Dynamic Object Tracking and 3-D Visualization from Big Visual Data

Kuan-Hui Lee

A dissertation

submitted in partial fulfillment of the
requirements for the degree of:

Doctor of Philosophy

University of Washington
2016

Reading Committee:
Jenq-Neng Hwang, Chair
Linda G. Shapiro
James W. Pitton

Program Authorized to Offer Degree:
Department of Electrical Engineering

University of Washington

Abstract

Dynamic Object Tracking and 3-D Visualization from Big Visual Data

Kuan-Hui Lee

Chairperson of the Supervisory Committee:
Professor Jenq-Neng Hwang
Department of Electrical Engineering

We propose an automatic system which dynamically tracks video objects (human and vehicle) and create their 3-D visualization from big visual data. Big visual data implies that all videos are collected from either static or mobile surveillance cameras. Our goal is to track a video object within such a surveillance camera network. To achieve this goal, several tracking scenarios must be carefully dealt with. In this work, we focus on tracking under a single static camera, tracking under a single moving camera, and tracking across multiple moving cameras.

In the case of tracking under a single static camera, our proposed work is mainly based on constrained multiple-kernel tracking framework. For human tracking, the system adopts a Kalman filter to predict and refine the tracking results. A pre-trained human detector is further applied to solve initial merging issues. For human tracking across multiple static cameras, a self-organized and scalable multiple-camera tracking system that tracks human across cameras with nonoverlapping views is proposed. For vehicle tracking, our proposed approach regards each patch of the 3-D vehicle model as a kernel, and track the kernels under certain constraints facilitated with the 3-D geometry of the vehicle model. Meanwhile, a kernel density estimator is designed to fit the 3-D

vehicle model during tracking. By elegant application of the constrained multiple-kernel (CMK) tracking facilitated with the 3-D vehicle model, the vehicles are able to be tracked efficiently and located precisely.

As for tracking under a single moving camera, we propose a robust moving platform based object tracking system, and apply to human tracking. Our work effectively integrates Visual Simultaneous Localization And Mapping, pedestrian detection, ground plane estimation, and kernel-based tracking techniques. The proposed system systematically detects the pedestrians from recorded video frames and tracks the pedestrians in the V-SLAM inferred 3-D space via a tracking-by-detection scheme. In order to efficiently associate the detected pedestrian frame-by-frame, we propose a novel tracking framework, combining the CMK tracking and the estimated 3-D (depth) information, to globally optimize the data association between consecutive frames. By taking advantage of the appearance model and 3-D information, the proposed system not only achieves high effectiveness but also handles efficiently occlusion in the tracking.

Based on the results of tracking under a single moving camera, we propose a new framework to track on-road pedestrians across multiple driving recorders. More specifically, we treat the problem as a multi-label classification task, determining whether a specific pedestrian belongs to one or several cameras' field of views by considering the association likelihood of the tracked pedestrians. The likelihood is calculated based on the pedestrians' motion cues and appearance features, which are necessarily transformed via brightness transfer functions obtained by some available spatially overlapping views for compensating for the diversity of the cameras. When a pedestrian is leaving a camera's field of view, the proposed framework predicts and interpolates the possible moving trajectories facilitated by an open map service which can provide routing information. Moreover, based on the GPS locations, we can also reconstruct a 3-D visualization on a 3-D virtual real-world environment, so as to show the dynamic scenes of the recorded videos.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to first express my sincere gratitude to my adviser, Prof. Jenq-Neng Hwang, for his guidance throughout my Ph.D. study. He is always supportive and encouraging to help me pursue the goals and overcome the difficulties in research. From him I have learned a lot about how a successful scholar tackles open problems and gives good presentations.

I am grateful to my committee members, Prof. Mark Ganter, Prof. Linda Shapiro, Prof. Ali Farhadi, Prof. James Pitton, and Prof. Zicheng Liu for the valuable suggestions they have given since my General Exam.

I am thankful to the alumni I have met at the Information Processing Lab (IPL), Chun-Te Chu, Shian-Ru Ke, Po-Han Wu, Meng-Che Chuang, Pei-An Lee, Youngdae Lee and Ruizhi Sun, for their help and discussions during my first couple years in this group.

I would like to thank the current colleagues at IPL: Xiang Chen, Younggun Lee, Jounsup Park, Renshu Gu, Arash Tarkhan, Zheng Tang, Qiuyu Chen, Tsung-Wei Huang, and Gaoang Wang. They make this research group an excellence environment to work. Best wishes to all of them on their graduate study and future career.

