The Dissertation Committee for Jong Taek Lee
certifies that this is the approved version of the following dissertation:

Recognition of Human Interactions with Vehicles using 3-D Models and Dynamic Context

Committee:

J. K. Aggarwal, Supervisor

Alan C. Bovik

Wilson S. Geisler

Kristen Grauman

Gustavo de Veciana

Recognition of Human Interactions with Vehicles using 3-D Models and Dynamic Context

by

Jong Taek Lee, B.S., M.S.E

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2012

Dedicated to my parents.

Acknowledgments

First of all I would like to express my deepest gratitude to my advisor Professor J. K. Aggarwal for his all-round support. Without his help, I would not be able to pursue my studies. I sincerely appreciate his great guidance on my research and life. I am highly honored to be his 45th student to earn a PhD. I also would like to thank my committee members, Prof. Alan Bovik, Prof. William Geisler, Prof. Kristen Grauman, and Prof. Gustavo de Veciana, for sharing their time and insights on my research.

I also want to thank my current lab, the Computer and Vision Research Center, members Chia-Chih Chen, Josh Harguess, Birgi Tamersoy, Suyog Jain, and Lu Xia, for the numerous research discussions and for cheering me up all the time. My appreciation goes to former lab member Dr. Michael Sahngwon Ryoo for his great guidance and insights on my work. I also want to thank Medha Bhargava and Matthew Riley as well. I would like to extend my thanks to Ms. Selina Keilani for the great help on revising many papers.

I sincerely appreciate the support and the prayers of my father and mother, who always encourage me to pursue my goals. I also thank my sister and brother, who always trust me.

Recognition of Human Interactions with Vehicles using 3-D Models and Dynamic Context

Jong Taek Lee, Ph.D.
The University of Texas at Austin, 2012

Supervisor: J. K. Aggarwal

This dissertation describes two distinctive methods for human-vehicle interaction recognition: one for ground level videos and the other for aerial videos. For ground level videos, this dissertation presents a novel methodology which is able to estimate a detailed status of a scene involving multiple humans and vehicles. The system tracks their configuration even when they are performing complex interactions with severe occlusion such as when four persons are exiting a car together. The motivation is to identify the 3-D states of vehicles (e.g. status of doors), their relations with persons, which is necessary to analyze complex human-vehicle interactions (e.g. breaking into or stealing a vehicle), and the motion of humans and car doors to detect atomic human-vehicle interactions. A probabilistic algorithm has been designed to track humans and analyze their dynamic relationships with vehicles using a dynamic context. We have focused on two ideas. One is that many simple events can be detected based on a low-level analysis, and these detected events must contextually meet with human/vehicle status tracking results. The other

is that the motion clue interferes with states in the current and future frames, and analyzing the motion is critical to detect such simple events. Our approach updates the probability of a person (or a vehicle) having a particular state based on these basic observed events. The probabilistic inference is made for the tracking process to match event-based evidence and motion-based evidence. For aerial videos, the object resolution is low, the visual cues are vague, and the detection and tracking of objects is less reliable as a consequence. Any method that requires accurate tracking of objects or the exact matching of event definition are better avoided. To address these issues, we present a temporal logic based approach which does not require training from event examples. At the low-level, we employ dynamic programming to perform fast model fitting between the tracked vehicle and the rendered 3-D vehicle models. At the semantic-level, given the localized event region of interest (ROI), we verify the time series of human-vehicle relationships with the pre-specified event definitions in a piecewise fashion. With special interest in recognizing a person getting into and out of a vehicle, we have tested our method on a subset of the VIRAT Aerial Video dataset [37] and achieved superior results.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the last decade, considerable effort has been devoted to recognition of human activities. However, it still remains a challenging problem in computer vision due to errors in low-level processing, scene changes by the camera viewpoints, and the complexity of semantic representations [1, 53]. In addition to simple human (e.g. single person) activity recognition, researchers have proposed methodologies for the recognition of human-human (person-to-person) interactions [38, 44], human-object interactions [33], and group activities [43]. Human-vehicle interactions may be categorized as human-object interactions.

The problem of the recognition of human-vehicle interactions has not received the same level of attention as other interactions, but it is now of significant interest in many applications, such as automated surveillance, abnormal activity detection, video annotation, and crime detection. Vehicle-related activity recognition is a challenging problem in computer vision. In the study of most human-object interactions recognition, objects are smaller than humans (e.g. books, cups, phone, and so on [33]). These interactions are simple such that people carry objects or stand near objects. On the other hand, most ve-

hicles are larger than humans in general, and humans can easily be occluded by vehicles. The occlusion varies significantly as the viewpoint of a camera changes. Furthermore, a person may change the appearance (i.e. shape) of a vehicle by opening and/or closing its doors, and the motion of the same human action looks very different due to the change of the location and orientation of the corresponding vehicle. These characteristics of human interactions with a vehicle make the problem more challenging.

Even for the same objective such as recognizing human-vehicle interactions, the solution can differ significantly because different filming condition restricts the applicable approaches. This dissertation explains two distinctive methods for human-vehicle interaction recognition: one for ground level videos and the other for aerial videos. For the ground level videos, we usually have higher quality imagery and a background model from stationary cameras. Furthermore, a number of training samples are available and real-time processing is not mandatory. On the other hand, for the aerial videos, we have the opposite situation. Namely, training samples are difficult to obtain. We introduce these two methods in the following separate subsections.

1.1 Human-vehicle interaction recognition from ground view

The automated and continuous analysis of humans, objects, and their status has been a long-time goal of artificial intelligence, robotics, and computer vision. Particularly, in the field of computer vision, the detection and

tracking of humans from closed circuit television (CCTV) videos recorded in various environments have been studied in the last several decades, and numerous promising approaches have been proposed.

However, human tracking itself is insufficient to analyze interactions between humans and vehicles: in order to annotate and retrieve videos containing activities involving humans and vehicles, complex movements of humans/vehicles and their relationships in a dynamic environment (e.g. a crowded parking lot) must be analyzed. The system must be able to identify detailed 3-D status and motion of all objects appearing in each frame. Such an analysis is particularly essential for the construction of many important applications including surveillance and military systems.

This dissertation presents a novel methodology which is able to estimate a detailed status of the scene involving multiple humans and vehicles. The system tracks their configuration even when they are performing complex interactions with severe occlusion such as when four persons are exiting a car together. The motivation is to identify the 3-D states of vehicles (e.g. status of doors), their relations with persons, which is necessary to analyze complex human-vehicle interactions (e.g. breaking or stealing a vehicle), and the motion of humans and car doors to detect atomic human-vehicle interactions. In addition, our methodology aims to identify the regions where each person enters the vehicle (e.g. the driver's seat or passenger seat), anticipating his/her role and position even when the person is invisible. The challenges are derived from significant human-human occlusion, human-vehicle occlusion, which ear-

lier human tracking systems had difficulties handling, and motion changes due to the camera view. Fig.1.1 illustrates the difficulties which appearance only based human-vehicle interaction recognizing systems can have.

A probabilistic algorithm has been designed to track humans and analyze their dynamic relationships with vehicles using a dynamic context. We have focused on two ideas. One is that many simple events can be detected based on a low-level analysis, and these detected events must contextually meet with human/vehicle status tracking results. That is, simple events (e.g. a person approaching a vehicle) detected during the interactions can be used as key features (e.g. it may be an indication of the person opening the nearby door) for more robust tracking. The other is that the motion clue interferes with states in the current and future frames, and analyzing the motion is critical to detect such simple events. Our approach updates the probability of a person (or a vehicle) having a particular state based on these basic observed events. The probabilistic inference is made for the tracking process to match event-based evidence and motion-based evidence. The event influences an interval of states, making a certain set of states more probabilistically favorable than the others for each time frame. For example, tracking a person occluded by a door is difficult without any contextual knowledge, but the detection of the event 'a person opening the door and going into the car' may help the system analyze his/her movements in these frames.

Even though there have been previous attempts to process videos of humans and vehicles, they have focused on recognition of simple human-vehicle

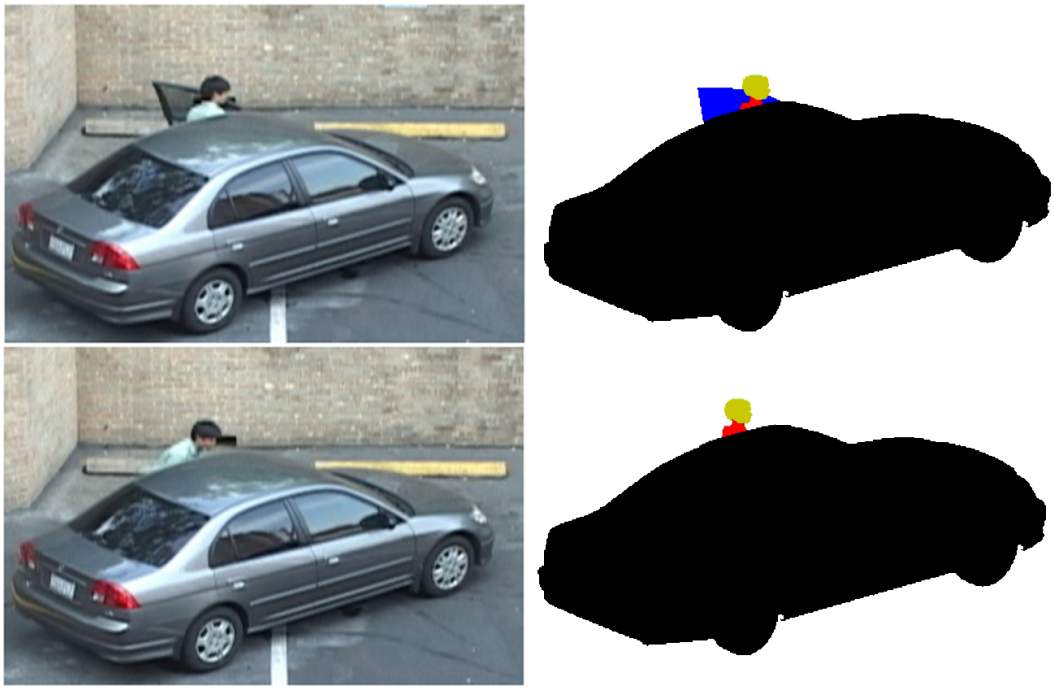

Fig. 1.1: Example 3-D scene models of two different image frames. Detailed analysis of configurations of humans and vehicles using event context must be performed, correctly distinguishing states of two images with similar appearances.

interactions. Instead of performing detailed scene analysis in a complex environment, they either assumed that the interactions are performed in simple environments which have no (or little) occlusion [22, 49], or assumed that human-manual corrections of tracking objects [24] are provided. The system proposed by Lee *et al.* [28] was able to perform view-independent recognition of a single person getting out of (or into) a vehicle, but it was limited in processing crowded human-vehicle interactions with two or more people. This is due to their inability to analyze states of scenes composed of multiple objects, failing to process complex events composed of several fundamental human-vehicle movements (e.g. 'door open,' 'person get in,' or 'person get out').

Our tracking problem is formulated as a Bayesian inference of finding the sequence of scene states with the maximum posterior probability. The scene state includes individual object states (humans and vehicles), object-object occlusions, and specific parameters of objects (e.g. door position and status). Our system estimates and tracks scene states frame-by-frame using Markov Chain Monte Carlo (MCMC), measuring the appearance similarity between hypothetical 3-D scene models and the observed image. The appearance of the scene state is described in terms of joint 3-D models and its projection is compared with the real image. In addition, as mentioned above, our probabilistic framework uses event-based and motion-based cues to update the prior probability of object states, tracking highly occluded human-vehicle interactions (e.g. a person opening a door) reliably. In order to handle an

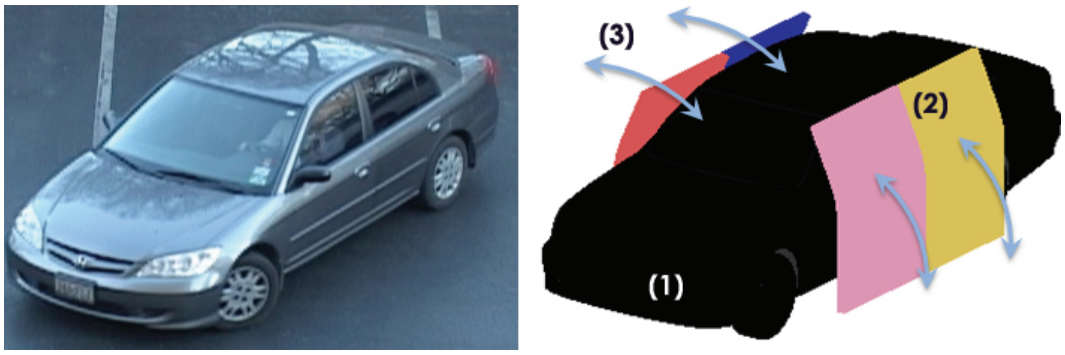

Fig. 1.2: A raw image of a vehicle (left) and its matching synthetic 3-D vehicle model (right). The 3-D model is used for 1) the shape based matching by a vehicle-only model (black color), 2) extraction of regions-of-interest (ROIs) by four door regions (rectangular shape), and 3) transformation of motion features by the direction of door opening/closing on each door (double arrows).

event which is only detected 'after' its occurrence, we propose an algorithm to correct past frames by traversing past time frames. The motion-based cue is view-dependent in the human-vehicle interactions. To solve these difficulties, our approach uses synthetic 3-D vehicle models for detecting vehicle location, orientation, and door regions, and estimating patterns of motion represented in the optical flow field (see Fig. 1.2). The transformation is accomplished by measuring the direction of a door opening or closing that fits the optical flow field. As a result, our system is able to extract view-independent features. We train a Support Vector Machine (SVM) [9] classifier with the view-independent features for the classification of interactions.

This view-independent system recognizes complex human-vehicle interactions using 3-D vehicle models. The system processes a dataset taken from

various viewpoints. The proposed approach has several benefits over previous approaches. First, our approach is able to extract view-independent features from human motion. Consequently, the system requires less training data from various viewpoints to achieve the same performance as previous systems which use view-dependent features. Second, our approach is able to reduce computation time and to recognize multiple occurrences of interactions. This advantage is possible because we specify ROIs based on localization of vehicles and their fitted 3-D models, while most of the previous approaches specified ROIs based on detection of humans.

1.2 Human-vehicle interaction recognition from aerial view

Recognizing human-vehicle interactions from an aerial view is a challenging problem in computer vision. It is of increasing interest in security, automated surveillance, and military operations. For example, the detection of a person getting into a vehicle may provide the first level alert of abnormal events. The discovery of frequent human-vehicle interactions from aerial videos may help pinpoint a warehouse or signify the migration of a group of people. As shown in Fig. 1.3, due to limited image resolution, air turbulence, cloud coverage, objects temporarily out of field of view, and the constantly moving aerial vehicle, the recognition of human-vehicle interactions from aerial views is a much more challenging task than those in normal scenarios. In this work, we propose a general framework to recognize human-vehicle interactions from

Fig. 1.3: (a) The aerial image of a person approaching the front door of a vehicle. The bounding box of the person is magnified to illustrate this challenging scenario. (b) The snapshots of a vehicle taken from an UAV (Unmanned Aerial Vehicle) in every 5 seconds.

an aerial video. More specifically, we illustrate our framework using the cases of recognizing a person getting into and out of a vehicle.

With careful and sometimes repeated inspections, a human observer can recognize human-vehicle interactions from aerial videos without seeing any examples from the same setup. This is because humans are capable of constantly tracking objects in low quality imagery and are proficient at reasoning about the underlying event without seeing it in its entirety. However, there are two major difficulties for machine vision to perform the similar task as people do. First, most machine learning algorithms require a sufficient number of training samples to perform reliable recognition; however, the cost is high for taking aerial videos and annotating example sequences. Second, the key moments of human-vehicle interactions always happen when persons are in close proximity to the vehicle; as a result, a human tracker is easily subject to drift due to overlapped object structures in blurry low-resolution imagery.

Our method is a temporal logic based approach which does not require the tracking of human objects nor event-level training examples. Our system starts with processing the bounding box sequences of the tracked vehicles. To estimate the location and the orientation of a vehicle, we train SVM classifiers with samples rendered from 3-D vehicle models and ray tracing. Then we search for the optimal solution of vehicle states in a sequence of frames using dynamic programming under a Markovian assumption. Given the aligned 3-D vehicle models, we use the localized door (or trunk) regions together with local human detection results to reason about their interactions over time. We define the temporal flow of a human-vehicle interaction based on the sub-events of particular changes in their spatial relationships. Weights are manually assigned to the interaction associated sub-events according to their relative importance to the composition of the interaction. The likelihood of individual interactions is computed by matching an observation sequence with the formal event representations and binning the weighted votes of matched sub-events.

1.3 Dissertation outline

The rest of the dissertation is organized as follows: Chapter 2 discusses the relevant previous works. Chapter 3 and Chapter 4 present methodologies for human-vehicle interactions from ground level videos. Chapter 3 explains Bayesian formulation of scene states with event-based context and Chapter 4 extends the framework with motion-based context. We introduce a very

distinctive human-vehicle interaction method exclusively for aerial videos in Chapter 5. We conclude in Chapter 6.

Chapter 2

Related work

2.1 Vehicle detection and tracking

There has been a considerable amount of research related to the special tasks of vehicle detection and tracking. Vehicle detection and tracking has been done using a stationary single camera [20, 21, 25, 48, 51], stationary multiple cameras [26], a moving single camera [5, 6, 41], and moving multiple cameras [4]. The systems for vehicle detection and tracking using stationary cameras are the most typical, and our first approach falls into this category. The systems for vehicle detection with moving cameras [4–6] use various methods such as template matching, temporal differencing, and specific patterns searching instead of using a background model. Our second approach falls into this category. Rajagopalan and Chellappa [41] applied an image stabilization algorithm to detect motion regions. After stabilizing the images, they applied thresholding and morphological operations to detect motion regions. Gupte *et al.* [21] tracked blobs for vehicle tracking using an association graph to connect objects in continuous frames. They used Kalman filtering to predict the position of a vehicle in each consecutive frame. Sun *et al.* [50] presented a review of the problem of vehicle detection and integrating detection with tracking. Jun *et al.* [25] and Tamersoy and Aggarwal [51] proposed sys-

Fig. 2.1: Collected vehicle training samples.

tems that detect vehicles in order to count the number of vehicles in highway traffic. Jun *et al.* [25] proposed a method of segmentation for vehicles under severe occlusions. Their system first finds feature points of vehicles using scale-invariant feature transform (SIFT) and tracks those features to compute motion vectors. Oversegmented image fragments are then clustered based on motion vectors of the fragments, and occluded vehicles are separated finally. Tamersoy and Aggarwal [51] proposed SVM training with positive and negative training sets which can be obtained in an unsupervised manner and an efficient tracking solution by a simplified multi-hypothesis approach. Recently, Feris *et al.* [17] proposed a generic vehicle detection approach. They implemented a semi-automatic data collection system. From collected videos, they manually define one or more regions-of-interest for training (see Fig.2.1). Synthetic generation of occluded vehicles scene improves the detection of vehicles in crowded scenes. Large-scale learning is performed with Haar-like features and a cascade of Adaboost classifiers.

Background subtraction has most widely been used for foreground detection with one or multiple stationary cameras. In background subtraction, it is desirable to obtain good background models. Many papers address this

problem using an adaptive background mixture model [49] which works well in outdoor environments. Zivkovic [59, 60] proposed an improved adaptive background mixture model for background subtraction. The algorithm provides decent performance in various environments, which works especially well for shadow removal. Morphological operations are performed on the foreground to reduce errors in foreground and background models after processing [13]. We use the methods from [13, 59, 60] for successful low-level processing of videos taken by a stationary camera.

Specialized vehicle activities, such as detection of illegally parked vehicles, had not been studied in depth with one possible exception of [34] that presents a system that detects and warns of an illegally parked vehicle. In 2007, however, the i-Lids vehicle detection challenge dataset was released, and a large number of papers provided distinctive solutions on this challenge: Bevilacqua and Vaccari [7], Boragno *et al.* [8], Guler *et al.* [19], Lee *et al.* [29], and Porikli [40], and Venetianer *et al.* [54]. Lee *et al.* [30] presented a novel system to detect illegally parked vehicles using 1-D transformation and compared the system with state-of-art systems. Specialized vehicle activities, particularly the detection of illegally parked vehicles has been intensely studied since 2007, when the i-Lids vehicle detection challenge dataset was released.

