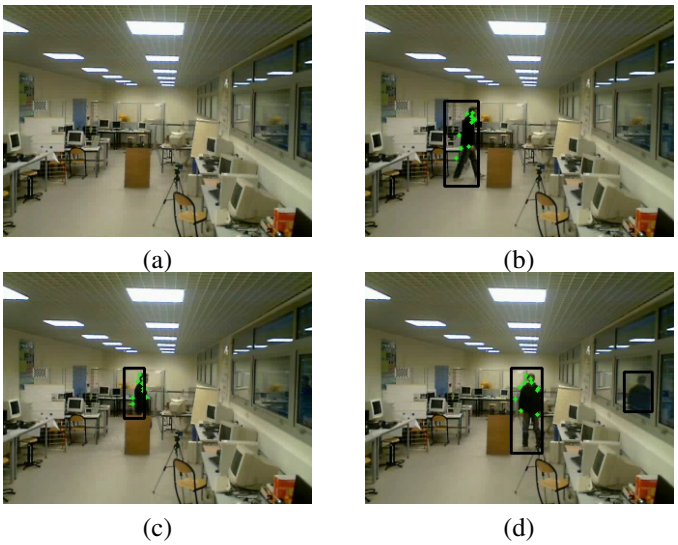


Figure 8. Representation of the problem of occlusion

applied to the global model, especially to the human motion model. Another strategy is to determine, by image processing, obstacle dimensions and to treat it as a visual constraint in RHE. However, this last approach needs more computational time.

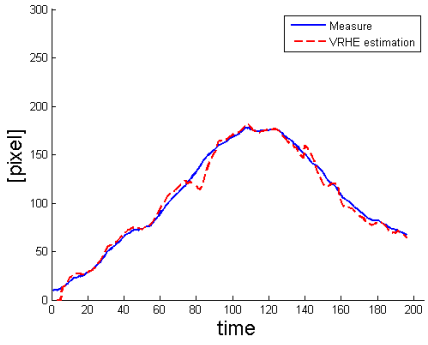


Figure 9. Time history of the measured trajectory and the estimated trajectory according to u

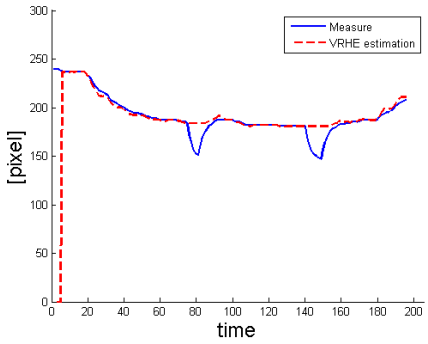


Figure 10. Time history of the measured trajectory and the estimated trajectory according to v

V. Conclusion

In this paper, a method for the visual estimation of the position of a moving human has been proposed. The approach is based on the extension of RHE principle to visual estimation. The main advantage is its capability to take into account constraints. The minimization of the cost function is performed in the image plane. The visual estimates are obtained by the knowledge of a global model combining the human motion model and the camera one. The experimental results confirm the feasibility and the efficiency of the proposed approach. A first approach to avoid problem of deviant measures due to occlusion has been presented. In future works, the residual generation, error between the measures and the estimates, will be used to generate an alert signal.

VI. Acknowledgement

We especially thank all our partners involved in the CAPTHOM project. This work was realized with the financial help of the French Industry Ministry and local collectivities, within the framework of the CAPTHOM project of the Competitiveness Pole S^2E^2, www.s2e2.fr.

References

[1] I. Masri, T. Boudet and A. Guillot, *Hyperfrequency detection method and detector using said method*, patent nb. EP1818684, August 2007.

[2] C.F. Tsai and M.S. Young, *Pyroelectric infrared sensor-based thermometer for monitoring indoor objects*, Review of Scientific Instruments 74 (12), pp. 5267-5273, 2003,doi:10.1063/1.1626005.

[3] N.A. Ogale, *A Survey Of Techniques For Human Detection From Video*, scholarly papers for the Master's of Science degree in Computer Science of the University of Maryland, http://www.cs.umd.edu/Grad/scholarlypapers/papers/neetiPaper.pdf.

[4] Y. Benezeth, B. Emile, H. Laurent and C. Rosenberger, *A Real Time Human Detection System Based on Far Infrared Vision*, ICISP 2008, Elmoataz et al. (Eds.), LNCS 5099, Springer, pp. 76-84, 2008.

[5] T. Broida and R. Chellappa, *Estimation of object motion parameters from noisy images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1, pp. 90-99, 1986.

[6] C. Harris, *Tracking with rigid models*, in A. Blake and A. Yuille (Eds.), Active vision, Cambridge, MA: MIT Press, pp. 57-73, 1992.

[7] S. Lee and Y. Kay, *An accurate estimation of 3-D position and orientation of a moving object for robot stereo vision: Kalman filter approach*, proceedings of 1990 IEEE international conference on robotics and automation, pp. 414-419, 1990.

[8] J. Wang and W.J. Wilson, *3D relative position and orientation estimation using Kalman filter for robot control*, proceedings of 1992 IEEE international conference on robotics and automation, pp. 2638-2645, 1992.

[9] B. Leibe, K. Schindler and L. Van Gool, *Coupled Detection and Trajectory Estimation for Multi-Object Tracking*, Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on Volume , Issue , 14-21, pp. 1-8.

[10] V. Lippiello, B. Siciliano and L. Villani, *Visual motion estimation of 3D objects: an adaptive extended Kalman filter approach*, Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Volume 1, pp. 957-962, 2004.

[11] E.L. Haseltine and J.B. Rawlings, *Critical Evaluation of Extended Kalman Filtering and Moving-Horizon Estimation*, Industrial and Engineering Chemistry Research, 44 (8), pp. 2451-2460, 2005.

[12] C.V. Rao, J.B. Rawlings and D.Q. Mayne, *Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations*, IEEE Trans. Auto. Cont., 48(2):246-258, February 2003.

[13] E. Pastor et al., *Computing the rate of spread of linear flame fronts by thermal image processing*, Fire safety Journal 41, pp. 569-579, 2006.

[14] G. Arechavaleta, J.P. Laumond, H. Hicheur and A. Berthoz, *The nonholonomic nature of human locomotion : a modeling study*. In 1st IEEE/RAS-EMBS Int. Conf. on Biomedical Robotics and Biomechatronics. Pisa, Italy, 2006.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Multiple People Detection from a Mobile Robot using Double Layered Laser Range Finders

Alexander Carballo, Akihisa Ohya and Shin'ichi Yuta

***Abstract*— This work presents our method for people detection on the surroundings of a mobile robot by using two layers of multiple LRFs, allowing to simultaneously detect two set of different features for every person: chest and legs areas. A person model is created according to the association of these features and a volume representation allows to estimate the current person position. We present experimental results of multiple people detection in an indoor environment. The main problem of our research the development of a mobile robot acting as member of a group of people, simple but accurate people detection and tracking is an important requirement.**

I. INTRODUCTION

Companion robots are becoming more part of daily life and are designed to directly interact with people. One necessary subsystem for such robots is detection, recognition and tracking of people as well as obstacles in the environment.

Laser Range Finders (LRF), besides being used for obstacle detection are also an important part of people tracking systems. The Tour-Guide robots Rhino and Minerva by Burgard *et al*[1] and Thrun *et al*[2] featured LRFs for people detection and collision avoidance. LRF present important advantages over other sensing devices like high accuracy, wide view angles, high scanning rates, etc., and are becoming more accessible and safer (meaning class 1 lasers) for usage in human environments.

Most approaches based on LRFs ([3], [4], [5], [6], [7], [8]) place the sensors in the same height (single row or scan plane) to detect and track some feature of the human body. Due to laser safety regulations, applications using non class-1 lasers are mostly limited to a low position, mostly about knee height or below. Thus legs are widely used as features for human detection and tracking.

In Fod *et al* [3] a row of several LRFs on different positions in a room were used for tracking moving objects, future positions are estimated according to a motion model. Montemerlo *et al* [4] also uses LRF from a mobile robot for people tracking and simultaneously robot localization by using conditional particle filters. Xavier *et al* [5] focused on people detection using a fast method for line/arc detection but from a fixed position. Zhao *et al* [6] proposed a walking model to improve position prediction by including information about leg position, velocity and state. The later model was then used by Lee *et al* [7] and by Zhao *et al* [8] but this time from a mobile robot.

Intelligent Robot Laboratory, Graduate School of Systems and Information Engineering, University of Tsukuba, 1-1-1 Tennoudai, Tsukuba City Ibaraki Pref., 305-8573, Japan, +81-29-853-6168. {acs,ohya,yuta}*@roboken.esys.tsukuba.ac.jp*

A common problem is how to correctly identify people features from laser measurements. Arras *et al* [9] using range data and Zivkovic *et al* [10] using range data and images, employ a learning method, particularly *AdaBoosting*, to determine which properties and in what amounts to consider to improve detection. However, detection of multiple people in cluttered environments is difficult especially considering occlusion cases of people walking side by side.

Most tracking applications can deal with temporal occlusions due to obstacles, such as the temporal disappearance of the legs behind a dust bin. Multiple target tracking in cluttered environments including crossings tracks is part of most current works [11], [12], [13]. Mucientes *et al* [11] extends the problem of single person tracking by considering clusters of tracks (people) using Multiple Hypothesis Tracking (MHT). Arras *et al* [12] also uses MHT for tracking without a leg swinging-motion model but introducing an occlusion state, low level tracks (legs) are associated to a high level track (people). Kondaxakis *et al*[13] present also a multi-target approach using JPDA with a grid map where occupancy counters of each cell decrease with time to identify background objects.

One limitation still present in those systems is occlusion of the tracked body feature for an extended time, for example if the person stopped behind the dust bin. MHT based systems will delete of the occluded track if it is missing for more than some maximum time. The usage of additional features can overcome this problem, provided that they are separated over some distance (height) where occlusion stops. Instead of a single layer system one can consider a multi-layered arrangement of class-1 LRFs on a mobile platform. Multiple features have the additional benefit of complementarity for detection and tracking: a person can be described by the union of a set of small swinging segments at low height (legs), a bigger segment at medium height (waist) and a larger segment at a high position (chest). This idea was proposed in our previous work [14]. A multi-layered system to extract multiple features is of course possible as long as the person height is over some minimum value.

A multi-layered system has also being considered previously [15], [16]. Gidel *et al*[15] used a 4-layer laser sensor for pedestrian detection, scanning planes are not parallel so that slight tilting of the vehicle do not affect detection. However, the vertical distance between features on the target pedestrian depend on the distance from the sensor. Hashimoto *et al*[16] use 3 LRFs around a wheelchair for 360^o scanning at 3 different heights, each sensor with its own processing computer performing detection and tracking.

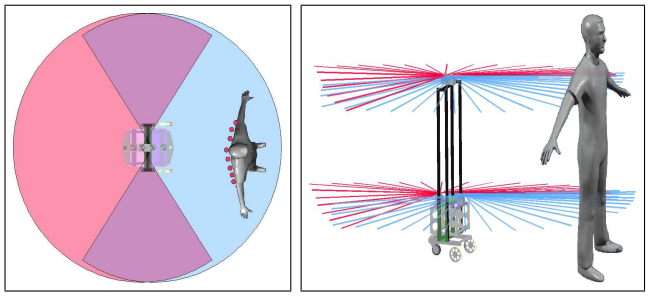
For every sensor, scan data is mapped into a occupancy grid map, then target tracking and tracks association is performed. Tracking in overlapping areas is done by cooperation of respective computers and covariance intersection.

Our approach is then similar to Hashimoto's[16]: we have sensors are arranged in two parallel planes for 360^o scanning, separated at different heights from the ground depending on the features to detect. However, we perform all computing in a single computer, sensors in the same layer are fused to combine their individual readings and then layers are also fused for people detection.

The rest of the paper is organized as follows. In section II present an overview of our current system. Section III presents our approach for fusion of multiple sensor layers, including feature extraction, people detection and position estimation. Section IV presents experimental results for the different fusion steps and for people detection. Finally, conclusions and future work are left for section V.

II. System Overview

Fig. 1 represents our layered approach, every layer has two sensors facing opposite directions for 360^o scanning (Fig. 1(a)), and two layers are used to extract features from upper and lower parts of a person's body (Fig. 1(b)).



(a) (b)

Fig. 1: Scanning from a double layered approach: (a) opposite facing sensors (top view) and (b) two layers of sensors (lateral view).

The processing pipeline of our system is best understood referring to Fig. 2. Our people detection approach (before tracking) involves four steps: fusion of sensors, segmentation, feature extraction and layer fusion. The outputs for some of the steps are depicted as inlets in the figure: Fig. 2(a) is the result of fusion of sensors (the top layer represented in red and the lower in green), Fig. 2(b) corresponds to geometrical feature extraction (features of people is shown), and in Fig. 2(c) the detected people around the robot.

Our method involves two fusion steps: fusion of sensors in a single layer and then fusion of layers. In the first step, sensors facing opposite directions in the same layer are fused to produce a 360^o representation of robot's surroundings. There is overlapping of scan data from both sensors (darker areas in Fig. 1(a)) so this fusion step must deal with data duplication. Then, in the multiple layer fusion step, raw data from every layer is processed to extract features

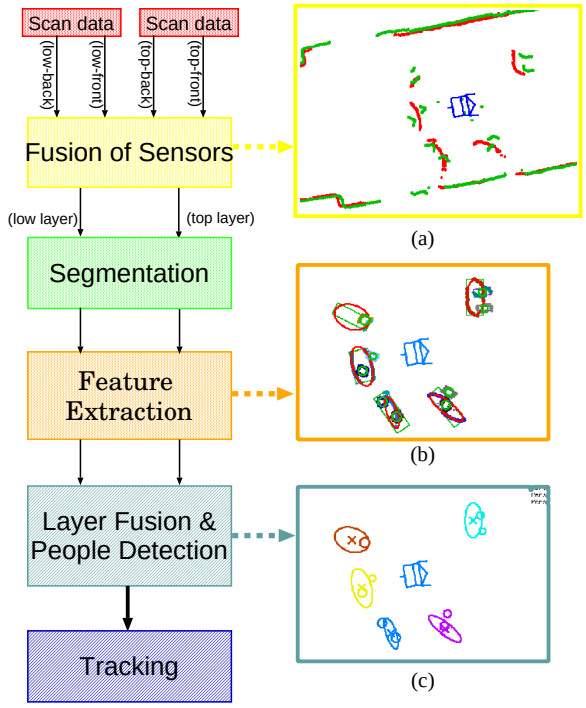


Fig. 2: System overview.

corresponding to people, then a people model is computed and from it allowing people detection and person position and direction estimation.

After fusion of sensors in every layer, geometrical features are extracted: large elliptical shapes corresponding to chest areas and smaller circular shapes for legs. Fusion of extracted features allows creating a cylindrical volume and from it the estimated person position is computed. A simple yet logical assumption here is that an elliptical shape corresponding to a chest is always associated to one or two circular shapes corresponding to legs (if no occlusions due to clutter are considered), and that the large elliptical shape (chest) is *always over* the set of small circles (legs). Fig. 3 illustrates this concept, here we present a sequence of continuous scan images from a person walking (as seen from above), both upper layer (large arc-like shape, chest) and lower layer (small arc-like shapes, legs) are visible.

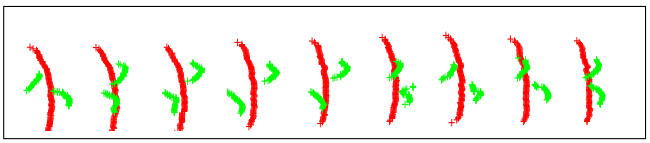
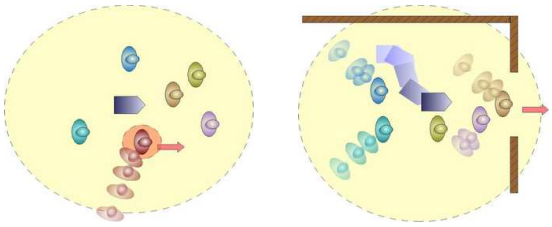


Fig. 3: A sequence of walking steps using actual scan data using sensors from both upper layer (darker points on large curve) and lower layer (smaller curves).

Our main research goal aims to develop a companion robot with the objective to study the relationship of an autonomous mobile robot and a group of multiple people in a complex environment like public areas, where the robot is to move and behave as another member of the group, while

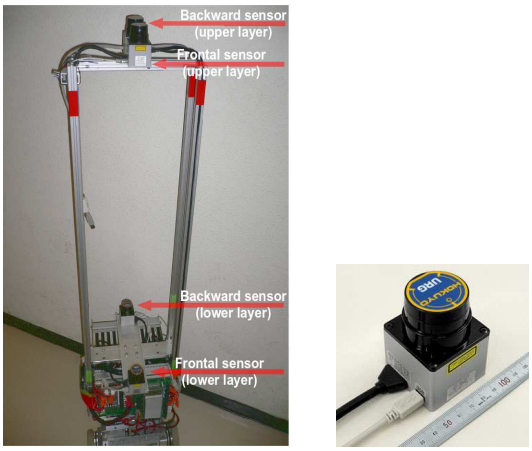
achieving navigation with obstacle avoidance. Some of the basic functions of such companion robot are depicted in Fig. 4, while the robot acts as another group member it has to detect, recognize and track the fellow human members (Fig. 4(a)) and also move in the environment like the rest of the members do (Fig. 4(b)).



(a) (b)

Fig. 4: Companion Robot with a group of people: group members recognition (a) and obstacle avoidance (b).

The robot used for our research is depicted in Fig. 5. The robot (Fig. 5(a)) is based on *Yamabico* robotic platform [17]. Two layers of LRF sensors are used, the lower layer is about 40*cm* from the ground while the upper layer is about 120*cm*. Every layer consists of 2 LRF sensors, one facing forwards and another facing backwards for a 360^o coverage (Fig. 1 and 5). The sensors used in our system are the *URG-04LX* laser range scanners (Fig. 5(b), [18] provides a good description of the sensor's capabilities).



(a) (b)

Fig. 5: Our robot system for multiple people detection and tracking (a), four *URG-04LX* are used (b).

III. Fusion of double layered LRF sensors

Sensors in the same layer are facing opposite directions, individual scan data are combined into a 360^o representation. The next step is fusion of both sensor layers, here data will be divided into clusters with a segmentation function and then clusters will be classified according to their geometrical properties. Finally only those segments that match people features will be selected and joined into a 3D model from where people position is obtained.

A. Segmentation

Data clustering can be considered as the problem of break-point detection and finding breaking points in scan data can be considered as the problem of finding a threshold function $\mathscr{T}$ to measure separation of adjacent points. Every pair of neighboring points p_j and p_k are separated by an angle α which is proportional to the sensor's angular resolution (true for points of two adjacent scan steps) and by a distance $\mathscr{D}(p_j, p_k)$. Points are circularly ordered according to the scanning step of the sensor.

A cluster $\mathscr{C}_i$, where $\mathscr{C}_i = \{p_i, p_{i+1}, p_{i+2}, \cdots, p_m\}$, is defined according to a cluster membership function $\mathscr{M}$

$$\mathscr{M}(p_j, p_k) = (\theta_k - \theta_j) \leq \alpha \wedge \mathscr{D}(p_j, p_k) \leq \mathscr{T}(p_j, p_k) \quad (1)$$

such that for every pair $\langle p_j, p_k \rangle$ of adjacent points, the Euclidean distance $\mathscr{D}(p_j, p_k)$ between them is less than a given threshold function $\mathscr{T}(p_j, p_k)$ for p_j, p_k. A new point p_n is compared to the last known member p_m of a given cluster $\mathscr{C}_i$ as $\mathscr{M}(p_m, p_n)$.

Now, the threshold function $\mathscr{T}$ is defined for a pair of points, as in the work of Dietmayer [19], as:

$$\mathscr{T}(p_i, p_j) = C_0 + C_1 min(r_i, r_j) \quad (2)$$

with $C_1 = \sqrt{2(1 - cos(\alpha)}$. Dietmayer's work includes the constant C_0 to adjust the function to noise and overlapping. In our case C_0 is reemplaced by the radius $\mathscr{R}$ of the accuracy area for p_i as base point plus a fixed threshold value (10*cm* in our case). $\mathscr{R}$ is defined according to the *URG-04LX* sensor specifications [18], [20] as:

$$\mathscr{R}(p_i) = \begin{cases} 10 & \text{if } 20mm \leq r_i \leq 1000mm \\ 0.01 \times r_i & \text{otherwise} \end{cases} \quad (3)$$

The proposed threshold function $\mathscr{T}$ uses this accuracy information $\mathscr{R}$ when checking for break points, if two neighboring points have a large range value, it will be most probable that they form part of the same cluster for their bigger accuracy areas.

There is also a cluster filtering step that will drop segments very small to be considered of significance.

B. Feature Extraction

The idea of *feature extraction* is to match the sensor readings with one or more geometrical models representing expected behaviour of the data. For example if a LRF sensor data scanning a wall, then the *expected behaviour* of *a wall* scan data is *a straight line*. Also if the same sensor is to scan *a person* then the expected behaviour is a set of points forming *an arc*. So in order to identify walls a first requirement is to correctly associate the scan data with some straight line model, for people the same: associate a set of scan points to an arc shape (a circle or an ellipse).

Before applying any fitting method, it is important to have some information about the shape of the cluster that allows selecting the method. The information about clusters is extracted as a set of indicators like number of points, standard deviation, distances from previous and to next clusters, cluster curvature, etc.

One of the indicators is the cluster's *linearity*; our approach here is to classify the clusters into *long-and-thin* and those rather *short-and-thick*. The rationale behind this is that, straight line segments tend to be long and thin, round obstacles, irregular objects, etc., do not have this appearance.

Linearity is achieved by computing the covariance matrix Σ for the cluster $\mathscr{C}_i$ and then its eigenvalues λ_{max} and λ_{min} that define the scale and its eigenvectors $\mathtt{v}_1$ and $\mathtt{v}_2$ orientation (major and minor axes) of the dispersion of $\mathscr{C}$. The ratio $\ell = \lambda_{max}/\lambda_{min}$ defines the degree of longness/thinness of the cluster. We set threshold values for ratio $\mathscr{L}$ and for λ_{max}.

The *ellipticality* factor ε is computed as the standard deviation σ of the residuals of a ellipse fitting processes using the Fitzgibbon method [21]. The distance between a cluster point and an ellipse is computed using *Ramanujan's* approximation.

Only clusters with good ellipticality value are selected and segments passing the linearity criteria (that is lines) can be easily rejected since they do not belong to people.

We assign a weight value $\mathtt{w}$ to every indicator i and compute an scoring function $\mathbb{S}$ for every segment j in in layer Ψ, where $\Psi \in \{top, low\}$, as:

$$\mathbb{S}^j = \sum_i^n \mathtt{w}_i^\Psi \mathscr{H}_i^\Psi(I_i^j) \tag{4}$$

where $\mathscr{H}_i^\Psi : \mathbb{R} \to \{-1, 1\}$ is a binary classifier function for the i-th indicator which compares whether the given indicator is under some threshold value. Table I presents an example of indicators and their classifiers, the actual list of indicators is similar to that presented by Arras *et al* in [9]. Weight values $\mathtt{w}_i$ and thresholds for every indicator i were defined after experimental validation.

TABLE I: Example of indicators and their classifiers

Indicator	Classifier	Meaning
width w	$w \leq W_{max}^\Psi$	a leg or a chest has a width no bigger than the threshold
linearity ℓ	$\ell \leq \ell_{max}^\Psi$	leg and chest features are not linear
curvature k	$k \geq k_{min}^\Psi$	leg and chest features are curved
ellipticality ε	$\varepsilon \leq \varepsilon_{max}^\Psi$	the fitting error of ellipse for chest under the threshold

C. People Model and Position Detection

3D projection of two planes of scan data from the layered sensors can be used to represent the position and direction of a person. The set of geometrical features extracted from the former step are mostly ellipses and circles. If they belong to a person another important criteria should be meet: the large elliptical segment should come from the upper layer and the small circles from the lower layer. No large ellipses are possible for a person in the leg area. The small circles can not be over the large ellipse (the person height is restricted according to the height of the upper layer).

To properly establish the previous requirements, it is necessary to associate segments in the upper layer with those in the lower layer, this is to find the corresponding legs for a given chest. Latt *et al.* [22] present a study about how human motion, step length, walking speed, etc. are selected to optimize stability. Their study present data about different speeds people prefer when walking. If the average values of step length are used then it is possible to define the limits of motion of the legs with respect to the projected chest elliptical area. Figure 6 helps understanding this idea. The average leg height h is about $84cm$, and the height of the lower layer of sensors l is fixed at $40cm$. s is the step length which depends on the speed, for example $73cm$ for an average speed of $1.2m/s$ [22]. d is calculated as:

$$d = 2(H-l)tan(\theta), \text{ where } \theta = sin^{-1}(\frac{s}{2h}). \tag{5}$$

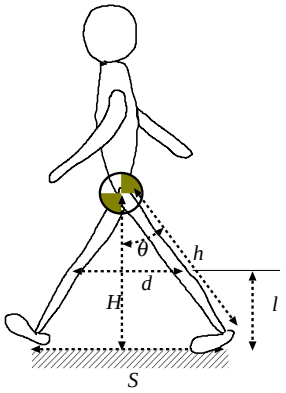


Fig. 6: Simple representation of human step to compute the distance d between leg segments while walking.

According to [22] the step lengths for three different walking speeds are presented in Table II. In this table we include the parameter d from Fig. 6 about the distance between leg segments when walking at the different speeds.

TABLE II: Step length according to speed and distance between leg segments d

Mode	Speed[a]	Step Length[b]	Distance between leg segments d [c]
normal	$1.2 \pm 0.04m/s$	$73.0 \pm 3cm$	$34.40cm$
very slow	$0.5 \pm 0.05m/s$	$47.0 \pm 3cm$	$22.29cm$
very fast	$2.1 \pm 0.1m/s$	$86.0 \pm 6cm$	$39.64cm$

[a,b] Values according to Latt *et al.* [22].
[c] estimated from Eq. 5.

With an estimation of the maximum value for d, the separation of legs at the lower layer height, we can set a search radius of $\frac{d}{2} \pm \xi$ at the center of the chest elliptical area projected into the lower layer to search for the corresponding legs for the chest. We use average walking step length from Latt *et al.* [22], at normal walking speed, to compute the value for d.

IV. Experimental Results

The robot used for our research was presented in Fig. 5, the computer operating the robot is a Intel Pentium Core Duo based notebook running (Linux kernel 2.6.24) as

operating system and robot control board is powered by a Hitachi SH-2 processor. The robot system uses *4 URG-04LX* range scanners from *Hokuyo Automatic Co., Ltd.*[20], small size (50*x*50*x*70mm), covers distances up to 5.6*m*, distance resolution of 10*mm* and angular resolution of 0.36^o, angular range of 240^o operating at 10*Hz*. Scan data from each sensor consists of 682 points circularly ordered according to scanning step.

Data from each sensor is read every 100*ms* by a driver processes and registered in parallel into a shared memory system (*SSM*[23]) based on IPC messaging and multiple ring-buffers with automatic timestamping, one driver process per sensor. SSM also allows to record raw sensor data into log files and to play it back with the same rate as the sensor (10*Hz* in this case).

Client processes read scan data from the ring-buffers according to sensor's pose (those in the top layer and those on the low layer), pairs of LRF sensors are processed in the fusion step, sensor layers are further fused and finally people position is computed. The processing time for the two layers (4 sensors), from single layer fusion to people position detection, was below 40*ms*, fast enough given the sensor's scanning speed.

We performed an experiment for people detection and position estimation from a mobile robot. In the experiment, 5 persons walked around the robot and additional person was taking the experiment video. Log data from each sensor was recorded, people position detection tests were performed off-line by playing back this log data using our SSM system. Fig. 7 corresponds to the group of people surrounding the robot.

Fig. 7: An experiment for multiple people position estimation using the proposed method.