I also thank all the friends I have met in Seattle. You have not only given me support but also enriched my life of studying abroad in the beautiful Emerald City. I will remember the laughter every time we got together.

With deepest love, I would like to thank my wife, I-Jen Wang, for her selfless support in my whole PhD. career. I am lucky to have such a great backing on my adventure to PhD. Without her taking good care of my life, I cannot achieve the goal smoothly.

Last but not least, I am grateful to my dearest family: my father, my mother, my brother; and also my father in law and mother in law. For their endless love, support and encouragement for everything I do. They are the ones who make me strong and confident to conquer the hurdles set in front of me. This dissertation is dedicated to them.

Dedication

To my family

Chapter 1 – Introduction

1.1 Motivations and Objectives

Recently, the development of intelligent surveillance system has attracted much attention. Tracking of video objects is one of the major issues in video surveillance systems. In a traditional video surveillance system, video objects are tracked under a single static camera, or across multiple static cameras. By successfully tracking the objects, it is possible to collect their trajectories in videos for high level analytics. Therefore, researchers are motivated to develop effective object tracking systems, which are not only robust and fast but also able to collect the information efficiently and accurately.

The lessons learned from the 2013 Boston Marathon bombing indicate that intelligent surveillance systems become more and more important for crime investigation or even tragedy prevention. Most of the surveillance cameras are installed at fixed locations, which currently reduce the flexibility of camera views and may create monitoring blind-spots. Therefore, the idea of developing mobile surveillance is thus introduced. Fortunately, an emerging application of video analytics in autonomous/smart vehicles is the usage of the driving recorders (or dash-cameras), which are devices that record video in a vehicle to create a record of driving [30]. Currently, driving recorders are widely used in many applications of video analysis for Intelligent Transport Systems (ITS). A driving recorder, which can be regarded as a moving camera on the roads, gradually becomes a necessity in a vehicle. Besides recorded videos, a vehicle can also obtain other driving information such as location, time, and speed, by global positioning system (GPS) or other electronic devices. A driving recorder provides a new mechanism to extend applications for video analyses.

Inspired by the growing usage of driving recorders and advanced wireless infrastructure, we can soon expect a mobile surveillance platform with a cloud server

being connected to all devices via a wireless wide area network (WWAN), such as 3G, WiMAX, or LTE. Based on the collected data, the cloud server can thus systematically perform video analytics, such as tracking on-road pedestrians or vehicles, to better monitor the on-road situation dynamically and cooperatively share the on-road information. Such a mobile camera platform would then be combined with a traditional surveillance system to achieve a larger and wider surveillance camera network: a platform collecting large amounts of videos, i.e., *big visual data*. As shown in Figure 1.1, three vehicles equipped with driving recorders can upload the surveillance videos to a server via a wireless network, while two static surveillance videos are linked to the server through a wired network.

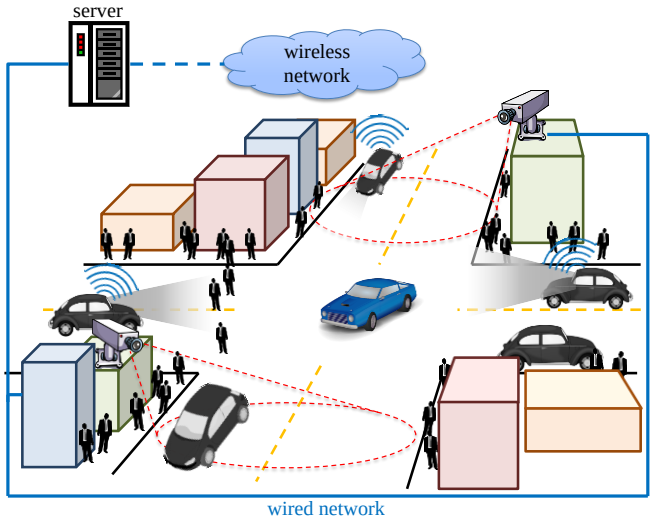


Figure 1.1: An illustration of big visual data

In our work, the goal is to track video objects, and further reconstruct scenes recorded from the collected surveillance videos. More specifically, the objects are walking pedestrians and moving vehicles recorded by the surveillance cameras. In other words, we intend to develop a human/vehicle tracking system, which operates not only under the static surveillance cameras, but also across different moving surveillance cameras. Finally, with the tracking results, the scene recorded by the surveillance cameras can be dynamically reconstructed.