Several researchers have tried to use 3-D models to recognize more complex activities from arbitrary viewpoints. The usage of 3-D models is focused on human representations for human recognition [31] or object representations for object detection [26, 47]. Kim and Malik [26] used a semi 3-D vehicle

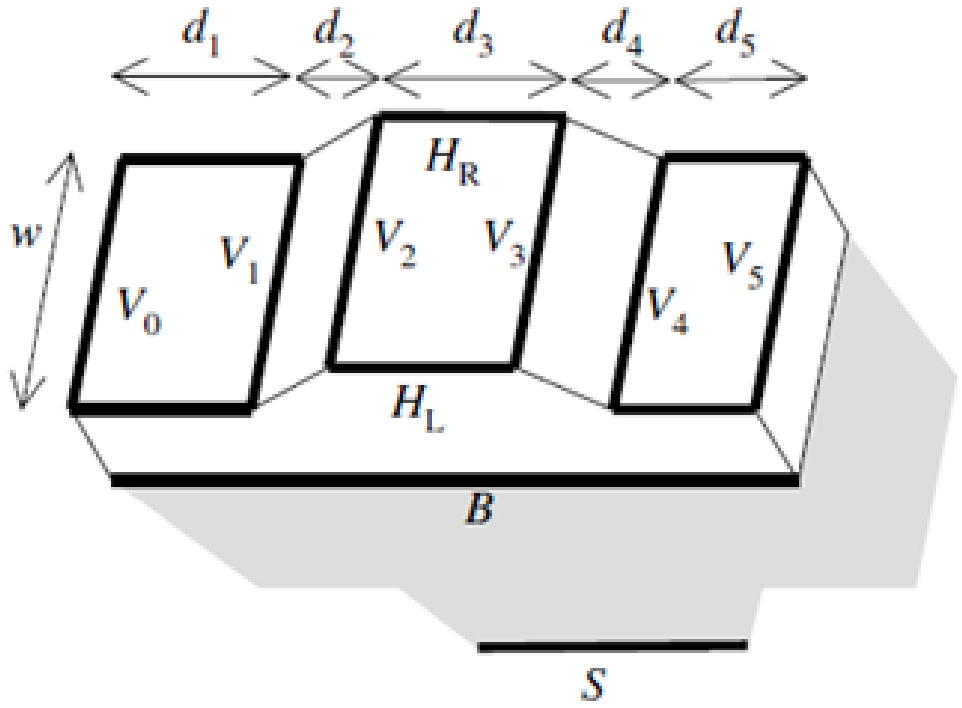

Fig. 2.2: The car model. H_L and H_R are the left and right lines, V_0, . . . , V_1 are the vertical lines, B is the base line, and S is the shadow line. w and d_i are in the world coordinates

Fig. 2.3: An example detection result. The view is very limited.

model (see Fig. 2.2) and proposed a probabilistic line feature based vehicle description to detect vehicles in limited views (see Fig. 2.3). To track vehicles in three cameras, they proposed Zero-mean cross-correlation matching based tracking. Song and Nevatia [47] developed a methodology to detect and track vehicles using 3-D models from various viewpoints. They extracted 2-D silhouette shape templates from 3-D models and matched the templates with observed foregrounds. They formed a hypothesis for vehicle information and refined it by a data driven MCMC process. Their approach relies on the distinctiveness of the vehicle shape cue, which is viable for side-view cameras. For front-view or rear-view cameras, it might not work well as shown in their experiments. They also used Kalman filtering for tracking detected vehicles. Yang *et al.* [56] proposed a vehicle detection and tracking system with low-angle cameras. They used two semi 3-D vehicle models (sedan and bus, see Fig. 2.4). Windshield and other feature points are detected, and they fused these detections to improve the vehicle detection rates. They classified the type of vehicles and implemented Markov chain Monte Carlo based vehicle tracking. We present a comparison among the distinctive vehicle detection and tracking approaches in Table 2.1

2.2 Bayesian framework for tracking

In previous tracking solutions following a Bayesian framework, trajectories of objects are modeled as a sequence of scene states describing the location of objects [6, 14, 57, 58]. Zhao *et al.* [58] presented a model-based approach for

Methods	Kim	Song	Feris	Yang	Ours (ground)	Ours (aerial)
Model	Semi 3D	3D	None	Semi 3D	3D	3D
Training	Manual	Automatic	Semi automatic	Manual	Automatic	Automatic
Feature	Line	Silhouette (edge)	Haar	Edge and color (for windshield)	Silhouette	Histogram of Gradient
Testing environment	Aerial, limited view, Stationary camera	Ground level, limited view, stationary camera	Semi-aerial, Stationary camera	Semi-aerial, Stationary camera	Ground level, Stationary camera	Aerial, Moving camera, Low resolution
Tracking	Zero-mean cross-correlation matching	Kalman filter	X	MCMC	MCMC	Dynamic Programming
Orientation Estimation	X	O	X	X	O	O
Etc.			Adaboost	Classify type of vehicle	Detect door and its motion	No background modeling, fast, easy to extend

Table 2.1: Comparison of vehicle detection and/or tracking approaches

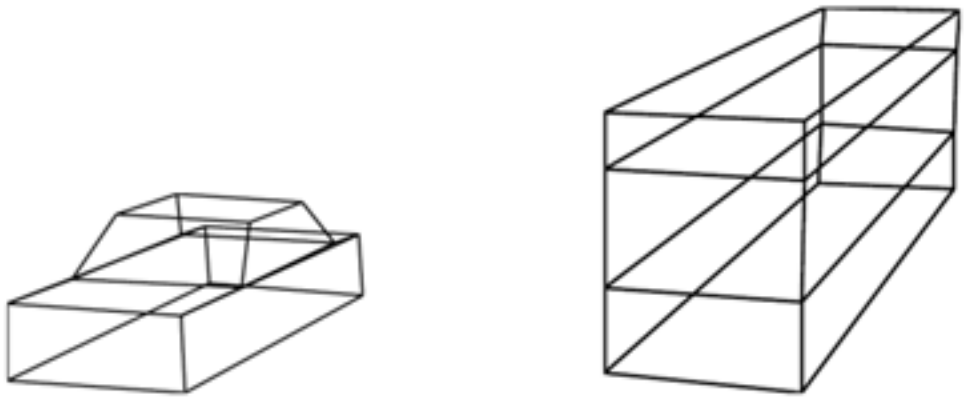

Fig. 2.4: two vehicle models (left: sedan, right: bus)

the segmentation and tracking of humans in crowded situations following a Bayesian framework. They computed prior probabilities and joint likelihoods using 3-D human models and calculated the posterior probability. Because of the enormous complexity of the solution space, they used a data-driven MCMC for efficient sampling of the posterior probability to search for the optimal solution.

Ryoo and Aggarwal [42] presented a new paradigm for optimal tracking under severe occlusion. The limitation of most of the previous human tracking systems following the hypothesis-and-test paradigm [14, 58] is that they are required to maintain an exponentially growing number of hypotheses over frames if they do not apply pruning. Under severe occlusion, pruning can result in significant tracking performance reduction, and the system was able to overcome such limitations. However, the system only tracks humans without

considering any other objects or their relations. The system was unable to analyze interactions between humans and vehicles.

2.3 Human-vehicle interaction recognition

There have been several attempts to analyze human and vehicle interactions. Ivanov and Bobick [23] developed an automatic surveillance which consists of two levels, independent probabilistic temporal event detectors and stochastic context-free grammar parsers. In the surveillance application, their system recognizes human activities involving vehicles. Joo and Chellappa [24] recognized activities in a parking lot such as picking up and dropping off a person. The activities of "dropping off" and "picking up" are similar to "getting out of a car" and "getting into a car". For the recognition of such activities, they used attribute grammars to represent the activities. Their contribution was on the representation of specific activities using the attribute grammars, not on the accurate detection of objects or motions. Therefore, their system was neither fully automatic nor view-independent. Similarly, Tran and Davis [52] proposed an approach using Markov logic networks to recognize vehicle-related events for surveillance by integrating common sense reasoning with uncertainty from computer vision algorithms. However, these works have focused on the detection of simple events rather than analyzing complex scenes with severe occlusion. Park and Trivedi [39] presented an approach to analyze moving-object interactions between humans and vehicles by using planar homography domain and semantic event grammar, but the scenes they analyzed

were limited to simple interactions with little occlusion as well.

Lee *et al.* [30] proposed a system to recognize human-vehicle interactions such as exiting and entering. Shape-based matching with a 3-D vehicle model is performed to detect a vehicle, and regions-of-interest(ROIs) are extracted from four door regions of the detected vehicle next. Under the assumption that the interactions occur in the ROIs, their system extracts motion and shape features in ROIs and analyzes them to classify interactions. However, since they did not consider spatial organizations (e.g. occlusion) between door ROIs, their system was unable to process interactions such as 'two persons coming out of the car from doors on the same side'. Furthermore, similar to [22, 24, 52], they did not attempt to analyze detailed scene configurations of objects. They did not take advantage of event context and were unable to analyze human-vehicle interactions in complex environments.

The descriptors based on the histogram of oriented gradient (HOG) and histogram of oriented optical flow (HOOF) have popularly been used for object recognition and action classification [10, 15, 32]. Dalal and Triggs [15] used a dense grid of HOG to detect humans. Chaudhry *et al.* [10] recognized 10 basic human actions including running, galloping sideways, waving, and jumping by classifying a HOOF time-series. Training sets of both systems [10, 15] are, however, taken from the limited viewpoints (front-and-back views or side views). Marszalek *et al.* [32] recognized 12 complex human actions from various viewpoints by using a variety of descriptors. The labeled actions consist of "getting out of a car," "driving a car," "shaking hands," and "hugging a

person." The descriptors include HOG, HOOF, SIFT, and 3-D/2-D Harris detectors. The problem with their system was that the recognition rate of "getting out of car" is about 15 %, which is lower than the recognition rates of other actions. They identified scene classes and combined them with the descriptors in order to improve performance. The precision of the recognition of "getting out of a car" did not improve.

2.4 Human activity recognition from aerial view

There has been an emerging interest in recognizing human activities from aerial views in the past few years. The pioneer work by Efros *et al.* [16] characterizes human actions at a distance by using an optical flow based descriptor. They use the rectified optical flow components to describe the motion patterns between pairs of figure-centric bounding boxes and a k-nearest-neighbor classifier to perform action recognition and synthesis. On the same subject, Chen and Aggarwal [11] present a joint feature action descriptor, which combines features selected from human poses and motion in a supervised manner. They represented poses and movements by continuous frames of HOG and HOOF, respectively. Supervised Principle Component Analysis (SPCA) is used to reduce the dimension of feature space and SVM classifier is trained to classify actions. Later in their work [12], they propose a novel representation called an *action spectrogram*, which characterizes human activities using both local video content and the occurrence likelihood spectra of body parts' movements. Their method has been shown to further the recognition

accuracy on two low-resolution human activity datasets [37, 45].

The approaches [22, 24, 28, 46, 52] mentioned in Section 2.3 are mostly not applicable to the analysis of aerial videos, where the interactions are filmed from a moving platform and the accurate characterization of object contours and motion is not possible. For the evaluation of human activity and human-object interaction recognition algorithms, the newly published VIRAT Video Dataset [37] includes videos collected from stationary ground cameras as well as unmanned aerial vehicles (UAV). This large-scale benchmark dataset features 6 types of human-vehicle interactions in both camera settings.

Chapter 3

Human-vehicle interaction recognition using event context

In order to recognize human-vehicle interactions in crowded conditions, we design a probabilistic algorithm to track humans and analyze their dynamic relationships with vehicles using *event context*. Our tracking problem is formulated as a Bayesian inference of finding the sequence of scene states with the maximum posterior probability. The scene state includes individual object states (humans and vehicles), object-object occlusions, and specific parameters of objects, e.g. door position and status. Our system estimates and tracks scene states frame-by-frame using MCMC, measuring the appearance similarity between hypothetical 3-D scene models and the observed image. The appearance of the scene state is described in terms of joint 3-D models and its projection is compared with the real image. In addition, as mentioned above, our probabilistic framework uses event-based cues to update the prior probability of object states, tracking highly occluded human-vehicle interactions (e.g. a person opening a door) reliably. In order to handle an event which is only detected 'after' its occurrence, we propose an algorithm to correct past frames by traversing past time frames.

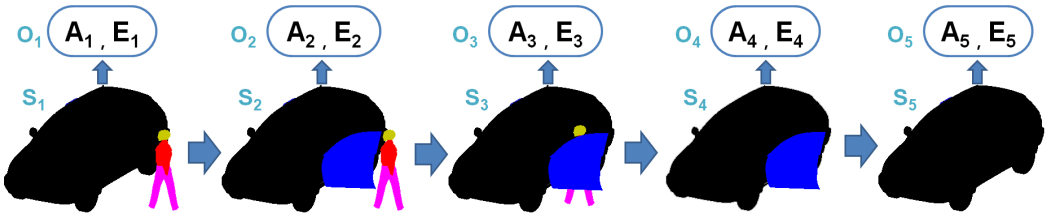

Fig. 3.1: Example scene state transitions of 'a person entering a car'. Each S_i is a scene state, and (A_i, E_i) corresponds to an observed image frame. The goal of our system is to identify a sequence of states correctly describing the video.

3.1 Definition of scene states

In this section, we define the 'state', S, of each scene. A state is a complete description of object's locations, their internal parameters, and relationships among them in each scene image. The level of detail in the scene state definition directly influences the system's level of understanding image frames, and is important for constructing a scene analysis system. Throughout this work, our system interprets a video as an observation generated by a particular sequence of scene states (Fig. 3.1), and searches for the sequence that best describes the dynamics of objects and their relationships in the video.

In many of the previous tracking paradigms (e.g. [14, 57]), each state is modeled as a set of independent objects (with particular parameters) present at each frame. Recently, the tracking paradigm has been extended to explicitly consider occlusion among humans [42, 58]. However, these previous systems only consider relative depth-ordering among humans, limiting themselves on analyzing detailed states of human-vehicle interactions such as "one car is

parked in a parking lot, its front left door is fully opened, and a person is in the middle of getting out of the car through the door."

In our approach, we extend the definition of the scene state so that it can describe scene conditions more specifically. A scene state, S, is composed of the term C describing individual object states and the term R specifying object-object spatial relationships: $S = (C, R)$. The object state C is a set of c_ks, each describing the object class, tracking ID., and the class-specific parameters of the kth object: $c_k = (cls_k, id_k, param_k)$ where cls_k is the class of the object and id_k is its ID. Because there are two classes of objects (i.e. a human and a vehicle) and they have different object properties, the parameters for two classes ($param_k$) are defined differently. R is defined as a spatial relationship of all objects in C. R is composed of multiple rs, each describing the spatial relationship between two different objects (i.e. whether they are occluded, they are close to each other, or they have any spatial relationships): $R = \cup_{c_i \neq c_j} r_{(i,j)} = \cup_{c_i \neq c_j} (type, c_i, c_j)$. For example, $r_{(1,2)} = (occ, c_1, c_2)$ illustrates that the object c_2 is occluded by c_1. As a result, our scene state not only describes the locations of individual objects but also specifies their relative dynamics.

3.2 Bayesian formulation

We formulate the tracking process of human-vehicle interactions as a Bayesian inference of computing the posterior probabilities of scene states:

$$S^{max}_{(1,2,...)} = argmax_{S_{(1,...,n)}} P(S_{(1,...,n)}|O_{(1,...,n)}),$$

where S_i is a scene state at frame i, O_i is an observation at frame i, and n is the number of frames observed. That is, we want to compute the optimum sequence of scene states that matches with the observations best. $P(S_{(1,...,n)}|O_{(1,...,n)})$ can further be enumerated as the multiplication of prior probability and image likelihood:

$$P(S_{(1,...,n)}|O_{(1,...,n)}) \propto P(O_{(1,...,n)}|S_{(1,...,n)}) \cdot P(S_{(1,...,n)}). \tag{3.1}$$

For an efficient searching of the maximum-a-posteriori (MAP) of a scene state in all frames, $S^{max}_{(1,...,n)}$, we make a Markov assumption:

$$\begin{aligned} &P(O_{(1,...,n)}|S_{(1,...,n)}) \cdot P(S_{(1,...,n)}) \\ &\quad = P(O_1, ..., O_n|S_1, ..., S_n) \cdot P(S_1, ...S_n) \\ &\quad = P(O_n|S_n) \cdot P(O_1, ..., O_{n-1}|S_1, ..., S_{n-1}) \qquad (3.2) \\ &\qquad \cdot P(S_n|S_{n-1}) \cdot P(S_1, ...S_{n-1}) \texttt{(refer Appendix I for the details)} \qquad (3.3) \\ &\quad = P(O_n|S_n) \cdot P(S_n|S_{n-1}) \\ &\qquad \cdot P(O_{(1,...,n-1)}|S_{(1,...,n-1)}) \cdot P(S_{(1,...,n-1)}) \qquad (3.4) \end{aligned}$$

Therefore,

$$
\begin{aligned}
&argmax_{S_{(1,\ldots,n)}} P(S_{(1,\ldots,n)}|O_{(1,\ldots,n)}) \\
&\quad = \{argmax_{S_n} P(O_n|S_n) \cdot P(S_n|S_{n-1}), \\
&\qquad argmax_{S_{(1,\ldots,n-1)}} P(S_{(1,\ldots,n-1)}|O_{(1,\ldots,n-1)}) \qquad (3.5)
\end{aligned}
$$

From Equation 3.5, $P(O_n|S_n) \cdot P(S_n|S_{n-1})$ needs to be calculated to search MAP scene state in frame n. Intuitively, $P(O_n|S_n)$ is the likelihood between the observed image and the scene state at frame n, and $P(S_n|S_{n-1})$ describes the transition probability.

We further extend our Bayesian formulation to take advantage of 'event context' for reliable and detailed tracking of scene states. As mentioned in the previous sections, event detection results can be treated as an important feature that benefits the tracking process greatly. The state tracking problem must be formulated so that it takes into account the fact that occurrences of events must meet with the states of the scenes during the event. For example, if the event of the person getting out of the car is clearly occurring, then there is little possibility that the person was out of the scene during this event.

While an observation O corresponds only to an image appearance A in most of the previous systems, we extend the Bayesian tracking formula so that certain events between a vehicle and a human change the prior probabilities of objects. Therefore, observation O is defined to include both appearance A

and event E.

$$\begin{aligned} P(O_n|S_n) \cdot P(S_n|S_{n-1}) &= P(A_n, E_n|S_n) \cdot P(S_n|S_{n-1}) \\ &= P(A_n|S_n) \cdot P(E_n|S_n) \cdot P(S_n|S_{n-1}) \\ &\propto P(A_n|S_n) \cdot P(S_n|E_n) \cdot P(S_n|S_{n-1}) \end{aligned} \tag{3.6}$$

That is, we assume $P(E_n)$ is uniformly distributed. In Equation 3.6, $P(A_n|S_n)$ represents the similarity between an input image and an object model. $P(S_n|E_n)$ represents the prior probability of an image frame n in a particular state S_n, given an occurrence of an event E. If object states are assumed to be independent on events as in previous systems, $P(S_n|E_n)$ is the same as $P(S_n)$. $P(S_n|S_{n-1})$ shows the conditional probability of scene states in continuous two frames.

By solving the formulated Bayesian inference problem, we are able to estimate the most probable sequence of scene states. Each of the probability terms described in this section is modeled more explicitly in the following sections.

3.3 Probabilistic modeling

In this section, we present the method to compute Bayesian probability 'given' each scene state and image frame (e.g. Fig. 3.2). We present a 3-D scene (human and vehicle) model which is used for calculating appearance likelihood, and introduce our 'event context' that influences states' prior probabilities for contextual inference. The methodology to search for the optimal scene state

based on these models will be discussed in Section 3.4.

3.3.1 Appearance likelihood, $P(A_n|S_n)$

Our comprehensive definition of a scene state enables the system to construct a virtual appearance of the scene given its state. We use a 3-D model of a human (or a vehicle) to represent an appearance of each individual object c_k. The motivation is to estimate the optimal appearance of an individual object c_k in the scene as a 2-D projection of its 3-D model, so that it can be compared with the real image to measure the appearance likelihood. Furthermore, the appearance of multiple overlapped objects are modeled by considering the spatial relationship of the objects R. Fig. 3.2 shows an example 2-D projection of a 3-D scene model consists of several 3-D human and vehicle models. We take advantage of such appearance model to compare it with a real image to measure the state likelihood. The camera parameters for the projection are assumed to be known.

3.3.1.1 3-D vehicle model

Our system assigns a 3-D model for each vehicle appearing in the scene. Based on the parameters of the vehicle state, a snapshot of the 3-D vehicle model is computed at each frame to obtain its virtual appearance. A vehicle is described with the following parameters: (x, y, $size$, $orient$, $tilt$, $type$, $door$). x and y are the center xy-coordinates of the vehicle, $size$ is the resize factor

Fig. 3.2: An example appearance of a projected 3-D scene state (right image) corresponding to an input image (left one). The 3-D scene model is constructed based on S_n, and is used for the appearance likelihood computation.

of an 3-D template image, *orient* is the orientation of the vehicle, *tilt* is the tilt angle of the vehicle, *type* is the type of the vehicle (e.g. sedan and sport utility vehicle), and *door* is the parameters of all doors to describe how far the doors are open (closed, partially opened, and fully opened). The orientation and tilt angle of a vehicle are quantized and sampled for 5 degrees. Sample 2-D projection images of a 3-D vehicle model with an opened door are shown Fig. 3.3(a).

3.3.1.2 3-D human model

Similar to our 3-D vehicle model, a 3-D model is assigned per person in the scene. A human is described with the following parameters: (x, y, *size*, *orient*, *tilt*, *type*, *color_histogram*, *velocity*). x, y, *size*, *orient*, and *tilt* of a human are defined similar to those of a vehicle. Two *type*s of human models

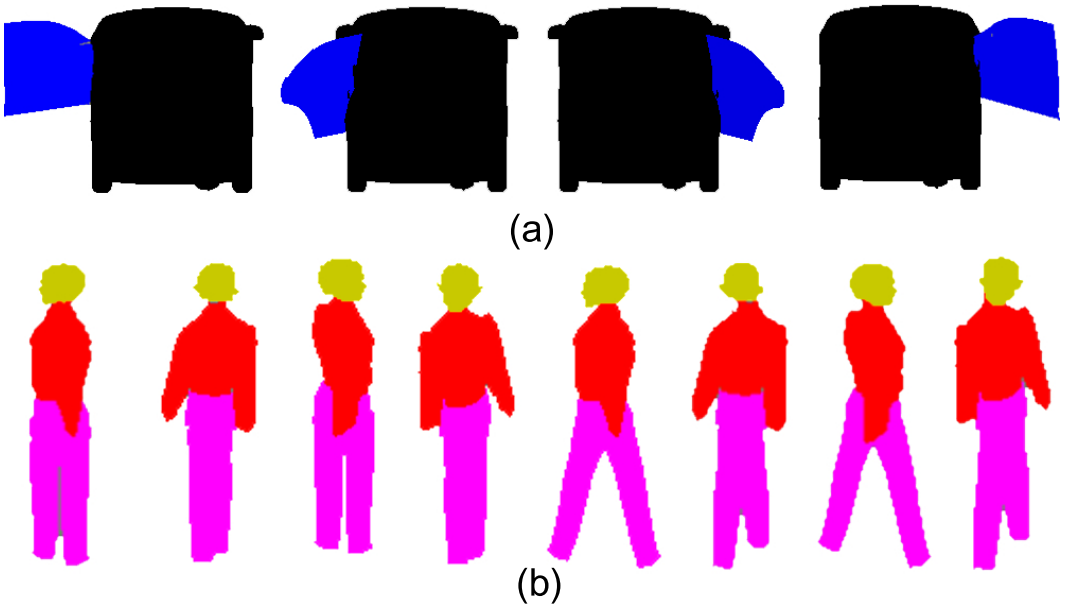

Fig. 3.3: (a) 2-D projections of 3-D vehicle models representing door opening states. (b) 2-D projections of 3-D human models. The left four images are from a standing model and the right four images are from a walking model.

are used: walking and standing. In addition to the 3-D human shape model, a color histogram is used to detect and distinguish human objects [58] in order to handle non-rigid human appearances. For human objects, we calculate *color_histogram* on three regions of humans such as a head, an upper body, and a lower body. The *velocity* is also calculated for tracked human objects to be applied in Kalman filtering. The orientation and tilt angle of a human are digitized and sampled for 90 degrees and 5 degrees, respectively. Each 3-D human model at a frame is generated based on these parameters. Sample 3-D human models of two types are presented in Fig. 3.3(b).