Fig. 8 shows results of LRF data segmentation and feature extraction: raw data from each layer (top layer in Fig. 8(a)) is divided into clusters (Fig. 8(b)) and each cluster's indicators analyzed to extract those segments with human-like features and average sizes (Fig. 8(c)).

In this figure, arrows in Fig. 8(a) represent the location of people in the environment, most of them were successfully detected in the results of feature extraction (8(c)). However one of them has a height below the standard so top-level sensors were actually scanning his neck area, accordingly his chess ellipse is smaller than the allowed values, therefore was rejected. Another interesting case is the segment marked as "column" in Fig. 8(a), although its curvature and linearity indicators classify it as person, the boundary length and segment width were far bigger than the allowed values, reducing its scoring and marking it for rejection.

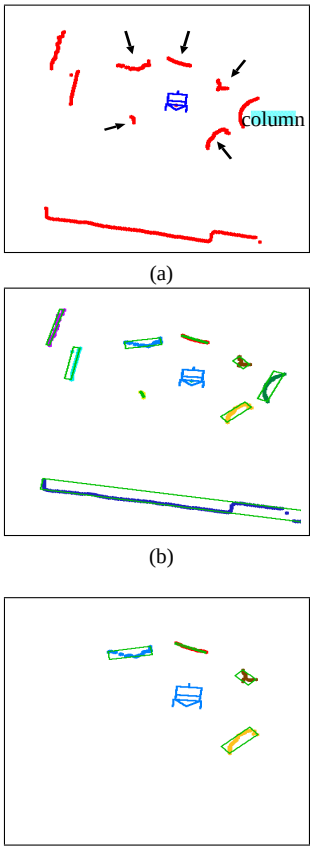


Fig. 8: Results of LRF data segmentation and feature extraction: raw data ((a)) is segmented ((b)) and then classified ((c)).

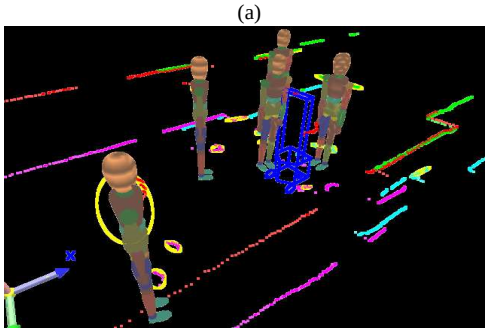
Fig. 9 shows the results of an experiment for people detection and position estimation from a mobile robot. In the experiment, 5 persons walked around the robot and additional person was taking the experiment video (Fig. 9(a)). Log data from each sensor was recorded, people position detection tests were performed off-line by playing back this log data using our SSM system.

A 3D tool was created to visualize inspect how the people detection worked; in Fig. 9(b) chest ellipses and leg circular ellipses are detected then we place a 3D wooden doll, as a representation of a person, in the estimated position the person should have. Results were verified by human operator comparing the experiment video with results.

The members have varied body sizes, from broad and tall to thin and short. Some of the members have a height a little under the average, as result their chest ellipses were not correctly detected in the people detection step. As presented in Fig. 9(b), the person to the right of the robot (represented with blue line segments) is missing although circles from legs are present.

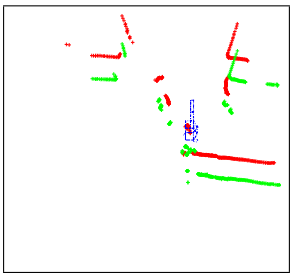
Additional snapshots of experimental results are presented

(a)

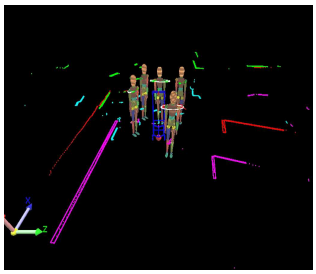


(b)

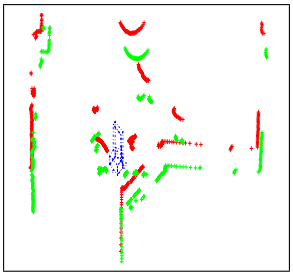
Fig. 9: Results of the people detection, (a) snapshot (b) 3D models in estimated positions of people in the experiment.



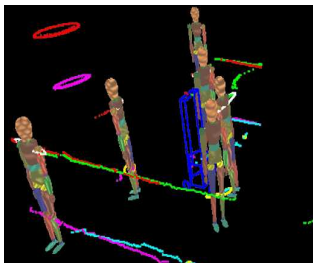
(a)



(b)



(c)



(d)

Fig. 10: Experimental results with raw scan data ((a) and (c)) and the corresponding people detection and position estimation ((b) and (d)).

in Fig. 10, the robot is represented in all cases as blue line segments. Fig. 10(a) and 10(c) shows raw scan data from both layers (red for the upper layer and green for the lower one), and in Fig. 10(b) and 10(d) a 3D representation of the human detection and position estimation. In the cases of 3D representation, the raw scan data is plotted together with wooden dolls enclosed in the estimated people positions represented with elliptical shapes, a large one for the chest area and smaller ones for the extracted leg areas.

In Fig. 10(c) there are two rather large arc-like segments in the raw scan image and two large elliptical shapes in the 3D representation in Fig. 10(d), in both layers. That is the column inside the indoor environment, as already explained in Fig. 8(a). The people detection method discards this elliptical object because its dimensions are larger than the expected for people, those elliptical objects are represented with red color in this figure. Also we do not expect large elliptical objects from the lower layer so discarding this column as a non human object was simple.

V. Conclusions and Future Works

The problem of multiple people position detection by fusion of multiple LRF sensors arranged in a double layer structure was presented in this paper. Instead of using different sensors of complementary capabilities, we used the same type but at different heights (layers), this gives a different perspective which also helps solving simple cases of occlusion where one sensor is occluded and the other is not.

The addition of an extra layer of LRFs to detect chest elliptical areas improve the estimation of people position as the lower part of body (the legs) move faster and wider than the chest area. The combination of both areas creates a 3D volume which helps locating the position of the person more closely related to the center of this 3D volume and as a measure of the possible direction the person is facing. Although research exists in the area of detection and tracking, the proposed approach is simple and fast enough to be used for real time detection of people in robot's surroundings.

As future work, multiple people tracking will be considered. Also the effectiveness of our method in cluttered environments will be studied. Future steps of our research include understanding people group motion and recognition of group members.

References

[1] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "The interactive museum tour-guide robot," in *Fifteenth National Conference on Artificial Intelligence*, (Madison, WI), pp. 11–18, July 1998.

[2] S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Häehnel, C. R. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "Minerva: A tour-guide robot that learns," in *23rd Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence*, (Pittsburgh, PA), pp. 14–26, September 1999.

[3] A. Fod, A. Howard, and M. J. Matarić, "Laser-based people tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Washington D.C.), pp. 3024–3029, May 2002.

[4] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Washington, D.C.), pp. 695–701, May 2002.

[5] J. Xavier, M. Pacheco, D. Castro, A. Ruano, and U. Nunes, "Fast line, arc/circle and leg detection from laser scan data in a player driver," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Barcelona, Spain), pp. 3941–3946, April 2005.

[6] J. Cui, H. Zha, H. Zhao, and R. Shibasaki, "Robust tracking of multiple people in crowds using laser range scanners," in *IEEE 18th International Conference on Pattern Recognition (ICPR)*, (Hong Kong, China), pp. 857–860, August 2006.

[7] J. H. Lee, T. Tsubouchi, K. Yamamoto, and S. Egawa, "People tracking using a robot in motion with laser range finder," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Beijing, China), pp. 2936–2942, October 2006.

[8] H. Zhao, Y. Chen, X. Shao, K. Katabira, and R. Shibasaki, "Monitoring a populated environment using single-row laser range scanners from a mobile platform," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Roma, Italy), pp. 4739–4745, April 2007.

[9] K. O. Arras, Ó. Martínez Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Roma, Italy), pp. 3402–3407, April 2007.

[10] Z. Zivkovic and B. Kröse, "Part based people detection using 2d range data and images," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, (San Diego CA), pp. 214–219, October 2007.

[11] M. Mucientes and W. Burgard, "Multiple hypothesis tracking of clusters of people," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Beijing, China), pp. 692–697, October 2006.

[12] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Pasadena CA), pp. 1710–1715, May 2008.

[13] P. Kondaxakis, S. Kasderidis, and P. E. Trahanias, "A multi-target tracking technique for mobile robots using a laser range scanner," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Nice, France), pp. 3370–3377, September 2008.

[14] A. Carballo, A. Ohya, and S. Yuta, "Fusion of double layered multiple laser range finders for people detection from a mobile robot," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, (Seoul, Korea), pp. 677–682, August 2008.

[15] S. Gidel, P. Checchin, C. Blanc, T. Chateau, and L. Trassoudaine, "Pedestrian detection method using a multilayer laserscanner: Application in urban environment," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Nice, France), pp. 173–178, September 2008.

[16] M. Hashimoto, Y. Matsui, and K. Takahashi, "People tracking with in-vehicle multi-laser range sensors," in *2007 Annual Conference of the Society of Instrument and Control Engineers (SICE)*, (Kagawa, Japan), pp. 1851–1855, September 2007.

[17] S. Yuta, S. Suzuki, and S. Iida, "Implementation of a small size experimental self-contained autonomous robot-sensors, vehicle control, and description of sensor based behavior," *Lecture Notes in Control and Information Sciences: Experimental Robotics II, The 2nd International Symposium, Toulouse, France, 1991*, vol. 190, pp. 344–358, 1993.

[18] H. Kawata, S. Kamimura, A. Ohya, J. Iijima, and S. Yuta, "Advanced functions of the scanning laser range sensor for environment recognition in mobile robots," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, (Heidelberg, Germany), pp. 414–419, September 2006.

[19] K. C. Dietmayer, J. Sparbert, and D. Streller, "Model based classification and object tracking in traffic scenes from range-images," in *IEEE Intelligent Vehicles Symposium (IVS)*, (Tokyo, Japan), May 2001.

[20] Hokuyo Automatic Co. Ltd., "Hokuyo Scanning Range Finder (SOKUIKI) Sensor." `http://www.hokuyo-aut.jp`.

[21] M. Pilu, A. W. Fitzgibbon, and R. B. Fisher., "Ellipse-specific direct least-square fitting," in *IEEE International Conference on Image Processing (ICIP)*, vol. 3, (Lausanne, Switzerland), pp. 599–602, September 1996.

[22] M. D. Latt, H. B. Menz, V. S. Fung, and S. R. Lord, "Walking speed, cadence and step length are selected to optimize the stability of head and pelvis acceleration," *Experimental Brain Research*, vol. 184, pp. 201–209, January 2008.

[23] E. Takeuchi, T. Tsubouchi, and S. Yuta, "Integration and synchronization of external sensor data for a mobile robot," in *Society of Instrument and Control Engineers (SICE)*, vol. 1, (Fukui, Japan), pp. 332–337, August 2003.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Estimation of Pedestrian Distribution in Indoor Environments using Multiple Pedestrian Tracking

Muhammad Emaduddin and Dylan A. Shell

Computer Science Department
University of Southern California
Los Angeles, CA 90089, USA

emaduddi@usc.edu

Abstract - We propose a two-tier data analysis approach for estimating distribution of pedestrian locations in an indoor space using multiple pedestrian detection and tracking. Multiple pedestrian detection uses laser measurement for sensing pedestrians in a heavily occluded environment which is usually the case with most indoor environments. . We adapt a particle filter based multiple pedestrian tracker to address the constraints of a limited number of sensors, heavy occlusion and real-time execution. Under these conditions any detection and tracking technique is likely to encounter a degree of error in cardinality and position of pedestrians. A completely new approach is employed which measures the error in tracker output due to occlusion and uses it to estimate a probability density function which represents the probable number of pedestrians located at a particular exhibit at a particular time. The end result of the system is a variable representing cardinality of pedestrians at a particular exhibit. This variable follows a distribution which is approximately normal where the variance of the probability distribution function is directly proportional to the error encountered by the tracker because of occlusion. The accuracy of our detection and tracking algorithm was tested both separately and in conjunction with the second-tier pedestrian distribution analysis and found marked improvement making our average pedestrian counting accuracy to at least 90% for all the pedestrian position data that we gathered with average pedestrian density at 0.34 pedestrians per sq. meter. Since the environment constraints for our system are unprecedented, we were unable to compare our result to any previous experiments. We recorded the number of people at each exhibit manually to establish the ground truth and compare our results.

I. Introduction

Indoor detection and tracking of pedestrians has a wide spectrum of applications ranging from architectural design of walkways to controlling pedestrian flow at public places like theatres, museums, airports, sports arenas, conventions centers and parks. Our effort in this paper is to devise a system capable of tracking and counting pedestrians in real-time using minimal resources. The word "minimal" here refers to the fewest possible laser measurement sensors with constraints on their orientation and placement. In real life applications, (e.g. narrow walkways, mounting on vehicles etc) the set of feasible locations for deploying sensors can be severely constrained. In our experience the requirements for non-intrusiveness of sensors i.e. reliable electrical power and maximum sensor coverage, limit the number and placement of sensors. Among the set of sensors that are available for tracking pedestrians, laser-range finders (LADAR) are presently among the most reliable and accurate; they reliably provide sub-centimeter accuracy at millisecond frequencies in range of environments. But even with the high fidelity that laser sensors provide, circumstances exist in which laser-based techniques fail to produce dependable pedestrian tracking results. While the techniques introduced in [1], [3], [4], [5], [6] and [7] are among the most successful in terms of tracking accuracy, they are significantly limited when dealing with occlusions [2] and many have a computational complexity that means they remain unsuitable for real-time applications. While our developed system is not as accurate as the online-learning tracker described in [4], it produces dependable results in heavily occluded environments while not compromising its real-time applications.

This work was partially supported by.US National Science Foundation under their Crosscutting Human and Social Dynamics (HSD) program.

II. Experiment Setup

Our test-bed for the detection and tracking algorithm consists of a tunnel like pathway which has five exhibits along its path and two access doorways to an unobserved theatre exhibit close to the centre of the pathway as shown in Figure. 1. Pedestrians can enter and exit the section of museum under discussion using any of the two accesses to the pathway. Pedestrians can also enter in and out from any of the doorway accesses to the unobserved exhibit. This pathway was chosen to be our test case as it allows various situations that can introduce complications in indoor pedestrian detection and tracking to be tested. These situations include: (i) Pedestrians move in a narrow tunnel like space thus there exists a high probability of occlusion due to close proximity of people: (ii) The pathway contains sections that can help us observe completely distinct behaviour of pedestrians e.g. at the exhibits where we expect pedestrians to stop and gather, away from exhibits where we expect pedestrians to walk with a relatively longer stride and at entrances where pedestrians are usually in an exploratory mode and tend to change walking direction very quickly: (iii) The two only access doorways to the circular theatre are observed by our laser scanners thus we were able to keep track of people present within the theatre without even directly observing them by simple count-keeping of people leaving and entering the theatre, (iv) Pedestrians visiting the exhibits were both adults and children which required us to tune detection to accept a relatively wide range of values for stride of a pedestrian, (v) Pedestrian groups,

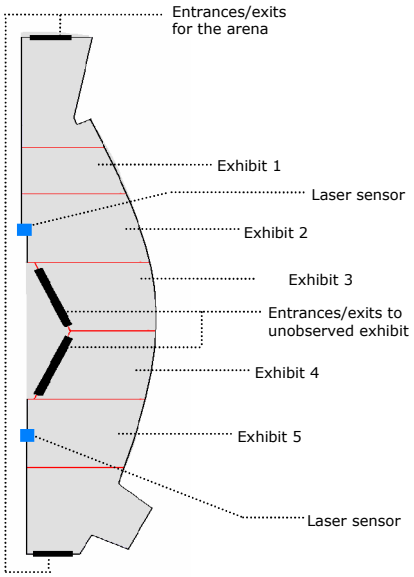


Fig.1. Test arena

which were usually a group of students lead by a teacher were a frequent occurrence at our test bed.

In order to meet our objective of tracking a fairly large number of people utilizing minimum possible resources, we decided to place two SICK Laser Measurement Sensors (LMS) 200 at a distance of approximately 8 meters from each other to cover an area of roughly 70 sq. meters. Ranges of our laser sensors overlapped for almost 16 sq. meters of area out of the total thus giving us a relatively accurate count in the overlapped area. The total area was divided into 5 cells each representing an exhibit (as shown with red lines in Figure 1). These cells will be later used to gather count of pedestrians visiting each exhibit at any given time. The off-the-ground height of rotating mirror within laser sensor was set at 29.9cm for all observations during the project. This height plays a crucial role in detection and association of clusters to the pedestrians since lowering the sensor height gives us discrete clusters representing feet but at the same time decreases our chances of detection of feet since we raise our feet while walking. On the other hand increase in height tends to ignore discrete clusters from feet of children or people with short heights. The effective scanning frequency of laser sensors is about 39Hz. The foreground points from the laser sensors were extracted easily by background learning and subtracting it from laser sensor readings.

III. The System

We present a system that is capable of detecting, tracking and the giving us the probability of pedestrian count at required locations. It comprises of two tiers explained in detail below

Tier 1: Detecting and Tracking Pedestrians

As will be shown, this involves a non-trivial adaptation and extension of the techniques developed in [1]. We describe the three parts below.

A. Clustering: Our algorithm starts by clustering incoming points from laser sensors using mean shift clustering algorithm. The system needs the size of cluster parameter at this point which is equivalent to the average area A of foot-print of an adult foot i.e. 0.04 sq. meters [10].

B. Temporal Correlation Analysis: After classification of points into clusters we iterate through clusters and establish which clusters belong to which pedestrian based on the notion that each pedestrian can be associated with a maximum of two clusters in *nth* frame which lie closest to the pedestrian in (*n-1)st* frame, we call this step as temporal correlation step. We divide this step into two phases (i) Phase one starts with identification of potential feet of pedestrians by calculating closest clusters and separating these as pairs. Only those clusters qualify as feet pair which lie within a parameter know as inter-feet distance *I* and have sizes in the vicinity of A sq. meters.

$$test\_pair(C_i, C_j) = \{Pair^t(C_i, C_j) \mid \begin{array}{l} distance(C_i, C_j) < I \\ \wedge \min(\sum_i \sum_j distance(C_i, C_j)) \\ \wedge \max(size(C_j), size(C_i)) \le 0.04 \\ \wedge \neg Pair^{t-1}(C_i, C_j) \end{array} \} \quad (1)$$

The remaining unpaired clusters are thought to be clusters which are formed due to the fact that we cross our feet while walking thus rendering a single cluster in the laser sensor readings. The area of such clusters can be at most twice the footprint area of an average human foot (ii) Second phase consists of determining whether each cluster pair belongs to a newly detected pedestrian or it should be considered an update for an already tracked pedestrian *P* on the scene. This is done using association distance *D* that is the maximum distance that a pedestrian can travel between readings collected by laser sensors. Therefore the value of *D* is dependent upon the maximum walking speed of pedestrians in the arena.

$$associate\ (Pair_i^t, P_j^{t-1}) = \{update\ (Pair_i^t, P_j^{t-1}) \mid \begin{array}{l} distance(Pair_i^t, P_j^{t-1}) < D \\ \wedge \min(distance(Pair_i^t, P_j^{t-1})) \end{array} \} \quad (2)$$

Introducing above condition limits the distance travelled by pedestrians while being occluded and still being effectively tracked as a unique pedestrian.

We observed that the periodic motion of pedestrian feet described in [1] remains undetectable most of the time in environments cluttered with occlusions. Algorithm in [1] defines merge as a stage during walk when clusters of both feet of a pedestrian come close together and their clusters merge while split is described as a case when the pedestrian continues to walk after a merge and clusters of both feet split part. While merge and split cases were occasionally encountered during

our experiment, we found out that detection of pedestrians in this manner is both inaccurate and computationally burdensome. The reason of inaccuracy lies in following notions (a) Most of the time we observe pedestrians walking in close proximity to other pedestrians or in the shape of groups, this tends to produce merges and splits that involve feet of two different pedestrians (b) Due to frequent occlusion (see Figure 2) We are likely to miss splits and merges belonging to a pedestrian thus rendering our split/merge detection mechanism useless under this situation (c) Pedestrians may not always walk, they might just stand for a while. Our solution to these problems as evident by (1) and (2) is to ignore the merge and split cases completely thus reducing the time complexity of temporal correlation step to $(n^2 logn + (nm).log(nm))/3$ where n is the number of clusters and m is the number of pedestrians on the scene. After this step, detected pedestrians along with their associated clusters are provided to the tracker i.e. our next step in sequence.

C. Tracking: The tracker is the component of our system that is responsible for estimating the parameters of motion and location attached with our pedestrian based on given updates from temporal correlation step. It uses a particle filter to estimate the position p, stride s, direction d and phase ph of a pedestrian as already employed in [1]. In brief the tracker keeps track of the pedestrians in three sub-steps (i) Update Step: Tracker weighs each pedestrian's particles proportional to their distance to the points belonging to its associated clusters: (ii) Sampling Step: After update step, the tracker randomly samples the weighted particles where the likelihood of any particle to be chosen is proportional to its weight. Thus a certain predefined number of particles M are chosen: (iii) Propagation Step: In the last step of tracking the sampled M particles are propagated through a multidimensional space representing the motion of the tracked pedestrian according to the walk model described in detail in [1]. This step modifies the position, stride, direction and the walking phase of a pedestrian and is performed without taking into account whether a pedestrian has received updates or not. The propagation of pedestrians that do not receive updates helps our tracker to track occluded pedestrians up till a certain amount of distance D.

During tracking each foreground point belonging to the pedestrian is used for calculating its distance with each of M particles belonging to the same pedestrian in tracker. For a maximum density of 1.8 pedestrians per sq. meter under which our tracker can perform optimally, it performs on the average nearly 504,000 calculations to update, sample and propagate 126 pedestrians through a single iteration. Given such high a penalty in terms of execution time, we deemed it extremely important for our algorithm to produce results with nearly same accuracy using fewer less computational resources in order to remain useful in real-time applications. Considering this requirement, we were able to successfully track pedestrians with very little degradation of accuracy by skipping unnecessary observations from laser sensors (See Table 1). The laser sensors provide our system with observations effectively after every 0.025 seconds. We forced our system to consider observations after every 0.05 seconds i.e. in effect dropping every second observation. This reduced the output accuracy by a very negligible value but the performance gain was more than 2 times. Since our system is specifically designed to handle occlusion, skipping an observation makes our system behave as if the skipped observation is due to an occlusion, thus by increasing the D parameter in temporal correlation module it compensates for most of the loss in accuracy.

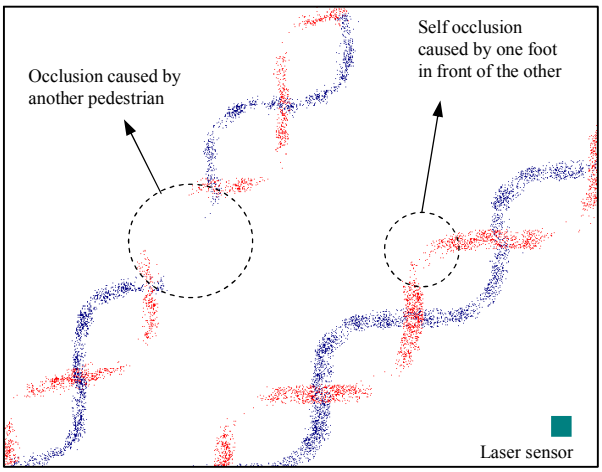


Fig. 2 An S-T representation of observable feet data

The resultant system described up till now is relatively robust and accurate means of detecting and tracking pedestrians given the fact that we are performing these steps in real-time.

Tier 2: Pedestrian Distribution Analysis

Although reliability in the results could have been achieved by integrating techniques like online-supervised learning [4], Multiple Hypothesis Tracking [3] or Auxiliary Particle Filter switching [2] in the first tier, but doing so will exclude our tracker completely from the realm of real-time systems. Thus, the second tier of our system is designed to further enhance the reliability of the pedestrian count output for each exhibit while keeping the computational complexity growth nearly constant. We term this tier as the pedestrian distribution analysis tier as it is concerned with keeping track of pedestrians crossing in and out of each cell cells within the environment. A cell comprises of area in front of an exhibit defined using cell boundaries (as marked in Figure 1). By maintaining information about the distribution of people over cells, although the system cannot answer questions about where particular pedestrians are, one may still investigate questions about the flow of people and how their (average) route selection depends on the (average) presence or absence of people.

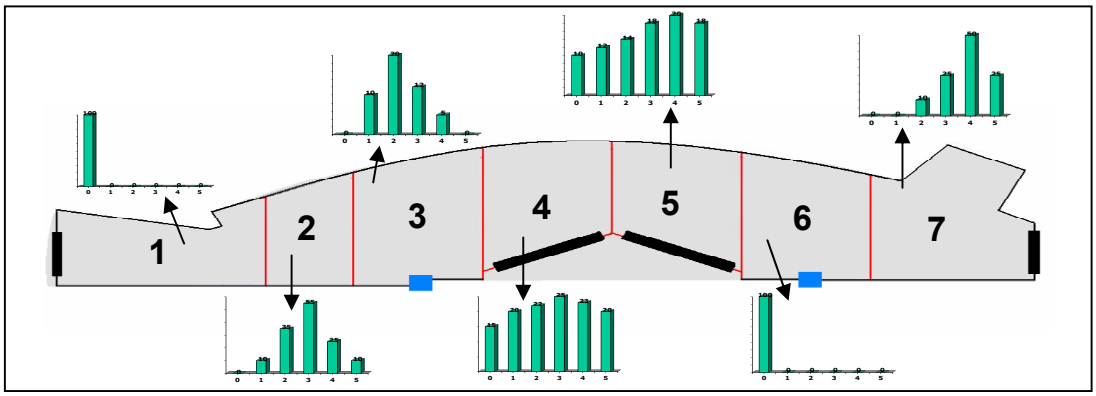


Fig.3. Pedestrian Distribution Analysis tier Output

Detecting number of people crossing into and out of each cell we were able to deduce the number of people N_i^t in each cell *i* at each time-step *t*. This number contains a certain error directly proportional to the percentage of the cell boundary hidden from laser sensors due to the pedestrians standing/walking very close to the laser sensors. In order to factor-in the error present in this number, we choose to represent the output of the system for each cell as a distribution over the number of people. A distribution variable X_i^t for each cell *i* at any given time *t* is a state of our belief that represents all past observations including the current one. This is achieved via updating the distribution variable X_i^t for each exhibit at each time-step. Variable X_i^t is defined as

$$X_i^t = \left\{ \pi_u^t, \quad u = N_i^{t-1} - r, N_i^{t-1} - (r-1), \ldots, N_i^{t-1} + r \right\} \qquad (3)$$

Here *u* is an index that runs through the range of weights π which represent our probability density function *(pdf)*. Most generally the range adjustment value *r* is subject to the requirement of the analyst which differs with the application of our system. (We used the physical capacity of the exhibits to place limits on this range of values.) Changing the value of *r* increases or decreases the domain of our distribution function. N_i^{t-1} is a number that has the maximum weight π_u^{t-1} associated to it in the distribution X_i^{t-1} from previous time-step. Following steps update the variable X_i^t at each time-step via a Gaussian update U_i^t whose variance is determined by the percentage of cell boundary occluded at any given moment. The update step is given below.

$$X_i^{t+1} = X_i^t + \delta^{t+1}(U_i^{t+1} - X_i^t)$$

$$\textit{where} \qquad U_i^t = \left\{ \pi_u^t, u = N_i^{t-1} - r, \ldots, N_i^{t-1} + r \right\} \qquad (4)$$

$$\textit{and} \qquad \delta^{t+1} = \frac{\sigma_t^2}{\sigma_t^2 + \sigma_{t+1}^2}$$

Here if U_i^{t+1} has high variance relative to U_i^t then δ^{t+1} is small thus it has little impact on value of X_i^{t+1}. This ensures that updates which have more chance of error are factored-in less into our current belief X_i^{t+1}. σ_t^2 is the adjusted-variance in update distribution U_i^t and is determined using this intuitive criteria :

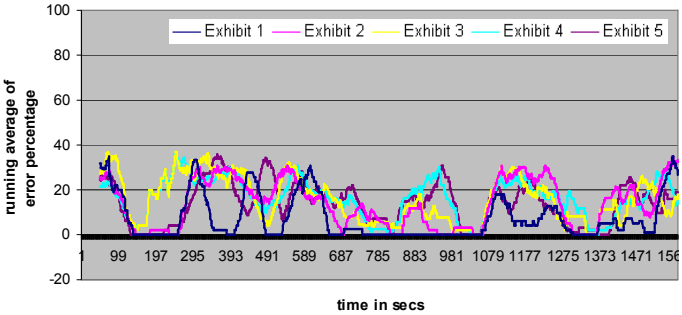
$$\sigma_t^2 \quad \propto \quad (g\sigma_t^2)\left(\frac{o_i^t}{l_i}\right) \qquad (5)$$

Here $g\sigma_t^2$ is the Gaussian variance of update U_i^t. The criteria described in (5), sets the variance to be directly proportional to the ratio of length of occluded boundary of cell o_i^t (calculated at every time-step) to the total visible length l_i of the cell boundary.