1.2 Video Object Tracking under Different Scenarios

In general, there are several different scenarios of video object tracking, including: tracking under a static camera, tracking under a moving camera, tracking across multiple static cameras, and tracking across multiple moving cameras. These scenarios are discussed as follows.

1) Tracking under a single static camera: A surveillance camera is fixed at a location, and with a fixed camera's view, in the sense that the field of view (FOV) of the camera is limited. When an object enters into the FOV, it can be easily extracted and tracked by using its features, such as colors, textures, etc. This scenario is a traditional topic of a video surveillance system.
2) Tracking under a single moving camera: A moving camera such as a driving recorder or dash-camera is set on a vehicle, shooting in the direction in which the vehicle moves forward. Thus, the FOV is the view in front of the vehicle, and dynamically changes when the vehicle moves. Thus, the features of an object recorded by a driving recorder are still consistent, and the object can be detected and tracked until the object exits the FOV.
3) Tracking across multiple static cameras: Multiple surveillance cameras are fixed at different locations, with different camera's FOVs. To track an object across different cameras, we have to associate one tracked object from one single camera with that from another single camera. Since the features of the same object extracted from different cameras are inconsistent, the problem becomes how to normalize the extracted the features.
4) Tracking across multiple moving cameras: Multiple moving cameras can move around and record videos simultaneously. In summary, there are two basic cases of this tracking problem: object in overlapping cameras' FOVs, which implies that an object simultaneously appear in two or more different cameras views; object in non-overlapping cameras' FOVs, which means that an object enters in a camera's FOV

then exits, and later enters into other cameras' FOV(s). Generally, a static camera can be regarded as a moving camera which always stays in a location, thus the scenario 3 is treated as a special case of the scenario 4.

By combining all the tracking scenarios, the overall system can automatically and possibly track each object within the areas which are covered by the surveillance camera network. In this dissertation, we develop static camera tracking, including human/vehicle tracking under a single static camera and human tracking across multiple static cameras. Our work also focuses on moving camera tracking, including human tracking under a single moving camera and human tracking across multiple moving cameras.

1.3 Contributions

The contributions of this dissertation are summarized as follows.

- For human tracking in a single static camera, we adopt the Constrained Multiple-Kernel (CMK) tracking framework and use a pre-trained human detector to solve the initial merging issue.
- For human tracking across multiple static cameras, we propose a framework which utilizes information from Google Maps to facilitate the larger scalability of the out-door camera network.
- For vehicle tracking under a single static camera, a novel model-based vehicle localization approach is proposed. The proposed approach efficiently locates and tracks the vehicles by combining the CMK tracking with a 3-D deformable vehicle model.
- For tracking under a single moving camera, we propose a robust moving platform based human tracking system, which effectively integrates Visual Simultaneous Localization And Mapping (V-SLAM), human detection, ground plane estimation, and kernel-based tracking techniques.
- A novel human tracking framework combines CMK tracking and the estimated 3-D information (depth), to globally optimize the data association between consecutive

frames.

- For tracking across multiple moving cameras, a new framework to track on-road pedestrians across multiple driving recorders is proposed. The proposed approach predicts and interpolates its possible moving trajectories facilitated by an open map service which can provide routing information.
- The tracking problem across multiple moving cameras is treated as a multi-label classification task, determining whether a specific pedestrian belongs to one or several cameras' FOVs by considering association likelihood of the tracked pedestrians.
- Based on GPS locations of moving cameras, a 3-D visualization on a 3-D virtual real-world environment can be reconstructed, so as to show the dynamic scenes of the recorded videos.

1.4 Dissertation Organization

The rest of this dissertation is organized as follows.

Chapter 2: related work on object tracking under different scenarios is reviewed: 1) tracking under a single static camera, 2) tracking under a single moving camera, and 3) tracking across multiple static/moving cameras. Finally, the related work of 3-D visualization is briefly discussed.

Chapter 3: our proposed video object tracking approach under a single static camera is described and is applied to human and vehicle tracking. The basic concept of CMK tracking is described in Section 3.2. In Section 3.3, we develop a CMK-based human tracking which uses a pre-trained human detector to solve the initial merging issues. Section 3.4 depicts the proposed vehicle tracking approach, which integrates a 3-D generic vehicle model with the CMK tracking framework. Experimental results have shown favorable performance of the proposed approach in several scenarios, which efficiently tracks vehicles while maintaining knowledge of the 3-D geometry of the tracked vehicles.