3.3.1.3 Human-vehicle joint model

A human-vehicle joint model is constructed per scene by considering the spatial relationship (e.g. occlusion) R of humans and vehicles. We construct a complete 3-D scene model composed of multiple 3-D object models, so that its 2-D projection may be compared with the real image. A 3-D scene model essentially is a set of 3-D human and vehicle models whose relative spatial relationships are described with R.

The process to obtain a projection of a joint scene model (given a particular scene state) is as follows: 1) Build a blank canvas whose size is the same as the real image for representing a scene model. 2) Choose object c_k which does not occlude any non-chosen object, based on R. 3) Draw the 2-D projection of the object c_k. 4) Repeat 2) and 3) until all objects are drawn. That is, we are essentially drawing all objects into a blank image in a particular order so that an occluded object is drawn before the object occluding it. Drawing each object can be done using the 3-D human/vehicle individual models. Note that spatial relationship R specifies which object is occluded by which, enabling the overall joint model projection process.

To construct a complete projection of a 3-D scene model, object-object spatial relationship (R) should be calculated. The spatial relationship between humans or between vehicles can be obtained based on xy-coordinates of the objects. Based on the following criteria, we build R for each S_n using its C value. The two criteria for deciding relations of human-human occlusion and vehicle-vehicle occlusion are: 1) If the feet of person c_{k1}^p are located under

Fig. 3.4: Example occlusion types generated based on the simulation. Representative occlusion types describing relationships among human, door, and vehicle body are presented.

the feet of person c^p_{k2} and two people are overlapped in an image, person c^p_{k1} occludes person c^p_{k2}. 2) If the center of vehicle c^v_{k1} is under the center of vehicle c^v_{k2}, vehicle c^v_{k1} occludes vehicle c^v_{k2}. Human-vehicle occlusion is more complex to process compared to the other two types of occlusion, due to the existence of doors. A relation between an overlapped human and vehicle (i.e. which is occluding which) is estimated by comparing C with several simulated occlusion types. As shown in Fig. 3.4, we construct several representative occlusion types with a rough simulation, and compare which occlusion type matches the given C of the scene S_n best. The depth order of the best matching occlusion type is chosen to be the relation between the human and the vehicle.

3.3.1.4 Joint image likelihood

Here, we present how we actually compute the appearance likelihood based on the projection of the joint model described above. We compare the expected appearance (i.e. 2-D projection) generated from the 3-D scene model with a real image. We measure the distance between the image and the model for each object c_k, and sum them to compute the state-image dis-

tance. That is, assuming conditional independence among appearances of non-occluded object regions given the 3-D scene model, we can calculate $P(A_n|S_n)$ as $\prod_{c_k} P(A_n|M(c_k))$, where $M(c_k)$ is a non-occluded region of object c_k obtained in the previous section. $P(A_n|M(c_k))$ can be measured by calculating the ratio of the number of foreground pixels of $M(c_k)$ to the number of foreground pixels on the region ($P(FL_k\ |M(c_k))$) and pixel-wise color distances ($P(CL_k\ |M(c_k))$). Thus, $P(A_n|S_n)$ can be calculated as shown in Equation 3.7.

$$\begin{aligned} P(A_n|S_n) &= \prod_{c_k} P(A_n|(c_k, R)) = \prod_{c_k} P(A_n|M(c_k)) \\ &= \prod_{c_k} \{P(FL_k\ |M(c_k)) \cdot P(CL_k\ |M(c_k))\} \end{aligned} \tag{3.7}$$

3.3.2 Dynamics Likelihoods, $P(S_n|E_n) \cdot P(S_n|S_{n-1})$

In this subsection, we model two probability terms that influence the posterior probability, $P(S_n|E_n)$ and $P(S_n|S_{n-1})$. Intuitively, the former corresponds to the probability of the 'event context' supporting the states, and the latter specifies the influence of the previous frame state to the current state. We discuss how we model each of these terms describing scene dynamics.

3.3.2.1 Event Context, $P(S_n|E_n)$

As we have formulated in Section 3.2, the probability of the scene in a particular state S_n is highly dependent on its event context. The occurrence of an event at a particular time interval (i.e. a pair of a starting time and

an ending time) suggests that the states within the interval must follow a particular distribution; the state sequence must contextually agree with the event. Here, we model such probabilistic distribution of the interval's states for each event class (i.e. type). The goal is to assign scene states that match event detection results with higher probability values.

Let a pair (t_s, t_e) be a time interval of an event e. Then, we model the distribution $P(S_n|E_n = e)$ for all states of $t_s < n < t_e$ to have a distribution learned from training examples of the event e. Similar to the case of appearance likelihood computation, we assume conditional independence among objects in the scene as follows:

$$P(S_n|E_n = e) = \prod_{c_i} P(c_i|E_n = e) \cdot \prod_{c_j} P(c_j|E_n = null) \tag{3.8}$$

where c_i are the objects involved in the event e, and c_j are the other objects. We assume that the event time intervals do not overlap, meaning that there's only one (or no) event going on at a particular time frame.

We model each $P(c_i|E_n = e)$ based on training data. We assume that all states within the event's interval show an identical probability distribution, ignoring their temporal order. Given a set of example state sequences corresponding to the event intervals, $P(c_i|E_n = e)$ is learned by considering all observed ground truth states to be sampled from the same distribution. More specifically, we model $P(c_i|E_n = e)$ to have a 3-dimensional distribution where the first dimension specifies whether the object c_i is in the scene and the other two dimensions specify the relative XY-coordinates of the object. As a result,

the system makes certain spatial locations more probabilistically preferable than others for the object during the event interval. Our event context has an effect of narrowing down the state search space, making the scene state tracking process more efficient and reliable.

In principle, our proposed methodology is able to cope with any number of events as long as their state distributions can be learned. However, in this work, we have chosen the three events which most effectively benefits the scene tracking process for computational efficiency. The defined events are 1) a person gets out of a vehicle, 2) a person approaches and opens a door of a vehicle, and 3) a person is sitting inside a car. For example, in the case of the third event, the distribution of the locations of the person c_k during the event's time interval will be modeled to be centered at the seat. Thus, our event context consideration process will update all $P(S_n|E_n)$ within the interval so that it penalizes the states representing the location of the c_k to be somewhere else. All of this is done by learning the distributions based on training examples.

We discuss more about how we actually detect events' time intervals and take advantage of them in Section 3.4.

3.3.2.2 Previous State, $P(S_n|S_{n-1})$

The term $P(S_n|S_{n-1})$ describes the probability of the objects (i.e. humans and vehicles) in a certain scene state S_n, given their state at the previous frame $n-1$. Our system's consideration on the previous state is done in a

straight forward fashion. Similar to previous tracking algorithms [42, 58], our system assumes linear movements of objects. Based on the XY velocity of the object, the distribution of $P(S_n|S_{n-1})$ is modeled to have a Gaussian distribution centered at the expected location using the previous state.

3.4 MAP searching by MCMC

In this section, we present an algorithm to search the scene state S_n^{max} providing the highest posterior probability at time frame n. What we presented in Section 3.3 is a method to compute the posterior probability of each scene state S_n, and we now search for the optimum state among them. A trivial approach is to perform brute force searching. However, the high dimensionality of our solution space requires a fast maximum-a-posteriori (MAP) searching algorithm. Markov Chain Monte Carlo (MCMC) has been widely used in complex tracking systems for efficient MAP searching. We apply the following three procedures to search MAP.

3.4.1 Markov Chain Monte Carlo Dynamics

Our MCMC algorithm searches for the best scene state at each frame. It randomly applies one of the predefined moves to S_n, iteratively updating the S_n for hundreds of rounds while searching for the one with the highest probability. We have adopted a Metropolis-Hastings algorithm with reversible jumps [18]. At each iteration, our Metropolis-Hastings algorithm applies a randomly selected move to an individual object state c_k of S_n to obtain S',

which will either be discarded or accepted as the new S_n. The initial value for S_n is set to be S_{n-1}, and is iteratively updated. The prior probability of selecting a human as c_k to update is 0.9 and that of selecting a vehicle is 0.1. The list of MCMC sampling moves are as follows:

1. **Object addition hypothesis.** Randomly select a vehicle or person to be added in the scene. All parameters of an object are randomly chosen from prior object parameter distribution, except for the position (x, y). The center position of an object will be randomly located on the foreground pixels.

2. **Object update hypothesis.** Change parameters of objects based on their prior probability distributions. For human objects, the values of x, y, *size*, *type*, and *orient* are updated. The other parameters are automatically calculated using the knowledge of the camera model and the ground plane. For vehicle objects, the values of x, y, *size*, *orient*, *type*, and *door* are updated as well.

3. **Object removal hypothesis.** Randomly select a vehicle or a person to be removed from the scene.

At every iteration of the dynamics, the system updates object-object spatial relationship (R) from the updated individual object states (C). Therefore, the system can obtain a new scene state (S') and calculate $P(O_n|S') \cdot P(S'|S_{n-1})$. We accept the scene state S' for S_n if the $P(O_n|S') \cdot P(S'|S_{n-1})$ is

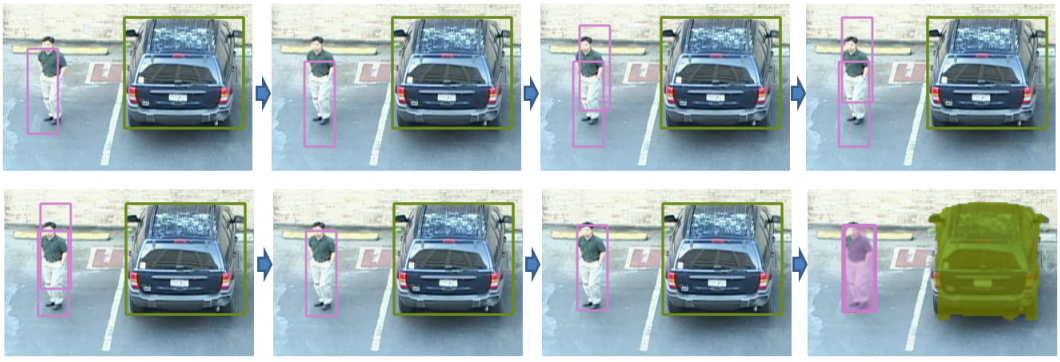

Fig. 3.5: Example candidate scene states, S', obtained during our MCMC iteration. Various MCMC sampling moves have been sequentially applied to search for an optimal scene state, S_n^{max}.

larger than $P(O_n|S_n) \cdot P(S_n|S_{n-1})$. The experimental results are obtained after 200 iterations. Fig. 3.5 shows an example iteration of our MCMC process.

3.4.2 Event Detection

In order to search for the scene state providing the maximum posterior probability, events occurring during human-vehicle interactions must be detected. The detected events will enable the computation of the dynamics likelihood probability using event context (i.e. Section 3.3.2), making our system able to track detailed scene states. In principle, any of the existing activity recognition methodologies can be adopted for the detection of events. In our implementation, events are recognized using a rule-based elementary detector with a simple criterion; our elementary detector is activated (i.e. it detects an event) by checking whether the previous state S_{n-1} satisfies the encoded rules of the event. That is, we say that the event is occurring if the rules are

satisfied and use this information as an event context to compute the state probabilities.

Note that the detector is activated at a particular time point, instead of fully providing events' intervals. In general, the detector is activated either at a starting time or an ending time of the event depending on its characteristics. No exact time interval is provided, and most events are detected 'after' the event has occurred. This implies that the probability computation using the event context presented in Section 3.3.2 is difficult in a standard forward inference process. It is not capable of recalculating the past states even if the system later finds that an event has occurred in the past frames. This situation occurs commonly for the detectors which are difficult to compute exact occurring time intervals (e.g. traditional hidden Markov models), and hence we present a forward/backward probability updating process in the following subsection. The motivation is to dynamically update future (or past) frames that are expected to be within the time interval until the event conditions are violated.

The detailed detection criteria of our three events, "a person getting out of a car," "a person approaching and opening a door of a vehicle," and "a person sitting inside a car" are as follows:

1. **A person getting out of a car.** The event of "a person getting out of a car" is detected at time t_e, which is the ending frame of the event's time interval. The detection rules are 1) a new person appears near a

door d and 2) the door d is open. That is, we assume that the new person came out from the door.

2. **A person approaching and opening a door of a vehicle.** The event of "a person approaching and opening a door of a vehicle" is detected at time t_s, which is the starting time frame of the event's time interval. The detection rules are 1) a person from outside the scene boundary approaches a door d (i.e. their distance becomes small) and 2) the door d was closed at t_s. The event continues until the person disappears or the distance between the person and the vehicle becomes larger than a threshold.

3. **A person sitting inside a car.** The event of "a person sitting inside a car" is detected at frame t_s (i.e. starting time), when the following conditions are satisfied: 1) a person c_k disappears near a door d at frame t_s and 2) the door d was opened at frame t_s. The event continues until the person reappears from the door.

3.4.3 Updates with Backward Tracking

As mentioned in the previous subsection, many events tend to be detected 'afterwards', making the MCMC-based MAP state computation with event context difficult. What we present in this subsection is a methodology to support our event context-based scene state tracking by compensating for such late detections using a backward re-tracking process.

We say that an event has a forward characteristic if it is detected at its starting time, and has a backward characteristic if it is detected at its ending time. Basically, unless an event having a backward characteristic occurs, our system progresses the computation of MAP states in a forward direction using the MCMC-based algorithm presented in Section 3.4.1. This process is similar to hidden Markov models or other sequential state models. The system assumes that no event is going on, if no forward event has been detected (it may later correct it if an event with a backward character is detected afterwards). If a forward event e is detected at frame t_s, the system records that the event is starting to occur from the frame t_s and considers the event context for each frame n such that $t_s < n$. This event context consideration (i.e. $E_n = e$) is applied for future frames, as long as the conditions of the event are satisfied, influencing $P(S_n|E_n)$.

The backward probability update process is described as follows. Once a backward event is detected, our system initiates the tracking process in the backward direction, starting from the frame t_e where the event is detected. That is, we update (or re-estimate) the scene states of frame n such that $n < t_e$. Leaving non-related objects c_js unchanged, the system recalculates $P(c_i|E_n = e)$ for event related objects c_is at frame n and recomputes $P(S_n|O_n)$ to search for the MAP state. This backward traversal process is continued until the event conditions are violated. For example, in the case of the event 'a person getting out of a vehicle', the backward traversal is continued until the person disappears in the backward tracking process (i.e. until the system

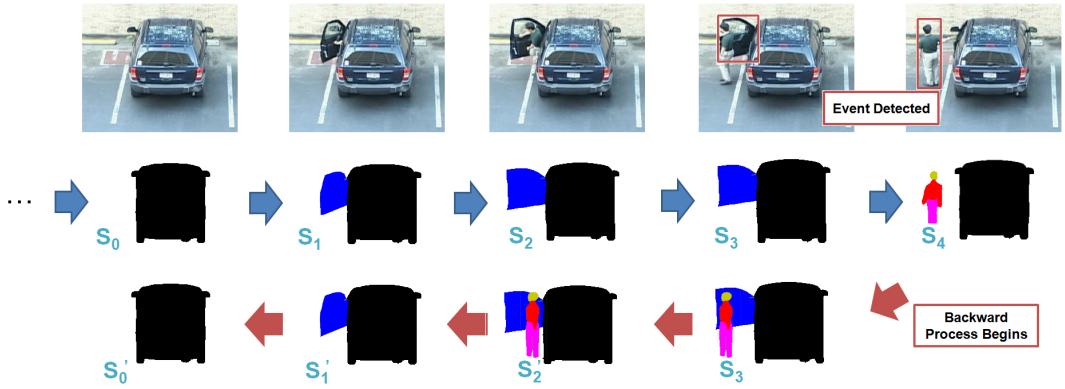

Fig. 3.6: An example backward tracking process initiated by the event 'a person exiting a car'. The event triggers the backward tracking, successfully correcting previous scene states to contextually agree with the event.

reaches the frame where he/she comes out of the vehicle for the first time). Fig. 3.6 shows an example backward tracking process. For computational efficiency, we concatenate the backward process for a certain amount of frames (i.e. delays the initiation of the backward tracking mode), so that the backward updates can be done at once without having a duplicate update process.

3.5 Experimental results

We tested the system's ability to track scene states from videos of humans interacting with a vehicle. We generated a dataset of 20 video sequences for our experiments. Each video sequence includes one vehicle and one to four interacting persons. Each person either enters into or gets out of the car (or both) in a video at least once. The videos were filmed at 12.5 frames per second with the resolution of 360 by 240 pixels. Five different actors participated

in the experiment, and a total of 2535 frames have been collected.

In each sequence, an actor interacts with a vehicle at least once and at most twice. In the first 12 sequences, each actor appearing in the scene (note that there can be 1 to 4 actors) performs both 'entering' and 'exiting' interactions. In the other 8 sequences, only one interaction is performed per actor. Among 20 sequences, 6 videos were taken with a single actor, another 6 videos contain two actors, and the other 8 videos were taken with four actors. As a result, a total of 36 entering and 35 exiting interactions are performed. The videos with four actors are particularly challenging, since multiple persons participate in the interaction with the vehicle body and doors, occluding each other as we can observe from Fig. 3.7 (c). We have measured the tracking accuracies of all persons appearing in the videos. For each person, the system estimates his/her trajectory using our approach and compares it with its ground truth trajectory. The tracking process at each time frame is said to be correct if the tracked bounding box of the person overlaps more than 75% of the ground truth bounding box. For each estimated trajectory, we find the longest interval in which the object is correctly tracked. We define the tracking accuracy as the length of this longest interval divided by the length of the entire ground truth trajectory. The tracking accuracies of persons are averaged to measure the mean accuracy of our system.

We have compared our system, scene state with event context (SSEC), with a baseline system similar to [58], which considers occlusion among persons and uses MCMC to solve the tracking problem. This system does not take

Table 3.1: Tracking accuracy results

Scene condition	Avg. Tracking Accuracy		Number of Frames
	Baseline*	SSEC	
1 person	85.4 %	92.0 %	852
2 persons	85.2 %	93.3 %	788
4 persons	67.5 %	81.5 %	895
Total	**79.1 %**	**88.7%**	2535

Table 3.2: *Baseline: MCMC based human tracking using only 3D human model. SSEC: Scene state with event context

advantage of a 3-D vehicle model or event context, and tracks objects purely in terms of human models. The objective of this implementation is to compare our system with others to confirm the advantage of our new system using event context.

Table 3.2 shows the overall tracking accuracies of the two systems. Our approach clearly outperforms the baseline. The previous method performed particularly worse for videos with four persons. This is due to its inability to analyze detailed scene states with severe human-vehicle occlusion. We are able to observe that the use of 3-D scene state models and event context benefits the system greatly. The tracking accuracy of one-person scenes and that of two-person scenes are observed to be similar. This result is because of the fact that the occlusions in two-person scenes are not severe: each of them usually gets in or out of the car from a different direction. Therefore, the difficulty of tracking humans in one-person scenes was similar to the one in two-person scenes. Fig. 3.7 shows example tracking results of human-vehicle interaction

videos. Actors appearing in the videos are tracked very accurately by our improved tracking system. Tracking of one person in Fig. 3.7 (c) failed at the beginning of his appearance, but the system was able to recover quickly.

In addition to measuring tracking performance, We have compared the activity recognition rate of our previous approach - ROI / view-independent features (ROI/VIF) [30] and this approach - scene state and event context (SSEC). Three more exiting sequences are added (39 exiting, 36 entering). Total of 75 sequences are temporally segmented and each sequence contains one action of getting into or getting out of a car. True negatives are not counted in this experiment.

Table 3.8 and Table 3.9 show the interaction recognition results of ROI/VIF and SSEC, respectively. SSEC approach performed 78.6% overal accuracy rate and ROI/VIF approach performed 54.0% overal accuracy rate.

Fig. 3.10 and Fig. 3.11 show comparison results of our two methods, ROI/VIF and SSEC. "Person getting into" activity shows more accurate results than the other activity. ROI/VIF method shows low precision due to false positives from overlapped door regions.