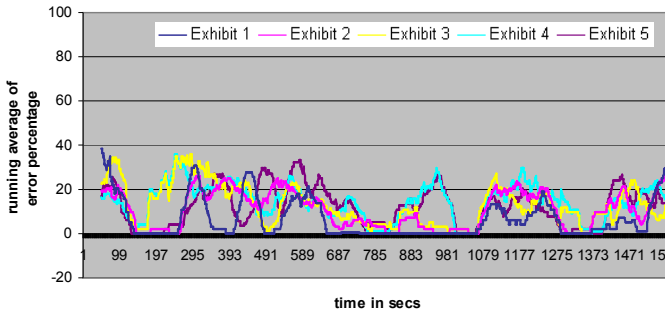
Pedestrian Distribution analysis tier thus represents snapshots of *pdfs* for each cell at each time-step which gives us a measured idea about the confidence that we can place on the pedestrian count in each cell (see Figure 3).

IV. Discussion

Tracking pedestrians at exits and entrances proved to be one of the trickiest parts during the system design. We know that the tracker output grows accurate with increase in the time for which a target is observed since tracker gets more chances to update and propagate its particles so that these can match target dynamics. Thus the places the tracker tends to be most inaccurate are the entrances to the observed area where the observed time for entering targets is limited. In order to estimate by what margin our tracker fails to track entering pedestrians, we performed an experiment by first measuring the number of pedestrians crossing east to west across a line dividing the observed area into two halves. We did this because our tracker is relatively accurate about pedestrians in the middle of observed area since the tracker had enough time to track these pedestrians. Then we considered the same line as an entrance and ran the tracker for the second time on the same



(a) Error before Tier-2 application



(b) Error after Tier-2 application

Fig. 4 System counting error comparison

set of observations for people entering in east to west direction considering updates only from one half of the observed area and ignoring the rest. The difference between the numbers of people crossing east to west in both cases provided us with the bias the tracker had in tracking pedestrians near the entrances. We used this bias b_i in following manner to adjust the number of people in cells that are situated at the entrances:

$$N_i^t = N_i^t + b_i$$

Using updated cardinality as an input to the second tier of our system proved to be beneficial in terms of accuracy but we restrained to declare it a formal part of our system since it would make tedious experimentation to learn bias, a prerequisite for deploying our system thus limiting its applications.

V. Results

We tested our system in terms of accuracy and computational efficiency. In data collection phase we manually recorded the pedestrian crossings over certain episodes of time observed via laser data stream for each of the cells. These time-stamped recordings were accurate up to 1 second resolution and served as our ground truth. For accuracy measurement we computed following two errors. (i) ($N_i^t - ground\_truth_i^t$) for exhibits *i=1 to 7* (Figure 4a shows a single episode depicting the error for each of the cells). Here error is calculated using pre tier-2 measurement i.e. N_i^t from tier-1. Here the cumulative average counting error for all our observations for all the exhibits totalled to be 13.8%. (ii) ($\mu_i^t - ground\_truth_i^t$) for exhibits *i=1 to 7* where μ_i^t is the value with highest probability in the *pdf* representing X_i^t (Figure 4b shows the same episode as shown in fig. 4a depicting the error for each of the cells). This error is computed using output from tier-2 of our system. The average counting error for all our observations for all the exhibits in this case stood at 9.83% which shows marked improvement as a result of applying tier-2.

By applying our tier approach to laser data collected by recording over 50 hours of museum visitors, we are able to plot locations of high-traffic. This is shown in Figure 5 using a colour coded scheme in which red highlights reflect the positions that people spend most of their time in. In a sense, this represents the time-averaged distribution from tier-2.

VI. Conclusion

Techniques described in [1], [3], [4] and [6] stress the tracking accuracy. Our effort is focused on retrieving analysable results using fast tracking techniques in order to get reliable pedestrian count in heavily occluded environments. Our pedestrian detection and tracking algorithm is extremely computationally intensive as is the case with all other multiple target tracking algorithms [7] and this happens in our case due to computations like inter-cluster, cluster to pedestrian distance calculation and propagation of a high number of particles in particle filter at each time-step. During our experiment phase we were able to produce sufficiently accurate results in a more reliable format for scientific analysis of pedestrian distribution in indoor environments.

Acknowledgements

Support from Interaction lab, University of Southern California (USC) is gratefully acknowledged. Also support from all undergraduate students who worked under NSF's Research Experience for Undergraduates (REU) program is appreciated. Scholarship grant from Fulbright Commission is acknowledged and appreciated as it funded the research assistantship for one the authors. We thank Professor Kristina Lerman from USC Information Sciences Institute for her constant advice and mentoring during all phases of our research. Lastly this work was made possible by the motivation given to us by our ever helpful Professor Maja Mataric.

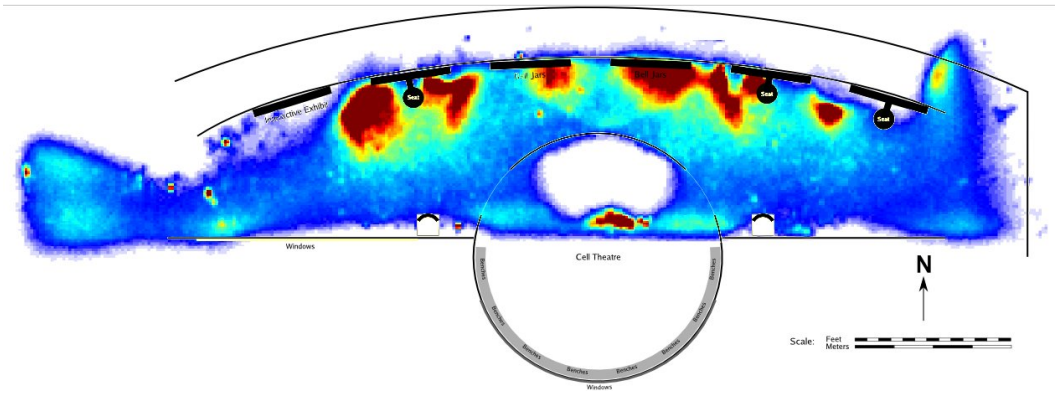


Figure 5: Locations of high traffic within the museum exhibit

TABLE I
COMPUTATIONAL EFFICIENCY FOR VARYING PEDESTRIAN DENSITY (System: Ubuntu 8.04, kernel 2.6.24-18, Intel Pentium Mobile 1700 MHz Processor)

Frame skip rate	Average execution time for 1 sec of frames	Peak density encountered (people per sq. m)	Average density (people per sq. m)	Average counting error % (error/truth*100)
Every 2 out of 3	0.58 sec	0.33	0.10	11.8
Every 2 out of 3	0.8 sec	1.94	0.35	11.9
Every 2 out of 3	0.92 sec	0.72	0.54	13.6
Every other	0.71 sec	0.33	0.10	9.7
Every other	0.94 sec	1.94	0.35	9.4
Every other	1.07 sec	0.72	0.54	10.2
None skipped	1.94 sec	0.33	0.10	8.5
None skipped	2.6 sec	1.94	0.35	8.4
None skipped	3.12 sec	0.72	0.54	9.3

REFERENCES

[1] Shao .X, Zhao .H, Nakamura .K, Katabira .K, Shibasaki .R, "Detection and Tracking of Multiple Pedestrians by Using Laser Range Scanners" in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, April 2007.

[2] Bando .T, Shibata. T, Doya. K, Ishii. S, "Switching Particle Filters for Efficient Real-time Visual Tracking" in *Proceedings of the 17th International Conference on Pattern Recognition 2004*, vol. 2, pp. 720-723, Aug 2004.

[3] Arras .K, Grzonka .S, Luber .M, Burgard .W, "Efficient People Tracking in Laser Range Data using a Multi-Hypothesis Leg-Tracker with Adaptive Occlusion Probabilities" in *2008 IEEE International Conference on Robotics and Automation*, pp. 1710-1715, May 2008.

[4] Song .X, Cui .J, Wang .X, Zhao .H, Zha .H, "Tracking Interacting Targets with Laser Scanner via On-line Supervised Learning" in *2008 IEEE International Conference on Robotics and Automation*, pp. 2271-2276, May 2008.

[5] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, pp. 843–854, Dec 1979.

[6] Wang .J, Makihara .Y, Yagi .Y, "Human Tracking and Segmentation Supported by Silhouette-based Gait Recognition" in *2008 IEEE International Conference on Robotics and Automation*, pp. 1698-1703, May 2008.

[7] Khan .Z, Balch .T, Dellaert .F, "MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets" in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, Issue. 11, pp. 1805-1819, Nov. 2005.

[8] Thrun, S., "Particle filters in robotics", *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*, 2002.

[9] Hollinger .G, Djugash .J, Singh .S, "Tracking a Moving Target in Cluttered Environments with Ranging Radios", in *2008 IEEE International Conference on Robotics and Automation*, pp. 1430-1435, May 2008.

[10] Hawes, Michael R., "Quantitative morphology of the human foot in a North American population" in *Ergonomics*, Vol. 37, Issue. 7, pp 1213, 1994.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Improved Human Detection Using Image Fusion

E. Thomas Gilmore III*, Preston D. Frazier+, M. F. Chouikha*
*Department of Electrical and Computer Engineering Howard University, Washington, DC 20059, USA
+General Dynamics Corporation, Hanover, MD 21076, USA

***Abstract*— Image fusion is fundamental to several modern day image processing applications. It is often a vital preprocessing procedure to many computer vision and image processing tasks which are dependent on the acquisition of imaging data via sensors, such as infrared and visible. One such task is that of human detection. In this paper, we present improvements to our shape and heat flow-based technique of detection and classification of humans in unrestricted poses with the addition of image fusion. We focus on both rural and urban environments and demonstrate the effectiveness of using image fusion as a preprocessing procedure for improved human detection and classification. Extensive simulations using MWIR images were conducted and promising results are obtained. Receiver Operating Characteristic (ROC) analysis also showed excellent performance of the SVM-based human classification.**

I. Introduction

Infrared (IR) sensors have been applied to human detection applications such as vehicle safety, night vision, and military applications. They directly detect targets with warm temperatures in an image, providing a potentially simpler and quicker solution to human detection, especially during nighttime. However, IR sensors are much more expensive compared to optical cameras with comparable resolutions, making it less affordable for many applications. IR-based human detection has been investigated by a number of groups. Most existing research on IR-based human detection is focused on pedestrian detection in an urban environment on the street or on campus to provide assistance to the drivers or for surveillance purposes, especially during the evening [1]-[8]. Compared to non-urban environments where terrain, mountain, and/or forest scenes are the main background, urban scenes usually have artificial objects in their background such as buildings and streetlights whose temperatures are generally elevated during the evening. Vehicles also generate heat that can show up as hotspots in IR images. These background noises can make IR-based human detection more complicated. On the other hand, however, pedestrians on the street are generally in simple walking or standing upright poses which are easier to model than other complicated poses such as stretching (e.g., running and bending) or hiding, which can often occur in a non-urban environment such as in the battlefield. Human detection is obviously a more challenging situation and new methods have to be introduced, especially in dealing with the unrestricted human poses. Since little research has been done for human detection in such a non-urban environment, we have analyzed many existing algorithms designed for pedestrian detection in urban environment, and experimentally evaluated them against non-urban IR images [9]. It is shown that, as expected, these existing algorithms performed especially poorly on humans in stretching or hiding poses because they rely heavily on features of standing or walking human shapes and appearances. As a result, humans with stretching poses or partial occlusions (such as behind the trees) in the IR images are mostly missed.

In this paper, we investigate the application of Image Fusion for the purpose of improving our human detection algorithm previously presented [10]. Image fusion has been investigated by many research groups and a number of algorithms have been developed [11] - [14]. The purpose of Image fusion is to integrate images of the same target or scene from multiple sensors to produce a composite image or images that will inherit most salient features from the individual images. The fused image usually has more information about the target or scene than any of the individual images used in the fusion process. The images used for fusion here are MWIR and visible. This new method of combining image fusion to the human detection algorithm represents a natural yet powerful extension from existing pedestrian detection methods.

In section II, we briefly review the heat flow and shaped-based human detection algorithm. The application and background on Image Fusion is described in section III. Section IV presents experimental results and simulations, and section V discusses conclusions, respectively.

Manuscript received January 15, 2009. This work was supported in part by the U.S. Army Robotics Collaborative Technology Alliance (RCTA) program.

Erwin Gilmore is with the Electrical Engineering Department, Howard University, Washington, DC 20059 USA (e-mail: ethomasg@gmail.com).

Mohamed Chouikha is with the Electrical Engineering Department, Howard University, Washington, DC 20059 USA (e-mail: mchouikha@howard.edu.

II. Review of IR Based Human Detection/Classification Algorithm

A. IR Spectrum for Human Detect

Based on the distribution of infrared radiation spectrum, an IR sensor can be classified as one of the following four categories according to its wavelength [12]:

- Short Wave IR (SWIR): 0.7 - 3 μm
- Mid Wave IR (MWIR): > 3 - 6 μm
- Long Wave IR (LWIR): > 6 - 15 μm
- Far IR (FIR): > 15 - 1000 μm

IR energy is emitted by all materials and objects above 0°K as thermal radiations. The upper limit of FIR occurs in a region where it is difficult to envision the output from a source as heat (peak radiation occurs at 3°K). At normal temperature, human body radiates most strongly in the IR range at about 10 μm, which apparently corresponds to the wavelength range of LWIR. As a result, LWIR, MWIR and some FIR sensors are usually used for human detection in most applications.

B. Shape-based Feature Selection

The process of human candidate selection consists mainly of three steps: first preprocessing such as histogram equalization and segmentation by thresholding the image to obtain the hotspots, then morphological operations to suppress background noises, and finally selection of human candidates using metrics such as aspect ratio constraint, local histogram filtering, and/or morphological human model matching.

Thresholding is a technique often used to separate foreground targets from background environment based on their differences in image intensity. In a simple thresholding process, a single intensity threshold is used to generate a binary image from the original image. For example, the intensity threshold can be determined using the following equation [2].

$$Threshold = \alpha I_{mean} + \beta I_{max} \qquad (1)$$

where α and β satisfy $\alpha + \beta = 1$ and represent weights assigned to the mean intensity I_{mean} and the maximum intensity I_{max} of the original image. The best threshold setting will depend on the camera settings and the ambient conditions, e.g. temperature distribution of background objects; hence it will have to be tuned to the conditions. Determination of appropriate values for the weights, however, is not a trivial task. It is usually dependent on the specific setting of the IR camera such as brightness and/or contrast. By extensively testing our IR images using different weights, it is shown that weights $\alpha = 0.4$ and $\beta = 0.6$ perform the best, as shown in Figure 1, where 456 MWIR images with forest background were used to plot the relationship between rate of correct human selection vs. average number of non-humans selected per image with different weight values and in different aspect ratio ranges.

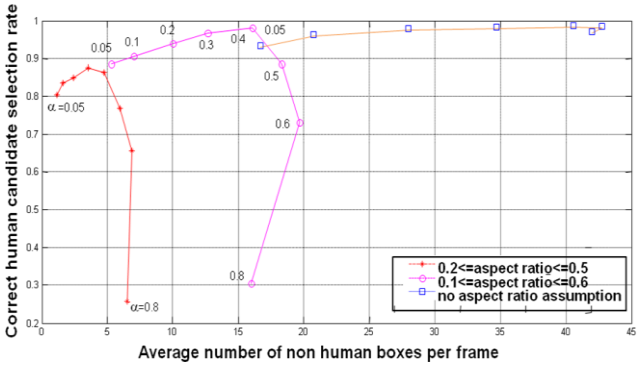
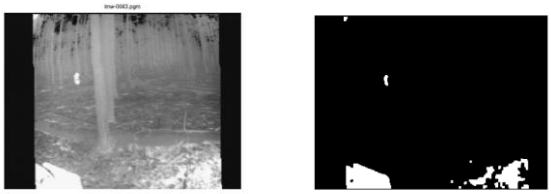
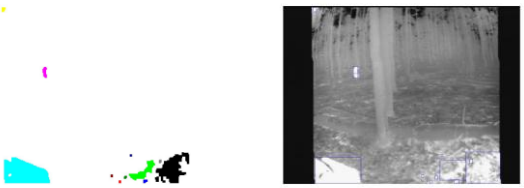


Figure 1: Threshold vs. Aspect Ratio Optimization

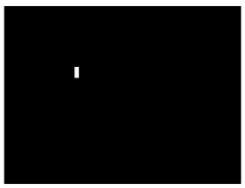
To remove isolated noises in the thresholded binary images, morphological operations of combined erosion and dilation are effectively used. Further, it is shown that local histogram of the selected human candidates can be used as a powerful filter for the elimination of false human candidates such as tree branches or electric poles [1]. This is primarily based on the fact that the intensity values of a human body in an IR image are far less uniform compared to those cylinder shaped objects. For example, the middle portion of the histogram of a bounding-box for a hotspot resulting from an electric pole is often either empty (i.e., concentration on both dark (background) and bright (pole) pixels with little gray pixels between them), or narrowly concentrated (i.e., with little or no dark or bright pixels) when the pole fills up the whole bounding-box. An example of a 'spread-out' local histogram of a human candidate is shown in Figure 2(g). Figure 2 shows an example of the process of human candidate selection.



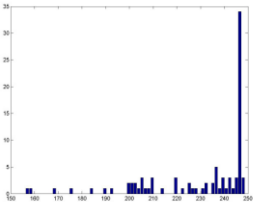
(a) Original Image (b) Thresholding/Morphological Operation



(c) Grouping of Hotspots (d) Bounding Boxes of Hotspots

(e) Applying Aspect Ratio (f) Bounding Box for Human Candidate



(g) Local Histogram of the Human Candidate

Figure 2: Example of Human Candidate Selection Process

Overall, with shape-based features we have achieved a maximum correct human candidate selection rate of 96% with a false alarm rate (or false positive rate) of around 20% in our initial experiments using the 456 MWIR images with forest background [9].

C. Heat Flow-based Feature Selection

Heat flow is a similar concept as optical flow in motion analysis using optical images [15]. Optical flow estimates motion information at pixel (x, y) at time t and $t + \delta t$ between two consecutive frames of a video camera by assuming a near-constant pixel intensity value I, which results in the following partial differential equation:

$$I_x v_x + I_y v_y = -I_t \qquad (2)$$

where (v_x, v_y) represents the motion of the pixel (x, y) or its optical flow vector.

In IR images, pixel values represent heat levels emitted by the corresponding physical points in the scene being monitored by an IR sensor, as compared to optical levels in the optical images reflected by the similar points. As a result, pixel motion in an IR image represents flow of the heat caused by motion of a warm target such as a human in the scene. We can thus use heat flow to detect relative motion of a human in an IR image.

In our method, heat flow is primarily used to locate those hotspots or bounding boxes, for reexamination, that failed to qualify the shape-based feature criteria described above. Those bounding boxes represent hotspots that were first picked up by the thresholding process, but were subsequently screened out and discarded primarily because their shape features did not fall in the range of a standing or walking human in the IR image. They were mostly treated as hotspots or noises of the background. If relative motion can be detected from those bounding boxes, however, it is strongly implicated that the targets can be human candidates in stretching poses or with partial occlusions. As a result, they will be 'rescued' from the 'trashcan' and reexamined for potential human candidates.

Relative motion of a human candidate can be detected when the magnitudes of heat flow vectors of a group of pixels inside a hotspot are larger than a threshold value determined in a similar way as that used in shape-based feature selection above. Figure 3 shows a number of examples of selection of human candidates, in both MWIR and LWIR images, in stretching poses or with partial occlusions using the proposed combined shape and heat flow method.

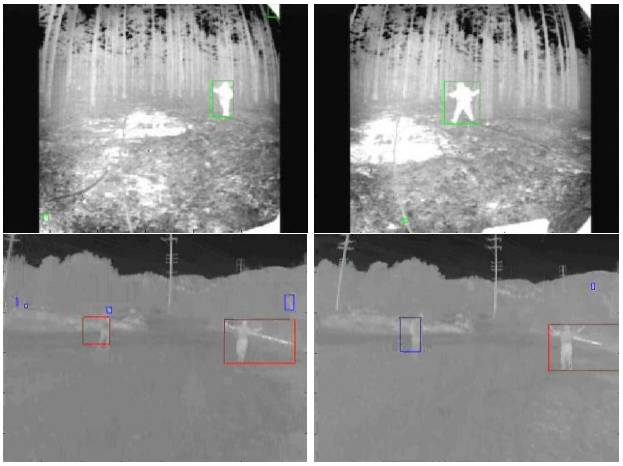


Figure 3: Example of Initial Human Candidate Selection

Preliminary experiments were performed comparing performance of initial human candidate selection using shape-based features vs. using combined shape and heat flow-based features. A total of 198 LWIR images with mountain background were used. The shape-based algorithm achieved a maximum sensitivity of 64%, but the combined shape and heat flow algorithm achieved a significantly higher maximum sensitivity of over 90% while keeping the false alarm rate at the similar level.

D. Classification

Human candidates selected above are fed to a classifier for final classification into either a human or a non-human class. We have implemented the SVM classification method [16]-[18] on our IR images, and used small templates (18x45 in size) of both gray level IR images and their edge maps as training and testing samples. A number of such training samples are shown in Figure 4.

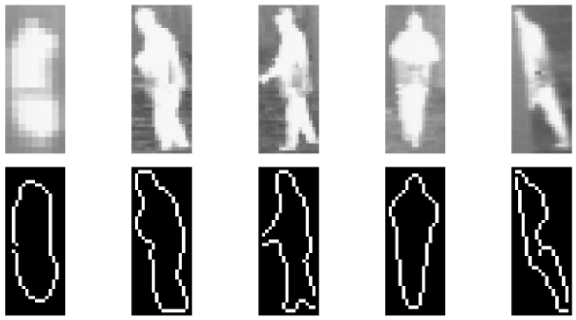


Figure 4: SVM Training Samples

III. Image Fusion Application

Multi-resolution image fusion schemes were developed to overcome the limitations of the previously introduced pixel averaging methods. The goal of these methods is to extract the salient features of each source image, e.g. edges, texture, etc., at various levels of decomposition from coarse to fine, and then aggregate them to create a fused image. The pyramid based schemes first put these concepts into practice. These methods generally produce sharp, high-contrast images that are clearly more appealing and have more information content that the simpler weighted pixel averaging techniques.

First investigated in the early 1980's, the concept of the image pyramid was used as a fast method of representing the multi-resolution information contained within an image in a manner that reflects the multiple scales of processing in the human visual system [19]. The image pyramid is basically a data structure made of a series of low-pass or band-pass copies of the image, each depicting pattern information of a different scale.

The most common example is the image pyramid, whose construction begins by convolving a source image G_0 with a Gaussian kernel K. The filtered image is then sub-sampled by selecting only every other row and column to generate a new image G_1 with half the width and height of the original image G_0. This combination of sub-sampling and convolution is known as a REDUCE operation and defined by:

$$G_1(x,y) = \sum_{u=-p}^{p}\sum_{v=-p}^{p} K(u,v)G_0(2x+u, 2y+v) \quad (3)$$

where the Gaussian kernel K is usually small, i.e. 3×3 or 5×5, for rapid execution. This process is then repeated with G_1 to develop G_2, and so on, until a pyramid of images $G_0, G_1, G_2, \ldots, G_N$ are produced. High spatial frequencies are lost when stepping from one level of the pyramid to the next due to the reduction in the resolution and sampling density. This is interpreted as a loss of salient image detail.

To compare the various image contents now available at each level of the Gaussian pyramid, the EXPAND operator is used. Basically, this consists of duplicating each row and column in the image G_{k+1} and convolving the result with the Gaussian kernel K to generate the new image E_K of the same dimensions as G_k. The EXPAND operator can be expressed by the following:

$$E_k(x,y) = \sum_{u=-p}^{p}\sum_{v=-p}^{p} K(u,v)G_{k+1}(\text{floor}[(x+u)/2], \text{floor}[(y+v)/2]) \quad (4)$$

A new image is then created from the difference between reduced image G_k and expanded image E_K:

$$L_k(x,y) = G_k(x,y) - E_k(x,y) \quad (5)$$

which captures the high frequency spatial details of the *kth* level of the Gaussian pyramid. Thus, new pyramids of varying resolutions are determined from the different Gaussian pyramids, i.e. $L_0, L_1, \ldots, L_{N-1}$, which represent salient information in the original image. This structure is known as the Laplacian Pyramid due to the Laplacian operator that is utilized and was first used for image compression applications [20], [21] and then as an image fusion scheme [22].

A. *Fusion using Laplacian Pyramid*

The Laplacian pyramid fusion method consists of an iterative process of calculating Gaussian and Laplacian pyramids of each source image, fusing the Laplacian images at each pyramid level by selecting the pixel with larger absolute values, combining the fused Laplacian pyramid with the combined pyramid expanded from the lower level, and expanding the combined pyramids to the upper level.

IV. Experiments and Simulations

We employed a Laplacian Pyramid fusion scheme to generate a series of fused images to be utilized as input to the Detection and Classification scheme. Figure 5 depicts a representative test image where the green bounding boxes indicate a correct detection and red bounding boxes indicate a correct rejection, respectively. Initial training and testing experiments were performed with sample fused images (bounding boxes) of selected human candidates (426 for training – 253 humans and 173 non-humans, and 1146 for testing – 654 humans and 492 non-humans) using the SVM classifier. All sample images (bounding boxes) were scaled to the same template size of 18x45 before being fed to the classifier.

The Receiver Operating Characteristic (ROC) curves for an SVM classifier with a quadratic kernel function were generated and are shown in Figure 6, which demonstrated significant improvement over the non-fusion based human detection scheme.

Figure 5: Sample Test Image fused via Laplacian Pyramid (Green, correct detection; Red, correct rejection)

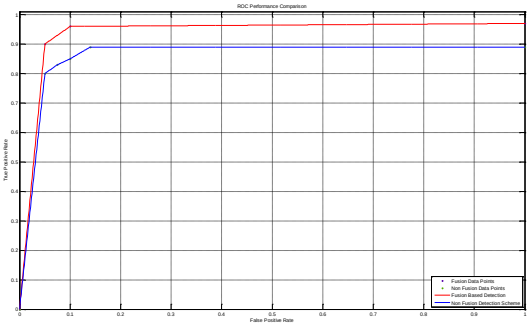


Figure 6: ROC Performance Comparison

V. Conclusion

We have developed an improvement to our original method of human detection using IR images. This method incorporated image fusion as a preprocessing task to the combined shape and heat flow-based detection scheme. Preliminary experiments using a large number of IR images have shown that this new method has achieved significant performance improvement over the original algorithm. The ROC curves also confirmed the excellent performance of the SVM-based human candidate classification.

VI. References

[1] M. Bertozzi, A. Broggi, A. Fascioli, T. Graf, and M.M. Meinecke, "Pedestrian detection for driver assistance using multiresolution infrared vision", IEEE Trans. on Vehicular Technology, 53 (6), 2004.