Chapter 4: the proposed approach to track humans on a moving platform, effectively integrating V-SLAM, human detection, ground plane estimation, and kernel-based tracking techniques. In Section 4.1, the overview of the proposed system, including adopted algorithms, is briefly described. Section 4.2 depicts the proposed human tracking, which combines CMK tracking with 3-D information to formulate the association of the detected targets. Experimental results in Section 4.3 show favorable performance of the proposed system which efficiently tracks humans in a camera equipped on a ground moving platform such as a dash-camera or an unmanned ground vehicle.

Chapter 5: a new framework for tracking on-road pedestrians across multiple driving recorders is proposed. In Section 5.2, the overview of the proposed framework is provided. Section 5.3 depicts the methodologies used in the proposed framework. In Section 5.4, pedestrian association likelihood based on appearance features is specifically discussed. The experimental results are shown in Section 5.5, showing the robustness and effectiveness of the proposed framework in tracking pedestrians across several recorded driving videos and a 3-D visualization on a 3-D virtual real-world environment.

Chapter 6: a conclusion of this dissertation is given by summarizing the main contributions and discussing some extensions to this work that lead to potential research topics in the future.

Chapter 2 – Background and Related Work

2.1 Tracking under a Single Static Camera

Video object tracking under a single static camera was has been developed over the past two decades. Several approaches have been proposed to deal with the tracking problems. The following briefly shows the three major categories of tracking methods [1]:

1) Point tracking: The target is expressed as a point in the frame, and the previous target state is utilized to make the association between targets and points. The Kalman filter is one of the most well known trackers in this category.
2) Kernel tracking: Targets are tracked by computing the motion of the kernels which represent the appearance or shape of the targets. Mean shift tracker is a kind of kernel tracking.
3) Silhouette tracking: Given the target model, the target is tracked by estimating the region in each frame, for instance, by contour matching and shape matching.

Among the object tracking algorithms, kernel-based object tracking has recently gained more popularity for better and more robust tracking performance due to its fast convergence speed and relatively low computation. The basic concept is to maximize the similarity between the target's and the candidate's appearance models (i.e., color histogram), which are built by spatially masking the object with a kernel function [2]. Based on kernel-based tracking framework, many approaches are proposed and applied to human tracking [3]–[8].

The mean-shift method, one of the most popular kernel-based tracking methods, was applied to tracking problems to find the most similar location around the local neighborhood area [2] [3]. Based on this framework, many methods have been proposed to improve the tracking performance. Collins [4] used the difference of Gaussian and Lindeberg's theory to track the objects through the scale space. Yilmaz's approach [5] employed asymmetric kernels that can adaptively change the scale and orientation to

track the target. Furthermore, in order to better represent the tracked video object, multiple kernels have also been adopted in recent years. In [6], the video object represented by multiple kernels, which denoted various body parts, was tracked by using a two-step approach. Global movement and local movement were alternatively applied to locate the target. Fan *et al*. [7] linked the multiple collaborative kernels by using predefined constraints. In their approach, tracking was formulated as solving a least square problem. Chu *et al.* [8] generalized constrained multiple-kernel (CMK) tracking by adaptively adjusting kernels' weights according to their similarity, so as to improve the reliability when occlusion happens.

As for vehicle tracking, the approaches using surveillance cameras are roughly divided into two categories [9]: 2-D based and 3-D based approaches. In 2-D based approaches, 2-D motions and trajectories are commonly used in the Kalman-filtering or particle-filtering frameworks, mainly based on 2-D geometry features, such as edges, lines, and contours [10]–[13]. Hsieh *et al*. [14] used area, length and a set of rules, and a linearity feature to measure the vehicle silhouette roughly. Alternatively, some approaches [15]–[17] include color histograms as tracking features, since they are not only invariant to vehicle rotations and translations but also robust to occlusions and noise. The 3-D based approaches can be roughly categorized into two types [18]: feature based and intensity based. The feature based approaches utilize 2-D geometry features to evaluate the constructed 3-D models. The approaches in [19] [20] track vehicles in an extended Kalman- filtering framework with a 3-D model built by edges and corresponding features. In [21], the approach evaluates the distance between extracted edge points and model projection. Liebelt *et al.* [22] extend the idea used in [48] to detect the 3-D poses of vehicles from 2-D contours. Approaches that evaluate the similarity of the projected contours have also been proposed in [23], [24]. Since 2-D feature extraction is sensitive to image noise and occlusion, on the other hand, the intensity-based approaches evaluate image intensities or gradients in terms of projecting the 3-D models onto the images. Ferryman *et al*. [25] present a deformable model with 29 parameters,