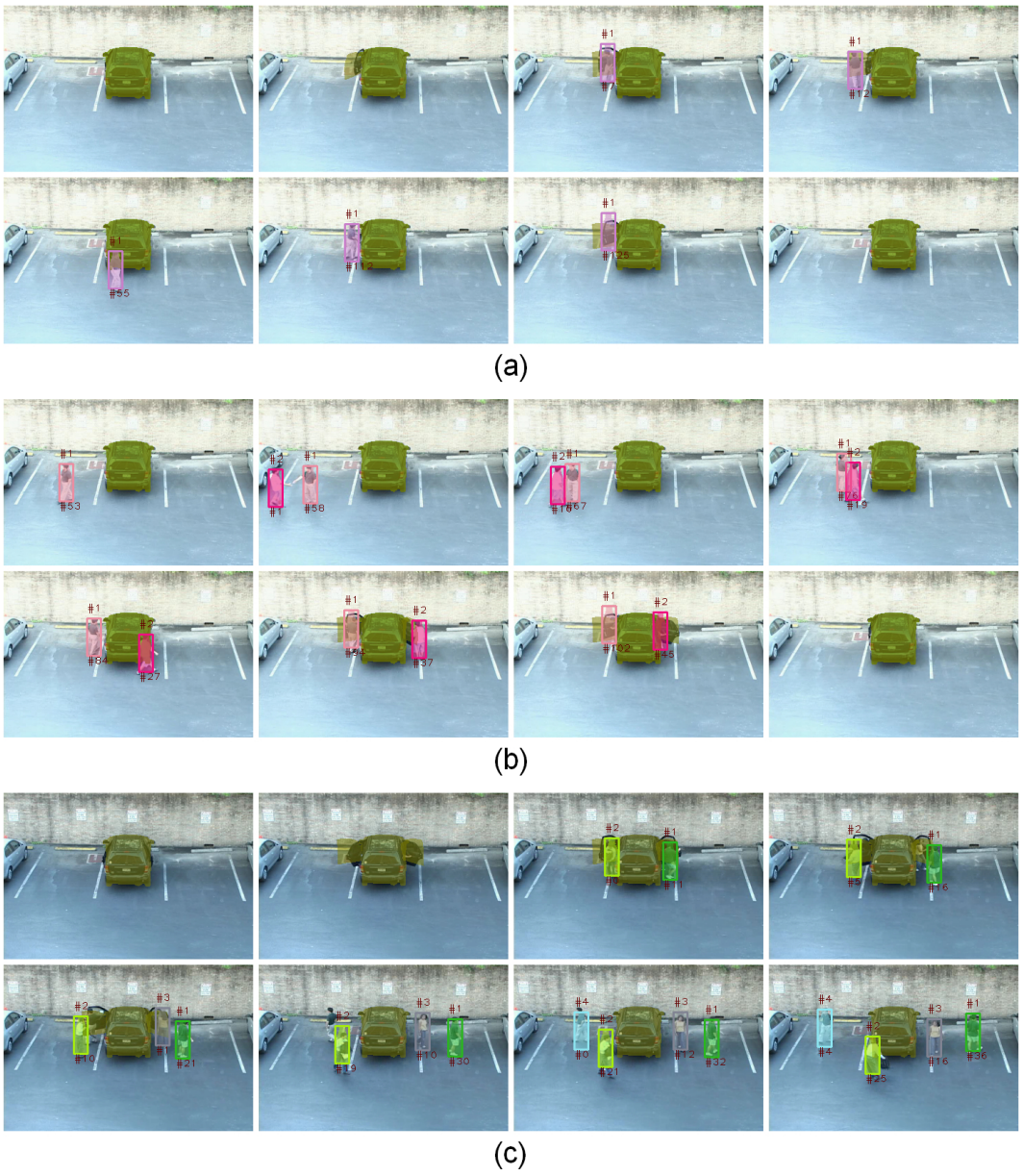

Fig. 3.7: An example of tracking results on humans interacting with a vehicle in various environments: (a) one person exits and enters a car, (b) two people enter a car, and (c) four people exit a car.

ROI/VIF	TP	FP	FN	ACC(%)
Out of	30	21	9	50.00
Into	30	15	6	58.82
Total/Avg	60	36	15	54.05

Fig. 3.8: Human-vehicle interaction recognition results of ROI / view-independent features approach

SSEC	TP	FP	FN	ACC(%)
Out of	29	0	10	74.36
Into	30	0	6	83.33
Total/Avg	59	0	16	78.67

Fig. 3.9: Human-vehicle interaction recognition results of scene state with event context approach

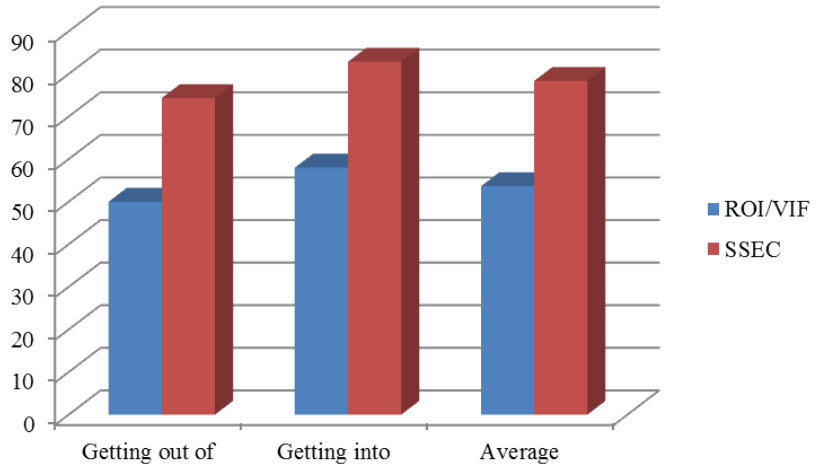

Fig. 3.10: Comparison results of ROI / view-independent features approach and scene state with event context approach in each activity.

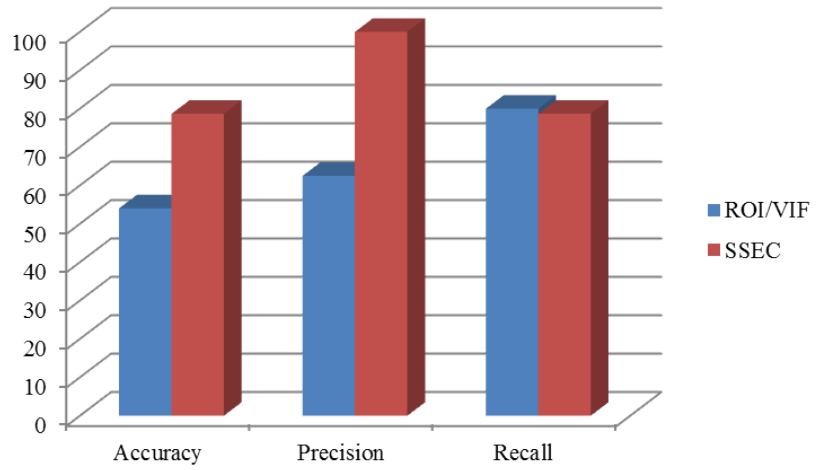

Fig. 3.11: Comparison results of ROI / view-independent features approach and scene state with event context approach on accuracy rate, precision, and recall.

Chapter 4

Human-vehicle interaction recognition using dynamic context

In this chapter, we design a probabilistic algorithm to track humans and analyze their dynamic relationships with vehicles using *dynamic context* including motion context and event context. We also present a robust vehicle alignment method. The motion of interactions between humans and a vehicle cannot be analyzed without reliable vehicle alignment. Furthermore, we extract view-independent features which allow for the system to analyze motion context with a lower number of training samples.

4.1 Bayesian formulation

We have formulated the tracking process of human-vehicle interactions as a Bayesian inference of computing the posterior probabilities of scene states in Section 3.2:

$$S^{max}_{(1,2,...)} = argmax_{S_{(1,...,n)}} P(S_{(1,...,n)}|O_{(1,...,n)}),$$

where S_i is a scene state at frame i, O_i is an observation at frame i, and n is the number of frames observed. Again, we want to compute the optimum sequence of scene states that matches with the observations best.

Here, we extend our Bayesian formulation to take advantage of not only 'event context' but also 'motion context' for reliable and detailed tracking of scene states. As mentioned in the previous sections, motion analysis results can be treated as an important feature that benefits the tracking process greatly.

While our previous formulation described in Chapter 3 is not able to integrate motion analysis and event context to the framework, we extend the Bayesian tracking formula as shown in Equation 4.2 so that certain events between a vehicle and a human and their motion/action change the prior probabilities of objects. Furthermore, event detection results can be probabilistically integrated. Observation O is defined to include appearance (A) and dynamics (D) from motion(M) and event(E). We make a second-order Markov assumption for dynamic context:

$$P(O_n|S_1, ..., S_n) \cdot P(S_n|S_{n-1}) \tag{4.1}$$

$$= P(A_n|S_n, S_{n-1}) \cdot P(D_n|S_n, S_{n-1}) \cdot P(S_n|S_{n-1})$$

$$= P(A_n|S_n) \cdot P(S_n|D_n, S_{n-1}) \cdot P(E_n|S_{n-1}) \tag{4.2}$$

In Equation 4.2, $P(A_n|S_n)$ represents the similarity between an input image and an object model. $P(S_n|D_n, S_{n-1})$ represents the probability of an current frame n in a particular state S_n, given an occurrence of dynamics D_n and a previous frame $n-1$ in a scene state S_{n-1}. $P(E_n|S_{n-1})$ shows the conditional probability of an event E_n given a previous scene state S_{n-1} (refer Appendix I for the details).

By solving the formulated Bayesian inference problem, we are able to

estimate the most probable sequence of scene states. New or modified probability terms from our previous formulation are described more explicitly in the following sections.

4.2 Appearance likelihood, $P(A_n|S_n)$

Calculating an accurate appearance likelihood for 'a vehicle' is one of the important steps in human-vehicle interaction recognition. Without robust vehicle alignment, our system can easily fail to recognize any interactions between humans and a vehicle. Here, we present the proposed vehicle alignment methods without shadows and with shadows in the scene.

4.2.1 Vehicle alignment without shadow

After a blob is classified as a vehicle, we extract its geometric parameters and type by the shape based matching of 3-D vehicle models. We build synthetic 3-D vehicle models of a sedan and an SUV (sports utility vehicle), then extract 2-D templates from the 3-D models. We adopt Song and Nevatia's approach [47]. They extracted 2-D images for 72 bins from a 360° orientation and 19 bins from a 90° tilt angle for optimal processing. Each 3-D vehicle model has 1368 extracted 2-D templates. Sample images of 2-D templates from our 3-D vehicle models are shown in Fig. 4.1.

For the shape based matching, the system scales the 2-D vehicle templates to have a similar size as the foreground blob. The system calculates an area matching score and a contour matching score. The area matching score

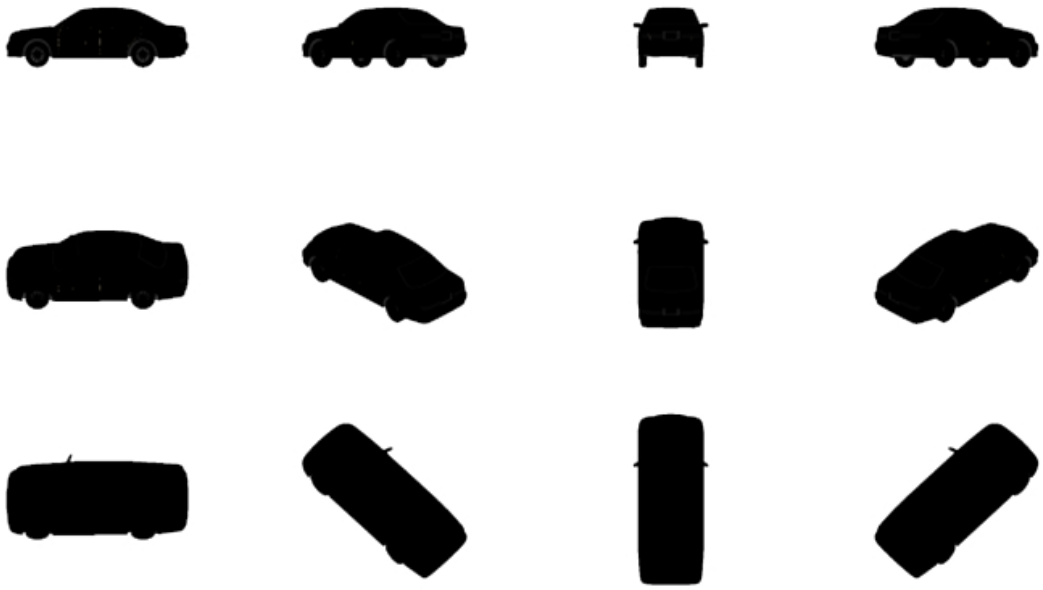

Fig. 4.1: Extracted 2-D templates from a 3-D vehicle model (sedan)

is the number of overlapped pixels of blobs and 2-D templates. The contour matching score is obtained by chamfer matching [3] on edges of blobs and contours of 2-D templates. The system calculates the final matching score by multiplying the two matching scores. The geometrical parameters and type of a vehicle can be extracted from the 2-D template which has the maximum value of the final matching score. The detection of an SUV from eight different orientations is shown in Fig. 4.2.

Vehicle blobs are tracked through frames. In general, the orientation, tilt angle, and class of the vehicle blobs do not change abruptly. We use this knowledge in order to improve the match process. While a vehicle blob is

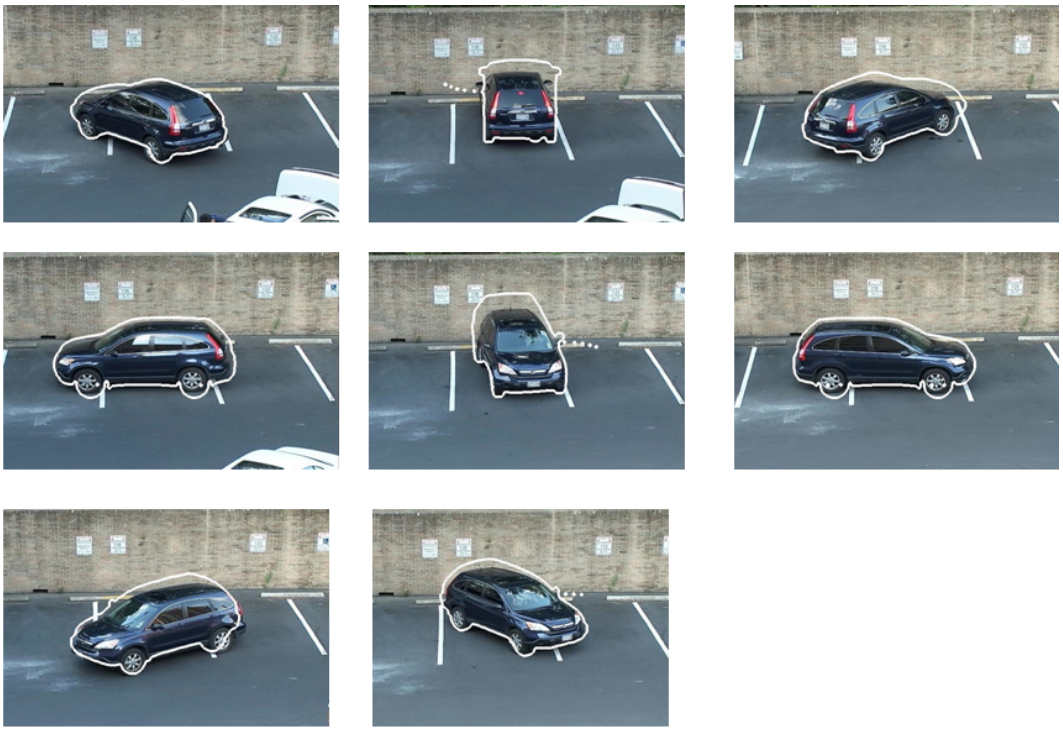

Fig. 4.2: A vehicle from various viewpoints is detected, and its silhouettes are marked by white color lines. The silhouettes are generated by a 3-D vehicle model (SUV).

detected and tracked, our system calculates a match score on each frame and the transient cost on continuous frames. We find a solution that maximizes the summation of the match score and minimizes the summation of the transition cost through the frames. Dynamic programming produces an optimal solution without excessive computation.

4.2.2 Vehicle alignment with shadow

If a video is taken in a cloudy day or indoor, the effect of shadows is minimal as shown in Fig. 4.2. However, the existence of vehicle shadows can sometimes be a serious problem in vehicle alignment because a silhouette of vehicle blob changes significantly as shown in the Fig. 4.3. In this circumstance, shape based matching of 2-D vehicle templates in Section 4.2.1 can be no longer successful. Surveillance videos often provide metadata such as time, weather, and so on. Based on a given metadata, we model the sun light on our vehicle models and generate synthetic shadow casted on a ground as shown in Fig. 4.4. Our shape based matching introduced in the previous subsection can become more accurate with this new templates. Examples vehicle alignment results with original vehicle templates and shadow-casted vehicle templates are shown in 4.5.

4.3 Dynamics likelihoods, $P(S_n|D_n, S_{n-1}) \cdot P(E_n|S_{n-1})$

In this section, we model two probability terms that influence the posterior probability, $P(S_n|D_n, S_{n-1})$ and $E(S_n|S_{n-1})$. Intuitively, the former corresponds to the probability of the 'dynamic context from event context and motion context' supporting the states, and the latter specifies the chance of an event occurrence given the previous frame state. We discuss how we model each of these terms by describing the scene dynamics.

Fig. 4.3: (a) Source images of cars with shadows on the ground (b) Its foreground blob detected by background subtraction. Because of the casted shadow, the shape of vehicle blobs changed

4.3.1 Dynamic context, $P(S_n|D_n, S_{n-1})$

In this subsection, we will describe a methodology to calculate the dynamic context. $P(S_n|D_n, S_{n-1})$ is proportional to $P(S_n|E_n, S_{n-1}) \cdot P(S_n|M_n, S_{n-1})$ (refer Appendix I for the details). Since the event context ($P(S_n|E_n, S_{n-1})$) is explained in Chapter 3, we will explain the motion context ($P(S_n|E_n, S_{n-1})$) of our new framework in the following.

4.3.1.1 View-independent feature extraction

The extraction of appropriate features is an important step that enables the system to operate fast and robustly. We extract view-independent

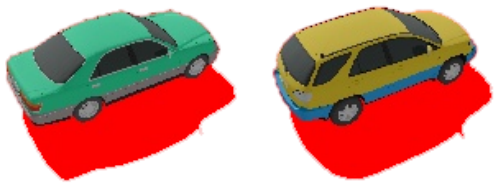

Fig. 4.4: Given light condition, shadows are synthetically generated with vehicle templates, and we can apply shape based matching with this new templates on a blob of a vehicle with a shadow.

Fig. 4.5: (a) Vehicle detection using 2D templates from 3D vehicle models without shadow (b) Vehicle detection using 2D templates from 3D vehicle models with shadow.

features after a vehicle is correctly localized. By using 3-D vehicle models, regions-of-interest (ROIs) are specified. On each ROI, we extract the optical flow and gradient. The optical flow field is not view-independent. To make it view-independent, the system transforms it using 3-D vehicle models. The transformation is accomplished by measuring the direction of a door opening or closing that fits the optical flow field. We illustrate these processes below. The careful specification of regions is an important step in ROI analysis. In the human-vehicle interactions of "a person getting into/out of a vehicle," a vehicle is parked so it does not change its location and orientation. Therefore, the system specifies ROIs only once in a human-vehicle interaction. People can get in or out of a vehicle through specific regions (door regions). Thus, the extraction of features on the ROIs can be enough for the recognition of interactions. By maintaining multiple ROIs, the system is also able to recognize several interactions simultaneously (e.g. a driver and a passenger getting out of a vehicle at the same time). We can specify ROIs of a vehicle using 3-D vehicle models with movable doors after accurate localization of a vehicle. The ROIs are correctly sized and located on the vehicle by using the 3-D vehicle models as shown in Fig. 4.6.

4.3.1.2 Transformation of the optical flow field

To recognize human-vehicle interactions, it is critical to understand the motion of humans. Because we use optical flow to detect and analyze motion, accurate optical flow calculation is important for motion analysis. Ogale and

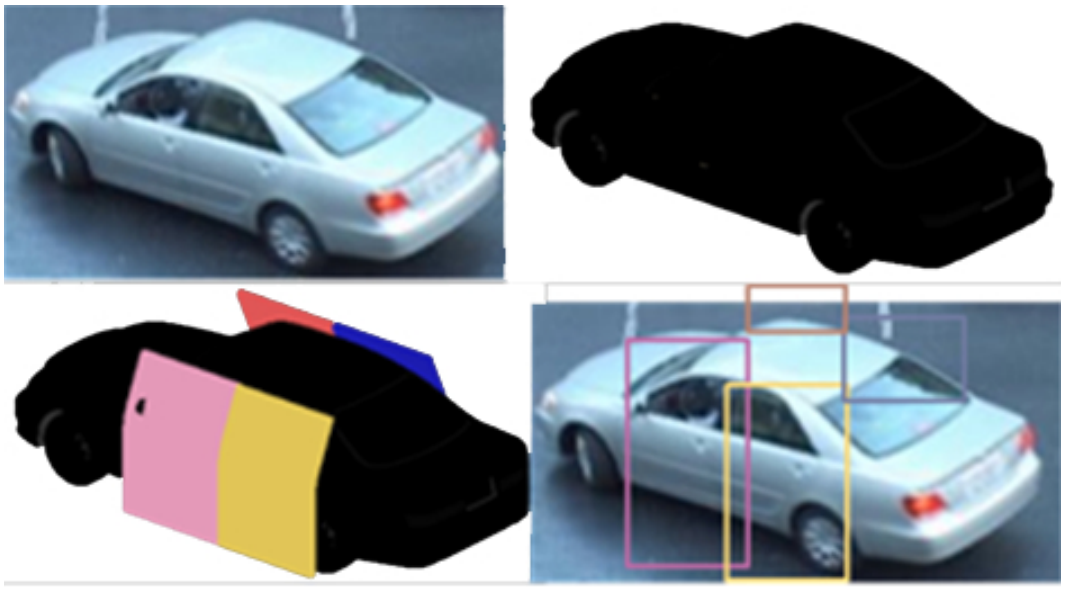

Fig. 4.6: ROI extraction. The regions of four doors are extracted separately.

Aloimonos [35, 36] proposed an advanced optical flow detection algorithm and presented its implementation. We apply their implementation to extract the optical flow accurately. However, relying on the raw extracted optical flows may cause problems because the optical flow field appears different as the viewpoint changes. Particularly, the raw optical flow field cannot distinguish whether a human opens or shuts a door.