[2] F. Xu, X. Liu, and K. Fujimura, "Pedestrian detection and tracking with night vision", IEEE Trans. on Intelligent Transportation Systems, 6 (1), 2005.

[3] M. Yasuno, N. Yasuda, and M. Aoki, "Pedestrian detection and tracking in far infrared images", Proc. 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'04), 2004.

[4] Y. Fang, K. Yamada, Y. Ninomiya, B. K. P. Horn, and I. Masaki, "A shape-independent method for pedestrian detection with far-infrared images", IEEE Trans. on Vehicular Technology, 53 (6), 2004.

[5] X. Liu and K. Fujimura, "Pedestrian detection using stereo night vision", IEEE Trans. on Vehicular Technology, 53 (6), 2004.

[6] C. Dai, Y. Zheng, and X. Li, "Layered representation for pedestrian detection and tracking in infrared imagery", Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 3, pp. 20-26, 2005.

[7] M. Bertozzi, A. Broggi, C. Hilario Gomez, R.I. Fedriga, G. Vezzoni, and M. Del Rose, "Pedestrian detection in far infrared images based on the use of probabilistic templates", Proc. 2007 IEEE Intelligent Vehicles Symposium, 2007.

[8] Yu-Ting Chen and Chu-Song Chen, "A Cascade of Feed-Forward Classifiers for Fast Pedestrian Detection", Lecture Notes in Computer Science, Volume 4843, pp. 905-914, Springer, Berlin, November 2007.

[9] J.Zeng, A. Sayedelahl, M. Chouikha, T. Gilmore, and P. Frazier, "Human detection in non-urban environment using infrared images", Proc. Sixth International Conference on Information, Communications, and Signal Processing, Singapore, December 2007.

[10] J. Zeng, A. Sayedelah, H. Laryea, and M. Chouikha, "Enhanced Human Detection in Non-urban Using Combined Shape and Heat Flow Features," 2008 IEEE International Conference on Robotics and Automation, ICRA 2008, Pasadena, California, on May , 2008.

[11] Z. Zhang and R.S. Blum, "A Categorization of Multiscale Decomposition-based Image Fusion Schemes with a Performance Study for a Digital Camera Applicatoin," Proceeding of the IEEE, vol. 87, no. 8, pp. 1315-1326, 1999.

[12] P. Scheunders and S. DeBacker, "Multispectral Image Fusion and Merging Using Multiscale Fundamental Forms," Proc. IEEE International Conference on Image Processing, 2001.

[13] D. Rajan and S. Chaudhuri, "Data Fusion Techniques for Super-Resolution Imaging," Information Fusion, 3, pp. 25-38.

[14] L.A. Chan and Z.S. Der, and N.M. Nasrabadi, "Dualband FLIR Fusion for Automatic Target Recognition," Information Fusion, 4, pp. 35-45.

[15] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", Proc. Imaging Understanding Workshop, pp. 121-130,1981.

[16] V.N. Vapnik, The Nature of Statistical Learning Theory (2nd Ed.), Springer, New York, 1999.

[17] [10] J. C. Platt, “Fast training of support vector machines using sequential minimal optimization”, Chap. 12 in Advances in Kernel Methods - Support Vector Learning, B. Scholkopf, C. Burges, and A. J. Smola, Eds., pp 185--208, MIT Press, Cambridge, MA, 1999.

[18] N. Cristianini and J. Shawe-Taylor, Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, Cambridge, 2000.

[19] E. Adelson, C.H. Anderson, J.R. Bergen, P.J. Burt, and J.M. Ogden. Pyramid Methods in Image Processing. RCA Engineer 29, 33-41, 1984.

[20] P.J. Burt and E.H. Adelson. The Laplacian Pyramid as a Compact Image Code. IEEE Trans. Commun. COM-31, 532-540, 1983.

[21] P.J. Burt. The Pyramid as a Structure for Efficient Computation. Multi-Resolution Image Processing and Analysis.

[22] P.J. Burt and E.H. Adelson. Merging Images through Pattern Decomposition. Applications of Digital Image Processing VIII, Proc. SPIE 575, 173-181, 1985.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Real-Time Object Tracking and Classification Using a Static Camera

Swantje Johnsen and Ashley Tews

Abstract— **Understanding objects in video data is of particular interest due to its enhanced automation in public security surveillance as well as in traffic control and pedestrian flow analysis. Here, a system is presented which is able to detect and classify people and vehicles outdoors in different weather conditions using a static camera. The system is capable of correctly tracking multiple objects despite occlusions and object interactions. Results are presented on real world sequences and by online application of the algorithm.**

I. INTRODUCTION

It is important for vehicle operators around worksites to be aware of their surroundings in terms of infrastructure, people and vehicles. When an operator observes an object moving in a way that will impact on their operations, they take the necessary steps to avoid undesired interaction. Their response depends on recognising the type of object and its track. This skill is also important for autonomous vehicles. An autonomous vehicle needs to be able to react in a predictable and rational manner, similar to or better than a human operator. Onboard sensors are the primary means of obtaining environment information but suffer from occlusions. However, offboard sensors such as webcams commonly deployed around worksites can be used for this purpose. We present our system for offboard dynamic object tracking and classification using a static webcam mounted outside a building that monitors a typical open work area. As the preliminary step towards integrating the extracted information to improve an autonomous vehicle's situational awareness, information about the objects such as location, trajectory and type is determined using a tracking and classification system. The system consists of several existing subsystems with improvements in the detection and classification phases. The system is capable of working in different weather conditions and can distinguish between people and vehicles by identifying recurrent motion, typically caused by arm or leg motion in the tracked objects. Tests were conducted with different types and numbers of vehicles, people, trajectories and occlusions with promising results.

II. RELATED WORK

The common architecture of classification systems consists of the following three main steps: motion segmentation, object tracking and object classification [1] [2]. The steps are described as follows.

S. Johnsen is with the Institute for Reliability of Systems, Hamburg University of Technology, Eissendorfer Str. 40, 21073 Hamburg, Germany {swantje.johnsen}(at)tu-harburg.de

A. Tews is with the Commonwealth Scientific and Industrial Research Organisation (CSIRO) in Brisbane, Australia {ashley.tews}(at)csiro.au

In the motion segmentation step, the pixels of each moving object are detected. Generally, the motion segmentation consists of background subtraction and foreground pixel segmentation. Stauffer and Grimson [3] use the mixture of Gaussians to perform background subtraction and apply a two-pass grouping algorithm to segment foreground pixels. Simple and common techniques are based on frame differencing [4] or using a median filter [5]. In this work a technique based on the Approximated Median Filter [6] was used. Better results were obtained by introducing a step factor in the filter.

Following background subtraction, the mobile objects are tracked. Tracking of objects is the most important but error prone component. Problems arise when objects of interest touch, occlude and interact with each other, and when objects enter and leave the image. Israd and Blake [7] introduced a method termed CONDENSATION to track objects. Chen *et al*. [8] construct an invariant bipartite graph to model the dynamics of the tracking process. Stauffer and Grimson [3] use a linearly predictive multiple hypotheses tracking algorithm. Yang *et al*. [4] use a correspondence matrix and a merging and splitting algorithm to relate the measured foreground regions to the tracked objects. Many algorithms have been proposed in the literature, but the problem of multiple interacting objects tracking in complex scene is still far from being completely solved. Model based algorithms [9] are computationally more expensive, because the number of parameters to estimate the model is usually large. They are also sensitive to background clutter. Overall, many of those algorithms can only deal with partial object occlusions for a short duration and fail to deal with complete object occlusions.

In the classification step, the object type is determined. Classification of 3-dimensional moving objects from 2-dimensional images for known object classes is a highly complex task. Toth and Aach [10] use a feed-forward neural network to distinguish between human, vehicles, and background clutters. Rivlin *et al*. [11] use a Support Vector Machine to distinguish between a vehicle, a human and an animal. Zhang *et al*. [2] distinguish between cars, vans, trucks, persons, bikes and people groups. They introduced the error correction output code as a classifier. These techniques need to be trained via test sequences of the objects. Javed and Shah [1] produced an algorithm that does not need to be trained.

III. SYSTEM OVERVIEW

A system that observes an outdoor environment by a single static camera is developed and tested. The goal is to track objects like walking people or moving vehicles in view of

the camera and to determine their type and position. In Figure 1 the flow diagram of the system is shown. The motion segmentation step detects the moving objects using the current image in the image stream. This output (the moving objects) is required by the object tracking algorithm that provides the motion history of each object.

A particular characteristic of the tracking algorithm is its ability to track objects with complete occlusion for a long duration without knowledge about their shape or motion. The output of the tracking algorithm is used by the classification system. Our classification algorithm is a modified version of the system presented in Javed and Shah [1]. The algorithm uses on the motion history of each object and by determining the type of motion. Motion type is determined by any repeated, recurrent motion of the object's shape. This property is used to classify between people and vehicles.

The motion segmentation, tracking and classification steps are dependent on each other. Thus, the classification system would deliver inappropriate results, if one of the previous steps does not achieve good performance.

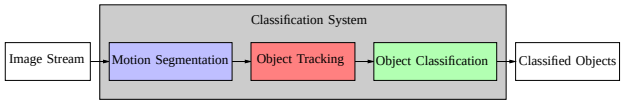


Fig. 1. Flow diagram of common classification systems.

The tests and experiments in this paper were conducted with a Canon VB-C50ir PTZ webcam. The maximal transmission rate of the camera is $25\,fps$ and it captures 768×576 resolution color images. Our system is developed in the c++ programming language on a 3.2 GHz Pentium D using the Open Source Computer Vision library (OpenCV).

IV. MOTION SEGMENTATION

An important condition in an object tracking algorithm as well as in an object classification algorithm is that the motion pixels of the moving objects in the images are segmented as accurately as possible. The common approach for motion segmentation consists of two steps: background subtraction and segmentation of foreground pixels.

A. Techniques of Background Subtraction

Background subtraction [12] identifies moving objects by selecting the parts of the image which differ significantly from a background model. Most of the background subtraction algorithms follow a simple flow diagram shown in Figure 2. Background modeling is a statistical description of the current background scene. Foreground pixel detection identifies the pixels in the current image that differ significantly from the background model and outputs them as a binary candidate foreground mask.

The Approximated Median Filter was chosen to perform background modeling. For our implementation, better results were obtained by scaling the increment and decrement by a step factor if the absolute difference between the current pixel and the median-modeled background pixel is bigger than a threshold.

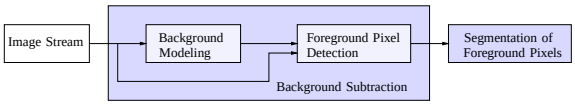


Fig. 2. Flow diagram of a general background subtraction algorithm.

Foreground pixels are detected by calculating the Euclidean norm at time t:

$$\|\mathbf{I_t}(x,y) - \mathbf{B_t}(x,y)\| > T_e \tag{1}$$

where $\mathbf{I_t}$ is the pixel intensity value, $\mathbf{B_t}$ is the background intensity value at time t and T_e is the foreground threshold or by checking

$$|I_{j,t} - B_{j,t}| > T_a \tag{2}$$

for $j = 1, ..., c$ where T_a is the foreground threshold,

$$\mathbf{I_t} = \begin{bmatrix} I_{1,t} & \dots & I_{c,t} \end{bmatrix}^T, \quad \mathbf{B_t} = \begin{bmatrix} B_{1,t} & \dots & B_{c,t} \end{bmatrix}^T \tag{3}$$

and c is the number of image channels. The foreground thresholds T_e and T_a are determined experimentally. The foreground pixels were detected by determining the threshold T_a.

B. Segmentation of Foreground Pixels

In the next step, foreground pixels are segmented into regions. Using the two-pass connected component labeling method [3], a bounded box is applied to the connected regions. After this step, only grouped regions with bordered rectangles are considered. Any remaining noise is removed in the second noise reduction step using a size filter [13]. Finally, blobs are merged if they intersect or if the distances between them are below a threshold depending on the object distance to the camera.

V. MULTIPLE OBJECT TRACKING WITH OCCLUSION HANDLING

The goal of tracking is to establish correspondences between objects across frames. Robust classification of moving objects is difficult if tracking is inaccurate. The flow diagram of the implemented object tracking algorithm is shown in Figure 3.

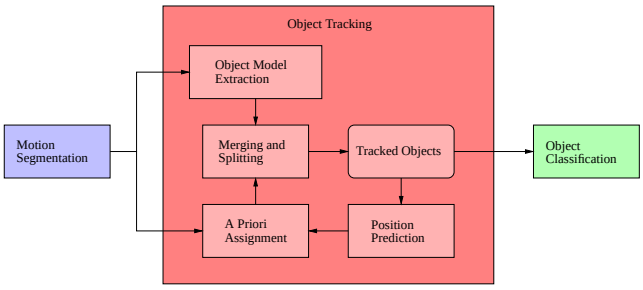


Fig. 3. Flow diagram of the multiple object tracking algorithm.

A. Object Model Extraction

A region-based model of the objects is extracted in this step. For every measured object, the normalized RGB color histogram is determined to uniquely identify an object. The histogram of an object was calculated by counting the number of pixels of the mask image within the rectangle that borders the object.

B. Position Prediction

In this step, the position of each tracked object on the plane is predicted by a Kalman filter. By using a homography the position measurement of each object is obtained. It is assumed that the objects are orthogonal to the plane and the lower points of the objects are touching the plane. Thus, the midpoint of the lower rectangle edge is chosen as the position and is projected onto the plane by the homography.

For the Kalman filter, a constant speed model is used. Thus, it is assumed that the accelerations of all objects are approximately zero except for noise to allow for non-constant object velocities. Each tracked object is modeled by one Kalman filter. The positions are also superimposed with noise since initially, the object velocities can not be estimated correctly due to absence of experience.

C. A Priori Assignment

In this step, the measured objects are *a priori* assigned to any existing tracks. Let $\hat{\mathbf{T}}_t^{1^-}, \hat{\mathbf{T}}_t^{2^-}, ..., \hat{\mathbf{T}}_t^{m^-}$ denote the predicted positions of tracked objects and $\mathbf{ME}_t^1, \mathbf{ME}_t^2, ..., \mathbf{ME}_t^n$ denote the positions of the measured objects on the plane at time step t. Then, the distance matrix $\mathbf{D}_t$ is computed based on the Euclidean norm as follows:

$$\mathbf{D}_t(i,j) = \|\hat{\mathbf{T}}_t^{i^-} - \mathbf{ME}_t^j\| < T_d, \tag{4}$$

for $i = 1,...,m$ and $j = 1,...,n$. It stores the distances between the predicted positions of the tracked objects and the positions of the measured objects. The rows of the distance matrix correspond to the existing tracks and the columns to the measured objects. If the distance is above threshold T_d, the element in the matrix will be set to infinity. The threshold T_d is determined experimentally. Based on analyzing the distance matrix, a decision matrix $\mathbf{J}_t$ at time step t is constructed. The number of rows and columns are the same number as in the distance matrix and all elements are set to 0. For each row in $\mathbf{D}_t$, find the lowest valued cell and increment the corresponding cell in $\mathbf{J}_t$. The same is done for the columns. Thus each cell in $\mathbf{J}_t$ has a value between zero and two.

Only if an element value of the decision matrix $\mathbf{J}_t$ is equal to two, the measured object is assigned to the tracked object and their correspondence is stored. All elements in the same row and column of the distance matrix $\mathbf{D}_t$ are updated to infinity and a new decision matrix $\mathbf{J}_t$ is constructed. This process is repeated until none of the elements in the decision matrix equals to two. The correspondence between the objects is calculated by the Bhattacharya distance:

$$BD(HT,HM) = \sum_{i=1}^{N_r \cdot N_g \cdot N_b} \sqrt{HT(i) \cdot HM(i)} > T_{co} \tag{5}$$

where HT is the color histogram of the tracked object and HM is the measured object with $N_r \cdot N_g \cdot N_b$ bins. The values $HT(i)$ and $HM(i)$ are the normalized frequencies of the bin i. If the Bhattacharya distance of the object histograms is below the correspondence threshold T_{co}, a correspondence between the objects is not given. The threshold is 1 for a correspondence and 0 for a non-correspondence.

After the *a priori* assignment the tracked and measured objects can be classified into the following three categories:

- matched tracked and measured objects,
- unmatched tracked objects and
- unmatched measured objects.

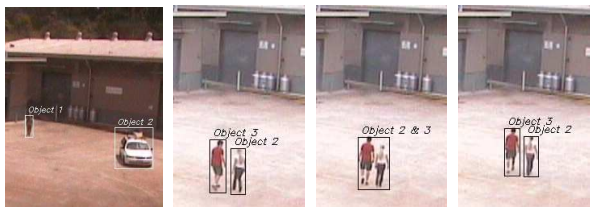
This step can not handle merging and splitting events, in which one measured object may be assigned to multiple tracks and one track may be assigned to multiple measured objects. A merging and splitting algorithm was developed to solve this problem.

D. Merging and Splitting

In this step, merging and splitting events are handled. Here, it is a valid assumption that as soon as objects touch each other, a large rectangle containing all objects is generated. Thus, the objects are not occluding each other at that time step. For tracked objects that are not matched to the measured objects, a merging detection algorithm is used to decide whether the track is merged with another track or it remains unmatched. If the track remains unmatched, its age increases until the object is assumed to be lost and therefore no longer significant. For unmatched measured objects, a splitting detection algorithm is developed. It decides whether the measured object is split from a tracked object or it is a new track.

E. Experimental Results

Three different scenes are chosen to represent the tracking algorithm. The first two scenes are demonstrated in Figure 4. A moving car and a walking person is shown in the leftmost figure. In the right three subfigures, two people merge and split. After the splitting, the individuals were identified correctly.



(a) (b) Before the merging. (c) After the merging. (d) After the splitting.

Fig. 4. Multiple object tracking (left). Merging and splitting of two people in a scene (right).

In figure 5, the third scene is demonstrated. In this scene, two people cross each other. During the crossing, one person occludes the other person. The persons are identified correctly after crossing. Note that complete occlusion of objects via other moving objects is handled correctly.

(a) Before the crossing. (b) During the crossing. (c) Occlusion. (d) After the crossing.

Fig. 5. Crossing of two people in a scene.

VI. OBJECT CLASSIFICATION

The goal is to classify each moving object visible in the input images as a single person, group or vehicle. Our approach to classify people and vehicles is based on [1]. The algorithm requires an appearance history of the object from the tracking algorithm by means of a bounding box (smallest possible rectangle bordering the mask of the object) and correspondence of each object over the frames. In most cases, the whole object is moving along with local changes in shape (mask of the object). Thus, the objects are classified by detecting repetitive *changes* in their shapes. In Figure 6, the flow diagram of the classification algorithm is presented.

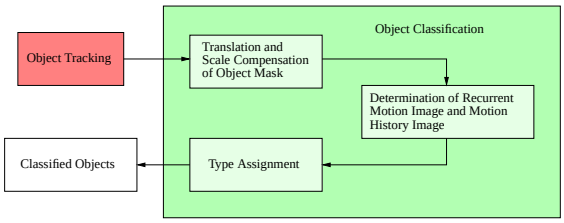


Fig. 6. The flow diagram of the classification algorithm.

These steps are explained in the following sections where an object mask is defined as the part of the mask image within the bounding box of the object.

A. Translation and Scale Compensation of Object Masks

A moving object often changes its position within the bounding box and its size. To eliminate effects of mask changes that are not due to shape changes, the translation and change in scale of the object mask over time needs to be compensated. The assumption is that the only reason for changes in the shape size is the variation of the object distance from the camera. The translation is compensated by aligning the objects in the images along its centroid. For compensation of scale, the object mask is scaled in horizontal and vertical directions such that its bounding box width and height are the same as of the first observation.

B. Determination of Recurrent Motion Image and Motion History Image

Let $A_t^i(x,y)$, for $i = 1,...,m$, be the pixel value of the translation and scale compensated object mask i at position (x,y) and at time t. Then, a difference image $D_t^i(x,y)$ is generated for each object $i = 1,...,m$ by using the exclusive-or operator $\oplus$ as follows:

$$D_t^i(x,y) = A_{t-1}^i(x,y) \oplus A_t^i(x,y). \quad (6)$$

The value $D_t^i(x,y)$ indicates the shape changes of the object. After this step, the Recurrent Motion Image (RMI) is calculated as follows:

$$RMI_t^i(x,y) = \frac{\sum_{k=0}^{\tau} D_{t-k}^i(x,y)}{\tau} \quad (7)$$

where τ is the time interval that should be large enough to capture the recurrent shape changes. The recurrent motion image has high values at those pixels whose shape changes repeatedly and low values at pixels where there are little shape changes or no shape changes at all.

Our classification algorithm is based on the work of Javed and Shah [1]. However, we found that it did not always correctly classify objects that change shape through turning. Henceforth, we enhanced their algorithm to increase robustness by providing a second metric for analysing motion - termed a 'Motion History Image'.

The Motion History Image (MHI) is a mask image that indicates where motion of the object occurred during the time interval τ. It is calculated as follows:

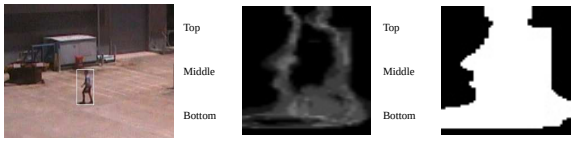
$$MHI_t^i(x,y) = \begin{cases} 0 & \text{if } \sum_{k=0}^{\tau} A_{t-k}^i(x,y) = 0 \\ MHI_{max} & \text{otherwise} \end{cases} \quad (8)$$

where MHI_{max} is the maximum value of the MHI.

C. Type Assignment

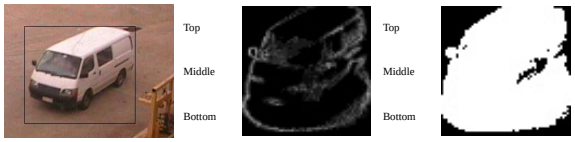
Once the recurrent motion and the MHI of the object is obtained, the type of the object needs to be classified. Therefore, the recurrent motion is divided into $o \times o$ equal sized square blocks and the mean value for each block is computed. The partitioning reduces the computation and the averaging reduces noise. Then, the corresponding MHI is computed by scaling it to an $o \times o$ image. In Figure 7, examples of averaged recurrent motion and scaled MHI are shown in three different scenes. As it can be seen, the ratio of recurrent motion to motion occurrence of the single person and the group in the bottom of the images is bigger than that of the van, because a van has no repeated changes in its shape.

The type assignment is also different to [1]. A Repeated Motion Ratio is introduced to distinguish between people and vehicles. The sum S_t^i of all mean values of the blocks in the bottom of the recurrent motion image at which the corresponding blocks of the MHI has its maximum value (motion has occurred) is determined for the objects $i = 1,...,m$ at time t. During this step, the number of the blocks $o_{p,t}^i$ in the bottom of the MHI with maximum value is counted. In the next step, the Repeated Motion Ratio is calculated by dividing the sum S_t^i by the number of blocks $o_{p,t}^i$ times the maximal value RMI_{max} of the recurrent motion image. The Repeated Motion Ratio is 1, if the recurrent motion image has its maximum mean value in every block at which the corresponding MHI indicates motion. That is, if the shape of the object changes repeatedly. If the recurrent motion image has its minimum mean value 0 in every block, the Repeated Motion Ratio is 0 as well which means that the shape of the object does not change repeatedly. Thus, the



(a) Single person walking. (b) Averaged RMI. (c) Scaled MHI.

(d) Two people walking. (e) Averaged RMI. (f) Scaled MHI.



(g) A moving van. (h) Averaged RMI. (i) Scaled MHI.

Fig. 7. Examples of RMIs and MHIs in different scenes.

object type single person or group is assigned to the object, if

$$RMR_t^i = \frac{S_t^i}{o_{p,t}^i \cdot RMI_{max}} > T_{ot} \tag{9}$$

where T_{ot} is the fixed decision threshold of the object type. If *RMR* is below that threshold, the object is classified as a vehicle. The threshold T_{ot} is determined experimentally.

D. Classification Results

The classification algorithm was applied to a variety of video sequences. They contain people walking and vehicles moving. Each sequence consists of 600 to 1000 frames. The tracking algorithm provides the bounding box and correspondence of each object over the images of each sequence. The classification algorithm was applied for each object after it has completely entered the image. The number of frames over which the recurrent motion and the motion history image were calculated is $\tau = 20$. Thus, a wrong data association do not have quite an impact on the recurrent motion and the motion history image The decision threshold of the object type is $T_{ot} = 0.12$. In Table I, the results of the classification algorithm distinguishing between people and vehicles are given. Even in presence of noisy mask images accurate classifications were obtained.

TABLE I

RESULTS OF THE OBJECT CLASSIFICATION ALGORITHM.

TYPE OF OBJECT	Classified as People	Classified as Vehicle
Single People	38	0
Vehicle	1	20

VII. ONLINE APPLICATION OF THE CLASSIFICATION SYSTEM

The classification system was applied online. The input image stream is handled by the DDX framework (Dynamic Data eXchange) developed by Corke *et al*. [14]. To acquire video live streams and controlling a camera the DDXVideo framework is used [15].

Three representative scenarios were chosen. In the first, a moving car enters the scene, stops, and a person egresses and both leave. Two people crossing each other are displayed in the second. During the crossing, one person occludes the other. In the third scenario, two people merge and split. The people occlude each other repeatedly when they are merged. The results are shown in Figures 8 to 10.

Fig. 8. First scene: Person and car.

In all tests, the objects are correctly tracked and identified. Further tests have shown that the classification system can achieve frame rates $33-50\,fps$.

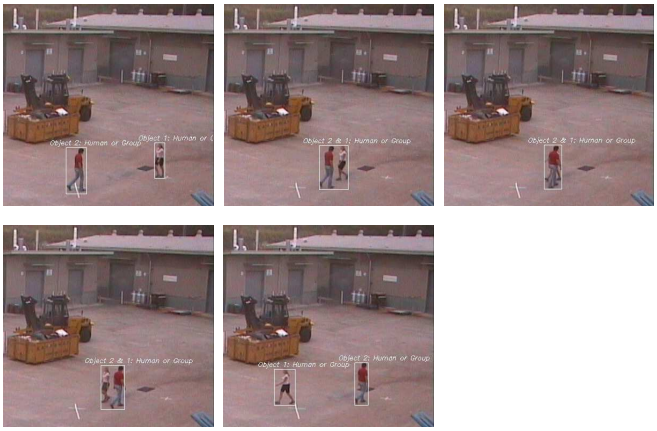


Fig. 9. Second scene: Two people cross each other.

We have also tested the algorithm on various vehicle types and in different types of weather. Figure 11 below show samples of a forklift in sunlight, and a bicycle rider in the rain - both mounted and unmounted. The bicycle rider case is interesting since the recurrent motion has a higher vertical component than in walking cases. The classifier gave the correct predictions in all cases.

VIII. CONCLUSIONS AND FUTURE WORK

We have demonstrated a vision based system for tracking and classifying dynamic objects in an outdoor environment. The system is based on [1] and shows improvements in the detection and classification of people and vehicles. The system can handle occlusions and has demonstrated good results over multiple objects in varying weather conditions. In each test case, the system accurately labeled the dynamic

(a) Before the merging. (b) After the merging. (c) During occlusion 1. (d) After occlusion 1.

(e) During occlusion 2. (f) After occlusion 2. (g) After the splitting.

Fig. 10. Third scene: Two people merge, occlude each other repeatedly and split.

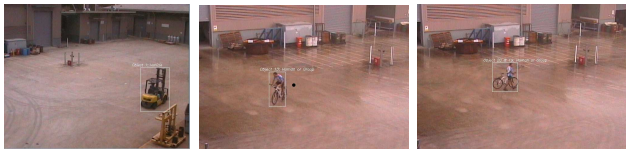


Fig. 11. Various types of dynamic objects have been used for testing the system in different weather conditions.

objects and tracked them correctly. The system works in real time and achieves a frame rate of $33-50\,fps$ for 768×576 resolution color images on a 3.2 GHz Pentium D computer. Our approach differs from existing approaches in that multiple objects are reliably tracked, even presence of occlusions, and the combination of using recurrent motion and Motion History Images improves classification and tracking performance.