combined with a PCA framework, to perform the vehicle tracking. Tan *et al*. [26] [27] estimate orientation of a vehicle by gradient vectors and evaluate the poses by intensity values. Zhang *et al*. [28] proposed a deformable-model based tracking approach to dynamically build 3-D vehicle models and locate vehicles in an iterative framework. Zheng *et al*. [18] proposed an efficient vehicle pose estimation scheme, to reduce the search range of the tracked poses. The approach in [29] develops a vehicle tracking system based on a highly complicated 3-D deformable model, which not only performs vehicle tracking but also determines vehicle geometry. However, these intensity-based approaches focus solely on the shape fitting, without taking color information into account, to perform the tracking. Furthermore, most of the proposed pose estimation schemes are time consuming due to the optimization of the parameters.

2.2 Tracking under a Single Moving Camera

Video object tracking under a single moving camera is a notoriously difficult task because of the combined effects of egomotion, blur, and rapidly changing lighting conditions. The introduction of a moving camera invalidates many elegant techniques used with a static camera, such as background subtraction and a constant ground plane assumption. Therefore, the challenge in this problem is to detect the objects successfully in a moving camera, and then apply the tracking techniques to them.

In general, object detection in a moving camera follows two basic steps [30] [31]: *candidate generation* (also referred to as foreground segmentation) and *object classification*. Candidate generation first extracts blobs of interest from the image, avoiding as many background regions as possible. Then, object classification classifies the extracted blobs as human/vehicle or nonhuman/nonvehicle objects. The approaches for candidate generation can be classified into two kinds: 1) image-based and 2) motion-based. The image-based approaches mainly rely on the color, intensity, edges, and gradient orientation of pixels [32]–[34]. The motion-based approaches utilize interframe motion and optical flow [35] [36]. Especially in vehicle detection, motion can provide

strong information, since approaching vehicles at an opposite direction produce a diverging flow, which can be quantitatively distinguished from the flow caused by the car ego-motion [37] [38]. The approaches to object classification are purely 2D, and can be broadly divided into two: 1) template-based and 2) appearance-based. The template-based approaches use predefined patterns of the human/vehicle class and perform correlation between the image and the template. The template is a human body in the case of human detection [39]–[41] and the rear/front view of a vehicle in the case of vehicle detection [42]. Appearance methods ([43], [44]–[48] for vehicles) define a space of image features (descriptors), and a classifier is pre-trained by positive examples (human/vehicle) and negative examples (nonhuman/nonvehicle) using various learning algorithms, such as neural networks, support vector machines (SVM), and AdaBoost.

After object detection, a tracking framework is applied to the detected objects. The work of human and vehicle tracking in a moving camera are discussed as follows. 1) Human: Kalman filters [41] [49] and particle filters [50]–[52] are also widely used in tracking. Some methods adopt Multi-Hypothesis Tracking (MHT) [53],[54] or Joint Probabilistic Data Association Filters (JPDAFs) [55] to optimize detected target association by considering information over several time steps. Ess *et al*. [56] perform multibody tracking by combining the ISM detector and the stereo-odometry-based tracker. Adnriluka *et al*. [57] detect people using a part-based detector [46], and then use a Gaussian process latent variable model to compute the temporal consistency of detections over time. 2) Vehicle: the tracking strategies are similar to the human tracking, except the interested features and templates/models. In [33], vehicles are tracked using multiple cues such as intensity and edge information. The system proposed in [58] combines the pre-trained deformable object model with particle filter tracking. In [59], the authors develop vehicle motion model and background model to detect and separate the vehicles from the background. 3) Human and vehicle: Leibe *et al*. [60] [61] proposed the use of a color model and what they refer to as the *event cone*, i.e., the time-space volume in which the trajectory of a tracked object is sought in 3-D space. Andreas *et al*. [62] proposed a

probabilistic generative model to understand the 3D scene layout as well as the location and orientation of objects in the scene. In addition to the frame-by-frame based target association techniques mentioned above, recently, more and more methods tend to find globally optimal solutions across the entire sequence.