We propose an approach to transform the optical flow field, so the system is able to extract view-independent features. Using the transformation, the system makes the direction of optical flow vectors extracted from the same interaction occurrences similar, regardless of its viewpoint. The direction of the optical flow ranges from 0 to 2π. By transformation, we normalize the

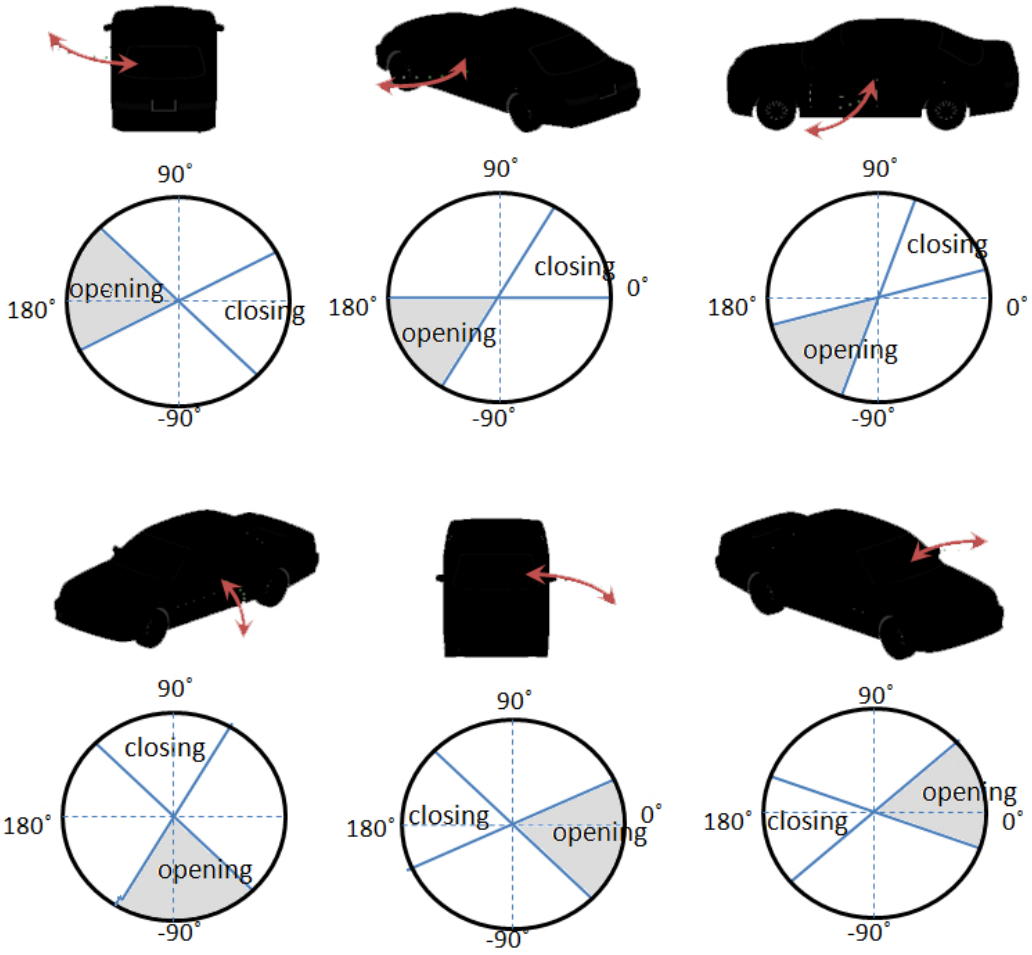

Fig. 4.7: 2-D templates from various viewpoints including door opening/closing directions and their graphs representing the range of direction. Optical flow is remapped by the direction of a vehicle. When a driver side door is opened (closed), optical flow vectors on the ROI are transformed so their angles range from 0° to 90° (from 180° to 270°).

range of direction of the optical flow on opening (or closing) a door to be from 0 to $\pi/2$ (or from π to $\pi \cdot 3/2$). In order to do that, we estimate the direction of a door opening (or closing) using 3-D vehicle models. In the 3-D vehicle models, we draw a curved line for each door to represent the direction. As we change the orientation of the vehicle models, the shape of the curved line changes also as shown in Fig. 4.7. We can estimate the range of direction for a door opening (or closing) from the selected 2-D templates (from Section 4.2.1) of the 3D models as shown in Fig. 4.8. Let $[\theta_1, \theta_2]$ (or $[\theta_3, \theta_4]$) be the range of direction for door opening from an assigned viewpoint. We can now transform the direction of the optical flow vectors by using the following equation:

$$\Theta' = \frac{\pi}{2} \cdot \left(\frac{mod(\Theta - \theta_i,\ 2\pi)}{mod(\theta_{i+1} - \theta_i,\ 2\pi)} + (i-1) \right) \tag{4.3}$$

$$if\ \ \{\theta_i \leq \Theta \leq \theta_{i+1}\}\, or\, \{\theta_{i+1} \leq \theta_i\ \&\ (\Theta \leq \theta_{i+1} \| \theta_i \leq \Theta)\}$$

$$for\ \ i = 1,\ 2,\ 3,\ 4\ (\theta_5 = \theta_1)$$

As a result, the system obtains a set of optical flow vectors whose direction is transformed to be view-independent. Using our 3-D vehicle model, the system adaptively transforms the vectors depending on the viewpoint.

4.3.1.3 Histogram of transformed & oriented optical flow and histogram of oriented gradient

We build two histograms to reduce the dimension of features. One histogram is of the optical flow field for analyzing motion and the other histogram is of the gradient field for analyzing shape. Both transformed optical

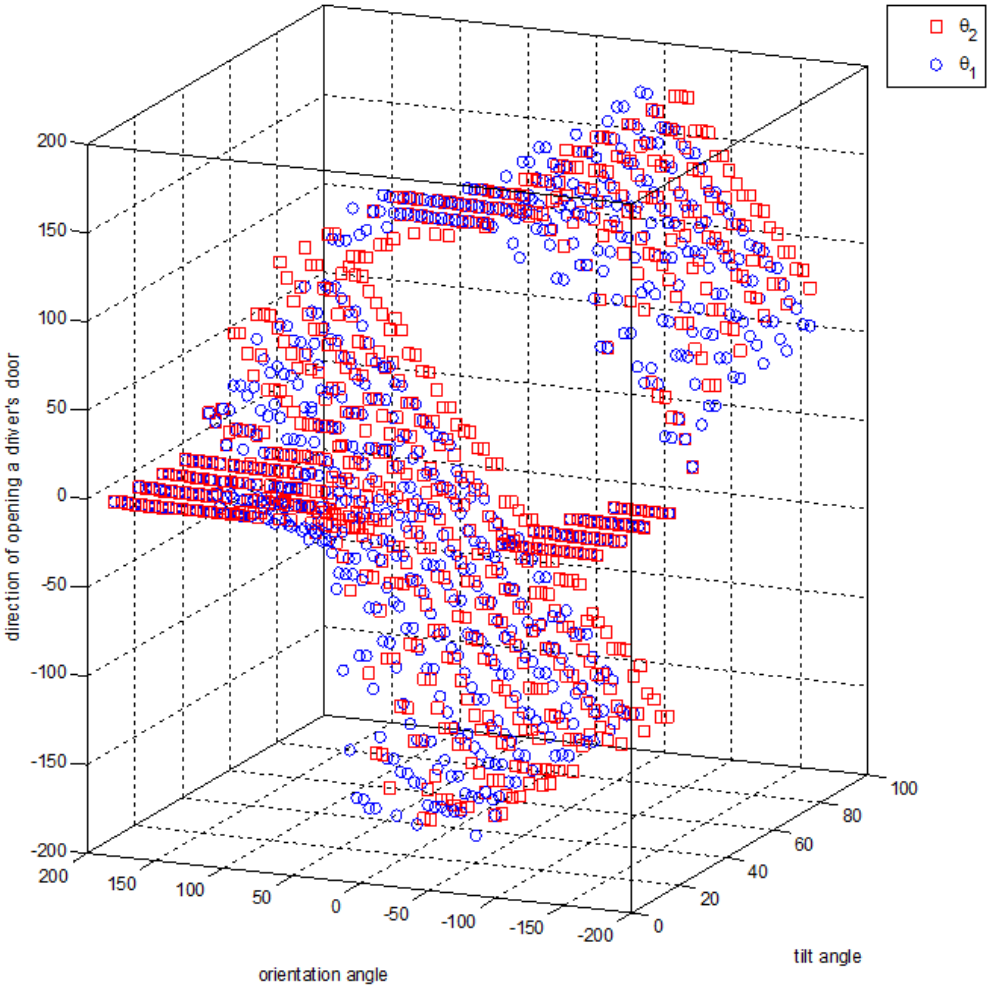

Fig. 4.8: Estimated direction of opening a driver's door from all orientation and tilt angles of a vehicle (θ_1, θ_2).

flow and raw image gradients are used to construct the histograms: motion is significantly dependent on viewpoints, while the shape of humans is not.

We classify sub-events (S_n^A) from the motion context (E^M) and the known vehicle state (S_{n-1}^V). We extract parameters (geometric parameters and a type), optical flow field (*opfl*), and gradient field (*grad*). After specification of ROIs and transformation of the optical flow field, we can obtain the transformed optical flow field (*T-opfl*) and gradient field on ROI. For dimensionality reduction, we build a histogram of transformed & oriented optical flow (T-HOOF) and a histogram of oriented gradient (HOG).

$$\begin{aligned}
\mathbf{P}(S_n^A \mid E^M, S_{n-1}^V) &\approx \mathbf{P}(S_n^A \mid param,\ I) \\
&\approx \mathbf{P}(S_n^A \mid param,\ opfl(I),\ grad(I)) \\
&\approx \mathbf{P}(S_n^A \mid T\text{-}opfl(ROI),\ grad(ROI)) \\
&\approx \mathbf{P}(S_n^A \mid \texttt{T-HOOF}(ROI),\ \texttt{HOG}(ROI)) \qquad (4.4)
\end{aligned}$$

Here, S_n^A refers to the sub-event of a human in frame n and *param* are the geometric parameters and type of vehicle. *opfl* (I) and *grad* (I) are the optical flow field and gradient field on image I, respectively. To build a histogram of transformed & oriented optical flow (T-HOOF), we create 9 bins for each direction (opening, closing, and the two others) so that we have 36 bins in 360° for the histogram. Each optical flow vector is weighted by its magnitude and is smoothed by Gaussian filter. To make the T-HOOF scale-

invariant, each bin is divided by the area of the ROI. Examples of T-HOOF and HOOF representations are shown in Fig. 4.9.

The second feature is HOG on ROIs. T-HOOF is a strong feature for detecting motion. However, the system may not distinguish the sub-event of "a person opening a door" from the sub-event of "a person appearing from a vehicle." To overcome this difficulty, we calculate the gradient field on pixels where the magnitude of the optical flow vectors is not zero. Because the shape of humans is more complex than the shape of doors (more edges), the magnitude of the gradient on humans is generally higher than the magnitude of the gradient on doors. We use the same number of bins for the HOG as the ones from the calculation of the T-HOOF.

4.3.1.4 Training human actions

When a person gets into or out of a vehicle, the person performs several distinguishable actions. In order to get in a vehicle, a person gets close to the vehicle, opens a door, disappears into the vehicle, and closes the door. Similarly, in order to get out of a vehicle, a person opens a door, appears from the vehicle, closes the door, and steps away from the vehicle. To represent these sub-events, we define six classes of actions as follows: "person appearing into / disappearing from a vehicle," "person opening / closing a door," "person walking around a vehicle," and "no movements."

We train an SVM classifier to classify these six sub-events (S_n^A) by using the motion context (E^M) and vehicle state (S_{n-1}^V). Several researchers [10, 32]

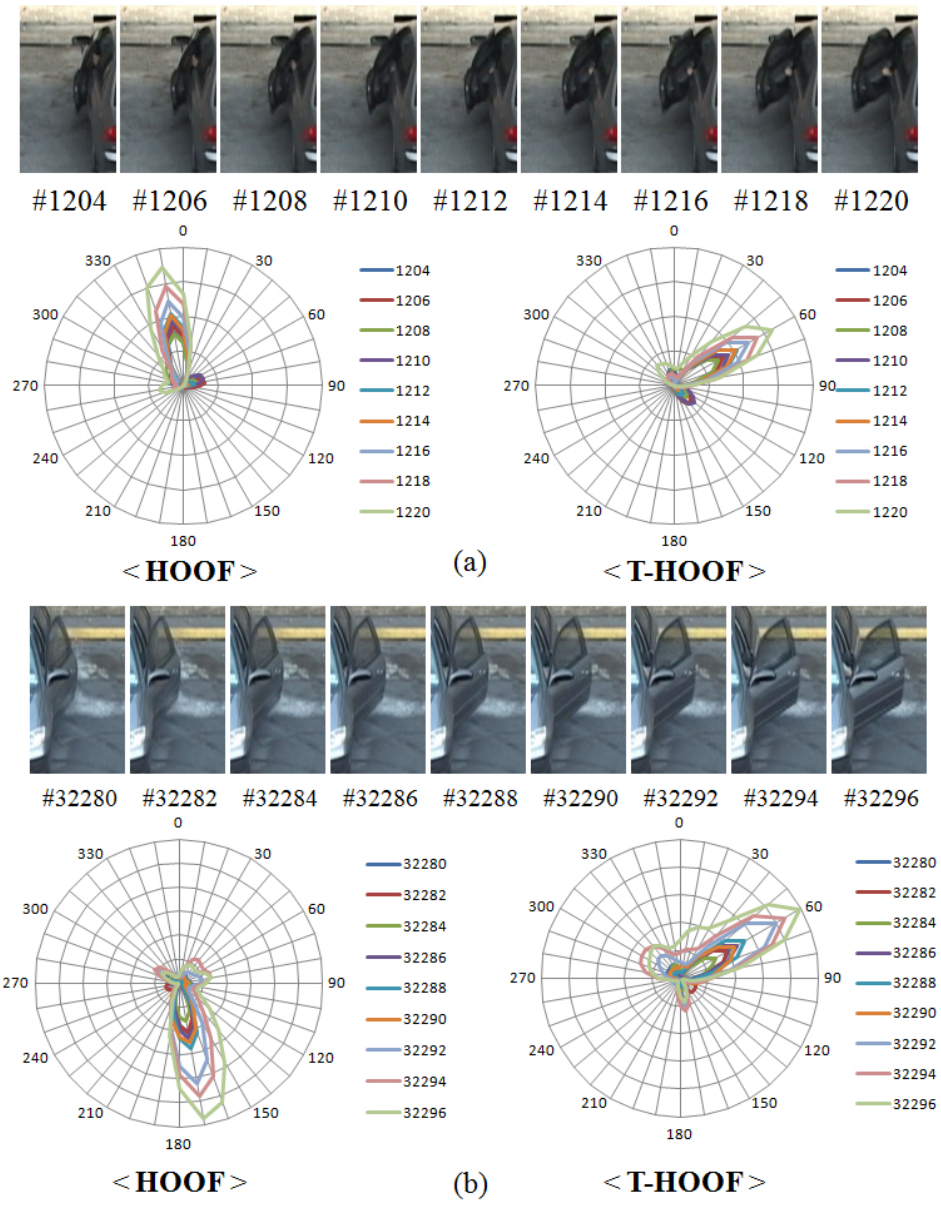

Fig. 4.9: Representations of HOOF and T-HOOF. (a) and (b) represent the same sub-event, "a person opens a door," but they are taken from different viewpoints.

used an SVM classifier with HOOF and/or HOG features, and they showed that an SVM classifier performs well with these features. Instead of training $\mathbf{P}(param,\ opfl(I),\ grad(I) \mid S_n^A)$, we train $\mathbf{P}(\texttt{T-HOOF}(ROI),\ \texttt{HOG}(ROI) \mid S_n^A)$ after extracting T-HOOF and HOG features. We use an SVM classifier with a RBF kernel to train the features simultaneously. We classify the six classes of actions on ROIs robustly using those two features in various viewpoints. More details on the classification results of these sub-events are presented in Section 4.4.

4.3.2 Event detection, $P(E_n|S_{n-1})$

Once our system is trained for six sub-events, our system is able to detect **event** based on the classification of actions in every frame. Temporal filtering is performed to improve initial sub-event classification performance and to cluster video frames which are classified as the same sub-event. Thus, we can have a series of sub-events which are composed of consecutive frames.

Ryoo and Aggarwal [44] proposed a general methodology for complex human activity recognition using Allen's event presentation [2]. Compared with their system, our system does not require the recognition of general human activities to solve the problem. All the actions are represented by one interval of temporal logic, "before." The representations of actions that a person gets into or out of a vehicle are as follows, where C_p denotes a person object and C_v denotes a vehicle object.

$$\textbf{person\_getting\_out\_of\_vehicle}(C_p,\ C_v) = ($$
$$list(\ def(o,\ \textbf{open\_door}(C_p,\ C_v)),$$
$$list(\ def(a,\ \textbf{appear\_from\_vehicle}(C_p,\ C_v)))),$$
$$and(\ before(o,\ a)\ )$$
$$);$$

$$\textbf{person\_sitting\_inside\_vehicle}(C_p,\ C_v) = ($$
$$list(\ def(o,\ \textbf{open\_door}(C_p,\ C_v)),$$
$$list(\ def(d,\ \textbf{disappear\_into\_vehicle}(C_p,\ C_v)))),$$
$$and(\ before(o,\ a)\ )$$
$$);$$

We represent the probability $P(E_n|S_{n-1})$ by using the detection of these six actions, Act_i under the assumption of conditional independency among actions:

$$P(E_n|S_{n-1}) = \prod_i P(Act_i|C_p, C_v)$$

Act_i indicates whether the action i occurred or not. $P(Act_i|C_p, C_v)$ can be estimated by averaging the probability of Act_i (from Section 4.3.1.4) among clustered frames from the temporal filtering.

4.4 Experimental results

In order to compare overall performance of our methods with other methods, we implemented a baseline method which uses *space-time interest points* in the *bag of words* framework. The classification is done using *SVM*s (STIP-BOW-SVM) [27]. We tested this method (STIP-BOW-SVM) and our two methods, Scene State with Event Context (SSEC) and Scene State with Dynamic Context (SSDC) on the new challenging dataset. The new dataset includes 15 sequences of "a person hiding near a car", 15 sequences of "a person appearing abnormally from a car (not from inside)", and 28 sequences of "A person getting into / out of one of two parallel cars". In total, 186 sequences of human-vehicle interactions are labeled as getting into (78 sequences), getting out of (78 sequences), hiding (15 sequences), and appearing abnormally (15 sequences).

Fig. 4.10, Fig. 4.11, and Fig. 4.12 show the ROC curves and confusion matrices of three approaches (STIP-BOW-SVM, SSEC, and SSDC, respectively) on our human-vehicle interaction dataset. In general, "getting into a vehicle" activity recognition rate is higher than other classes of activity recognition. While STIP-BOW-SVM approach shows ac accuracy rate of 50.5%, SSEC approach has an accuracy rate of 77.4%, and SSDC approach shows an accuracy rate of 80.6%.

As shown in Fig. 4.13 and Fig. 4.14, our approaches, SSDC and SSEC, showed superior results than the other approach. Furthermore, our improved approach using both motion context and event context show better results

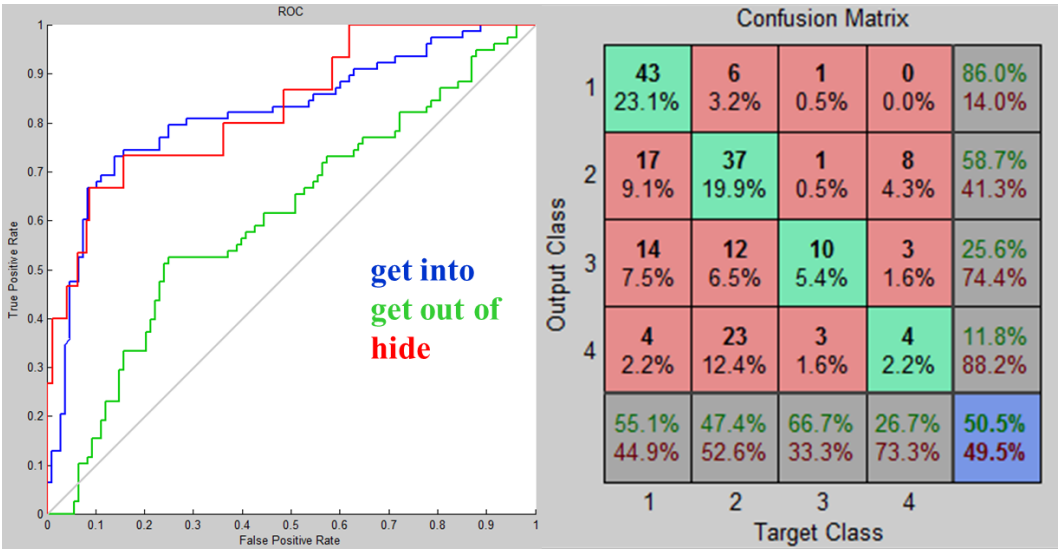

Fig. 4.10: ROC curves and a confusion matrix of STIP-BOW-SVM approach on human-vehicle interaction recognition. Class 1: get into, class 2: get out of, class 3: hide, class 4: appear abnormally. Percentiles in green color is from correct classification instances, and Percentiles in red color is from incorrect classification instances

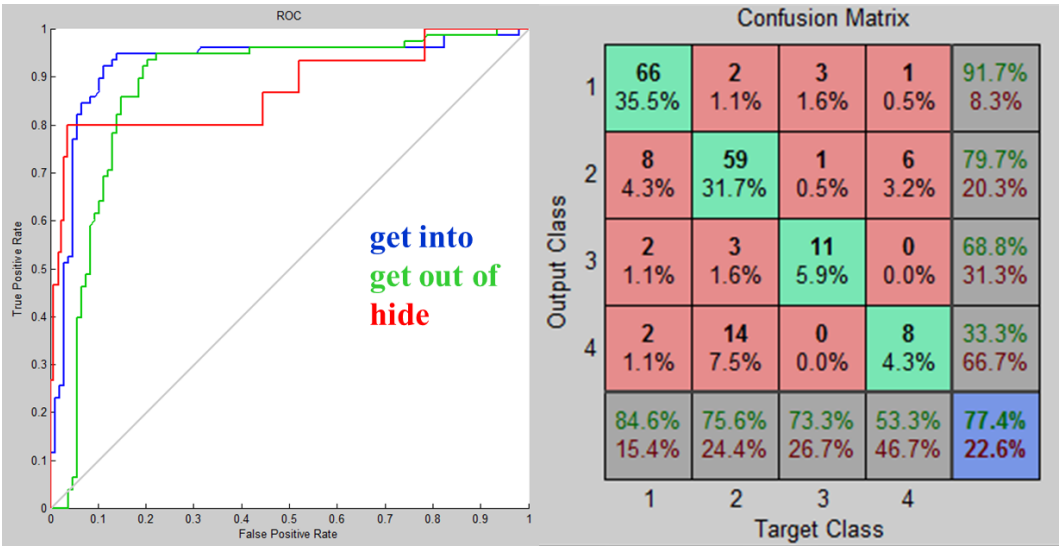

Fig. 4.11: ROC curves and a confusion matrix of scene state with event context approach on human-vehicle interaction recognition.