The system is a preliminary step towards improving the situational awareness of either human-operated or autonomous vehicles working in joint workspaces. Being more aware of the environment makes operations safer and improves efficiency since better local path planning can result from knowing where potential path conflicts will occur and anticipatory steps taken to avoid them.

Within this work a basis of classification system was created. It is very efficient in terms of computational and space requirements. The next step is to develop a cast shadow algorithm in the motion segmentation step to create a good prerequisite for object tracking and classification under all lighting conditions. During the course of this research, several cast shadow algorithms were tested [8], [16] but none were robust or reliable enough in our test environment.

The object classifier of the system is also a basis for investigating further improvements. For example, a classifier could be developed that distinguishes between the different types of vehicles like cars, vans, trucks etc. or between single persons and groups. Furthermore the system could be optimized in its implementation to improve its speed. Introducing multiple camera viewing the scene in different angles would improve the object tracking and classification performance and robustness of the system.

REFERENCES

[1] O. Javed and M. Shah, *Computer Vision - ECCV 2002*, ser. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, 2002, vol. 2353/2002, ch. Tracking And Object Classification For Automated Surveillance, pp. 439–443.

[2] L. Zhang, S. Z. Li, X. Yuan, and S. Xiang, "Real-time Object Classification in Video Surveillance Based on Appearance Learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[3] C. Stauffer and W. E. L. Grimson, "Learning Patterns of Activity Using Real-Time Tracking," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, pp. 747–757.

[4] T. Yang, S. Z. Li, Q. Pan, and J. Li, "Real-time Multiple Objects Tracking with Occlusion Handling in Dynamic Scenes," in *Proceedings of the 2005 IEEE Computer Society on Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 970–975.

[5] R. Cucchara, C. Grana, M. Piccardi, and A. Prati, "Detecting Moving Objects, Ghosts, and Shadows in Video Streams," in *Transactions on Pattern Analysis and Machine Intellegence*, 2003.

[6] N. McFarlane and C. Schofield, "Segmentation and Tracking of Piglets in Images," in *Machine Vision and Applications*, vol. 8, 1995, pp. 187–193.

[7] M. Israd and A. Blake, "CONDENSATION - Conditional Density Propagation for Visual Tracking," in *Int. J. Computer Vision*, 1998, pp. 5–28.

[8] H.-T. Chen, H.-H. Lin, and T.-L. Liu, "Multi-Object Tracking Using Dynamical Graph Matching," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2001, pp. 210–217.

[9] A. Senior, A. Hampapur, Y. Tian, L. Brown, S. Pankanti, and R. Bolle, "Appearance models for occlusion handling," in *Proceedings 2nd IEEE Int. Workshop on PETS*, 2001.

[10] D. Toth and T. Aach, "Detection and Recognition of Moving Objects Using Statistical Motion Detection and Fourier Descriptors," in *Proceedings of the 12th International Conference on Image Analysis and Processing*, 2003, pp. 430–435.

[11] E. Rivlin, M. Rudzsky, R. Goldenberg, U. Bogomolov, and S. Lepchev, "A Real-Time System for Classification of Moving Objects," in *16th International Conference on Pattern Recognition*, vol. 3, 2002.

[12] C. Stauffer and W. E. L. Grimson, "Adaptive Background Mixture Models for Real-time Tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999.

[13] S. Gupte, O. Masoud, and N. P. Papanikolopoulos, "Vision-Based Vehicle Classification," in *Proceedings 2000 IEEE Intelligent Transportation Systems*, 2000, pp. 46–51.

[14] P. Corke, P. Sikka, J. Roberts, and E. Duff, "DDX: A Distributed Software Architecture for Robotic Systems," in *Proceedings of the Australian Conference on Robotics and Automation*, 2004.

[15] E. Duff, "DDXVideo: A Lightwight Video Framework for Autonomous Robotic Platforms," in *Proceedings of the Australian Conference on Robotics and Automation*, 2005.

[16] J. C. S. J. Jr and C. Jung, "Background Subtraction and Shadow Detection in Grayscale Video Sequences," in *Proceedings of the XVIII Brazilian Symposium on Computer Graphics and Image Processing*, 2005.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

A dioptric stereo system for robust real-time people tracking

Ester Martínez and Angel P. del Pobil
Robotic Intelligence Lab
Engineering and Computer Science Department
Jaume-I University
Castellón
Spain
Email: { emartine, pobil } @icc.uji.es

***Abstract*—We address and solve a number of problems in the context of a robot surveillance system based on a pair of dioptric (fisheye) cameras. These cameras provide a hemispherical field of view that covers the whole robot workspace, with some advantages over catadioptric systems, but there is little previous work about them. Then, we had to devise and implement a number of novel techniques to achieve robust tracking of moving objects in dynamic, unknown environments from color image sequences in real time. In particular, we present a new two-phase adaptive background model that exhibits a robust performance when there are unexpected changes in the scene such as sudden illumination changes, blinking of computer screens, shadows or changes induced by camera motion or sensor noise. The system is also capable of tracking the detected objects when they are not in movement. We also deal with fisheye camera calibration to estimate both intrinsic and extrinsic parameters, as well as the estimation of the distance between the system and the detected objects with our dioptric stereo system. Experimental results are reported.**

Robotics research, from its begining, has been always focused on building robots which help human beings in their daily tasks while both of them coexist in the same environment. That means new robot generations have to deal with dynamic, unknown environments, unlike industrial robots which act in a restricted, controlled, well-known environment. For that reason, one of the key issues in this context is to be aware of what is happening around.

In fact, robot performance in any real environment requires to detect people and/or other objects, particularly if they are moving, in the robot's workspace. On the one hand, interaction tasks require detection and identification of the objects with which to interact. On the other hand, the safety of all elements present in the robot workspace should be guaranteed at any time, specially when they are human beings. Thus, it is important that the robot quickly detects the presence of any moving element to be able to properly react to the element movements.

So, among the available robot sensors, cameras might be suitable for this goal, since they are an important source of information. Nevertheless, it is not straightforward to successfully deal with a non-constrained environment by using traditional cameras due to its limited field of view. That constrain could not be removed by combining several images captured by rotating a camera or strategically positioning a set of them, because it is necessary to establish any feature correspondence between many images at any time. This processing entails a high computational cost which makes them fail for real-time tasks.

An effective way is to combine mirrors with conventional imaging systems [1] [2] [3]. The obtained devices are called catadioptric systems. Moreover, if there is a single viewpoint, they are referred as central catadioptric systems [4]. This is a desired feature in such imaging systems since it describes world-image mapping. In fact, a single viewpoint implies that all rays go through a 3D point and its projection on the image plane goes through a single point in the 3D space. Conventional perspective cameras are devices of a single viewpoint, for example. Although the central catadioptric imaging can be highly advantageous, they unfortunately exhibit a dead area in the centre of the image what can be an important drawback in some applications.

With the aim of overcoming all the above drawbacks, a dioptric system was used. Dioptric systems, also called fisheye cameras, are systems which combine a fisheye lens with a conventional camera [4] [5]. Thus, a conventional lens is changed by one of these lenses which has a short focal length what allows cameras to see objects in an hemisphere. Although fisheye devices present several advantages in front of catadioptric sensors such as no presence of dead areas in the captured images, a unique model for this kind of cameras does not exist unlike central catadioptric ones [6].

In this work, we have focused on dioptric systems to implement a robot surveillance application for fast and robust tracking of moving objects in dynamic, unknown environments. Although our final goal is to design an autonomous, mobile manipulation robot system, here we present the first stage: novel techniques for robust tracking of moving objects in dynamic, unknown environments from color image sequences such that manipulation tasks could be safely performed in real time when the robot system is not moving. For that, three different related problems have been tackled:

- moving object detection
- object tracking
- distance estimation from the system to the detected objects

First of all, a new robust adaptive background model has been designed. It allows the system to adapt to different unexpected changes in the scene such as sudden illumination changes, blinking of computer screens, shadows or changes induced by camera motion or sensor noise. Then, tracking process from two omnidirectional images takes place. Finally, the estimation of the distance between the system and the detected objects must be done by using an additional method. In this case, information about the 3D localization of the detected objects with respect to the system was obtained from a dioptric stereo system.

Thus, the structure of this paper is as follows: the new robust adaptive background model is described in Section 2, while in Section 3 the tracking process is introduced. An epipolar geometry study of a dioptric stereo system is presented in Section 4. Some experimental results are presented in Section 5, and discussed in Section 6.

I. Moving Object Detection

Research in human and object detection has taken a number of forms. Well-known segmentation techniques from a taken image are thresholding or frame subtraction. However, on the one hand, it is difficult to deal with threshold selection when it is working with an unknown, dynamic environment and targets to track can have different features. Actually, the uncertainty provided by those specified work conditions also makes that automatic threshold search methods, mainly based on histogram properties, fail [7]. In that way, other experiments to obtain a robotic assistant in which a person is detected and then followed by a mobile robot [8] [9] [10] [11] have been carried out. Nevertheless, in spite of the fact that existing algorithms are very fast and easy to use, image processing for object identification is very poor since it is color- and/or face-based. This restricts their utility because it is not viable to track objects of a particular color, which has also to be significantly different from the background, or it constrains people to always face the vision system.

On the other hand, although the image difference method provides a good detection of changing regions in an image, it is important to pay attention to several uncontrolled changes in the system environment which can produce multiple false negatives and make the system fail. These dynamic, uncontrolled changes can be divided into:

- minor dynamic factors, such as, for example, blinking of computer screens, shadows, mirror images on the glass windows, curtain movement or waving trees, as well as changes induced by camera motion, sensor noise, non-uniform attenuation or atmospheric absorption, among other factors
- sudden changes in illumination such as switching on/off a light or opening/closing a window

Different research has been developed to adapt to this changes. One of the most common is the background subtraction approach, which has been proposed by several researchers [12] [13] [14] [15] [16]. Basically, a background model, which is built after observing the scene several seconds, is used to identify moving objects by thresholding the new frame with respect to the built background model. However, this approach presents two important drawbacks:

- everything observed when the background model is being built is considered background
- no sudden change in illumination occurs during the whole experiment

It is important to take into account that, unlike most of them which each background pixel is represented by a Gaussian distribution, Stauffer and Grimson [17] presented adaptive background mixture models. However, as it was pointed out in [18], some issues have to be solved.

Therefore, a novel algorithm is proposed here. It is divided into two different phases, as can be seen in Figs. 1 - 2:

1) In the first phase, an initial background model is obtained by observing the scene during several seconds. However, unlike most background estimation algorithms, another technique for controlling the activity within the robot workspace is performed. With the aim of reducing the computational and time cost, this control is performed by means of a simple difference technique. In that way, there is no danger to damage people who approach the robot while this initial model is being built. Thus, basically, in this phase, a simple frame-difference approach is performed in order to detect moving objects within the robot workspace. Then, two consecutive morphological operations are applied to erase isolated points or lines caused by the dynamic factors mentioned above. In this point, two different tasks are carried out:
   - On the one hand, adaptive background model is updated with the values of the pixels classified as background in order to adapt it to some small changes which do not represent targets
   - On the other hand, a tracking process, which is explained in the next sections, is performed
2) In the second phase, detection and identification moving object process starts. When a human or another moving object enters in a room where the robot is, it is detected by means of a two-level processing:
   - pixel level, in which the adaptative background model, initially built in the previous phase, is used to classify pixels as foreground or background. It is possible because each pixel belonging to the moving object has an intensity value which does not fit into the background model. That is, the used background model associates a statistical distribution (defined by its mean color value and its variance) to each pixel of the image. Then, when an interest object enters and/or moves around the robot workspace, there will be a difference between the background model values and object's pixel values. Actually, a criterion based on stored statistical information is defined to deal with this classification and it can be expressed as follows:

$$b(r,c) = \begin{cases} 1 & \text{if } |i(r,c) - \mu_{r,c}| > k \times \sigma_{r,c} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $b(r,c)$ is the binary value of the pixel (r,c) to be calculated, $i(r,c)$ represents pixel brightness in the current frame, $\mu_{r,c}$ and $\sigma_{r,c}$ are the mean and standard deviation values calculated by the background model respectively and k is a constant value which depends on the point distribution

- frame level, whereby the raw classification based on the background model is improved as well as the model is adapted when a global change in illumination occurs. A proper combination of substraction techniques has been implemented. In that way, a different segmentation process is applied at frame level and it is used to improve the segmentation carried out at pixel level. Furthermore, this processing allows the system to identify global illumination changes. That is, it is assumed that an significant illumination change has taken place when there is a change in more pixels than a half of the image size. When an event of this type occurs, a new adaptative background model is built because if it was not done, the application would detect background pixels as moving objects, since the model is based on intensity values and a change in illumination produces a variation of them.

As in the previous phase, after properly segmenting an image, two consecutive morphological operations are applied to erase isolated points or lines caused by small dynamic factors. Later, pixels classified as background are incorporated to the adaptive background model, while foreground pixels are processed by applying a tracking method.

II. Tracking Moving Targets

Once targets to be tracked have been identified, the next step is to track them. For that, first of all, a connected-component labeling algorithm is performed. However, due to segmentation errors, it might be obtained more than one labeled component for the same target. Thus, a merge algorithm, based on neighbourhood and feature similarity, is applied. Then, a minimum rounded rectangles are generated. After that, with the aim of performing the corresponding tracking, a pattern is built from each of them. In this case, a pattern is intended as the data structure such that allows the system to track the moving objects by means of matching an object in two consecutive frames even when it suffers a partial or whole occlusion.

Thus, in our case, a pattern is composed of two different things:

- a representative image of the target, that is, it is not possible to directly compare two images of the same object when they are provided by an omnidirectional image. It is due to the fact that every object has different

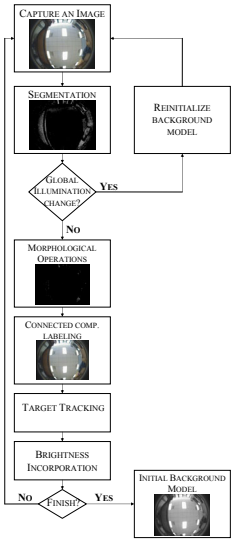


Fig. 1. Phase-1 flowchart of implemented two-phase algorithm for moving object detection in unknown, dynamic environments

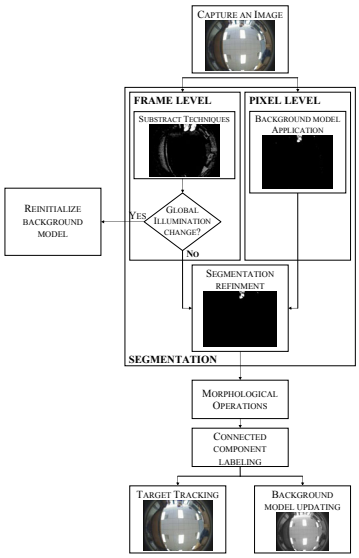


Fig. 2. Phase-2 flowchart of implemented two-phase algorithm for moving object detection in unknown, dynamic environments

orientation depending on its position inside the scene. So, several rotations would be necessary in order to correctly match the images of the same object in two different frames. Thus, it is necessary to apply a transformation from the circular omnidireccional image to a perspective one (see Fig. 3. This is done only for each region detected as object of interest since transformation of the whole image could become very high time-consuming.

- a feature array whose elements contain information about brightness and blob width and height, among other things, used to properly match to images of the same object in two consecutive frames as well as two stereo images

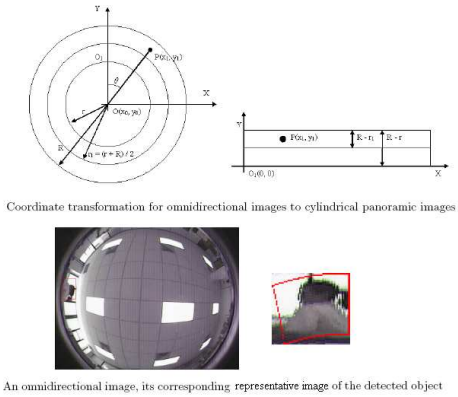


Fig. 3. Representative perspective image from the labeled omnidirectional image

Therefore, on the one hand, representative images are compared with the extracted from the previous frame or the another stereo image. In this way, a pixel-similarity likehood between representative images is obtained. On the other hand, a feature-similarity likehood is generated from feature array comparison. Both likehoods are properly combined to match two images from the same object in consecutive frames or frames taken from a dioptric stereo system.

III. Stereo System

For approximately determining the distance from the system to an objective we need to estimate the correspondence between omnidirectional images, that is, the epipolar equation.

From the point of view of stereo vision, an epipole is defined as the projection of the camera center on the image plane of another camera. Unlike traditional cameras, two epipoles are visible, since any camera is within the field of view of each other. For that reason, in the case of omnidirectional cameras, it is not necessary to use a third external object for stereo calibration. This is the idea in which Zhu et al [15] [16] based to implement a virtual stereovision system with a flexible baseline in order to detect, track and localize moving human subjects in unknown indoor environments. In the literature, other approaches developed for catadiotric systems can be found [16] [19] [20] [21]. However, even though there is almost no work with dioptric stereo systems, we have implemented a process to estimate the distance from the dioptric stereo system to the detected objects.

The guidelines of the distance estimation method are as follows:

- a matching process between images taken by different cameras is done. First, the adaptive background model at two levels is independently performed for the images captured by each camera. Then, each detected blob is described by means of a feature array whose elements contain information about brightness and blob width and height, among other things. Next, each feature array is compared with all the detected blobs in the frame taken by the other camera in the system, while the matching process included in the implemented moving object detection method is simultaneously performed. Thus, a similarity likelihood is calculated and a matching decision is made based on it.
- detection of the other camera in each frame, as it is depicted in Fig. 4. In fact, this step is necessary to be performed only once because the baseline of the stereo system is fixed.
- estimation of the distance with respect to each camera from triangulation geometry, as it is shown in Fig. 4. The triangle to solve is determined by the projection ray of a 3D point of the real world on each plane image as well as the projection ray of the center of the other camera of the system. It is possible thanks to the calibration step since the projection rays can be estimated, as mentioned above.

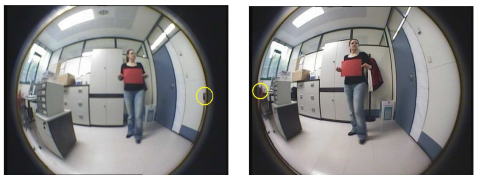


Fig. 4. Camera detection in images captured by the two cameras

IV. Experimental Results

Two different experiments have been carried out to check the designed application performance. First, the performance of the moving object detection was evaluated by using only one fisheye camera. After that, the obtained estimation of the detected object distance through our dioptric stereo system was analysed. In this section, some of these results are provided.

A. Experimental set up

For both kinds of experiments carried out, a mobile manipulator which incorporates a visual system composed of 2 fisheye cameras mounted on the robot base, pointing upwards to the ceiling, to guarantee the safety in its whole workspace. Figs. 5 depicts our experimental setup, which consists of a mobile Nomadic XR4000 base, a Mitsubishi PA10 arm, and two fisheye cameras (*SSC-DC330P third-inch color cameras* [22]

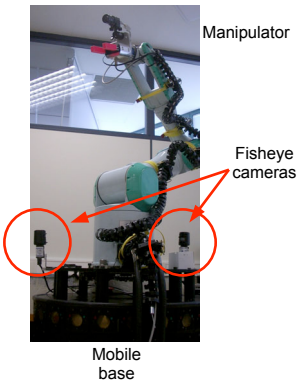


Fig. 5. Experimental setup: external view of the arm and cameras.

with *fish-eye vari-focal lenses YV2.2x1.4A-2*, which provide 185 degree field of view).

Thus, on the one hand, a single fisheye camera was used to evaluate the moving object detection performance. The robot system was located in the center of our laboratory where almost all the space was covered and where most of the uncontrolled, dynamic factors named above were present (e.g. blinking of computer screens, shadows, mirror images on the glass windows or variations in illumination due to the different time of the day or the switch on/off a light). On the other hand, the dioptric system was used. In both cases, the images to process were acquired in 24-bit RGB color space with a $640x480$ resolution.

B. Moving Object Detection Evaluation

As it was pointed out, the first series of experiments were to evaluate the performance of the novel adaptative, robust background model. For that, illumination conditions and object positions were changed. Two sequences of the images as a result of applying the novel updated background model under the same illumination conditions is depicted in Fig. 6. As it can be seen, the method is able to visually track moving objects without constraints such as clothes color or illumination.

In a similar way, illumination conditions were changed and, as it is shown in Fig. 7, the obtained results were also successful.

V. Conclusions

In this paper, a robust visual application to detect and track moving objects within a robot workspace has been presented based on a pair of fisheye cameras. These cameras have the clear advantage of covering the whole workspace without resulting in a time consuming application, but there is little previous work about this kind of devices. Consequently, we had to implement novel techniques to achieve our goal.

Thus, the first subgoal was to design a process to detect moving objects within the observed scene. After studying several factors which can affect the detection process, a novel adaptive background model has been implemented where contraints such as waiting a period of time to build the initial background or illumination conditions do not exist. In a similar

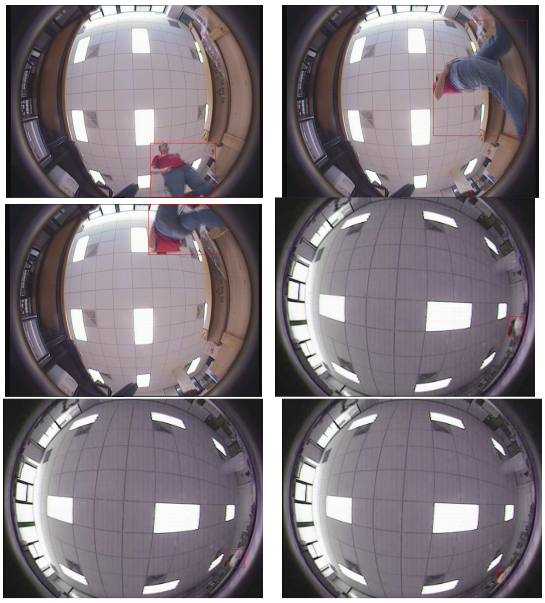


Fig. 6. Results of applying the novel adaptative background model with different subjects

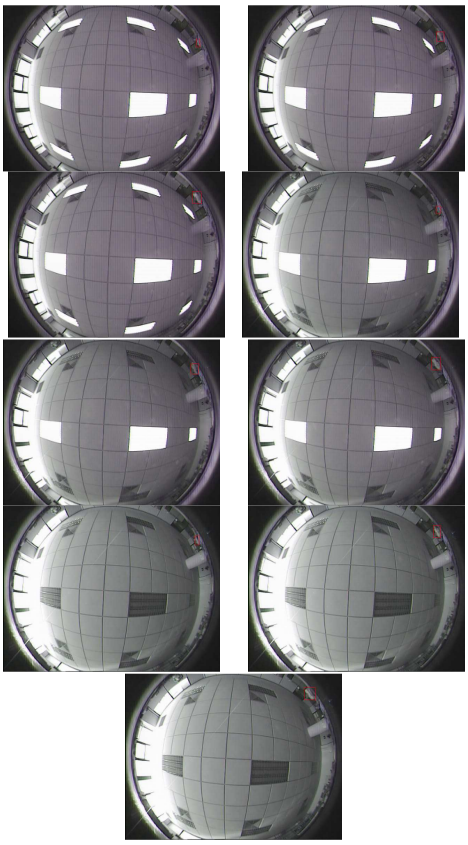


Fig. 7. Results of applying the novel adaptative background model under different illumination conditions

way, it is also capable of tracking the detected objects when they are not in movement. In addition, the designed method includes a matching process between two consecutive frames.

The next step is to estimate the distance from the detected objects to the system. For that, a stereo dioptric system with fixed baseline has been built. Therefore, it was necessary to perform a calibration process in order to obtain the fundamental matrix. Three different toolboxes were tested, but only two were used in the end. Finally, a method to estimate distance from the objects to the system was implemented. In this case, a triangulation technique is used. It is possible to perform because the cameras can see each other. It must be taken into account that epipolar geometry of the stereo dioptric systems was not used, although the combination of that with the current implementation in order to improve the accuracy in the matching process is part of our future work.

ACKNOWLEDGMENTS

This paper describes research carried out at the Robotic Intelligence Laboratory of Universitat Jaume I. Support for this laboratory is provided in part by the European Commission under project EYESHOTS (FP7 ICT-217077), by Fundacio Caixa-Castello under project P1-1B2005-28 and by Ministerio de Ciencia under project DPI2004-01920 and FPI grant BES-2005-8860.

REFERENCES

[1] T. Svoboda, T. Pajdla, and V. Hlavác, "Epipolar geometry for panoramic cameras," in *European Conf. on Computer Vision (ECCV'98)*, Freiburt Germany, July 1998, pp. 218 – 231.

[2] S. C. Wei, Y. Yagi, and M. Yachida, "Building local floor map by use of ultrasonic and omni-directional vision sensor," in *Int. Conf. on Robotics and Automation*, Leuven, Belgium, May 1998, pp. 2548 – 2553.

[3] S. Baker and S. K. Nayar, "A theory of single-viewpoint catadiptric image formation," *Int. Journal of Computer Vision*, vol. 35, no. 2, pp. 175 – 196, 1999.

[4] ——, "A theory of catadioptric image formation," in *Int. Conf. on Computer Vision (ICCV'98)*, Bombay, India, 5–8 January 1998, pp. 35 – 42.

[5] R. W. Wood, "Fish-eye views, and vision under water," *Philosophical Magazine*, vol. 12, no. Series 6, pp. 159 – 162, 1906.

[6] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical applications," in *European Conf. on Computer Vision (ECCV 2000)*, Dublin, Ireland, 26th June – 1st July 2000, pp. 445 – 461.

[7] S. L. G. and S. G. C, *Computer vision*, U. S. River, Ed. Prentice Hall, 2001.

[8] B. Kwolek, "Color vision based person following with a mobile robot," in *Third Int. Workshop on Robot Motion and Control (RoMoCo'02)*, November 2002, pp. 375 – 380.

[9] M. Tarokh and P. Ferrari, "Case study: Robotic person following using fuzzy control and image segmentation," *Robotic Systems*, vol. 20, no. 9, pp. 557 – 568, 2003.

[10] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera and a laser," in *2006 IEEE Int. Conf. on Robotics and Automation (ICRA'06)*, Orlando, Florida, May 2006, pp. 557 – 562.

[11] T. Yoshimi, M. Nishiyama, T. Sonoura, H. Nakamoto, S. Tokura, H. Sato, F. Ozaki, N. Matsuhira, and H. Mizoguchi, "Development of a person following robot with vision based target detection," in *2006 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Beijing, China, October 2006, pp. 5286 – 5291.

[12] K. Toyama, J. Krum, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Seventh IEEE Int. Conf. on Computer Vision*, vol. 1, Kerkyra, Greece, 1999, pp. 255 – 261.

[13] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809 – 830, August 2000.

[14] H. Liu, W. Pi, and H. Zha, "Motion detection for multiple moving targets by using an omnidirectional camera," in *IEEE Int. Conf. on Robotics, Intelligent Systems and Signal Processing*, vol. 1, Changsha, China, October 2003, pp. 422 – 426.

[15] Z. Zhu, K. D. Rajasekar, E. M. Riseman, and A. R. Hanson, "Panoramic virtual stereo vision of cooperative mobile robots for localizing 3d moving objects," in *IEEE Workshop on Omnidirectional Vision*, 12th June 2000, pp. 29 – 36.

[16] Z. Zhu, D. R. Karuppiah, E. M. Riseman, and A. R. Hanson, "Keeping smart, omnidirectional eyes on you. adaptive panoramic stereovision for human tracking and localization with cooperative robots," *IEEE Robotics and Automation Magazine*, pp. 69 – 78, December 2004, special Issue on Panoramic Robotics.

[17] S. C. and G. W.E.L., "Adaptive background mixture models for real-time tracking," in *Conference on Computer Vision and Pattern Recognition (CVPR'99)*, vol. 2, 23rd – 25th June 1999, pp. 246 – 252.