Some approaches formulate the tracking problem as a min-cost flow network problem, and others use iterative hierarchical methods to link tracklets. Zhang *et al*. [63] map the maximum-a-posteriori (MAP) data association problem into a cost-flow network with a non-overlap constraint on trajectories. In [64], the authors use a cost function with the objects' birth and death states, and show that the global solution can be obtained with a greedy algorithm. In [65], Wu *et al*. propose a coupling formulation to avoid the problem of error propagation in tracking-by-detection schemes, and further solve the partial/complete occlusions problem. The approach in [66] incorporates constraints of piecewise constant-velocity path smoothness based on the flow network framework. Li *et al*. [67] propose to progressively associate tracklets by using a ranking and classification algorithm (HybridBoost) to learn cost parameters for the tracklet function. Yang *et al*. [68] create a conditional random field from the set of tracklets to remove the assumption of independence between the tracked objects. These approaches globally optimize the trajectories of all objects, instead of locally optimizing for each object. However, the performance is highly dependent on the reliable detection. If missed detection or long-time occlusion happens, the performance deteriorates significantly.

Alternatively, several approaches based on the Structure-from-Motion (SfM) framework [69] have been developed. The SfM framework was originally used for static 3-D scene reconstruction in multi-view stereo applications. It can also be applied to moving cameras, combined with Visual Simultaneous Localization and Mapping (V-SLAM), to calibrate and to localize the 3-D positions of the camera and static features [70]–[72]. Based on SfM, many researchers develop approaches to detect, track, and reconstruct moving objects within the static background, so-called dynamic scene reconstruction. In [60] and [61], the authors reconstruct not only the static background

but also humans and vehicles, and track them by the tracking-by-detection scheme. Kundu *et al*. [38] presents a real-time, incremental visual SLAM system that allows choosing between full 3-D reconstruction or simply tracking of the moving objects. The approach in [73] estimates the ground plane by using sparse features, dense inter-frame stereo and object detection based on a real-time monocular SfM framework. The advantage of the SfM based approaches is to locate the objects in 3-D space, so as to deal with the occlusion problem during tracking. For this reason, we also choose to reconstruct the 3-D locations of the objects based upon the SfM framework in our proposed scheme.

2.3 Tracking across Multiple Static/Moving Cameras

The challenge of object tracking across multiple cameras is to understand the correlation and association of a moving object among multiple cameras, and then correctly identify and track the object from one camera to another (successful label handoff). Many papers in the literature [74]–[82] treat the problem as the person re-identification task, which commonly learns sets of descriptors and/or the metric functions to compute the similarities between the people using a pre-collected dataset. However, the tracking problem across multiple moving cameras is beyond the re-identification task. Since the mapping between two cameras keeps on changing when the cars move, such conditions will prevent the direct use of most person re-identification approaches.

Tracking across multiple static cameras with overlapping views has been well investigated [83]–[87], by exploring the spatial-temporal homography and association of color information between the cameras. In [88] and [89], the authors utilize a probabilistic occupancy map to deal with human tracking under multiple cameras with overlapping FOVs. As for non-overlapping scenarios, time-space cues and appearance features are normally used for the label handoff. The problem then becomes how to learn the time-space and appearance relationships between cameras effectively. In addition to the supervised [83],[90] and semi-supervised [91] learning schemes, there are several

approaches that effectively use unsupervised learning schemes. In [92], the authors came up with an approach for inferring the time-space relationship by measuring statistical dependence between observations in different cameras. In Gilbert's work [93], color information is used in building the transition time distribution. Matei *et al*. [94] proposed a method using joint kinematic and image appearance information for vehicle tracking. In [95], the authors utilize game theory to solve the label handoff issue in a collaborative PTZ camera network. In [10], a method was proposed to discover and remove the "weak link" which was defined as the redundant link between two cameras that are not directly connected. Recently, Chu and Hwang [97] propose a systematically estimated camera link model, in terms of solving a permutation matrix, by considering the transition time distribution along with and the holistic and regional human body color/texture appearance information.

To the best of our knowledge, there has been very little literature discussing the issue of tracking across multiple moving cameras. Zou and Tan [98] proposed a collaborative V-SLAM based on the SfM framework for multiple cameras. These cameras move independently and cooperate with each other to reconstruct the 3D trajectories of moving objects in a highly dynamic environment. However, Zou's work focuses on how to collaboratively reconstruct the 3-D objects from the multi-view videos; tracking problems such as trajectory prediction and association of the objects are not considered.

2.4 3-D Visualization

Earth can be visualized on the Internet with the growth of online Aerial Earth Maps (AEMs) services, such as Google Earth, Microsoft Virtual Earth, etc. We can visually browse through cities across the globe from our desktops or mobile devices and see 3D models of buildings, street views and topologies. However, such virtual environments only display static scenes, while dynamic scenes catch more people's attentions. Accordingly, several systems have been developed using static cameras. Sebe *et al*. [99] built a visualization system in which GPS devices, orientation sensors and video cameras

contained in a tracking backpack carried by a pedestrian were used to create augmented virtual environments. Kim *et al*. [100] analyzed videos of cities with pedestrians and cars under differing conditions, and then created augmented AEMs with live and dynamic information. In [101], the authors develop a distributed system for sensing, interpreting and visualizing the real-time dynamics of urban life within the 3D context of a city. Their work reconstructs a dynamic scene from a static surveillance videos or cameras; in contrast, our proposed work creates 3-D visualization from both static and moving cameras.