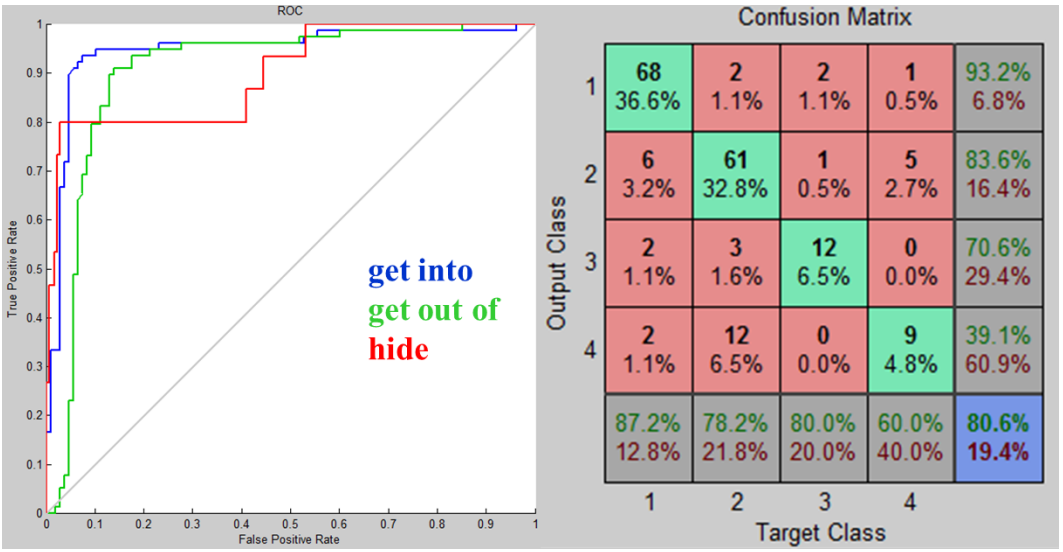

Fig. 4.12: ROC curves and a confusion matrix of scene state with dynamic context approach on human-vehicle interaction recognition.

than our approach using only event context.

We also test the performance of our view-independent features (T-HOOF) compared to basic features (HOOF) in human-vehicle interactions such as "a person getting into a vehicle" and "a person getting out of a vehicle." We generated two video datasets for our experiments. Each dataset includes four executions of two interactions performed by a driver from eight different views. Thus, each dataset has 64 human-vehicle interactions. We use 32 interactions for training and the other 96 interactions for testing. Vehicles used in the dataset are a sedan and an SUV. The videos were taken at 12.5 frames per second with the resolution of 720 * 480.

We present accuracy rates for the classification of actions to compare T-HOOF with HOOF as shown in Fig. 4.15 and Fig. 4.16. T-HOOF performed superior to HOOF in all provided conditions. The performance of HOOF and T-HOOF is improved by adding a feature, HOG and by processing temporal filtering.

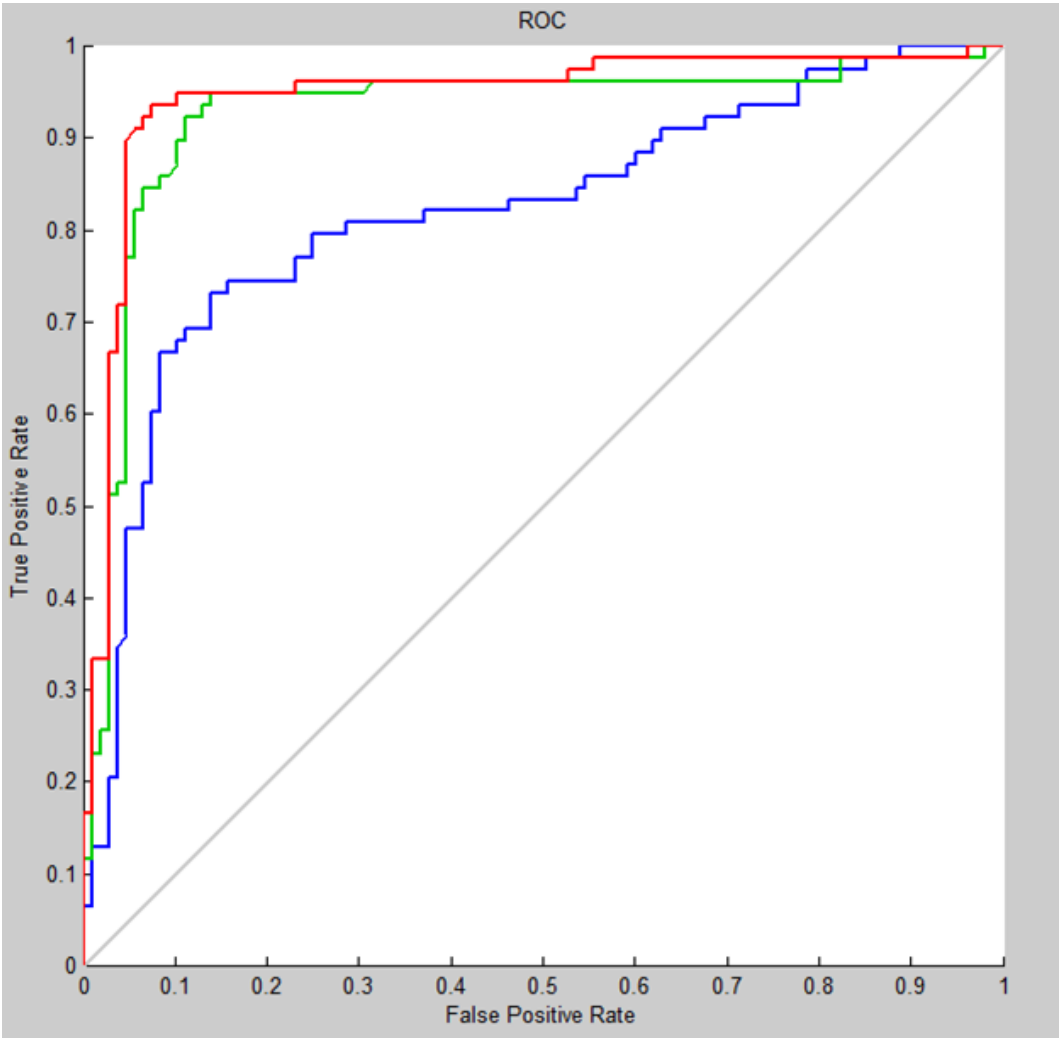

Fig. 4.13: ROC curves of three approaches on person getting into vehicle activity. Red, green, and blue curves indicate a result from scene state with dynamic context, scene state with event context, and STIP/BOW/SVM approaches, respectively.

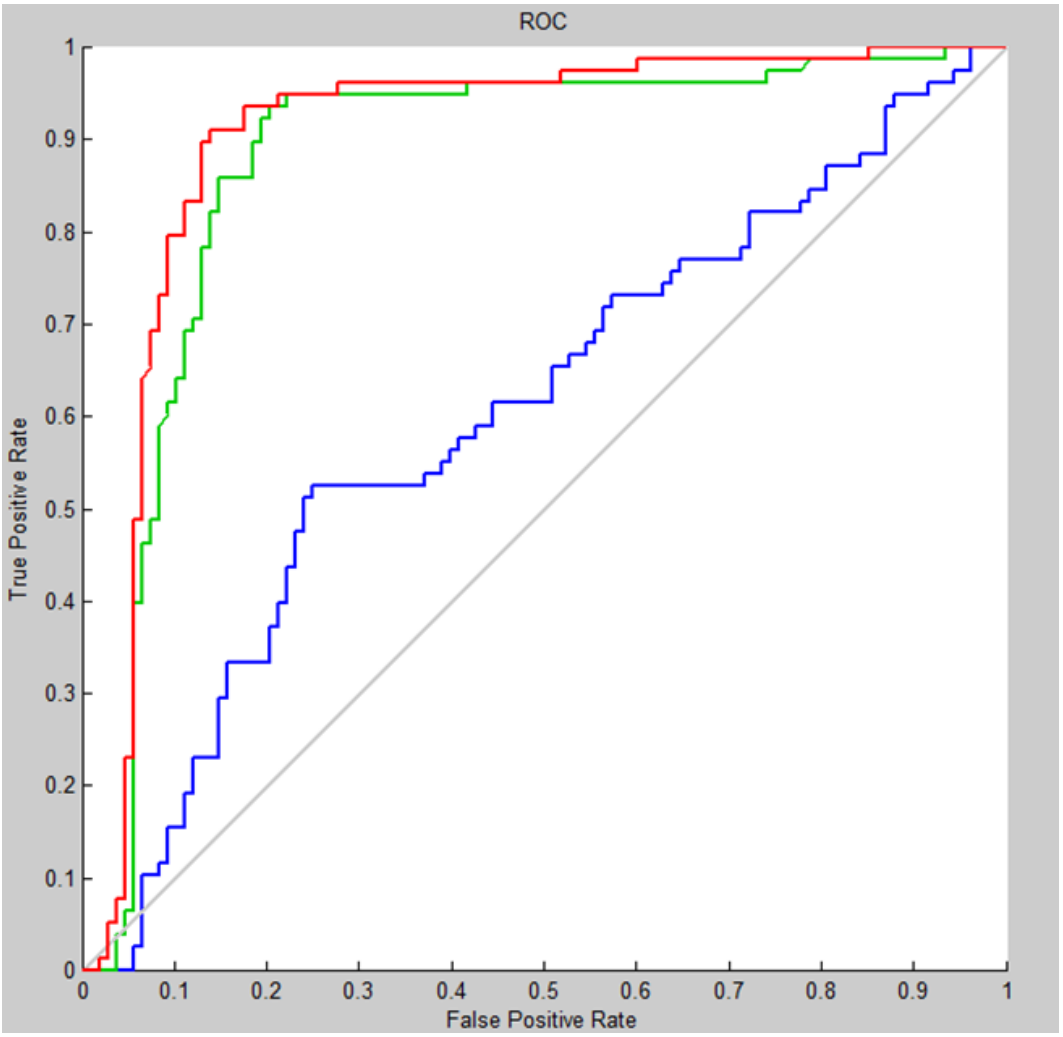

Fig. 4.14: ROC curves of three approaches on person getting out of vehicle activity. Red, green, and blue curves indicate a result from scene state with dynamic context, scene state with event context, and STIP/BOW/SVM approaches, respectively.

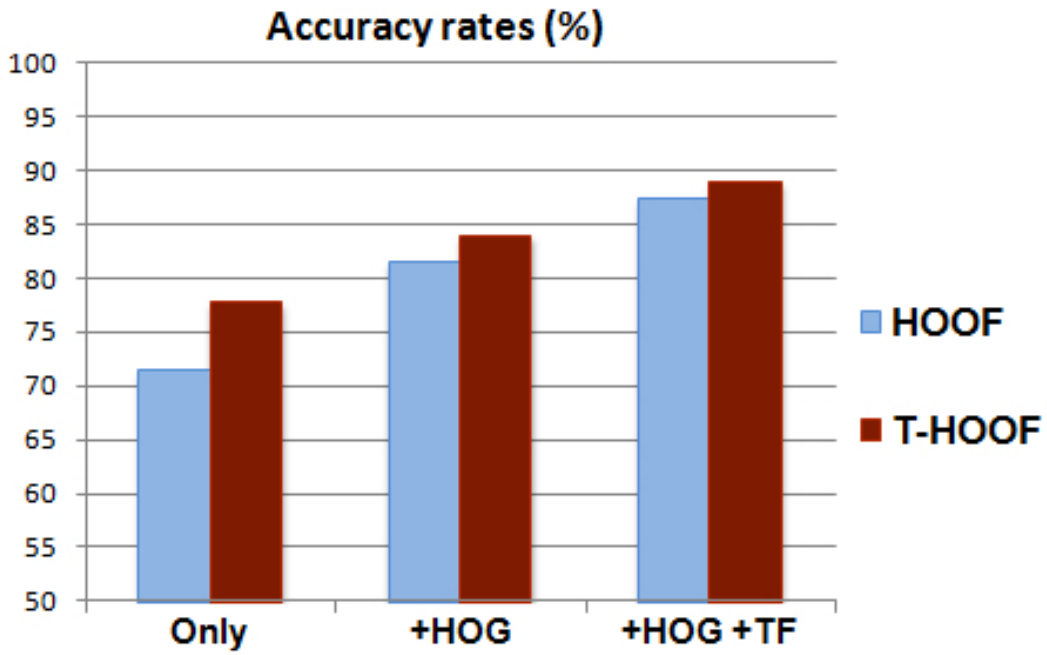

Fig. 4.15: Overall accuracy rates for the classification of actions to compare T-HOOF with HOOF. 'only,' '+HOG,' and '+HOG +TF' denote that HOOF/T-HOOF is used *without additional features or processing, with the HOG feature,* and *with the HOG feature followed by temporal filtering*, respectively.

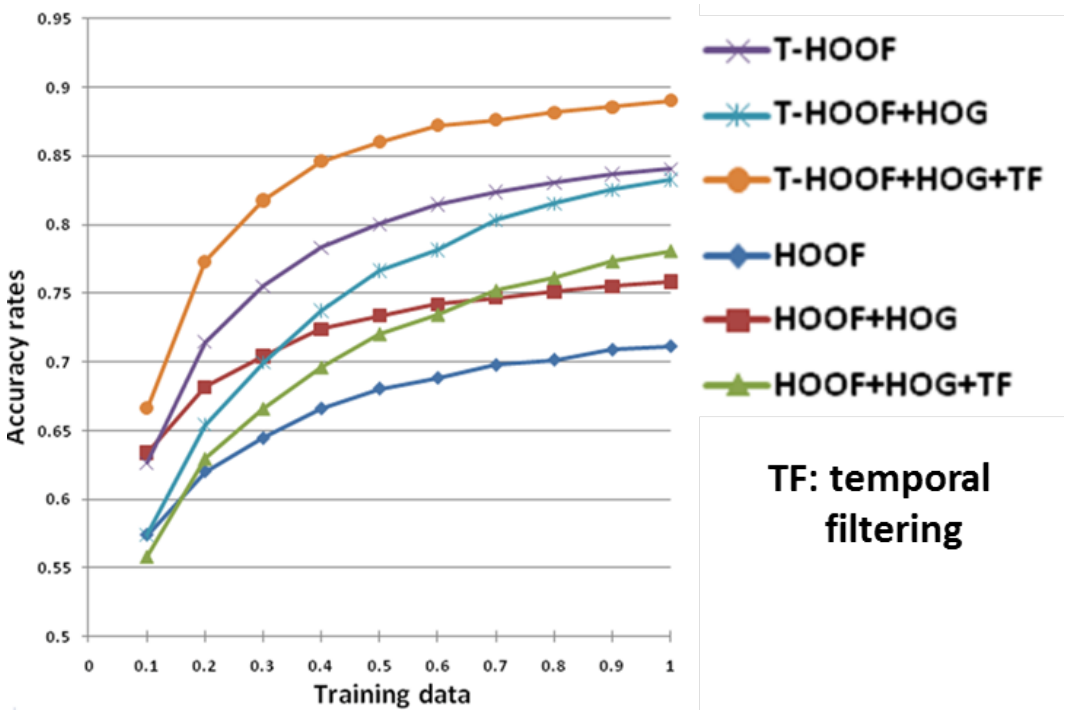

Fig. 4.16: Comparison results of T-HOOF with HOOF according to training data.

Chapter 5

Human-vehicle interaction from aerial view

In this chapter, we propose a method to recognize human-vehicle interactions from low resolution UAV videos. In this scenario, the object resolution is low, the visual cues are vague, and the detection and tracking of objects are less reliable as a consequence. Any methods that require the accurate tracking of objects or the exact matching of event definition are better avoided. To address these issues, we present an alignment method of 3-D vehicle model with synthetically generated training samples and a temporal logic based approach which does not require training from event examples.

5.1 Alignment of 3-D vehicle model

The robust alignment of a 3-D vehicle model is essential for the system to extract event ROI and to estimate the human-vehicle spatial relationship. In this section, we propose a novel and generic approach for the optimal search of vehicles states by the alignment of 3-D vehicle models. In the following subsections, we explain the details of our methodology from 1) 3-D model rendering, 2) localization of a vehicle centroid, 3) estimation of vehicle orientation, and 4) the optimal search of vehicle states using dynamic programming.

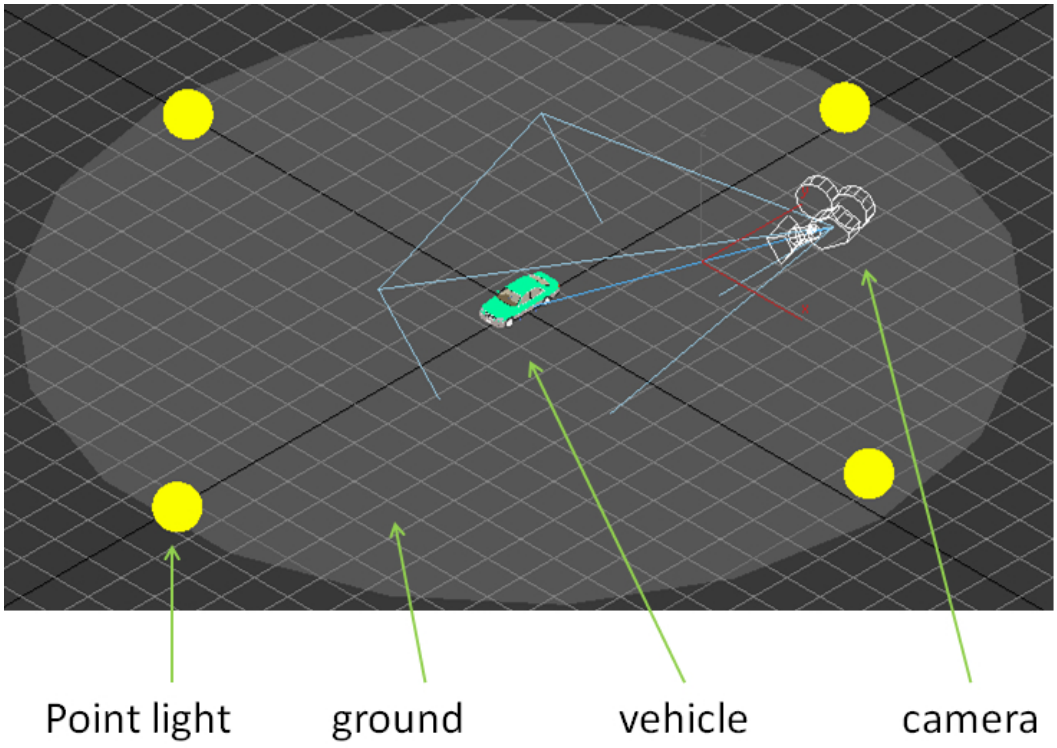

Fig. 5.1: A ray tracer with 3-D scene including a vehicle.

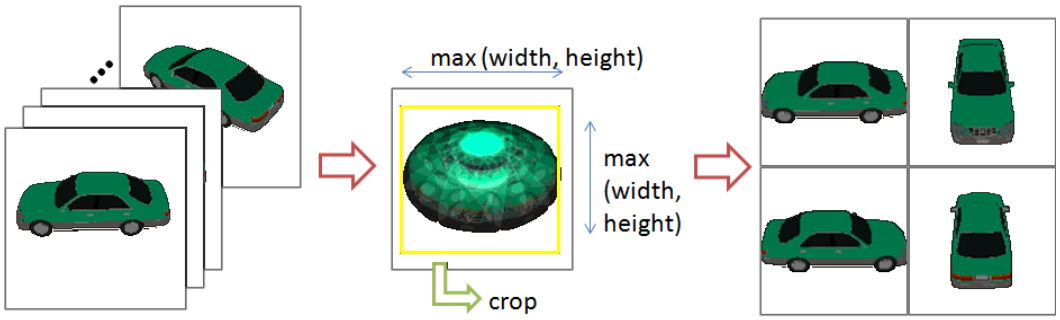

Fig. 5.2: Positive vehicle training sample generation.

5.1.1 3-D vehicle model

Collecting training samples for vehicle detection is a tedious task, and it is impractical to collect them in all possible view points. Therefore, we use ray tracing with 3-D vehicle models to generate controlled training images with detailed annotations. In order for our ray tracer to generate synthetic training samples, we create the scene of vehicles using the following descriptions: we place a vehicle model in the center of a 3-D space and a ground plane model below the vehicle model. Then, four point light sources are placed on the front, rear, left, and right of the vehicle model, respectively. Finally, a scene camera is added and controlled by the system as shown in Fig. 5.1. By adjusting the position and direction of the camera, our ray tracer can generate the projected images of a 3-D vehicle in different orientations.