[18] K. P. and B. R., "An improved adaptive background mixture model for real-time tracking with shadow detection," in *2nd European Workshop on Advanced Video Based Surveillance Systems (AVBS 01), VIDEO BASED SURVEILLANCE SYSTEMS: Computer Vision and Distributed Processing*, K. A. Publisher, Ed., September 2001.

[19] C. Geyer and K. Daniilidis, "Properties of the catadioptric fundamental matrix," in *The 7th European Conf. on Computer Vision (ECCV2002)*, vol. 2, LNCS 2351, Copenhagen, Denmark, 27th May – 2nd June 2002, pp. 140 – 154. [Online]. Available: http://link.springer.de/link/service/series/0558/tocs/t2351.htm

[20] Y. Negishi, J. Miura, and Y. Shirai, "Calibration of omnidirectional stereo for mobile robots," *2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2004)*, vol. 3, pp. 2600 – 2605, 28th September – 2nd October 2004.

[21] S. Li and K. Fukumori, "Spherical stereo for the construction of immersive vr environment," in *IEEE Virtual Reality (VR'05)*, 12th – 16th March 2005, pp. 217 – 222.

[22] http://www.infodip.com/pages/sony/camera/pdf/SSC-DC58AP.pdf.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Experimental Evaluation of a People Detection Algorithm in Dynamic Environments

Dario Lodi Rizzini, Stefano Caselli
RIMLab - Robotics and Intelligent Machines Laboratory
Dipartimento di Ingegneria dell'Informazione
University of Parma, Italy
E-mail {dlr,caselli}@ce.unipr.it

***Abstract*— People detection is an important capability both for human-robot interaction in service robotics and to distinguish the stable environment from the perturbation due to people motion in localization and mapping tasks. Several techniques have been proposed for different application contexts and sensors. Range data acquired by laser scanners are metrically accurate and suitable for computationally-inexpensive people detection. Furthermore, laser scans provide a geometric description of local environment that can be combined with the information about dynamic objects.**

In this paper, a previously proposed method for detecting people legs from laser scans is experimentally evaluated and exploited to improve scan matching by removing dynamic parts corresponding to people. This algorithm splits laser scans into beam segments and classifies each segment. Classifications of simple features are then combined into a boosted classifier with Adaboost. The fundamental assumption of scan matching is that consecutive scans can be aligned with a rigid body transformation, since they represent the same scene. When dynamic elements like human legs are removed from scans, such assumption holds. We also investigate the effectiveness of the proposed people detection algorithm in terms of its ability to generalize across different environments and to support track persistency across scans.

I. Introduction

The aim of service robotics is the execution of tasks for people care. As a mobile service robot moves in an environment populated by people, robot-human interaction is therefore a fundamental requirement. Furthermore, even if the tasks to be performed do not involve people care, the recognition of dynamic elements including people is required for localization and mapping. Indeed, localization and mapping algorithms usually assume the complete state hypothesis. According to this hypothesis, the evolution of the system consisting of the robot and the static environment is completely described by the state variables. State usually includes robot location and map descriptors, but it does not consider human presence. Thus, such assumption is strongly violated in populated environments. Solutions for this problem range from filtering the dynamic obstacles to classifying and tracking them.

Several approaches have been proposed for people detection depending on the available sensor data and the context of application. The most popular sensors are cameras and range finders. Range finders have the advantage of limited processing requirements. Limiting our survey to laser-based robot applications, the approaches can be divided into tracking oriented techniques and geometric rule classifiers. The first category includes simple extensions of localization or SLAM algorithms [1], [2], [3] or specifically designed techniques [4], [5]. The second category includes all the methods that perform a classification using the features extracted from laser scans [6], [7], [8]. However, the above categorization remains arbitrary since tracking and feature detection are typically mixed together.

In this paper, we experimentally evaluate the algorithm for detecting people proposed in [7] that combines several feature based classifiers to perform a more robust estimation according to Adaboost boosting technique. This method has the advantage of performing people detection on a single scan without depending on a specific tracking technique or on assumptions about motion of people.

Furthermore, we use this algorithm to improve scan matching performance in a populated environment and apply the concept of track persistency to the classification results. The fundamental assumption of scan matching is that consecutive scans can be aligned with a rigid body transformation, since they represent the same scene. As discussed before, this assumption is violated in a populated environment, but the people detection algorithm can be used to filter out people presence. Our contribution lies in the experimental evaluation of the robustness of a scan matching technique and of the improvement allowed by people filtering. A further application of people classification relies on the concept of persistency [8]. A track corresponding to a person is persistent if the segment associated to the given track in each scan is often classified correctly. The evaluation of the persistency of people tracks yielding the potential of speeding-up the training of the classifier is the final contribution of the paper.

The paper is organized as follows. Section II briefly describes the algorithm for people detection. Section III illustrates the application of the classifier to improve the scan matching problem and the possibility to exploit track persistency for semi-unsupervised training. Section IV presents the experimental results. Finally, section V summarises the paper drawing some conclusions and perspectives.

II. People Detector

This section illustrates the algorithm for detecting people and its application to recognize dynamic and stable elements in the environment. The basic people detection algorithm

has been adapted from [7], as described next. The algorithm operates on a single laser scan in order to find if any subset in the range readings of the scan corresponds to a person as described in the following. First, the scan is divided into groups of adjacent range values called segments. Second, the algorithm classifies the segments establishing their correspondence to people legs. The classifier is achieved by combining several elementary classifiers that operate by extracting a specific feature from the segment and evaluating the value of such feature.

In literature, the outlined method for combining weak decisors in order to reduce the classification error is known as *boosting*. Adaboost algorithm is one of the most extensively used boosting algorithm [9]. The input of the algorithm is the training set, a set of examples (the scan segments in this case) labeled with the result of correct classification. Adaboost builds the final classifier by iteratively executing a learning round. During each round, the weak classifiers are trained using the examples of training set and the classifier that minimizes the classification error is selected for the round. The classification error is computed by weighting the error of each misclassified example. Weights are larger for the examples that have been wrongly classified in previous rounds. The classification error is then used to compute the coefficient that measures the contribution of the weak classifier to the decision.

Adaboost is a meta-algorithm that does not impose the form of the weak classifier. For people detection based on laser scans, since the features extracted from each segment are represented by a scalar, the weak classifiers $h_j(\cdot)$ have the following fixed expression

$$h_j(e) = \begin{cases} true & \text{if } \mathrm{p_j f_j(e)} < \mathrm{p_j}\theta_\mathrm{j} \\ false & \text{otherwise} \end{cases} \quad (1)$$

where e is the item to be classified (the segment), $f_j(e)$ is the feature extracted from e, θ_j is the decision threshold and $p_j \in \{+1, -1\}$ gives the direction of inequality. This form is suggested in [10] and adopted for the people detector in [7].

A. Feature Definition

The features used in the described classifier are scalar values computed from a scan segment. As explained above, a segment is a set of consecutive range values of a laser scan approximately corresponding to a distinguishable object of the environment. Segmentation is an important step of the algorithm, which is sometimes neglected. In the experiments section, it will become apparent how segmentation affects the final result. In this paper, a simple splitting technique has been used. The range values of the scan are traversed in counterclockwise order and, when the jump distance of a range reading with respect to the previous reading is above a threshold, a new segment starts. Segments including only one range reading are discarted. Comparing with the original proposal in [7], it is unclear whether our segmentation technique exactly reproduces the original approach; if not, this is the only significant difference between the original algorithm and our implementation.

The range values of the segment are then transformed into cartesian coordinates with respect to the local reference frame fixed on the sensor. Depending on the feature, polar or cartesian coordinates are used. For each segment, we used the same set of 14 features proposed in the original paper, that are listed in the following.

1) *Number of points*.
2) *Standard deviation*: it is the mean distance from the mean value of the points of the segment.
3) *Mean average deviation from median*: it is a more robust version of the previous feature that uses the median point instead of the mean point. The median point coordinates are given by the 0.5 percentiles of the distribution of x and y coordinates of points.
4) *Jump distance from preceding segment*.
5) *Jump distance to succeeding segment*.
6) *Width*: it is the Euclidean distance between the first and the last point.
7) *Linearity*: it is the sum of square distances between each segment point and the regression line computed using the same points.
8) *Circularity*: it is the sum of square distances between each segment point and the regression circle computed using the same points. The regression circle is achieved according to least square criterion. When only two points are available, circularity is set to a large value.
9) *Radius*: it is the radius of the regression circle. When only two points are available, the radius is set to a large value.
10) *Boundary length*: it is the sum of the distances between consecutive segment points. It corresponds to the length of the boundary defined by the poli-line that connects each pair of points.
11) *Boundary regularity*: it is the standard deviation of the line
12) *Mean curvature*: it is the average value of the curvatures computed on triplets of consecutive points.
13) *Mean angular difference*: it is the average value of the angles computed on triplets of consecutive points.
14) *Mean speed*: it is the average speed of the range readings of the segment. The computation of range speed requires the value of the given range reading on the current and previous scans and the time interval between the acquisition of the two scans. Mean speed is the only feature that requires temporal correlation between two consecutive scans.

III. Multiple Scans Applications

The algorithm for detecting people described above has the remarkable advantage of performing a classification using only the geometric information available in a single scan, without requiring temporal correlations between scans. The only exception is represented by feature 14 that usually gives a negligible contribution to the boosted classifier as will be shown in section IV. Indeed, an algorithm that does not rely on temporal correlations is easier to implement and to test, since there is no specific constraint on the order

of the scans. Moreover, the detection is independent from the motion state of the people and of the robot carrying the laser scanner. Common experience suggests that a person usually moves in an environment, but a robust people detector cannot rely on this assumption. In contrast to other techniques exploiting tracking, the illustrated algorithm does not require arbitrary dynamic models. However, this method can be easily integrated into a tracking system. In this subsection, we describe the application of the people detector to two different problems both related to temporal proximity: alignment of scan pairs and scan segment tracking.

A. People Filtering

Scan matching is the problem of finding a rigid motion that makes a laser scan overlap another reference scan. The fundamental assumption is that the two scans to be aligned share the representation of a region of the environment. Such hypothesis usually holds when the second scan is collected from a location near to the reference location and the environment is static. However, if there are people moving in the environment, such assumption is clearly violated. While several scan matching algorithms may be robust to such violations, the people detector can improve the performance of the scan matcher by removing the perturbation caused by human presence. We call this operation *people filtering* hence after. The effects of such correction are not easy to illustrate and to evaluate. First, if the motion of an object is too slow when compared to the frequency of acquisition, the object appears still in two consecutive scans. Second, the scan matcher can recover the values of translation and rotation from the fixed background that often dominates the scans. More details and results on people filtering are reported in section IV.

B. Track Persistency

A second application of the people detection algorithm exploits temporal correlation between scan segments to minimize the acquisition cost of training set. The described boosted classifier learns the value of internal parameters (the thresholds of weak classifiers, the weights, etc.) in a supervised training phase. Currently, the examples in the training set are manually labeled, but manual classification is a tedious and time-consuming operation. It would be convenient to perform a partially automatic labelling of the collected segments, at least to expand the existing training set. The concept of *track persistency* proposed in [8] could be used for this purpose. The original aim of this proposal is the unsupervised training of moving obstacles classifiers in a multi-sensor architecture. First, moving obstacles are detected as persistent tracks in the data acquired from a given sensor source. Second, these data are labeled as moving obstacles and are used to train a classifier.

Persistency can be applied to improve the performance of the described feature-based classifier learnt from an initial training set. In a typical scenario, one or more people move in a trajectory and their legs are repeatedly observed by a range finder. A perfect leg detector would find a segment for each person (or two segments, if the legs are distinguishable) in every scan and it would be possible to associate such segment to another segment corresponding to the same person in the previous scan. Thus, a persistent track could be found for each person and for fixed obstacles. Such correspondences between segments are not found in one of these cases:

- when the segmentation is not properly done;
- when the tracked person exits the visibility area;
- when the tracking algorithm fails;
- when the leg detector wrongly classifies the current or previous segment.

The latter case is the most interesting one because, if a wrongly classified segment is detected, it can be added to the training set and used to train a better classifier.

IV. Results

The aim of this section is to report the experimental evaluation of the legs detector described in previous sections and of the correction on scan matching error achieved with such algorithm. The experimental setup consists of an ActivMedia Pioneer I equipped with a Sick LMS 200 laser scanner. The scanning plane is $29.7\ cm$ high from the ground floor.

Experiments reported in the following have been performed in the Computer Engineer building of the University of Parma. The robot moved in different positions to collect scans from different locations in the environment during both training and evaluation steps. Figures 1(a)-(b) show two settings used in the tests: the hallway of Computer Engineering building and the Robotics laboratory. The two settings capture different kinds of rooms: the hallway is long and narrow and allows robot motions; the laboratory is full of obstacles, table and chairs legs that can be mistaken for human legs. The choice of the two rooms is similar to the one suggested in [7]. The main hallway of the Faculty building in Figure 1(c) was used only for the scan matching tests, since a larger environment was required. The robot moved to several places for each locations to collect training set data, but it stayed at a fixed position during the acquisition. The robot moved only during scan matching tests. Since classification is performed on a single scan, the motion of the robot is not significant for performance assessment.

The method described in this paper has been implemented independently from the original version described in [7]. The illustration given in the paper was sufficient to reimplement the same algorithm. Thus, the results shown in this section provide an independent validation for such technique. The differences between the two versions may depend only on the value of few parameters and on the segmentation procedure. A scan is split into a new segment when the jump between two consecutive ranges is greater than a given threshold, that has been set to $18\ cm$ for these experiments. Such a simple solution works quite well in almost all the considered cases, even if segments representing legs are sometimes confused with the background.

(a) (b) (c)

Fig. 1. Views of the experimental environments: (*a*) hallway of the Computer Engineering building; (*b*) laboratory; (*c*) hallway of the Faculty building.

A. Experiments with People Detector

The first set of experiments has been devoted to the assessment of leg detector performance. Scans have been acquired from the hallway and the laboratory in Figure 1(a)-(b) with people inside as discussed above. A third collection of scans has been acquired in the hallway after inserting additional obstacles of several sizes and shapes, since the hallway contained few obstacles during the first acquisition. Thus, three settings will be initially considered: hallway, hallway with obstacles and laboratory. During the experiments one or two people moved in the area.

The acquired training set and test set contain respectively 300 and 341 scans. The number of segments extracted from the training set is 5798, but only 2713 segments contain more than one point. In the global test set, there are 3315 segments consisting of more than one point on a total amount of 10259. This result is a consequence of the simple segmentation technique that splits when the jump distance is above the threshold. In order to improve the efficiency of the classifier and to avoid classification of segments with a single range reading, we considered as eligible segments only those with more than one range reading value.

	Detected Label (Hallway Training Set)		
True Label	Person	No Person	Total
Person	454 (90.98%)	45 (9.02%)	499
No Person	84 (8.76%)	875 (91.24%)	959
	Detected Label (All Training Set)		
True Label	Person	No Person	Total
Person	424 (84.97%)	75 (15.03%)	499
No Person	94 (9.80%)	865 (90.20%)	959

TABLE I

RESULTS OF PEOPLE DETECTION IN THE HALLWAY WITH FEW OBSTACLES.

Tables I, II and III show the results for the hallway, the hallway with more obstacles and the laboratory. In these three tables, the top part provides the results obtained with the classifier trained with the data of the specific environment and the bottom part the results obtained with the timing data collected from all the environments. In all three settings, the latter classifier generally performs worse than the specifically trained one, which performs correct detection, in average, on 90% of cases. The globally trained classifier only seems to

	Detected Label (Hallway Obs. Training Set)		
True Label	Person	No Person	Total
Person	93 (83.78%)	18 (16.22%)	111
No Person	24 (4.26%)	539 (95.74%)	563
	Detected Label (All Training Set)		
True Label	Person	No Person	Total
Person	107 (96.40%)	4 (3.60%)	111
No Person	263 (46.71%)	300 (53.29%)	563

TABLE II

RESULTS OF PEOPLE DETECTION IN THE HALLWAY WITH OBSTACLES.

	Detected Label (Laboratory Training Set)		
True Label	Person	No Person	Total
Person	143 (89.94%)	16 (10.06%)	159
No Person	135 (13.18%)	889 (86.82%)	1024
	Detected Label (All Training Set)		
True Label	Person	No Person	Total
Person	146 (91.82%)	13 (8.18%)	159
No Person	277 (27.05%)	747 (72.95%)	1024

TABLE III

RESULTS OF PEOPLE DETECTION IN THE LABORATORY.

reduce the number of false negatives for the hallway with obstacles and the laboratory, but it increases the number of false positives. We remark that the statistics illustrated above do not include the one-point segments that are filtered before performing the classification. Otherwise, the number of correct "no people" classifications for hallway, hallway with obstacle and laboratory would increase respectively of 1638, 691 and 1021.

	Test Set		
Training Set	Hallway	Hallway Obs.	Laboratory
Hallway	91.15%	64.10%	72.44%
Hallway Obs.	83.20%	93.77%	73.37%
Laboratory	83.54%	68.99%	87.24%

TABLE IV

COMPARISON OF TRAINING SETS.

In order to gain some insight into the potential for environment generalization of the people detection algorithm, Table IV compares the percentage of correct classification achieved with classifiers learnt from different training sets. The hallway with obstacles and laboratory classifiers provide

the best global performance, hinting that richer environments should be used to favor generalization. The features that

Environment	Five best features
Hallway	9, 4, 4, 3, 7
Hallway Obs.	9, 7, 3, 11, 13
Laboratory	4, 3, 12, 9, 7
All	2, 7, 9, 7, 3

TABLE V

THE BEST FIVE FEATURES FOR EACH CLASSIFIER.

allow better results (Table V) are *radius* (9), *mean average deviation from median* (3), *jump distance* (4), and *linearity* (7). They are almost the same features reported in [7]. From a general viewpoint, our experimental results confirm with independent implementation and assessment, the results reported in [7].

B. Evaluation of People Filtering

The aim of the second set of experiments is the evaluation of the impact of people dynamics on operations that assume a static world. In particular, the described leg detector can be directly exploited for all the methods that extract geometric information from laser scans. For example, scan matching allows the estimation of local robot motion by aligning a pair scans acquired in two different locations. Scan matching presumes that two consecutive scans overlap on the common region when the correct rigid motion is applied. However, if the two scans contain segments corresponding to dynamic objects like people, the relative position between these segments and the environment may change.

In these experiments, the illustrated classifier is used to filter the scan segments corresponding to legs that should not be considered in scan alignment. The robot moved with a mean speed of $0.2\ m/s$ acquiring a laser scan approximatively every $100\ ms$. Experiments were performed in two environments. The first environment is the hallway of the Faculty shown in Figure 1(c). Since the people leg detector was not trained in this setting, we used the classifier trained with the Hallway Obs. dataset. The size of this environment allowed the robot to cover a path of about $25\ m$. The second environment is the hallway of the Computer Engineering building (Figure 1(a)), where classifier performance was tested. One or two people were walking in the environment at moderate speed. A standard scan matcher based on *iterative closest point* (ICP) algorithm [11] has been used. Since no ground truth information was available, the final robot pose estimated using scan matching on filtered scans has been compared with the final pose estimated on raw scans. These data do not represent a real error, but a displacement between two different evaluations.

Table VI illustrate the displacements for each coordinate of robot pose obtained in the experiments in the Faculty hallway. The overall position displacement is $9.3\ cm$ on a distance of about $25\ m$. Thus, the scan matcher is only affected to a limited extent by the presence of people. A second experiment was performed in a setting where the

Final Coordinate	$x\ (m)$	$y\ (m)$	$\theta\ (m)$
People Filtering	16.026	9.006	0.0114
No People Filtering	15.942	8.967	0.0180

TABLE VI

RESULTS OF SCAN MATCHING WITH PEOPLE FILTERING IN THE FACULTY HALLWAY.

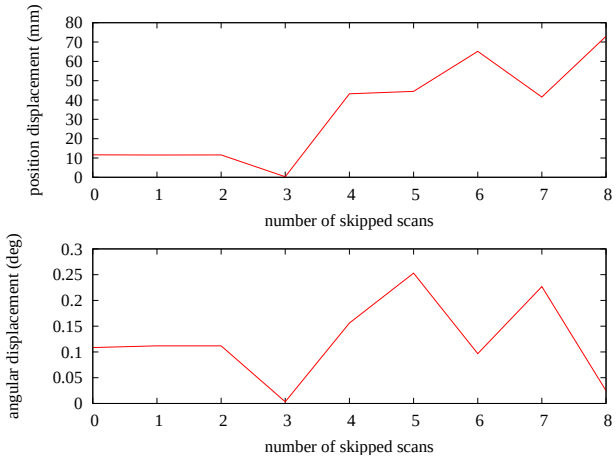


Fig. 2. Displacements between the final position (top) and between the final angle (bottom) of robot estimated by scan matching with leg detection enabled and disabled in Computer Engineering building. The displacement changes with the number of skipped scans.

people leg detector has been trained and tested. Figure 2 shows the position (top) and the angular (bottom) displacements varying with the number of skipped scans for a single experiment. In several cases, the human motion is slow when compared with the frequency of sensor acquisition and dynamic objects could be considered approximately static in two consecutive scans. Scans are then skipped to simulate systems subjected to computational load that cannot perform scan matching between all the pairs of scans or people moving at a faster rate. Note that the angular displacement is negligible even if the number of skipped scans is increased. Thus, orientation is not affected by people, at least in an environment like the considered hallway that has the strong reference provided by the parallel walls. On the other hand, the position displacement increases from about $1\ cm$ to $7\ cm$, when 8 scans are skipped, even though not monotonically. Thus, the evaluation of position is less robust than the evaluation of orientation, but scan matching is only marginally improved by people removal.

We interpret these results as follows. Motion estimation techniques like those based on scan matching may have the ability to filter out people presence, especially when only few slowly-moving people occlude a small portion of the scan. However, people filtering technique may play an important role for more cluttered and complex environments.

C. Towards Semi-unsupervised Labelling

The third set of experiments is devoted to the evaluation of track persistency as a criterion for semi-unsupervised segment labelling.

For this experiment, a simple tracking algorithm has been implemented. Each segment belonging to a scan is associated to the nearest segment of the previous scan. The considered distance is the distance between the centers of the two segments. Such naive association criterion is sufficient to achieve the results illustrated in the following, but a better segment matching would improve performance. For example, an accurate association rule should evaluate whether more segments correspond to the same object, e.g. two legs belonging to the same person. The robot moved in the hallway (Figure 1(a)) and acquired laser scans from the environment, while one or two people wandered in front of the range finder. Robot motion is estimated by matching pairs of consecutive scans and each scan is filtered removing the segments corresponding to legs as explained above. Such estimation is used to move the segments of the previous scan before performing the association.

	Track Persistency Percentage	
Sequence number	People	No People
1	79.13%	96.96%
2	83.23%	96.87%
3	92.27%	96.82%
4	89.29%	97.14%

TABLE VII

TRACK PERSISTENCY PERCENTAGE

The first parameter evaluated is the persistency of people tracks and no people tracks. A segment with a given classification is called persistent if it is associated to another segment with the same classification. The persistency of a category track can then be measured by the ratio between the number of persistent segments and the total number of segments belonging to this category. Table VII shows the persistency ratio of people tracks and of no people tracks for four scan sequences acquired in the hallway. People persistency is about 80% for two sequences and 90% in the other two sequences. Such high values demonstrate that both the classifier and the tracking system work quite well.

However, we are interested in the remaining $10 - 20\%$ of positively classified segments that are associated to negatively classified segments in the previous scan. Such negative segments are possibly false negative. Currently, the tracking algorithm is not sufficiently accurate to make a decision and to add them to the training set.

V. CONCLUSION

In this paper, we experimentally evaluated an algorithm for detecting people based on boosted features and tested two multiple scan applications. In particular, we found that the people detector performs a correct classification in about 90% of cases, when the training set has been acquired in the same environment of the test set. The percentage decreases when a different training set is used. Differences between the results illustrated in this paper and in [7], where the algorithm was proposed, may be related to the segmentation method.

Furthermore, the classifier has been applied to filter people presence and improve scan matching in a populated environment. Experimental results demonstrate that scan matching is robust to the violation of the static environment assumption and that people filtering marginally modifies the estimation. However, further experiments in cluttered and complex environments are likely to emphasize the benefits provided by the people detector.

The third contribution of this paper is the experimental evaluation of track persistency over a sequence of scans. A track corresponding to a person is persistent if the segment associated to the given track in each scan is often classified correctly. Experiments illustrate that the people detection algorithm recognizes tracks with high persistency and only in few cases a person segment is not associated to another segment classified in the same way. Such track interruptions are due to several reasons, but may correspond to false positives of the algorithm. Therefore, persistency may be used to improve people detection by collecting misclassified segments for further training.

VI. ACKNOWLEDGMENTS

The authors gratefully thank Filippo Agazzi, Paolo Bertacchini and Gabriele Novelli for their help in setting up the experiments. This reaserch is partially supported by laboratory AER-TECH of Regione Emilia-Romagna, Italy.

REFERENCES

[1] C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.

[2] D. Haehnel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.

[3] D. Schulz, W. Burgard, D. Fox, and A. Cremers, "People tracking with mobile robots using sample-based joint probabilistic data association filters," *Journal of Robotics & Autonomous Systems*, vol. 12, no. 3, pp. 99–116, 2003.

[4] A. Fod, A. Howard, and M. Mataric, "Laser-based people tracking," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2002.

[5] K. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.

[6] J. Xavier, M. Pacheco, D. Castro, and A. Ruano, "Fast line, arc/circle and leg detection from laser scan data in a player driver," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[7] K. Arras, O. Martinez Mozos, and W. Burgard, "Using boosted features for the detection of people in 2D range data," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2007.

[8] R. Katz, B. Douillard, J. Nieto, and E. Nebot, "A self-supervised architecture for moving obstacles classification," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008.

[9] R. Shapire and Y. Singer, "Improved boosting algorithms using confidence-rate predictions," *Machine Learning*, vol. 37, no. 3, pp. 297–336, 1999.

[10] P. Viola and M. Jones, "Robust real-time objects detection," in *Proc. of IEEE Workshop on Statistical Theories of Computer Vision*, 2001.

[11] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Robust Stereo-Based Person Detection and Tracking for a Person Following Robot

Junji Satake and Jun Miura
Department of Information and Computer Sciences
Toyohashi University of Technology

***Abstract*— This paper describes a stereo-based person detection and tracking method for a mobile robot that can follow a specific person in dynamic environments. Many previous works on person detection use laser range finders which can provide very accurate range measurements. Stereo-based systems have also been popular, but most of them have not been used for controlling a real robot. We propose a detection method using depth templates of person shape applied to a dense depth image. We also develop an SVM-based verifier for eliminating false positive. For person tracking by a mobile platform, we formulate the tracking problem using the Extended Kalman filter. The robot continuously estimates the position and the velocity of persons in the robot local coordinates, which are then used for appropriately controlling the robot motion. Although our approach is relatively simple, our robot can robustly follow a specific person while recognizing the target and other persons with occasional occlusions.**

***Index Terms*— Person detection and tracking, Mobile robot, Stereo.**

I. Introduction

Following a specific person is an important task for service robots. Visual person following in public spaces entails tracking of multiple persons by a moving camera.

There have been a lot of works on person detection and tracking using various image features and classification methods [1], [2], [3], [4], [5]. Many of them, however, use a fixed camera. In the case of using a moving camera, foreground/background separation is an important problem.

This paper deals with detection and tracking of multiple persons for a mobile robot. Laser range finders are widely used for person detection and tracking by mobile robots [6], [7], [8]. Image information such as color and texture is, however, sometimes necessary for person segmentation and/or identification. Omnidirectional cameras are also used [9], [10], but their limited resolutions are sometimes inappropriate for analyzing complex scenes.

Stereo is also popular in moving object detection and tracking. Beymer and Konolige [11] developed a method of tracking people by continuously detecting people using distance information obtained from a stationary stereo camera.