Chapter 3 – Static Camera Tracking

3.1 Overview

In this chapter, our proposed static tracking approaches are described, including human tracking under a single camera, human tracking across multiple cameras, and vehicle tracking under a single camera.

For human tracking under a single static camera, the proposed approach adopts Constrained Multiple-Kernel (CMK) tracking [8], which uses a Kalman filter to predict and refine the tracking results. A pre-trained human detector is additionally used in the CMK tracking system to solve the initial merging issues. For human tracking across multiple static cameras, a self-organized and scalable multiple-camera tracking system that tracks humans across cameras with nonoverlapping views is proposed. The system is mainly based on the work in [97], and further utilizes information from Google Maps, such as routing information and transition time. Thanks to the unsupervised pairwise learning and tracking in our system, the proposed system is able to be scaled up efficiently when more cameras are added into the network [86].

For vehicle tracking under a single static camera, CMK tracking is extended from 2-D space to 3-D space. The tracking system starts with a 3-D generic vehicle model, which can be deformed to different types of vehicles. Hence, based on the CMK tracking framework, our proposed approach regards each patch of the 3-D vehicle model as a kernel, and tracks the kernels under certain constraints facilitated by the 3-D geometry of the vehicle model. Meanwhile, a kernel density estimator is designed to fit the 3-D vehicle model during tracking. By elegant application of the constrained multiple-kernel tracking facilitated by the 3-D vehicle model, the vehicles are able to be tracked efficiently and located precisely [102]−[104].

The concept of the CMK tracking is introduced in Section 3.2, followed by the CMK based human tracking (Section 3.3) and vehicle tracking (Section 3.4).

3.2 Constrained Multiple-Kernel (CMK) Tracking

The objective of CMK tracking is to retrieve a candidate object, which can be described as multiple kernels with pre-specified constraints among these kernels, so that the maximum similarity between the tracked object and the candidate model can be reached. For N_κ kernels, the total cost function $J(\mathbf{x})$ is defined to be the sum of the N_κ individual cost functions $J_\kappa(\mathbf{x})$, which is designed to be inversely proportional to the similarity,

$$J(\mathbf{x}) = \sum_{\kappa=1}^{N_\kappa} J_\kappa(\mathbf{x}), \quad J_\kappa(\mathbf{x}) \propto 1/\, simi_\kappa(\mathbf{x}) \,, \tag{1}$$

where $simi_\kappa(\mathbf{x})$ is the similarity function at the location $\mathbf{x}$ in the state space domain.

Moreover, the constraint functions C(x) = 0 need to be considered to maintain the relative locations of the kernels. The constraint functions confine the kernels based on their spatial inter-relationships. Thus, the problem could be further formulated by

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} J(\mathbf{x}), \text{ subject to } \mathbf{C}(\mathbf{x}) = \mathbf{0}. \tag{2}$$

3.2.a Projected Gradient-based Multiple-Kernel Tracking

In order to gradually decrease the total cost function and maintain the constraints satisfied during the state search, the movement vector $\delta_{\mathbf{x}}$, i.e., the gradient vector of the $J(\mathbf{x})$, is needed for the projected gradient method [8] to iteratively solve the constrained optimization problem. The basic idea is to project the gradient vector onto two orthogonal spaces: one is related to decreasing the cost function, and the other corresponds to satisfying the constraints. (i.e., $\mathbf{C}(\mathbf{x}) = \mathbf{0}$):

$$\begin{aligned} \delta_{\mathbf{x}} &= \alpha(-\mathbf{I} + \mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^T\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^T)\mathbf{J}_{\mathbf{x}} + (-\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^T\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}(\mathbf{x})) \\ &= \quad \delta_{\mathbf{x}}^A \quad + \quad \delta_{\mathbf{x}}^B \quad , \end{aligned} \tag{3}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector; $\mathbf{C}(\mathbf{x}) = \begin{bmatrix} c_1(\mathbf{x}) \\ \vdots \\ c_m(\mathbf{x}) \end{bmatrix}$ is the matrix including m constraint functions, and $c_i(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is the *i*-th constraint function; $\mathbf{C}_\mathbf{x} \in \mathbb{R}^{n \times m}$ is the gradient matrix of constraint functions with respect to **x**; $\mathbf{J}_\mathbf{x}$ is the gradient vector of the total cost function with respect to **x**, and $\alpha > 0$ is the step size.