Without loss of generality, our ray tracer disables reflection and refraction. It is not possible for the system to simulate the detailed characteristics of the texture of vehicles and the ground from most aerial video data due to low resolution scenes and compression errors.

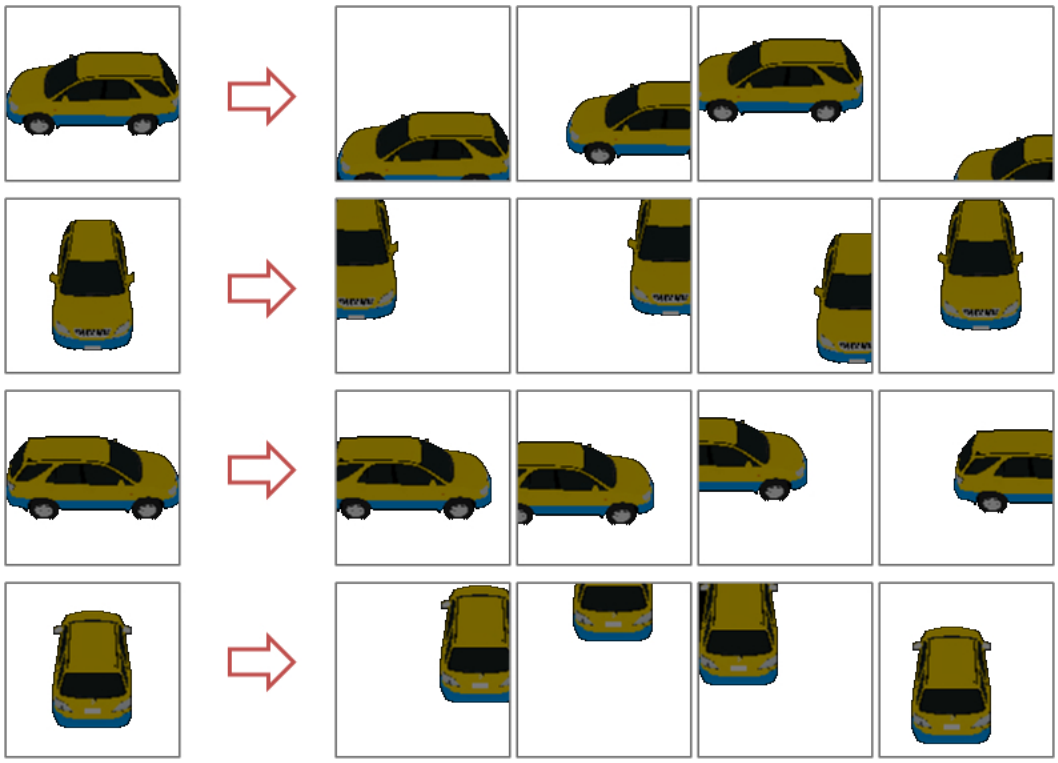

Fig. 5.3: Negative vehicle training samples.

5.1.2 Vehicle location detection

In this subsection, we explain the probabilistic approach to localize the centroid of the vehicle. Here, we assume that a vehicle is completely visible in the scene. We train an SVM classifier with the Histogram of Oriented Gradient (HOG) [15] features extracted from positive and negative vehicle sample images from 3-D vehicle models. The positive sample images have a vehicle at the center of the image and the negative sample images either have a vehicle near the boundary of the image or do not have a vehicle. Therefore, the trained binary SVM classifier can estimate the probability of the vehicle located at the center of a testing image.

The positive sample set has 720 images from 360 degree orientations and 2 vehicle types. The size of the projected image of a vehicle varies with respect to the camera views. These training samples are uniformly resized with a minimal margin as shown in Fig.5.2. In this process, we measure the maximum length of the height and width of a vehicle for all orientations, crop the margin, and resize the cropped image.

The negative sample set is generated from the positive sample set. For every positive set sample, we generate four negative samples by relocating the vehicle image of the positive sample. For the generality of the negative sample set, the relocation is processed randomly in x and y direction. The system chooses the sample if the center of the relocated vehicle is far enough from the center of the image. Fig.5.3 shows negative vehicle training samples.

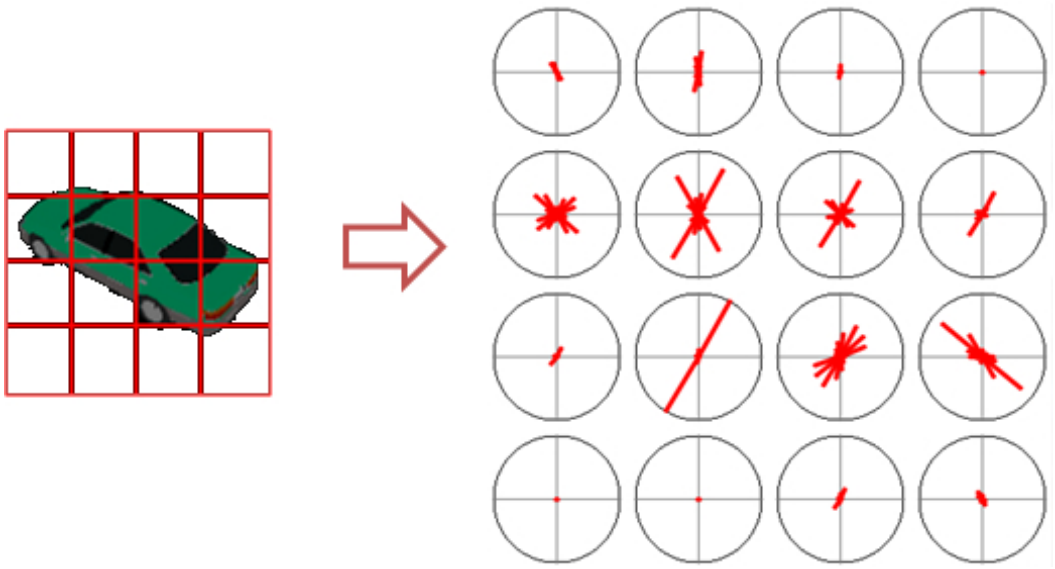

Fig. 5.4: The configuration of our HOG descriptors.

Extracting reliable features from the generated training samples is as important as generating robust training samples. The HOG descriptor has shown its excellence in detecting humans and vehicles. Here, we compute HOG descriptors from square image patches using 4x4 cell rectangular blocks, 9 orientation bins, and an unsigned gradient as shown in Fig. 5.4.

We train an SVM classifier with the HOG descriptor of generated positive and negative sample images. The classifier has two classes: 1) *positive*, a vehicle is located in the center of an image and 2) *negative*, a vehicle is not located at the center of an image [51].

In order to correct vehicle location in the given image with a tracked vehicle presence, we scan the image by sliding a window to extract the HOG and calculating the probability of vehicle existence in the center of the window

by the SVM classifier. The center of a window with the highest probability of vehicle existence ideally indicates the centroid of the vehicle in the given image.

5.1.3 Vehicle orientation estimation

Accurate vehicle orientation estimation enables the extraction of regions-of-interest (ROI) such as door regions after the vehicle location detection. This subsection explains the method to estimate 360 degree vehicle orientation in the order of 10 degree. The method of vehicle orientation estimation is similar to the method of vehicle location detection in that both methods use generated images from a ray tracer with 3-D vehicle models and extract the HOG descriptor from the synthetic images.

We train an SVM vehicle orientation classifier with the 720 images and their HOG descriptors from positive samples of vehicle location detection. The classifier has 36 classes for every 10 degrees so that each class has 20 training images.

Then, the SVM classifier estimates the probabilities of vehicle orientations in the testing images. Our SVM classifier can perform correctly if the vehicle is located in the center of testing images (Fig.5.5 (a)). If a vehicle is not correctly located (Fig. 5.5 (b)), or does not exist in the testing images (Fig.5.5 (c)), the estimation of our classifier cannot be valid. Therefore, we need to combine the results of vehicle location detection and vehicle orientation estimation for the valid estimation of a vehicle states.

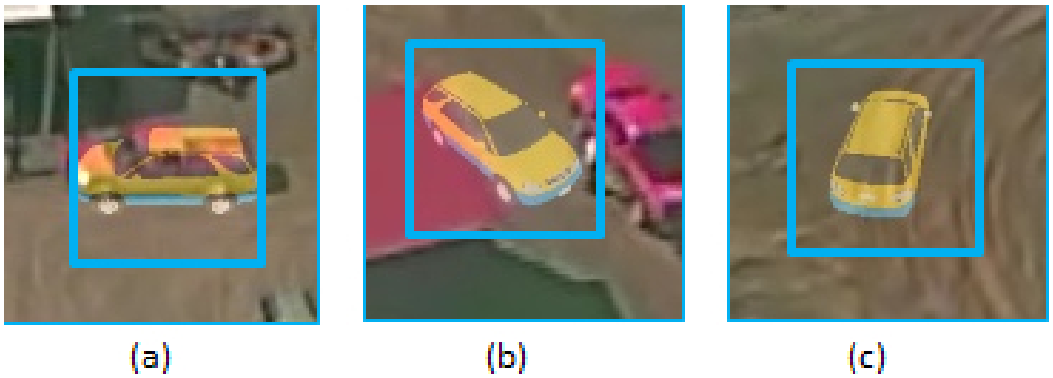

Fig. 5.5: Vehicle orientation estimation results.

5.1.4 Dynamic programming for the optimal search

In this subsection, we explain the method for the optimal search of vehicle states (location and orientation) in a sequence of frames using dynamic programming. For the event ROI extraction in Section 5.2, searching both the correct location and orientation of a vehicle is required. We first formulate the joint probability of vehicle location and orientation in a single frame under the assumption that vehicle location and orientation are conditionally independent. Then, we formulate the transition probability of vehicle states in two consecutive frames. With the formulated probability model and our dynamic programming solution, we are able to effcienly search the optimal vehicle states in every frame.

The joint probability of vehicle location (l) and orientation (o) given an image (I), $P(l, o|I)$, is represented as a product of the probability of vehicle location, $P(l|I)$, and vehicle orientation given vehicle location, $P(o|l, I)$ as shown in Equation 5.1. The estimation of $P(l|I)$ and $P(o|l, I)$ are explained

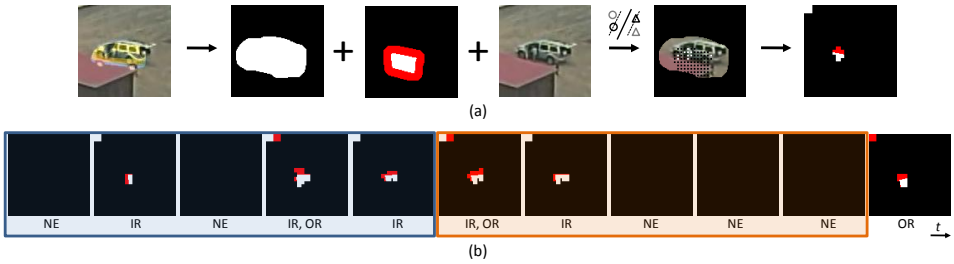

Fig. 5.6: (a) The illustration of our human detection process. (b) Our system extracts interaction associated sub-events from a labeled human-vehicle sequence using a two-sided sliding window. The sliding window detects *Meets*(IR,NE), which contributes a weighted vote to the interaction of a person getting into a vehicle.

in Subsection 5.1.2 and 5.1.3.

$$\begin{aligned} P(l,o|I) &= \frac{P(l,o,I)}{P(I)} = \frac{P(l,I)}{P(I)} \cdot \frac{P(l,o,I)}{P(l,I)} \\ &= P(l|I) \cdot P(o|l,I) \end{aligned} \tag{5.1}$$

We formulate the joint probability model of a sequence of the vehicle states given a sequence of corresponding images, $P(l_{\{1,t\}}, o_{\{1,t\}}|I_{\{1,t\}})$, under the Markovian assumption. Subscripts in equations indicate frame number(s) of variables. Let $S = \{l, o\}$, which indicates a vehicle state composed of l and o. Then, $P(l_{\{1,t\}}, o_{\{1,t\}}|I_{\{1,t\}})$ can be simplified as $P(S_{\{1,t\}}|I_{\{1,t\}})$. $P(S_{\{1,t\}}|I_{\{1,t\}})$ is expanded by using Bayes' Theorem as shown in Equation 5.2

$$\begin{aligned} P(S_{\{1,t\}}|I_{\{1,t\}}) &= \frac{P(S_{\{1,t\}}, I_{\{1,t\}})}{P(I_{\{1,t\}})} \\ &= \frac{P(S_t|S_{\{1,t-1\}}, I_{\{1,t\}})P(S_{\{1,t-1\}}, I_{\{1,t\}})}{P(I_{\{1,t\}})} \end{aligned} \tag{5.2}$$

In Equation 5.2, the term $P(S_{\{1,t-1\}}, I_{\{1,t\}})$ can be expanded as $P(S_{\{1,t-1\}}, I_{\{1,t-1\}}) \cdot P(I_t)$, and the term $P(S_t|S_{\{1,t-1\}}, I_{\{1,t\}})$ can be simplified as $P(S_t|S_{t-1}, I_t)$ by

the Markovian assumption. Also, $P(I_t)$ and $P(I_{\{}1, t\})$ are counted as constants given a sequence of images. Therefore,

$$\begin{aligned} & P(S_{\{1,t\}}|I_{\{1,t\}}) \\ & \quad \propto P(S_t|S_{t-1}, I_t)P(S_{\{1,t-1\}}, I_{\{1,t-1\}}) \end{aligned} \tag{5.3}$$

In Equation 5.3, the left term can be expanded as the following by using the Bayes' Theorem:

$$\begin{aligned} & P(S_t|S_{t-1}, I_t) \\ & \quad = P(S_t|S_{t-1})P(S_t|I_t)\frac{P(S_t)}{P(I_t)P(S_{t-1})} \end{aligned} \tag{5.4}$$

The right term can also be expanded as the following by using the Bayes' Theorem:

$$\begin{aligned} & P(S_{\{1,t-1\}}, I_{\{1,t-1\}}) \\ & \quad = P(S_{\{1,t-1\}}|I_{\{1,t-1\}})P(I_{\{1,t-1\}}) \end{aligned} \tag{5.5}$$

Under the assumption of the uniform prior probability distribution for S, Equation 5.3 can be represented as in Equation 5.6 by Equation 5.4 and Equation 5.5.

$$\begin{aligned} & P(S_{\{1,t\}}|I_{\{1,t\}}) \\ & \quad \propto P(S_t|S_{t-1})P(S_t|I_t)P(S_{\{1,t-1\}}|I_{\{1,t-1\}}) \end{aligned} \tag{5.6}$$

By induction, Equation 5.6 can be the product of a sequence of terms as shown

in Equation 5.7.

$$P(S_{\{1,t\}}|I_{\{1,t\}}) = P(S_1|I_1)\prod_{k=2}^{k=t}[P(S_k|S_{k-1})P(S_k|I_k)] \tag{5.7}$$

By replacing back S by l and o, we can derive the following equation:

$$P(l_{\{1,t\}}, o_{\{1,t\}}|I_{\{1,t\}}) = P(l_1, o_1|I_1)\prod_{k=2}^{k=t}[P(l_k, o_k|l_{k-1}, o_{k-1})P(l_k, o_k|I_k)] \tag{5.8}$$

$P(l_k, o_k|l_{k-1}, o_{k-1})$ implies the transition probability of vehicle states in two consecutive frames, k and $k-1$. $P(l_k, o_k|I_k)$ is derived from Equation 5.1. We assume that the transition probability model has an exponential distribution as follows:

$$P(l_k, o_k|l_{k-1}, o_{k-1}) = \lambda_l \cdot \lambda_o \cdot exp(-\lambda_l \cdot \|l_k, l_{k-1}\| - \lambda_o \cdot \|o_k, o_{k-1}\|) \tag{5.9}$$

After all, the problem of searching for an optimal sequence of vehicle states can be modeled as a Markov decision process. In order to have a finite set of states, locations are downsampled by every 5 pixels x 5 pixels windows, orientations are downsampled by every 10 degrees, and the original dataset with 30 fps (framesec) is downsampled in time to 2.5 fps.

Finding optimal states can be determined by a value iteration, V as follows:

Initialize $V(S_k)$ *arbitrarily*

loop for frame k

loop for states at k, $S_k = (l_k, o_k)$

loop for states at $k-1$, S_{k-1}

$$V(S_k) = max_{S_{k-1}} \{ S_P(l_1, o_1|I_1) \cdot \prod_{k=2}^{k=t} (P(l_k, o_k|l_{k-1}, o_{k-1}) \cdot P(l_k, o_k|I_k)) \}$$

end loop

end loop

end loop

Through dynamic programming, the optimal search improves with each frame. When real-time processing is required, our system provides the optimal solution in the current frame. Without the time constraints, the optimal vehicle states in previous frames can be updated using a backward search.

5.2 Temporal logic for human-vehicle interaction recognition

In this section, we introduce our temporal logic based approach, which derives the most likely human-vehicle interaction from low-level information. The low-level processing results include the localized event ROI and the locations of detected human objects, which are assigned with object states and parsed with modified temporal logic for interaction analysis.

5.2.1 Human Detection

After the process of 3-D vehicle model alignment, we perform human detection on the event ROI. As shown in Fig.5.6 (a), for the recognition of a person getting into and out of a vehicle, our 3-D vehicle alignment provides the binary masks of the vehicle and its door regions. We dilate both types of masks and apply the vehicle mask to the bounding box so that arbitrary image content around the vehicle will contribute less to the human detector. The door mask after dilation is marked with a different color to indicate the peripheral of the ROI, which is used to capture a person's approach of ROI.

We use HOG to characterize human objects in low-resolution imagery. Our SVM based human detector is trained with HOG features extracted from manually cropped figure-centric bounding boxes and negative samples from patches around the figures. To save computation, the SVM window classifier only performs detection on grid locations of the event ROI. We train linear SVM to compute calibrated likelihood values [55], which are thresholded to indicate the likely grid locations of human presence. However, the detection accuracy inevitably suffers from the blurry low-resolution imagery as in Fig.5.6 (a). Therefore, instead of taking the risk of missing true detections, a low threshold (< 0.5) is used to allow a certain amount of false positives. We perform connected component analysis on the detection grid coordinate to label the detected persons and remove unlikely blobs by area.

To identify the human-vehicle spatial relationship in each bounding box, the dilated mask of event ROI is applied to the mask of human blobs.

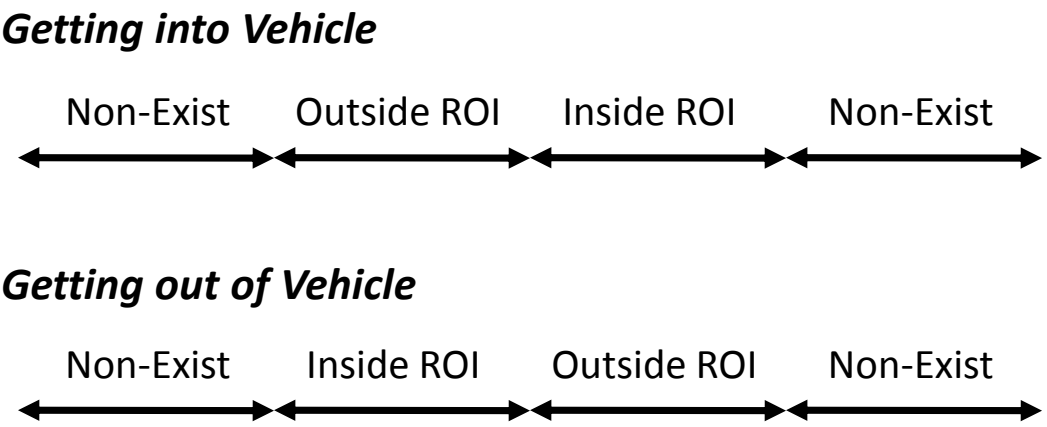

Fig. 5.7: The formal event representation of a person getting into and out of vehicle.

Based on the overlapped mask, our system estimates whether the person is inside the ROI (IR), outside the ROI (OR), or does not exist (NE) in the image patch. The specific permutations of these three event states are defined as the constituent sub-events of interactions.

5.2.2 Piecewise temporal logic

In Allen and Ferguson's classic temporal interval representation of events [2], an event is defined as having occurred if and only if the sequence of observations matches the formal event representation and satisfies the pre-defined temporal constraints. Temporal logic based approaches have been successfully applied for the recognition of human activities, human-human interactions, human-object interactions, and group activities [1]. Most importantly, instead of learning events from training examples, temporal logic allows the direct encoding of human knowledge. However, the recognition of interaction related

sub-events from aerial video is far less accurate than that in regular scenarios. Therefore, capturing human-vehicle interactions by matching them against their complete event representation is rarely a success in our experiments.

We adopt a modified temporal logic approach to mine the pieces of event evidence embedded in a human-vehicle sequence. We name our method piecewise temporal logic (PTL), which is different from the classic temporal logic in two major aspects. First, our interaction representation is defined based on event states, from which the higher level interaction associated sub-events are derived. Second, our method recognizes interactions by comparing the weighted sums of detected sub-events, the temporal relationships among which are *not* taken into account.

We found that in a human-vehicle sequence, the moments of interaction related primitive actions are not always observable and cannot be reliably recognized. Therefore, we define human-vehicle interactions in terms of the event states that lead to them. Fig.5.7 shows the formal event representation of a person getting into and out of a vehicle. Given the temporal flows of event states, interaction associated sub-events are defined in terms of the alternations of specific states. The set of predicates we used to describe the temporal relationships of event states include *Meets*, *Starts*, and *Finishes*. These sub-events are manually assigned with weights based on their relative importance to the actual occurrence of the interaction. For example, in Fig.5.6 (b), the alternation of event states from IR to NE is more informative than the change from NE to OR for the detection of a person getting into a vehicle. Table 5.1

shows the interaction associated sub-events and their corresponding weights. Note that the exact values of sub-event weights cause much less effect on the system performance than their relative values.