Howard et al. [12] proposed a person detection method which first converts a depth map into a polar-perspective map on the ground and then extracts regions with largely-accumulated pixels. Calisi et al. [13] developed a robot system that can follow a moving person. It makes an appearance model for each person using stereo in advance. In tracking, the robot extracts candidate regions using the model and confirms it using stereo. Occlusions between people are not handled in these works.

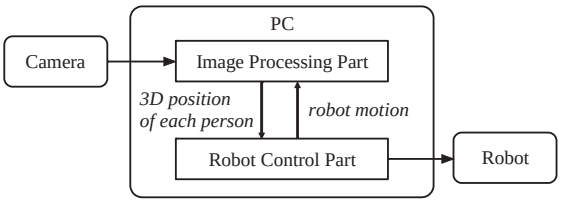


Fig. 1. Configuration of our system.

Ess et al. [14], [15] proposed to integrate various cues such as appearance-based object detection, depth estimation, visual odometry, and ground plane detection using a graphical model for pedestrian detection. Although their method exhibits a nice performance for complicated scenes, it is still costly to be used for controlling a real robot.

In this paper, we propose a person tracking method using stereo. We prepare several *depth templates* to be used for dense depth images and detect person regions by template matching, followed by a support vector machine (SVM)-based verifier. Depth information is very effective in data association with adjusting template size and values as well as occlusion handling. Person detection results are input to Extended Kalman Filter-based trackers. The robot continuously estimates the position and the velocity of persons in the robot local coordinates to appropriately control its motion. Fig. 1 shows the configuration of our system. The main contribution of the paper is to show that a simple depth template-based approach, combined with EKF and an SVM-based verifier, realizes a robust person following by a mobile robot.

II. Stereo-based person detection and tracking

To track persons stably with a moving camera, we use *depth templates*, which are the templates for human upper body in depth images (see Fig. 2); we currently use three templates with different direction of body. We made the templates from the depth images where the target person was at $2\,[m]$ away from the camera. A depth template is a binary template, the foreground and the background value are adjusted according to status of tracks and input data.

A. Tracking

For a person being tracked, his/her predicted scene position is available from the corresponding EKF-tracker (see

Fig. 2. Depth templates.

Sec. III-B). We thus set the foreground depth of the template to the predicted depth of the head of the person and search a region around the predicted image position for the person.

Concerning the background depth, since it may change as the camera moves, we estimate it on-line. We make the depth histogram of the current input depth image and use the Kth percentile as the background depth (currently, $K = 90$).

For a depth template $T(x, y)$ of $H \times W$ pixels ($x \in [-W/2, W/2]$, $y \in [-H/2, H/2]$) and the depth image $I_D(x, y)$, the 2D image position (x^*, y^*) is given as the position which minimizes the following SSD (sum of squared distances) criterion:

$$\sum_{p=-W/2}^{W/2} \sum_{q=-H/2}^{H/2} \left[T(p, q) - I_D(x + p, y + q)\right]^2 . \quad (1)$$

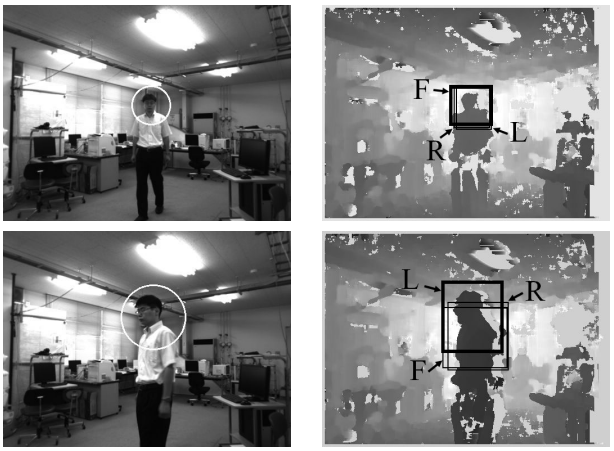
We use the three templates simultaneously and take the one with the smallest SSD value as the detection result if that value is less than some threshold.

Each template has the *position of the head* and the median value of the neighboring region of that position is used as the depth from the camera of the detected person. The accuracy of the depth value is empirically estimated as about one percent when a person is at about $3\,[m]$ distance.

B. Detection

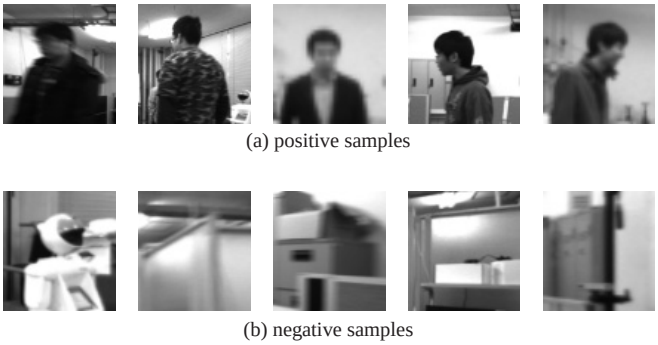
We continuously check if a new person appears in the image. In this case, we do not have any prediction and basically search the entire image. The foreground depth is set to the depth of each image position and the background one is set as in the same way as tracking.

We use the same SSD criterion (see eq. (1)) for judging if a person exists at an image position. Since applying this SSD calculation to the entire image is costly, we examine the three boundary points, on the left of, on the right of, and above the head position, and only when the depth values of the points are one-meter farther than the depth of the head, the SSD value is calculated and evaluated. We also set a *detection volume* to search in the scene; its height range is $0.7 \sim 2.0\,[m]$ and the range of the depth from the camera is $0.5 \sim 5.5\,[m]$. In addition, if the image position under consideration is in an already-detected person region, and unless the its depth is at least one-meter smaller than the depth of the region, the detection there is skipped. These techniques can reduce the search cost largely. After collecting pixels with qualified SSD values, we extract the mass centers of all connected regions as the positions of newly detected persons.



(a) Input images (b) Depth images

Fig. 3. Detection examples using depth templates.



(a) positive samples

(b) negative samples

Fig. 4. Training samples for the SVM-based verifier.

Figure 3 shows examples of detection using the depth templates. Three rectangles in each depth image are detection results with the three templates, and the one with the highest evaluation value is shown in bold line. Even when the direction of the body changed, it is possible to detect a person stably by using multiple templates.

C. Intensity-based false detection elimination

A simple template-based detection is effective in reducing the computational cost but at the same time may produce many false detections for objects with similar silhouette to person. To cope with this, we use an SVM-based person verifier using intensity images.

We collected many person candidate images detected by the depth templates, and manually examined if they are correct. Fig. 4 shows some of positive and negative samples. We used 438 positive and 146 negative images for training. The size of the sample images is normalized to 20×20. The SVM is the one with RBF kernel ($K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\gamma||\boldsymbol{x}_1 - \boldsymbol{x}_2||^2)$, $\gamma = 8.0$). We use an OpenCV implementation of SVM.

We examine the performance of the SVM-based verifier using three image sequences, which had not used for training. The numbers of persons appearing in the sequences are zero, one, and two, respectively. We used the image regions

TABLE I

PERFORMANCE SUMMARY OF THE SVM-BASED VERIFIER.

# of persons	results		
0		judged to exist	judged not to exist
	exist	—	—
	not exist	0	126
1		judged to exist	judged not to exist
	exist	414	5
	not exist	0	75
2		judged to exist	judged not to exist
	exist	391	31
	not exist	0	491

detected using the depth templates. Table I summarizes the results. It is noted that the rate of eliminating false positives is 100%. This is very important because a simple depth template-based person detection tends to produce many false positives. On the other hand, the verifier sometimes eliminates actual person regions; the false negative rate is about six percent. The EKF-based tracker can usually cope with such an occasional failure of person detection.

III. PERSON TRACKING AND ROBOT CONTROL

A. Configuration of our system

Figure 5 illustrates the coordinate systems attached to our mobile robot and stereo system. The relation between the robot and the camera coordinate system is given by

$$Z_c \left[\begin{array}{cccc} x & y & 1 \end{array} \right]^T = \boldsymbol{A} \left[\begin{array}{c|c} \boldsymbol{R} & \boldsymbol{T} \end{array} \right] \left[\begin{array}{cccc} X_r & Y_r & Z_r & 1 \end{array} \right]^T, \quad (2)$$

where $\boldsymbol{A}$, $\boldsymbol{R}$, and $\boldsymbol{T}$ show the intrinsic parameters matrix, the rotation matrix, and the translation vector, respectively.

B. Estimation of 3D position using EKF

1) State equation: In the robot coordinate system, the person's position at time t is defined as (X_t, Y_t, Z_t). The state variable $\boldsymbol{x}_t$ is defined as

$$\boldsymbol{x}_t = \left[\begin{array}{ccccc} X_t & Y_t & Z_t & \dot{X}_t & \dot{Y}_t \end{array} \right]^T,$$

where $\dot{X}_t$ and $\dot{Y}_t$ denote velocities in the horizontal plane.

We first consider the case where the robot does not move. The system equation is given by

$$\boldsymbol{x}_{t+1} = \boldsymbol{F}_t \boldsymbol{x}_t + \boldsymbol{G}_t \boldsymbol{w}_t \quad (3)$$

where $\boldsymbol{w}_t$ is the process noise and

$$\boldsymbol{F}_t = \left[\begin{array}{ccccc} 1 & 0 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array} \right], \quad \boldsymbol{G}_t = \left[\begin{array}{cc} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \Delta t & 0 \\ 0 & \Delta t \end{array} \right],$$

$$\boldsymbol{Q}_t = Cov(\boldsymbol{w}_t) = E\left[\boldsymbol{w}_t \boldsymbol{w}_t^T\right] = \sigma_w^2 \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right].$$

We then consider the case where the robot moves. Figure 6 shows how a wheeled mobile robot moves. The distance of two wheels is denoted as $2d$. When each wheel rotates with

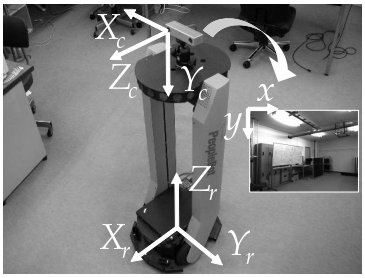


Fig. 5. Definition of coordinate systems.

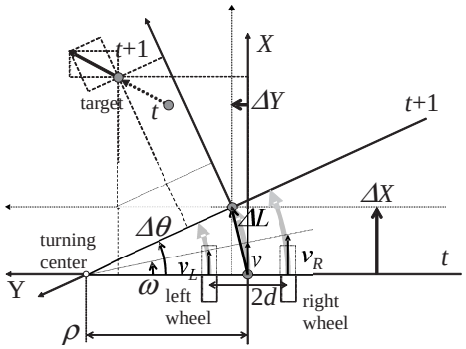


Fig. 6. Control of wheeled mobile robot.

speed v_L and v_R, the velocity v, the angular velocity ω, and the turning radius ρ of the robot have the following relations:

$$v = (v_R + v_L)/2, \quad \omega = (v_R - v_L)/2d,$$
$$\rho = d(v_R + v_L)/(v_R - v_L).$$

The rotation angle $\Delta\theta$ and the moved distance ΔL during time Δt are obtained respectively as

$$\Delta\theta = \omega \Delta t, \quad \Delta L = 2\rho \sin(\Delta\theta/2).$$

In addition, the robot movement ΔX and ΔY seen from the robot position at time t are obtained respectively as

$$\Delta X = \Delta L \cos(\Delta\theta/2), \quad \Delta Y = \Delta L \sin(\Delta\theta/2).$$

We then have the relationship between the position and the velocity of a person before and after the coordinate transformation from the robot coordinate at time t to that at time $t+1$ as follows:

$$X^{(t+1)} = (X^{(t)} - \Delta X)\cos\Delta\theta + (Y^{(t)} - \Delta Y)\sin\Delta\theta,$$
$$Y^{(t+1)} = -(X^{(t)} - \Delta X)\sin\Delta\theta + (Y^{(t)} - \Delta Y)\cos\Delta\theta,$$
$$\dot{X}^{(t+1)} = \dot{X}^{(t)}\cos\Delta\theta + \dot{Y}^{(t)}\sin\Delta\theta - v,$$
$$\dot{Y}^{(t+1)} = -\dot{X}^{(t)}\sin\Delta\theta + \dot{Y}^{(t)}\cos\Delta\theta.$$

By the combination of these equations and eq. (3), the state equation that considers the robot movement $\boldsymbol{u}_t = [v_L \ v_R]^T$ is expressed as

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_t(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{G}_t \boldsymbol{w}_t, \quad (4)$$

where

$$\boldsymbol{f}_t(\boldsymbol{x}_t, \boldsymbol{u}_t) = \begin{bmatrix} (X_t + \Delta t\dot{X}_t - \Delta X)\cos\Delta\theta + (Y_t + \Delta t\dot{Y}_t - \Delta Y)\sin\Delta\theta \\ -(X_t + \Delta t\dot{X}_t - \Delta X)\sin\Delta\theta + (Y_t + \Delta t\dot{Y}_t - \Delta Y)\cos\Delta\theta \\ Z_t \\ \dot{X}_t\cos\Delta\theta + \dot{Y}_t\sin\Delta\theta - v \\ -\dot{X}_t\sin\Delta\theta + \dot{Y}_t\cos\Delta\theta \end{bmatrix}.$$

2) Observation equation: The observed person's position in the robot coordinate system is denoted as $\boldsymbol{y}_t$. The observation equation is expressed as

$$\boldsymbol{y}_t = \boldsymbol{H}_t\boldsymbol{x}_t + \boldsymbol{v}_t, \quad (5)$$

where $\boldsymbol{v}_t$ is the observation noise and

$$\boldsymbol{y}_t = \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix}, \quad \boldsymbol{H}_t = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{R}_t = Cov(\boldsymbol{v}_t) = E\left[\boldsymbol{v}_t\boldsymbol{v}_t^T\right] = \sigma_v^2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

3) Extended Kalman filter: The Extended Kalman filter (EKF) are formulated using the the state eq. (4) and the observation equation (5). The EKF can estimate the position and the velocity of a person with their uncertainty estimates.

C. Data association and occlusion handling

3D position information is effective in data association. We use the predicted 3D position to adjust the size and the foreground depth of the depth templates to be used (see Section. II-A). If a person is detected, then its 3D position is tested with the Mahalanobis distance to see if the matching can be made between the detected person and the corresponding track.

3D information is also used for occlusion handling. In the case where an occlusion relation is reliably predicted between two persons, if an occluding one is correctly detected, only the prediction step in EKF is performed for the occluded person. Possible occlusion relationships are enumerated by examining the predicted 3D positions of tracks.

In an ordinary situation, persons pass each other with keeping a certain distance (say, one meter) between them. In our current setting, this distance difference can be detected as long as they are within about four meters from the camera; this is enough for the robot to correctly recognize the person motion in a local region around the robot.

Figure 7 shows an example of correctly tracking two persons under occlusion and depth change. In the middle row of the image, the person behind is completely occluded and only the prediction step in EKF is performed. After the occlusion, the track continues correctly.

D. Tracking algorithm

The image processing part (see Fig. 1) works as follows:
1) Stereo processing: The depth image is made with a stereo camera.
2) Person tracking: Each person is tracked by using the EKF described in Section III-B.

Fig. 7. Correctly tracking two persons in an occlusion case.

2.1) Prediction: The 3D position and its uncertainty at the current time t are predicted from the state variable at the previous time $t-1$. They are then projected to 2D image by eq. (2). The projected uncertainty region is used for determining the predicted region.
2.2) Observation: The predicted region is searched for the person by the method described in Section II-A. The templates used for search are made based on the depth to the person. After the search, the person's 3D position $\boldsymbol{y}_t$ is calculated by eq. (2) based on image coordinates (x, y) and distance from camera $Z_c = D$.
2.3) Data association: Correspondences are made between tracks and observations by the procedure described in Section III-C.
2.4) Update: The state variable is updated, if an observation is obtained.
3) Detection: The persons who appear newly in image are detected with depth templates.
4) Communication: The estimated position is sent to the robot control part, and the rotational speeds of the left and right wheels are received.

IV. Control to follow a specific person

The robot with two-wheel drive follows a circular trajectory from the current to the target position (path A in Fig. 8). In this case, the speeds for the wheels to move the robot at velocity v is calculated as follows. From the equation:

$$\rho^2 = \left\{(X/2)^2 + (Y/2)^2\right\} + \left\{(X/2)^2 + (\rho - Y/2)^2\right\},$$

we have

$$\rho = (X^2 + Y^2)/2Y.$$

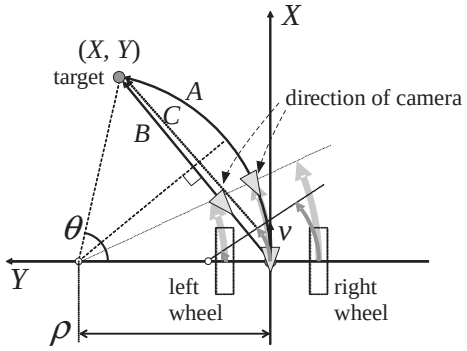


Fig. 8. Path to target position.

Then we can calculate the velocities as:

$$\begin{aligned} v_L &= v\left(1-\tfrac{d}{\rho}\right) = v\left(1-\tfrac{2dY}{X^2+Y^2}\right), \\ v_R &= v\left(1+\tfrac{d}{\rho}\right) = v\left(1+\tfrac{2dY}{X^2+Y^2}\right). \end{aligned}$$

When the robot follows this circular path, however, since the turning rate of robot orientation is relatively slow, the target person tends to go out of the field of view. On the other hand, the robot first turns and then moves straight toward the target like path B, the robot movement is not smooth. We thus use the one like path C, on which the robot turns to the target while moving ahead. In this case, the velocity of each wheel is adjusted as follows:

$$v_L = v\left(1-k\tfrac{2dY}{X^2+Y^2}\right), \quad v_R = v\left(1+k\tfrac{2dY}{X^2+Y^2}\right).$$

This means the turning radius ρ is reduced to ρ/k.

V. Experimental result

A. Experimental setup

We have implemented the proposed method on a PeopleBot (by Mobile Robots) with a Bumblebee2 stereo camera (by PointGrey Research) for the experiments (see Fig. 5). A note PC (Core2Duo, 2.6GHz) performs all processes including stereo calculation, person detection and tracking, and robot motion control. The processed image size is 512×384 and the processing time is about $90\,[msec/frame]$. Table II shows the breakdown of processing time; our system can process about eleven frames per second.

We implemented the software modules for person detection and tracking, motion planning, and robot control as *RT components* in the *RT-middleware* environment [16] for easier development and maintenance.

TABLE II
Breakdown of processing time.

Processing	Time
1) Image acquisition & stereo processing	40 [ms]
2) Person tracking (in the case of two persons)	20 [ms]
3) Person detection	10 [ms]
4) Communication, display, and save data	20 [ms]
Total	90 [ms]

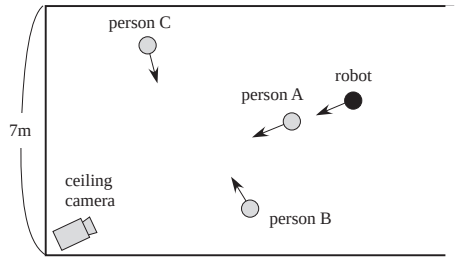


Fig. 10. Initial positions of the robot and the persons.

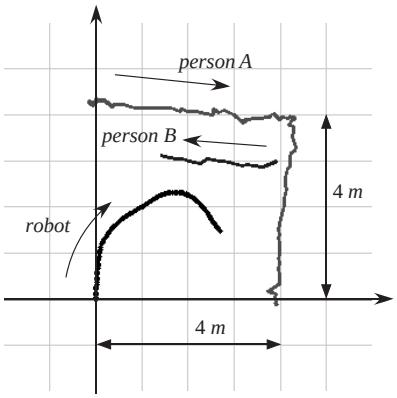


Fig. 11. Trace of two persons and the robot.

B. Person following experiments

Figure 9 shows a result of tracking. The left row images are the results of person detection. Each circle in the image shows the result of observation with depth templates, and each small point shows the 3D head position estimated using EKF. The right row images show the positions of the robot and the persons taken by a ceiling camera. In addition, the curves in the final frame (#156) shows the traces of the robot and the persons.

Figure 10 shows the initial positions of the robot and the persons. The robot moved toward person A who was detected first and considered the target. Even when person B and C passed between the robot and person A, the target person was correctly tracked.

C. Evaluation of person position estimation

We evaluated the quality of the person position estimation. Figure 11 shows the traces of the robot and two persons in the robot initial coordinates. Person A moved on two edges of a $4 \times 4\,[m]$ square drawn on the floor. Person B moved so that it temporarily occluded person A.

The robot followed person A while estimating the positions of the persons. The averaged and the maximum error in position estimation for person A were $125\,[mm]$ and $336\,[mm]$, respectively. This result shows that the position estimation is accurate enough for the robot to follow a specific person.

Fig. 9. Experimental result with one person to follow and the other two.

VI. Conclusions and Future Work

This paper has described a method of detecting and tracking multiple persons for a mobile robot by using distance information obtained by stereo. We presented an EKF-based formulation by which the robot continuously estimates the position and the velocity of persons. Distance information is effectively utilized for robust person detection, data association, and occlusion handling. We realized a robot that can robustly follow a specific person while recognizing the target and other persons with occasional occlusions.

The current algorithm does not consider the case where multiple persons are too close to be separated by depth information. To cope with such cases, it would be necessary to use other visual information such as color and texture. It is also necessary to manage static obstacles such as furniture as well as an effective path planning to realize a person following robot that can operate in more complex environments.

Acknowledgment

The authors would like to thank Yuki Ishikawa for his help in implementing the system. This work is supported by NEDO (New Energy and Industrial Technology Development Organization, Japan) Intelligent RT Software Project.

References

[1] P. Viola, M.J. Jones, and D. Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. *Int. J. of Computer Vision*, Vol. 63, No. 2, pp. 153–161, 2005.

[2] N. Dalal and B. Briggs. Histograms of Oriented Gradients for Human Detection. In *Proceedings of 2005 IEEE Conf. on Computer Vision and Patttern Recognition*, pp. 886–893, 2005.

[3] B. Han, S.W. Joo, and L.S. Davis. Probabilistic Fusion Tracking Using Mixture Kernel-Based Bayesian Filtering. In *Proceedings of the 11th Int. Conf. on Computer Vision*, 2007.

[4] D.M. Gavrila. A Bayesian, Exempler-Based Approach to Hierarchical Shape Matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 8, pp. 1408–1421, 2008.

[5] S. Munder, C. Schnorr, and D.M. Gavrila. Pedestrian Detection and Tracking Using a Mixture of View-Based Shape-Texture Models. *IEEE Trans. on Intelligent Transportation Systems*, Vol. 9, No. 2, pp. 333–343, 2008.

[6] C.-Y. Lee, H. Gonzàlez-Baños, and J.-C. Latombe. Real-Time Tracking of an Unpredictable Target Amidst Unknown Obstacles. In *Proceedings of the 7th Int. Conf. on Control, Automation, Robotics and Vision*, pp. 596–601, 2002.

[7] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers. People Tracking with a Mobile Robot Using Sample-Based Joint Probabilistic Data Association Filters. *Int. J. of Robotics Research*, Vol. 22, No. 2, pp. 99–116, 2003.

[8] N. Bellotto and H. Hu. Multisensor Data Fusion for Joint People Tracking and Identification with a Service Robot. In *Proceedings of 2007 IEEE Int. Conf. on Robotics and Biomimetics*, pp. 1494–1499, 2007.

[9] H. Koyasu, J. Miura, and Y. Shirai. Realtime Omnidirectional Stereo for Obstacle Detection and Tracking in Dynamic Environments. In *Proceedings of the 2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Sysetms*, pp. 31–36, 2001.

[10] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia. People Tracking and Following with Mobile Robot Using Omnidirectional Camera and a Laser. In *Proceedings of 2006 IEEE Int. Conf. on Robotics and Automation*, pp. 557–562, 2006.

[11] D. Beymer and K. Konolige. Real-Time Tracking of Multiple People Using Continuous Detection. In *Proceedings of the 7th Int. Conf. on Computer Vision*, 1999.

[12] A. Howard, L.H. Matthies, A. Huertas, M. Bajracharya, and A. Rankin. Detecting Pedestrians with Stereo Vision: Safe Operation of Autonomous Ground Vehicles in Dynamic Environments. In *Proceedings of the 13th Int. Symp. of Robotics Research*, 2007.

[13] D. Calisi, L. Locchi, and R. Leone. Person Following through Appearance Models and Stereo Vision using a Mobile Robot. In *Proceedings of VISAPP-2007 Workshop on Robot Vision*, pp. 46–56, 2007.

[14] A. Ess, B. Leibe, and L.V. Gool. Depth and Appearance for Mobile Scene Analysis. In *Proceedings of the 11th Int. Conf. on Computer Vision*, 2007.

[15] A. Ess, B. Leibe, K. Schindler, and L.V. Gool. A Mobile Vision System for Robust Multi-Person Tracking. In *Proceedings of the 2008 IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

[16] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W.-K. Yoon. RT-Middleware: Distributed Component Middleware for RT (Robot Technology). In *Proceedings of 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3555–3560, 2005.

Proceedings of the IEEE ICRA 2009
Workshop on People Detection and Tracking
Kobe, Japan, May 2009

Stream Field Based People Searching and Tracking Conditioned on SLAM

Kuo-Shih Tseng and Angela Chih-Wei Tang

***Abstract*—People searching and tracking (SAT) is a key technology for interactive robots since the tracked people are sheltered by environments frequently. For robots, it is a tracking problem given that the target is observable, but otherwise it is a searching problem. Traditional tracking algorithms may lead to divergent estimation of object position when moving objects are unobservable. Moreover, SAT conditioned on simultaneous localization and mapping (SLAM) is complex since it aims at estimating people position, robot position, and map under sensor uncertainty. Motivated by this, we propose a novel stream functions and Rao-Blackwellised particle filter based SAT algorithm in this paper. This laser based algorithm is conditioned on simultaneous localization and mapping (SLAMSAT) to search and track people. With this, the position of the targeted person sheltered by the environment can be successfully estimated by the virtual stream field in a mapped environment. Our experimental results show that this algorithm can search and track people effectively.**

I. Introduction

In a dynamic environment, the robot navigation problem becomes interactive and it includes leading, following, intercepting, and obstacle avoiding. For most applications, a robot should be capable of tracking, following, self-localization, and obstacle avoidance in an unknown environment. Most tracking algorithms aim at correctly estimating the position, velocity, and acceleration of moving objects based on the past and sensor measurement [1]. Object tracking can be realized with Kalman filter (KF) with constant velocity model and/or constant acceleration model [2]. With particle filter (PF), objects with nonlinear states, non-Gaussian probability distribution, and multi-hypotheses are tracked with higher accuracy although the price is its high computational complexity. SLAMMOT uses scan matching and EKF with laser range finders to simultaneously estimate robot position, map, and object state [3]. The conditional PF estimates people motion conditioned on the probability model of robot position with a previously mapped environment [4].

Manuscript received Feb. 2, 2009.

Kuo-Shih Tseng is with MSRL at the Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan. (phone: 886-4-22779363; fax: 886-4-22782670; e-mail: seabookG@gmail.com).

Angela Chih-Wei Tang is with the Communication Engineering Department, National Central University, Jhongli, Taiwan. (e-mail: cwtang@ce.ncu.edu.tw).

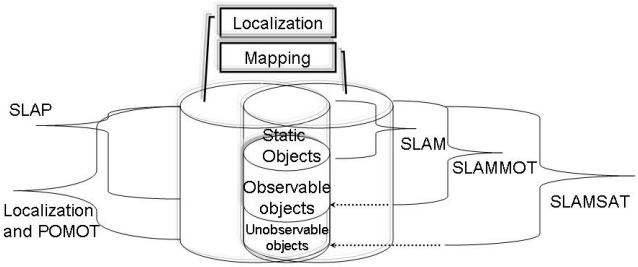


Fig.1. The relationship among SLAM, SLAP, SLAMMOT and SLAMSAT.