It is a difficult task to extract instances of sub-events from a noisy event state sequence such as Fig.5.6 (b). We propose to use a two-sided sliding window to detect interaction associated sub-events. As shown in Fig.5.6 (b), the sub-event *Meets*(IR,NE) extracted from rear and front sliding windows is compared with the human encoded list in Table 5.1. The matched sub-event contributes a weighted vote to the corresponding bin of an event histogram. We use the sum of absolute sub-event weights in an event histogram to determine if any of the two interactions have ever occurred. The normalized event histogram indicates the occurrence likelihood of interactions.

5.3 Experimental results

We test our methodology with the challenging VIRAT Aerial Video dataset [37]. The videos were taken in 30 frames per second with the resolution of 720 by 480 pixels. As shown in Fig.5.8, the challenges posed by this dataset include low image resolution, vague object appearance and motion (due to air turbulence and video compression artifacts), time-varying views, changing weather conditions, salient shadow, and cluttered backgrounds.

There are a number of human-vehicle sequences in this dataset. However, we can only find 7 instances of a person getting into and out of a vehicle. We manually select 20 other types of human-vehicle interaction sequences, in

Fig. 5.8: The snapshots of four true positive (TP), two true negative (TN), one false negative (FN), and one false positive (FP) sequence are shown. We treat the subject human-vehicle interactions (getting into vehicle, getting out of vehicle) as the positive class and all other events (others) as the negative class.

Interaction	Sub-event	Weight
Getting into vehicle	*Meets*(IR,NE)	2
	Meets(OR,IR)	1
	Meets(OR,NE)	0.5
	Finishes(IR)	-2
Getting out of vehicle	*Meets*(NE,IR)	2
	Meets(IR,OR)	1
	Meets(NE,OR)	0.5
	Starts(IR)	-2

Table 5.1: Interaction associated sub-events and their corresponding weights. IR, OR, and NE are shorts for human inside the ROI, outside the ROI, and does not exist (NE) in the image bounding box, respectively. *Meets*, *Starts*, and *Finishes* are the temporal predicates used to define their relationships.

which a person may be passing by or (un)loading the vehicle. Therefore, in our evaluation set, there are 4 sequences of a person getting into a vehicle, 3 sequences of a person getting out of a vehicle, and 20 other types of human-vehicle sequences. We use the same set of parameters for vehicle alignment and interaction analysis without any event-level training. Fig. 5.8 shows the snapshots of our testing sequences. Despite the differences in the types of vehicles, viewpoints, and interactions, our system is able to correctly detect the subject human-vehicle interactions from sequences such as the TP examples in Fig. 5.8. The FP and FN examples in Fig. 5.8 show the cases when our method fails. In the sequence of *"Getting into vehicle, FN"*, the approach of the person from the left was partially occluded by the building, and in the

sequence of *"Others, FP"* the departure of the person from the ROI misled the system.

Our system demonstrates superior results on the search of the optimal vehicle states. In 20 sequences out of 27 testing sequences (74.1%), both the orientation and location of vehicles are correctly estimated. In the 6 isntances out of 7 incorrect sequences (22.2%), the locations of the vehicles are correctly detected but the vehicle orientations are 180° reversed. In spite of that, the ROI in those sequences were correctly located because of the symmetry of vehicle shape. In the other 1 instance (3.7%), the estimation of the vehicle orientations is incorrect. For interaction recognition, we analyze sub-events in every 4-second long two-sided sliding window. The system classifies a sequence as the subject human-vehicle interactions if its sum of absolute sub-event weights exceeds 1 and there is no tie in the event histogram. A sequence is recognized as other events if the sum of absolute sub-event weights is less than 1 or there is a tie in its event histogram. Fig. 5.9 shows the confusion matrix. By treating the subject human-vehicle interactions as the positive class and all other events as the negative class, the accuracy of our method on this evaluation set is 77.78% ((TP + TN) / (TP + TN + FP + FN)), the precision is 53.85% (TP / (TP + FP)), and the recall is 100.0% (TP / (TP + FN)).

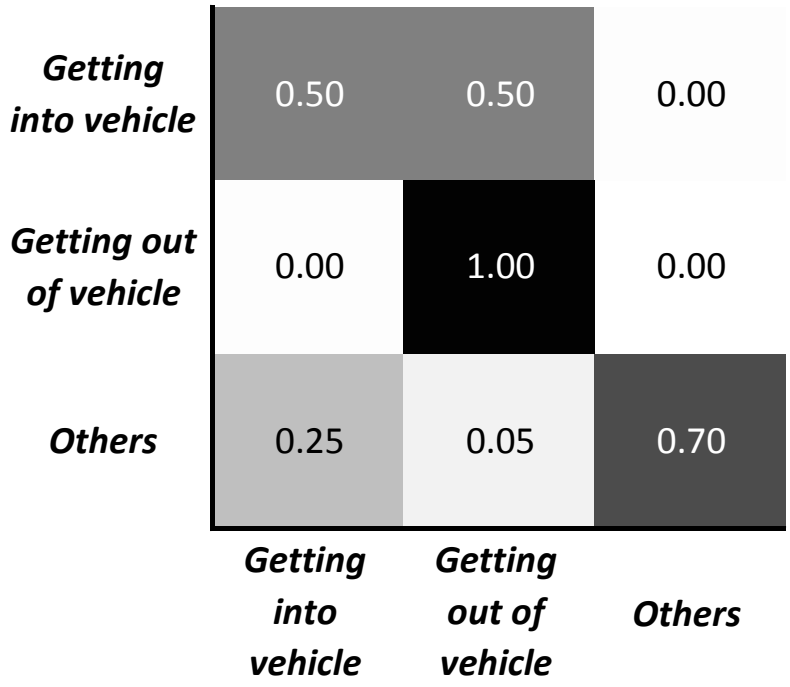

Fig. 5.9: The confusion matrix of our method on a subset of the VIRAT Aerial Video dataset.

Chapter 6

Conclusion

We presented a methodology for analyzing a sequence of scene states from videos of human-vehicle interactions. We developed a probabilistic framework for scene state tracking using 3-D scene models, identifying detailed configurations of humans and vehicles appearing in videos. Furthermore, we introduced the concept of event context which benefits the scene state analysis process greatly. Interplays between event detection and state tracking are explored probabilistically, providing better results in the experiments.

Furthermore, we extended the methodology to recognize complex human-vehicle interactions with a high degree of accuracy. The proposed methodology analyzes motion and actions from various viewpoints and improves the recognition rates of event detection. The main contributions of our work are: integration of motion based context with event based context to the framework, the extraction of view-independent features using 3-D vehicle models, and more reliable system requiring less training data. We showed that our approach is superior to the previous approaches.

Finally, we propose a general framework for the recognition of human-vehicle interactions from aerial view. Our method offers three major advan-

tages to better resolve the challenges posed in this scenario. First, we adopt a temporal logic based approach to avoid the cost of manually collecting and labeling the training examples. Second, we employ a dynamic programming based 3-D vehicle model alignment technique, which accurately locates event ROI with the consideration of the previous alignment results. Third, based on classic temporal logic, we introduce the concept of PTL, which significantly improves the recognition performance in our problem. PTL detects interaction sub-events by checking the temporal relationships between the event states. However, at the semantic-level, the temporal logics among the sub-events are not verified to induce the robustness against sequences of noisy sub-events. Furthermore, the proposed method can be generalized to recognize any kinds of human-vehicle interactions with the proper encoding and weighting of the temporal logics between event states. Most importantly, our method demonstrates high recognition accuracy on the challenging VIRAT Aerial Video dataset.

Appendix

Appendix 1

Derivations of equations

We present details of derivations of our fomulae in the appendix.

From Equation 3.2 in Chapter 3, the left term, $P(O_{(1,\ldots,n)}|S_{(1,\ldots,n)})$ can be expanded as follows.

$$P(O_{(1,\ldots,n)}|S_{(1,\ldots,n)}) = \frac{P(O_{(1,\ldots,n)} \cdot S_{(1,\ldots,n)})}{P(S_{(1,\ldots,n)})}$$

$$= \frac{P(O_n, S_n|O_{(1,\ldots,n-1)}, S_{(1,\ldots,n-1)}) \cdot P(O_{(1,\ldots,n-1)}, S_{(1,\ldots,n-1)})}{P(S_n|S_{(1,\ldots,n-1)}) \cdot P(S_{(1,\ldots,n-1)})} \tag{1.1}$$

$$= \frac{P(O_n, S_n|O_{(1,\ldots,n-1)}, S_{(1,\ldots,n-1)})}{P(S_n|S_{(1,\ldots,n-1)})} \cdot \frac{P(O_{(1,\ldots,n-1)}, S_{(1,\ldots,n-1)})}{P(S_{(1,\ldots,n-1)})}$$

$$= \frac{P(O_n, S_n|S_{(1,\ldots,n-1)})}{P(S_n|S_{(1,\ldots,n-1)})} \cdot P(O_{(1,\ldots,n-1)}|S_{(1,\ldots,n-1)}) \tag{1.2}$$

$$= \frac{P(O_n, S_n|S_{(1,\ldots,n-1)}) \cdot P(S_{(1,\ldots,n-1)})}{P(S_n|S_{(1,\ldots,n-1)}) \cdot P(S_{(1,\ldots,n-1)})} \cdot P(O_{(1,\ldots,n-1)}|S_{(1,\ldots,n-1)})$$

$$= \frac{P(O_n, S_{(1,\ldots,n)})}{P(S_{(1,\ldots,n)})} \cdot P(O_{(1,\ldots,n-1)}|S_{(1,\ldots,n-1)}) \tag{1.3}$$

$$= P(O_n|S_{(1,\ldots,n)}) \cdot P(O_{(1,\ldots,n-1)}|S_{(1,\ldots,n-1)}) \tag{1.4}$$

$$\propto P(O_n|S_n) \cdot P(O_{(1,\ldots,n-1)}|S_{(1,\ldots,n-1)})$$

Equations 1.1, 1.2, 1.3, and 1.4 can be derived by using Bayes' Theorem.

From Equation 3.2 in Chapter 3, the right term, $P(O_{(1,\ldots,n)}|S_{(1,\ldots,n)})$ can

be expanded as follows.

$$P(S_{(1,...,n)}) = P(S_n|S_{(1,...,n-1)}) \cdot P(S_{(1,...,n-1)})$$
$$= P(S_n|S_{n-1}) \cdot P(S_{(1,...,n-1)}) \quad (1.5)$$

Equation 1.5 can be derived by first order Markov assumption. Therefore, we derive Equation 3.3.

From Equation 4.1, we derive Equation 4.2 as follows.

$$P(O_n|S_1, ..., S_n) \cdot P(S_n|S_{n-1}) = P(A_n, D_n|S_1, ..., S_n) \cdot P(S_n|S_{n-1})$$
$$= \{P(A_n|S_1, ..., S_n) \cdot P(D_n|S_1, ..., S_n)\} \cdot P(S_n|S_{n-1}) \quad (1.6)$$
$$= \{P(A_n|S_n) \cdot P(D_n|S_{n-1}, S_n)\} \cdot P(S_n|S_{n-1})$$
$$= P(A_n|S_n) \cdot \frac{P(D_n, S_{n-1}, S_n)}{P(S_{n-1}, S_n)} \cdot \frac{P(S_{n-1}, S_n)}{P(S_{n-1})} \quad (1.7)$$
$$= P(A_n|S_n) \cdot \frac{P(D_n, S_{n-1}, S_n)}{P(S_{n-1}, D_n)} \cdot \frac{P(S_{n-1}, D_n)}{P(S_{n-1})}$$
$$= P(A_n|S_n) \cdot P(S_n|S_{n-1}, D_n) \cdot P(D_n|S_{n-1}) \quad (1.8)$$
$$\propto P(A_n|S_n) \cdot P(S_n|S_{n-1}, D_n) \cdot P(E_n|S_{n-1})$$

Equation 1.6 is derived by the assumption that appearance (A) and dynamics (D) are independent given scene states (S). Equation 1.7 is derived by the first-order Markov assumption for appearance (A) and the second-order Markov assumption for dynamics (D). Equations 1.7 and Equation 1.8 is derived by using Bayes' Theorem.

The following shows how dynamic context can be mathematically modeled by motion context and event context.

$$P(S_n|S_{n-1}, D_n) = P(S_n|S_{n-1}, M_n, E_n)$$

$$= \frac{P(S_n, S_{n-1}, M_n, E_n)}{P(S_{n-1}, M_n, E_n)} = \frac{P(M_n, E_n|S_n, S_{n-1}) \cdot P(S_n, S_{n-1})}{P(M_n, E_n|S_{n-1}) \cdot P(S_{n-1})} \tag{1.9}$$

$$\simeq \frac{P(M_n|S_n, S_{n-1}) \cdot P(E_n|S_n, S_{n-1}) \cdot P(S_n, S_{n-1})}{P(M_n|S_{n-1}) \cdot P(E_n|S_{n-1}) \cdot P(S_{n-1})} \tag{1.10}$$

$$= \frac{P(M_n, S_n, S_{n-1}) \cdot P(E_n, S_n, S_{n-1}) \cdot P(S_n, S_{n-1})^3}{P(M_n, S_{n-1}) \cdot P(E_n, S_{n-1}) \cdot P(S_{n-1})^3} \tag{1.11}$$

$$= P(S_n|M_n, S_{n-1}) \cdot P(S_n|E_n, S_{n-1}) P(S_n|S_{n-1})^3 \tag{1.12}$$

$$\propto P(S_n|M_n, S_{n-1}) \cdot P(S_n|E_n, S_{n-1})$$

Equations 1.9, 1.11, and 1.12 can be derived by using Bayes' Theorem. Equation 1.10 can be derived by the conditional independence assumption of motion (M) and event (E).

Bibliography

[1] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43:16:1–16:43, April 2011.

[2] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *Journal of Logic and Computation*, 4(5):531–579, 1994.

[3] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of the International Joint Conference on Arti cial Intelligence (IJCAI)*, 1977.

[4] D. M. Bertozzi, A. Broggi, A. Fascioli, and S. Nichele. Stereo vision-based vehicle detection. In *IEEE Intelligent Transportation Symposium*, 2000.

[5] M. Betke, E. Haritaoglu, and L. S. Davis. Multiple vehicle detection and tracking in hard real-time. In *IEEE Intelligent Vehicles Symposium*, 1996.

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

[7] A. Bevilacqua and S. Vaccari. Real time detection of stopped vehicles in traffic scenes. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[8] S. Boragno, B. Boghossian, J. Black, D. Makris, and S. Velastin. A dsp-based system for the detection of vehicles parked in prohibited areas. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[9] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[10] R. Chaudhry, A. Ravichandran, G Hager, and R Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[11] C.-C. Chen and J. K. Aggarwal. Recognizing human action from a far field of view. In *IEEE Workshop on Motion and Video Computing (WMVC)*, 2009.

[12] C.-C. Chen and J. K. Aggarwal. Modeling human activities as speech. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[13] M. L. Comer and E. J. Delp. Morphological operations for color image processing. *Journal of Electronic Imaging*, 8(3):279–289, 1999.

[14] Ingemar J. Cox and Sunita L. 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*, 18(2):138–150, 1996.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[16] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *International Conference on Computer Vision*, 2003.

[17] R. Feris, J. Petterson, B. Siddiquie, L. Brown, and S. Pankanti. Large-scale vehicle detection in challenging urban surveillance environments. In *Proceedings of IEEE Workshop on Applications of Computer Vision (WACV)*, 2011.

[18] P. J. GREEN. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[19] S. Guler, J. A. Silverstein, and I. H. Pushee. Stationary objects in multiple object tracking. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[20] S. Gupte, O. Masoud, , and N. Papanikolopoulos. Vision-based vehicle classification. In *IEEE Conference on Intelligent Transportation Systems*, 2000.

[21] S. Gupte, O. Masoud, R. Martin, and N. Papanikolopoulos. Detection and classification of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):37–47, 2002.

[22] Y. Ivanov, C. Stauffer, A. Bobick, and W. E. L. Grimson. Video surveillance of interactions. *Visual Surveillance, IEEE Workshop on*, 0:82, 1999.

[23] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852 –872, aug 2000.

[24] S. W. Joo and R. Chellappa. Attribute grammar-based event recognition and anomaly detection. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on*, 2006.

[25] G. Jun, J. K. Aggarwal, and M. Gokmen. Tracking and segmentation of highway vehicles in cluttered and crowded scenes. In *Proceedings of IEEE Workshop on Applications of Computer Vision (WACV)*, 2008.

[26] Z. Kim and J. Malik. Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *the Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003.

[27] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[28] J. T. Lee, M. S. Ryoo, and J. K. Aggarwal. View independent recognition of human-vehicle interactions using 3-d models. In *IEEE Workshop on Motion and Video Computing (WMVC)*, 2009.

[29] J. T. Lee, M. S. Ryoo, M. Riley, and J. K. Aggarwal. Real-time detection of illegally parked vehicles using 1-D transformation. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[30] J. T. Lee, M. S. Ryoo, M. Riley, and J. K. Aggarwal. Real-time illegal parking detection in outdoor environments using 1-D transformation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(7):1014–1024, July 2009.

[31] F. Lv and R. Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[32] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[33] D. J. Moore, I. A. Essa, and M.H. Hayes. Exploiting human actions and object context for recognition tasks. In *the Proceedings of the IEEE*

International Conference on Computer Vision (ICCV), 1999.

[34] K. Morimoto. System for detecting and warning an illegally parked vehicle (patent style). In *U. S. Patent 5,343,237*, 1994.

[35] A. S. Ogale and Y. Aloimonos. Shape and the stereo correspondence problem. *International Journal of Computer Vision*, 65(3):147–162, 2005.

[36] A. S. Ogale and Y. Aloimonos. A roadmap to the integration of early visual modules. *International Journal of Computer Vision: Special Issue on Early Cognitive Vision*, 72(1):9–25, 2007.

[37] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Roy-Chowdhury, and M. Desai. A large-scale benchmark dataset for event recognition in surveillance video. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[38] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[39] S. Park and M. M Trivedi. Analysis and query of person-vehicle interactions in homography domain. In *VSSN '06: Proceedings of the 4th ACM*

international workshop on Video surveillance and sensor networks, pages 101–110, New York, NY, USA, 2006. ACM.

[40] F. Porikli. Detection of temporarily static regions by processing video at different frame rates. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[41] A. Rajagopalan and R. Chellappa. Vehicle detection and tracking in video. In *ICIP*, 2000.

[42] M. S. Ryoo and J. K. Aggarwal. Observe-and-explain: A new approach for multiple hypotheses tracking of humans and objects. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2008.

[43] M. S. Ryoo and J. K. Aggarwal. Recognition of high-level group activities based on activities of individual members. In *IEEE Workshop on Motion and Video Computing (WMVC)*, 2008.

[44] M. S. Ryoo and J. K. Aggarwal. Semantic representation and recognition of continued and recursive human activities. *International Journal of Computer Vision*, 82(1):1–24, 2009.

[45] M. S. Ryoo, C.-C. Chen, J. K. Aggarwal, and A. Roy-Chowdhury. An overview of contest on semantic description of human activities 2010. In *International Conference on Pattern Recognition Contests (ICPR)*, 2010.

[46] M. S. Ryoo, J. T. Lee, and J. K. Aggarwal. Video scene analysis of interactions between humans and vehicles using event context. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, Proceedings of the ACM International Conference on Image and Video Retrieval(CIVR), pages 462–469, 2010.

[47] X. Song and R. Nevatia. A model-based vehicle segmentation method for tracking. In *the Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[48] X. Song and R. Nevatia. Detection and tracking of moving vehicles in crowded scenes. In *IEEE Workshop on Motion and Video Computing (WMVC)*, volume 0, page 4, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[49] 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 (CVPR)*, volume 2, page 2246, Los Alamitos, CA, USA, 1999. IEEE Computer Society.

[50] Z. Sun, G. Bebis, and R. Miller. On-road vehicle detection: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):694–711, 2006.

[51] B. Tamersoy and J. K. Aggarwal. Robust vehicle detection for tracking in highway surveillance videos using unsupervised learning. In *IEEE*

Conference on Advanced Video and Signal Based Surveillance (AVSS), 2009.

[52] S. D. Tran and L. S. Davis. Event modeling and recognition using markov logic networks. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 610–623, 2008.

[53] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE T-CSVT*, 18(11):1473–1488, Nov. 2008.

[54] P. L. Venetianer, Z. Zhang, W. Yin, and A. J. Lipton. Stationary target detection using the objectvideo surveillance system. In *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007.

[55] T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. In *Journal of Machine Learning Research*, volume 5, pages 975–1005, 2004.

[56] J. Yang, Y. Wang, A. Sowmya, Z. Li, B. Zhang, and J. Xu. Feature fusion for vehicle detection and tracking with low-angle cameras. In *Proceedings of IEEE Workshop on Applications of Computer Vision (WACV)*, 2011.

[57] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4):13, 2006.

[58] T. Zhao, R. Nevatia, and B. Wu. Segmentation and tracking of multiple humans in crowded environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1198–1211, 2008.

[59] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2004.

[60] Z. Zivkovic and F. van der Heijden. Recursive unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):651–656, 2004.

Vita

Jong Taek Lee was born in Daejeon, South Korea on July 3, 1983, the son of Yong Soon Lee and Hyang Suk Park. He received his Bachelor of Science degree in Engineering from Korea Advanced Institute of Science and Technology. He applied to the University of Texas at Austin for enrollment in their Electrical and Computer Engineering program. He was accepted and started graduate studies in August, 2005. He received his Master of Science in Engineering from the University of Texas at Austin in August, 2007. He is currently a graduate research assistant in the Computer and Vision Research Center.

Permanent address: Sinsun Maeul APT. 210-1003, Gwanjeo 2-dong,
Seo-gu, Daejeon, Republic of KOREA 302-783

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.