The tracking problem will turn into the searching problem if moving objects are unobservable. In [5], a map-based tracking algorithm using Rao-Blackwellised particle filter (RBPF) models the physical interaction between the ball and the wall even if the ball is unobservable. However, this algorithm can only track passive objects. Dynamic action spaces can be utilized to search and explore a moving object which goes toward one of the known destinations [6]. SAT techniques autonomously search and track objects using Bayesian estimator [7]. However, these algorithms cannot achieve simultaneous localization, tracking, and searching in an unknown environment. Motivated by this, a self-localization and partially observable moving object tracking (POMOT) algorithm is proposed in [8]. This algorithm is designed for a static and known environment. However, a robot has to localize itself, map, and search and track objects in most applications.

As shown in Fig. 1, objects can be static or dynamic. If the dynamic object is out of sight, it will be an unobservable object. Otherwise, it will be an observable object. Simultaneous localization and people tracking (SLAP) is to estimate robot and people position [4]. Localization and POMOT is to estimate robot and people position even if the person is unobservable [8]. SLAMMOT is to estimate robot, map and moving objects position [3].

In this paper, we propose a novel stream field based SAT algorithm conditioned on SLAM called SLAMSAT. SLAMSAT is to estimate robot, map and people position even if the person is unobservable. With stream field, we model interactions among goal position, updated environmental features, and people position. Traditional tracking algorithms deemed that objects move actively with velocity and acceleration generated themselves. But from the viewpoint of the stream field, object motion is passive due to the attraction and rejection forces resulted from the goal and environment. Based on this, we can still keep SAT the object

position based on the virtual stream field. The remainder of the paper is organized as follows. Section II describes the stream field based motion model for the RBPF based SAT proposed in Section III. Section IV gives our AdaBoost based leg detection. In Section V, we present the proposed SAT algorithm which combines the stream field and RBPF conditioned on the EKF SLAM algorithm. The experimental results are given in Section VI. Finally, Section VII concludes this paper.

II. Motion Model using Stream Field

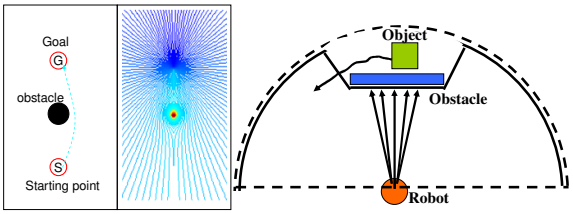


Fig.2. An example of a real environment and its virtual stream field. (a) Obstacle avoidance. (b) Stream field. (c) Real environment.

Complex potential is often employed to solve fluid mechanics and electromagnetism problems [9]. For an irrational and incompressible flow, there exists a complex potential consisting of the potential function $\phi(x, y)$ and stream function $\psi(x, y)$, where (x, y) is the 2-D coordinate. Although the complex potential has been studied quite extensively in motion planning and obstacle avoidance due to its high efficiency, it is seldom considered in object tracking. Motion model plays a key role in probability based tracking algorithms. To achieve on-line prediction of motion model according to the estimated map and virtual goal, we adopt stream field based motion model proposed in [8] for SAT algorithm in this paper. In the following, we give a brief description of this motion model. More details can be found in [8].

Stream field consists of a sink flow $\psi_{\sin k}(x, y)$ and a doublet flow $\psi_{doublet}(x, y)$ by

$$\begin{aligned}\psi(x, y) &= \psi_{\sin k}(x, y) + \psi_{doublet}(x, y) \\ &= -C\tan^{-1}\left(\frac{y - y_s}{x - x_s}\right) + C\tan^{-1}\left(\frac{\frac{a^2(y - y_d)}{(x - x_d)^2 + (y - y_d)^2} + (y_d - y_s)}{\frac{a^2(x - x_d)}{(x - x_d)^2 + (y - y_d)^2} + (x_d - x_s)}\right),\end{aligned} \quad (1)$$

where (x_s, y_s) is the center of sink, (x_d, y_d) is the center of doublet, a is the radius of doublet, and C is the constant proportion to the flow velocity. If the number of doublet flows is more than one, the stream field will be superposed by the sink flow and doublet flows. Details of stream fields can be found in [10]. Stream functions will be computed if the robot position, object goal, and obstacle positions are known. The object velocities are computed by the derivative of (1).

In typical tracking algorithms, the object position at time t is modeled by $\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{v}_{t-1})$, where $\mathbf{v}_{t-1}$ is object motion at time t-1, and f is the object motion model. A robot cannot track a moving object successfully when the object is unobservable. By (1), we assume that objects will avoid a known obstacle (doublet) and move toward a virtual goal (sink) as in the stream field (Fig. 2(c)). Since the stream field constructs an active field where an object is moved inactively by attraction and rejection forces, we can predict object and goal position and construct search path using the known stream field.

A stream field is constructed by a virtual sink and a doublet resulted from a known environment, and then the object motion is predicted by

$$\mathbf{V}_{t-1} = \begin{bmatrix} v_{o,t-1} \\ v_{o,t-1} \end{bmatrix} = \begin{bmatrix} \frac{\partial(\psi_{\sin k}(x_{o,t-1}, y_{o,t-1}) + \psi_{doublet}(x_{o,t-1}, y_{o,t-1}))}{\partial y_{o,t-1}} \\ -\frac{\partial(\psi_{\sin k}(x_{o,t-1}, y_{o,t-1}) + \psi_{doublet}(x_{o,t-1}, y_{o,t-1}))}{\partial x_{o,t-1}} \end{bmatrix}, \quad (2)$$

where $(x_{o,t-1}, y_{o,t-1})$ is the object position at time t-1. To estimate the position of virtual goal of an unobservable moving object, a probability based tracking algorithm with multi-hypothesis would work better than that with single hypothesis. Thus, Section III adopts RBPF to estimate N possible positions of an object goal.

III. RBPF based SAT using Stream Field Based Motion Model

To improve the accuracy of motion prediction in search case, we adopt stream field based motion model. However, the major problems of tracking with multi-hypothesis (e.g. PF) using this motion model are its heavy computational load and the requirement of precise probability distribution for prediction in the searching case. Since RBPF is capable of reducing the heavy load due to multi-hypotheses and approximating the probability distribution function more precisely [11], we adopt RBPF for SAT. Our RBPF based SAT algorithm using stream field based motion model is quite different from traditional ones where prediction will diverge if the moving object is unobservable.

Let the stream sample set $\mathbf{S}_k = \{\mathbf{s}_k^i, w_k^i \mid 1 \le i \le N\}$, and $\mathbf{s}_k^i = \langle O_k^i, G_k^i, D \rangle = \langle \langle O_{x,k}, O_{y,k}, \Sigma_{O,k} \rangle^i, \langle G_{\phi,k}, U_k \rangle^i, D \rangle$, where O_k^i is object state of the ith particle at time k including the mean $(O_{x,k}, O_{y,k})$ and covariance $\Sigma_{O,k}$. Object goal G_k^i consists of direction $G_{\phi,k}$ and intensity U_k. D is doublet position generated by map features. In our RBPF based SAT, PF estimates goal state G_k^i and KF estimates object state O_k^i.

We factorize stream distribution into goal set distribution, object set distribution, and stream set distribution at time k-1 as follows.

$$bel(\mathbf{S}_k) = P(\mathbf{S}_{1:k} \mid z_{1:k}) = P(O_k^i, G_k^i, O_{1:k-1}^i, G_{1:k-1}^i, D \mid z_{1:k})$$
$$\overset{DBN}{=} \underbrace{P(G_k^i \mid O_k^i, G_{k-1}^i)}_{Goal\ \text{set distribution}} \underbrace{P(O_k^i \mid O_{k-1}^i, D, G_{k-1}^i, z_k)}_{Objec\ set\ distribution} \times \underbrace{P(O_{k-1}^i, G_{k-1}^i, D \mid z_{k-1})}_{bel(\mathbf{S}_{k-1})} \quad (3)$$

After sampling goal positions, object set distribution is divided into two cases. It will be the tracking case if the robot detects object successfully, otherwise it will be the searching case.

$$P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, D, z_{1:k}) = \begin{cases} P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, D, z_{1:k}^O), & \text{tracking case.} \\ P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, D), & \text{searching case.} \end{cases} \quad (4)$$

For the tracking case, object set distribution is derived from Bayes theorem and updated by KF as follows.

$$P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, D, z_{1:k}) \overset{Bayes}{=} \eta \underbrace{P(z_k \mid O_k^i)}_{object\ Correction} \underbrace{P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, D, z_{1:k-1})}_{object\ \text{prediction}} \quad (5)$$

Finally, we resample stream sample set after computing the particle weightings. Based on the predicted object goal position, the algorithm can keep predict the object position when the object features are occluded or are fragmentation.

IV. Leg Detection

Detection is a necessary stage prior to tracking. Algorithms of laser based people detection usually work well only if the scan data of one or two of human legs is available. In [14], AdaBoost based leg detection is proposed for people detection. However, such algorithm will fail if the scan data of either leg is not available due to the sheltering effects resulted from environments. Another possible solution is to detect based on motion. However, it is difficult to distinguish slower legs from static objects.

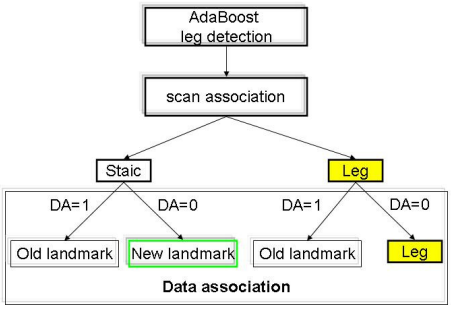


Fig.3. Decision flow of landmarks and leg.

In [15], the authors propose a multi-hypothesis leg-tracker for occlusion problem under known map. In SLAMSAT, a robot has to distinguish leg features (moving object) from static ones. Static features will be considered to be added into maps while leg features will be the candidates of tracked targets. The robot will search the dynamic feature based on the virtual stream field if there is not any dynamic feature detected.

Fig.4. Thresholds for feature classification in scan association.

In this section, we design a leg detection algorithm which is composed of AdaBoost, scan association, and data association of old landmarks. The decision flow of this detection algorithm is shown in Fig. 3. After applying AdaBoost based detection proposed in [14], all features are divided into leg features and non-leg features. Then, scan association determines whether a feature is static or dynamic by comparing scanned features at time t with those at time t-1. Let the Euclidean distance of a feature position at time t and a feature position at time t-1 be d. Please note that the feature at time t-1 could be static or dynamic. As shown in Fig. 4, a feature will be classified into a static one if d is smaller than the radius of black circle. Otherwise, this feature will be classified as a dynamic one.

Data association distinguishes whether a feature is for mapping or tracking based on the decision of scan association and AdaBoost based leg detection. If the static feature determined by scan association is not associated with any old landmark, it will be deemed as a new landmark. If a static or dynamic feature is associated with an old landmark, it will be deemed as an old landmark. Otherwise, the leg feature is still deemed as the leg.

It is a simplified task to distinguish static features from dynamic features using data association with a known map. However, such task will become difficult if a robot must simultaneously explore new landmarks with unknown map and search unobservable people. For example, when the dynamic features are sheltered by environments and a new feature is observable, it is difficult to distinguish a new landmark from a dynamic feature. A possible solution for such problem is employing multi-hypotheses based estimators instead of a single-hypothesis based one.

V. THE PROPOSED SAT ALGORITHM CONDITIONED ON SLAM

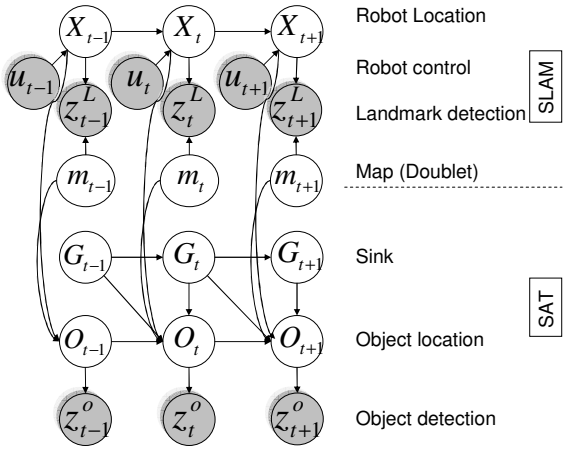


Fig. 5. Dynamic Bayesian Networks (DBNs) of SLAMSAT.

Effective search of sheltered object relies on robust localization, mapping, and tracking. To improve prediction accuracy, a robot has to move toward the sheltered zone and get more object information. This section proposes a scheme where the robot can simultaneously localize it, map environment features, and SAT a moving object (Fig. 5).

The SLAMSAT sample set is $\mathbf{X}_k = \{r_k, m_k, \mathbf{S}_k^i \mid 1 \le i \le N\}$, and

$$\mathbf{X}_k = \langle r_k, m_k, \mathbf{S}_k^i \rangle = \langle \langle r_{x,k}, r_{y,k}, r_{\theta,k}, \Sigma_{r,k}, m_k \rangle, \langle O_{x,k}, O_{y,k}, \Sigma_{O,k} \rangle^i, \langle G_{\phi,k}, U_k \rangle^i \rangle$$

where r_k is the robot state at time k and m_k is the map state at time k. Our SLAMSAT factorizes states into goal set distribution, object set distribution, robot state distribution, and the previous state set distribution as follows.

$$bel(\mathbf{X}_k) = P(\mathbf{X}_{1:k} \mid u_{1:k}, z_{1:k}) \overset{DBN}{=}$$
$$\underbrace{P(G_k^i \mid O_k^i, G_{k-1}^i)}_{Goal\ \text{set}\ distributi\ on} \times \underbrace{P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, r_k, m_{1:k}, u_{1:k}, z_{1:k})}_{Object\ \text{set distributi on}} \times$$
$$\underbrace{P(r_k, m_{1:k} \mid r_{1:k-1}, u_{1:k}, z_{1:k})}_{Robot\ set\ \text{distributi on}} \times$$
$$\underbrace{P(O_{1:k-1}^i, G_{1:k-1}^i, r_{1:k-1}, m_{1:k-1} \mid u_{1:k}, z_{1:k})}_{bel(\mathbf{X}_{k-1})}. \quad (6)$$

SAT conditioned on robot position and map is

$$P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, r_{1:k}, m_{1:k}, u_{1:k}, z_{1:k})$$
$$= \eta \underbrace{P(z_k^O \mid O_k^i)}_{Object\ correction} \underbrace{P(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, r_{1:k}, m_{1:k}, u_{1:k}, z_{1:k-1})}_{Object\ \text{prediction}}. \quad (7)$$

Thus, we simplify (6) to be an EKF localization problem

$$P(r_k, m_k \mid O_{1:k-1}^i, G_{1:k-1}^i, r_{1:k-1}, u_{1:k}, z_{1:k})$$
$$= \eta \underbrace{P(z_k^L \mid r_k, m_k)}_{Robot\ Correction} \underbrace{P(r_k, m_k \mid r_{1:k-1}, u_{1:k}, z_{1:k-1})}_{Robot\ \text{prediction}}. \quad (8)$$

Details of EKF localization can be found in [12].

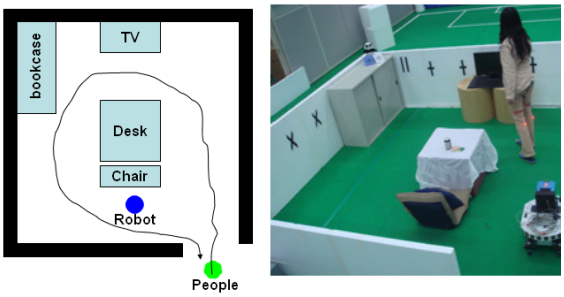
Table I : *S*LAMSAT *a*lgorithm

1. Inputs:
   $S_{k-1} = \{\langle G_{k-1}^{(i)}, O_{k-1}^{(i)}, D \rangle \mid i = 1,...,N\}$ posterior at time k - 1
   u_{k-1} control mesurement
   z_k observation
2. $S_k := \varphi$ // Initialize
3. $\bar{\mu}_k = g(u_k, \mu_{k-1})$ // Predict mean of robot and map postion
4. $\bar{\Sigma}_k = G_k \Sigma_{k-1} G_k^T + R_k$ // Predict covairance of robot and map postion
5. for m := 1,....,M do // EKFSLAM update
6. for c := 1,....,C do
7. if $d_m^L < d_{th}^L$ do // if $d_m < d_{th}$, z_i is landmark m
8. $K_k^c = \bar{\Sigma}_k H_k^{cT} (H_k^c \bar{\Sigma}_k H_k^{cT} + Q_k)^{-1}$
9. $\mu_k = \bar{\mu}_k + K_k^c (z_k^c - h_k^c(\bar{\mu}_k))$
10. $\Sigma_k = (I - K_k H_k^c) \bar{\Sigma}_k$
11. else do
12. $z_c^o = z_c$ // z_c is a dynamic feature
13. $w^{(i)} := 0$
14. for i := 1,....,N do // RBPF Tracking
15. $G_k^i \sim p(G_k^i \mid O_k^i, G_{k-1}^i)$ // virtual goal smapling
16. $O_k^i \sim p(O_k^i \mid O_{1:k-1}^i, G_{1:k-1}^i, r_{1:k}, D, u_{1:k}, z_{1:k-1})$ // see (1)
17. for j := 1,....,J do // data association
18. if $d_m^o < d_{th}^o$ do
19. O_k^i := kalman update(O_k^i) // update object
20. $w_k^i := p(z_{k,j}^o \mid O_t^i)$ // compute weighting
21. $S_k := S_k \cup \{\langle G_{k-1}^{(i)}, O_{k-1}^{(i)} \rangle\}$ // insert S_t into sample set
22. Discard smaples in S_t based on weighting w_t^i (resampling)
23. return S_t, μ_t, Σ_t

Our RBPF based SAT algorithm conditioned on SLAM is summarized in Table I. The EKF SLAM predicts and corrects robot position using EKF (lines 3-10). Laser measurements are represented as line features using the least square algorithm. The feature is either associated with a known landmark (line 7) or a leg feature (line 12). Goal states G_k^i are sampled and then the *N* kinds of object states O_k^i are predicted according to (1) (lines 15-16). Based on stream set distribution at time $k-1$, we assume the distance between the object and the goal is fixed at 200 cm so that we only randomly sample sink flow direction $G_{\phi,k}$ and sink flow intensity U_k for efficiency. If the *i*th particle is associated with a moving object, RBPF will update moving object position O_k^i. This is described as follows. First, the algorithm computes the weighting of the *i*th particle w_k^i and particles will be resampled (lines 20-22). In tracking case, the stream sample set S_k^i including the object sample set O_k^i and the goal sample set G_k^i will converge. In searching case, it will keep predict the object sample set O_k^i based on the previous stream field S_{k-1}^i

VI. EXPERIMENTAL RESULTS

We adopt UBOT with one SICK laser as the mobile robot platform and a 1.6 GHZ IBM X60 laptop with 0.5G RAM as the computing platform. The area of the experimental environment is 3.6m by 3.6m. We use PhaseSpace for the precise ground truth of people and robot trajectories [13]. The LEDs of PhaseSpace are mounted on two legs of the people and Ubot (Fig. 6(b)). The people walks along the line, and the robot follows the people through the remote control (Fig. 6(a)). The person is sheltered by the desk, bookcase and chair frequently so that tracking may turn into the searching case.



(a) (b)

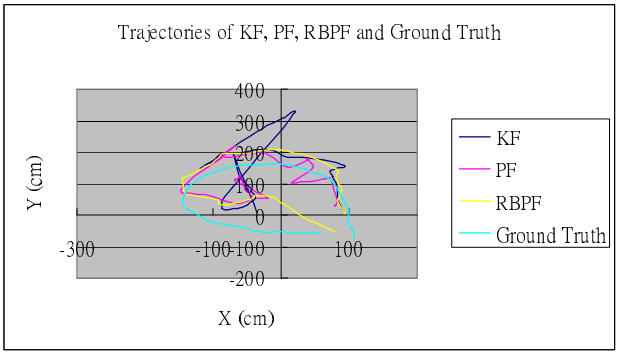
Fig.6. Environment setup: (a) Walking trajectory. (b) The experimental environment.

TABLE II.
Confusion matrix of AdaBoost detection.

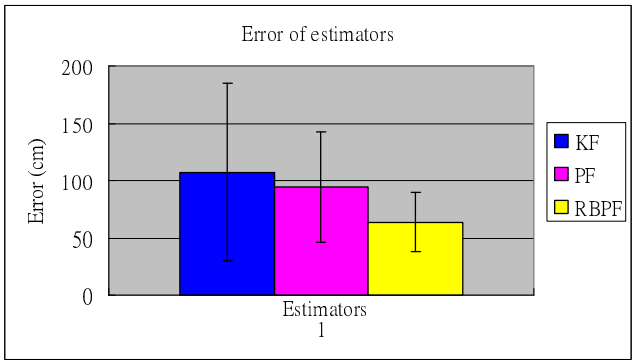
	Detected Label		
Groud Truth	Person	No Person	Total
Person	39 (88.8%)	5(11.2%)	44
No Person	45 (11.4%)	345 (88.6%)	390

TABLE III.
Confusion matrix of AdaBoost, scan association and data association.

	Detected Label		
Ground Truth	Person	No Person	Total
Person	39 (88.8%)	5(11.2%)	44



(a)

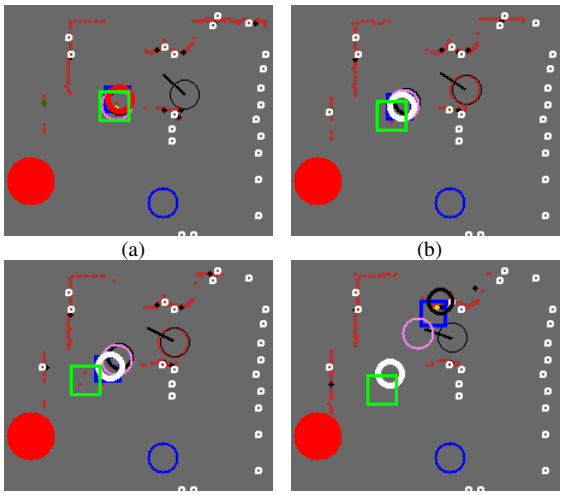


(b)

Fig.7. (a) Trajectories of KF, PF, RBPF, and ground truth. (b) Errors of KF, PF and RBPF.

TABLE IV.
Comparisons of tracking errors.

	Total mean (cm)	Total std. (cm)
KF	107.4	77.59
PF	94.0	48.39
RBPF	63.7	26.30



(a) (b) (c) (d)

Fig.8. The experimental results of SLAMSAT. Small white circles are mapped landmarks. Black and black points are static and dynamic features, respectively. Black circle and red circle are robot position of odometer and estimated robot position of EKF SLAM, respectively. Blue circle is the original point. Blue square is the estimated people position of KF. Pink circle is the estimated people position of PF. White/Red circle is estimated people position of RBPF when it is the searching/tracking case. Red solid circle is people goal (sink) position of stream field. Green square is the ground truth of people position. (a) 44th frame. (b) 45th frame. (c) 46th frame. (d) 47th frame.

The confusion matrices of detection based on AdaBoost only and detection based on AdaBoost, scan association and data association are presented in Tables II and III, respectively. Obviously, scan association and data association increase accuracy rate of detection.

The tracking trajectories are shown in Fig. 7. In searching case, KF diverges faster than PF while RBPF keeps

predicting the object position based on the stream field. Comparisons of average tracking errors among KF, PF, and RBPF are shown in Table IV. The average tracking errors of KF, PF, and RBPF are 107.4cm, 94.0cm, and 63.7cm, respectively.

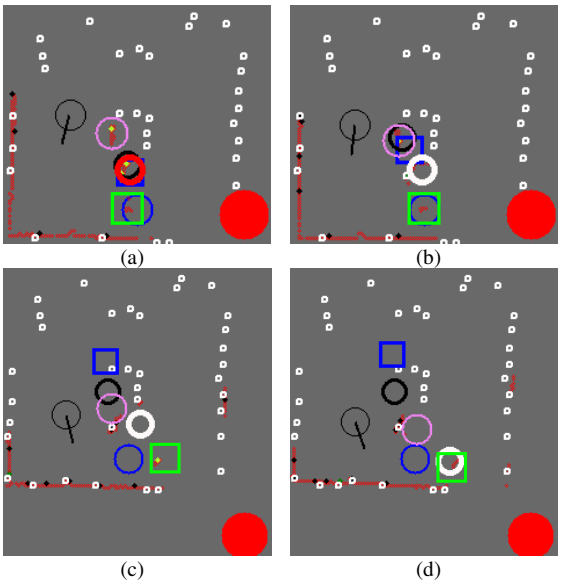


Fig.9. The experimental results of SLAMSAT with false data association. (a) 58th frame. (b) 59th frame. (c) 63th frame. (d) 64th frame.

The experimental results of SLAMSAT are shown in Figs. 8 and 9. As shown in Figs. 8(a) and 8(b), the KF and PF based tracking keep predicting people based on their motion model. However, RBPF can further keep searching people position based on the virtual goal. Figures 8(c) and 8(d) shows the incorrect KF estimation which is resulted from false detection. Parts of the PF particles are associated with the wall feature while others are not. Accordingly, the people position estimated by PF is between those by KF and RBPF. Nevertheless, RBPF can still keep searching people position based on the virtual goal.

In Fig. 9, the algorithm will infer that the chair is the person if the chair is deemed as leg features and the person is out of sight but near the chair. The false alarm of leg detection may result in the incorrect estimation of KF and RBPF (Fig. 9(a)). Also, the incorrect PF estimation is resulted from another false alarm of leg detection. As shown in Fig. 9(b), the incorrect estimation of KF and PF are resulted from false detection. However, RBPF can still keep searching people position based on the virtual goal. In Fig. 9(c), KF and PF diverge when there is no feature near the last estimation. When the leg feature is detected, the estimation of the people position by RBPF is near the feature so that RBPF can estimate the people position correctly. However, the feature position is far from that by KF estimation. Parts of PF particles are associated with the leg feature, but others are not so that the estimation of PF is inaccurate (Figs. 9(c) and 9(d)). Obviously, the experimental results show that our proposed RBPF algorithm is better than KF and SIR PF in the searching and tracking case.

VII. Conclusions

This paper proposes a novel SAT algorithm based on stream functions and RBPF conditioned on SLAM called SLAMSAT. SLAMSAT estimates the moving object position, robot position, and map under sensor uncertainty. The experimental results show our algorithm can search and track moving objects effectively.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.

[2] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS, Danvers, MA, 1995.

[3] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889-916, Sept. 2007.

[4] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 695-701, May 2002.

[5] C. Kwok and D. Fox., "Map-based multiple model tracking of a moving object,." *Robocup Symposium,* 2004.

[6] N. Roy and C. Earnest, "Dynamic action spaces for information gain maximization in search and exploration," in *Proc. the American Control Conference*, June 2006.

[7] B. Lavis, T. Furukawa, and H. Durrant-Whyte, "Dynamic space reconfiguration for Bayesian search and tracking with moving targets," *Autonomous Robots*, vol. 24, no. 4, pp. 387-399, May 2008.

[8] K.-S. Tseng, "A stream field based partially observable moving object tracking algorithm," *10th IEEE Intl. Conference on Control, Automation, Robotics and Vision*, Hanoi, Vietnam, 2008.

[9] W. Kaufmann, *Fluid Mechanics*, McGraw-Hill, 1963.

[10] S. Waydo and R. M. Murray, "Vehicle motion planning using stream functions," in *Proc. IEEE International Conference on Robotics and Automation,* vol.2, 2003.

[11] X. Y. Xu and B. X. Li, "Adaptive Rao–Blackwellized particle filter and its evaluation for tracking in surveillance," *IEEE Trans. Image Processing*, vol. 16, no. 3, pp. 838-849, March 2007.

[12] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics, MIT Press, 2005.

[13] http://www.phasespace.com/

[14] K. O. Arras, O. M. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2D range data," *Proc. IEEE International Conference on Robotics and Automation*, Rome, Italy, 2007.

[15] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities," in *Proc. IEEE International Conference on Robotics and Automation*, May 2008.