As proved in [8], $\delta_\mathbf{x}^A$ and $\delta_\mathbf{x}^B$ are orthogonal to each other. Moving along $\delta_\mathbf{x}^A$ decreases the total cost function $J(\mathbf{x})$ while keeping the same values of $\mathbf{C}(\mathbf{x})$. On the other hand, moving along the $\delta_\mathbf{x}^B$ can lower the absolute values of $\mathbf{C}(\mathbf{x})$. Owing to these characteristics, the optimal solution can be achieved in an iterative scheme. The iteration is stopped when either the cost function and the absolute values of constraint functions are both lower than some given thresholds ε_j and ε_c respectively, or the iteration count is larger than a threshold *T* (Algorithm 1 in [8]).

3.2.b Adaptive Cost Function

When occlusion happens, not all the kernels can be used for matching. To solve the issue, each kernel is assigned an adaptively adjustable weight w_κ in Eq. (1):

$$J(\mathbf{x}) = \sum_{\kappa=1}^{N_k} w_\kappa \cdot J_\kappa(\mathbf{x}). \tag{4}$$

Thus, the movement vector in Eq. (3) is modified to be:

$$\delta_\mathbf{x} = \alpha(-\mathbf{I} + \mathbf{C}_\mathbf{x}(\mathbf{C}_\mathbf{x}^T\mathbf{C}_\mathbf{x})^{-1}\mathbf{C}_\mathbf{x}^T)\mathbf{W}\mathbf{J}_\mathbf{x} + (-\mathbf{C}_\mathbf{x}(\mathbf{C}_\mathbf{x}^T\mathbf{C}_\mathbf{x})^{-1}\mathbf{C}(\mathbf{x})), \tag{5}$$

where $\mathbf{W} = \begin{bmatrix} w_1\mathbf{I} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_{N_\kappa}\mathbf{I} \end{bmatrix}$ and $w_\kappa \propto simi_\kappa(\mathbf{x})$; $\mathbf{I}$ is an $\frac{n}{N_\kappa} \times \frac{n}{N_\kappa}$ identity matrix, *n* is the dimension of the state space.

The value w_κ corresponding to the *κ-th* kernel is adaptively updated based on the similarity $simi_\kappa$ and is normalized to make the sum of the w_κ's equal to N_κ. The idea of the adaptation is that the movement vector will have higher confidence for a kernel with

higher similarity than for one with lower confidence. When a kernel is ineffective due to occlusion, the movement vector will adaptively count on the other effective kernels (i.e., kernels with higher similarity) [8].

3.3 CMK Based Human Tracking

3.3.a Single Static Camera

CMK based Human Tracking under a single static camera is well developed in [8]. In general, when the target is occluded or it is similar to the background, an error may occur. This can be avoided by applying multiple kernels just as shown in Figure 3.1, where the kernel is expressed as the rectangle. If occlusion happens, the tracking result can be severely affected since kernel 1 loses a large amount of information, as shown in Figure 3.1(a). However, once the kernel 2 is added in Figure 3.1(b), although kernel 1 is nearly non-observable, the well-observable kernel 2 is used to recover the missing information resulting from the occlusion after introducing some constraints which can link the two kernels.

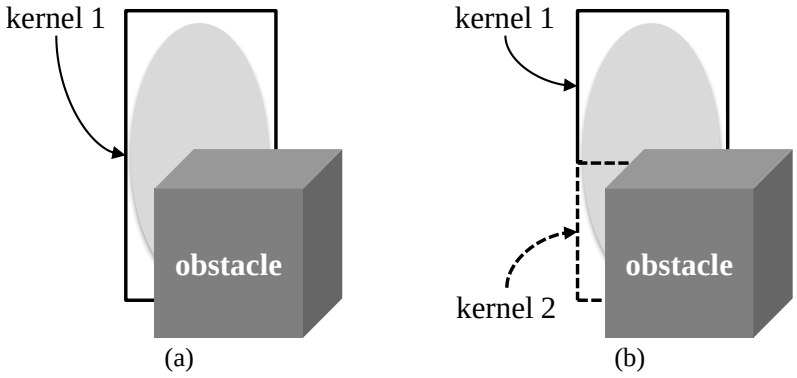
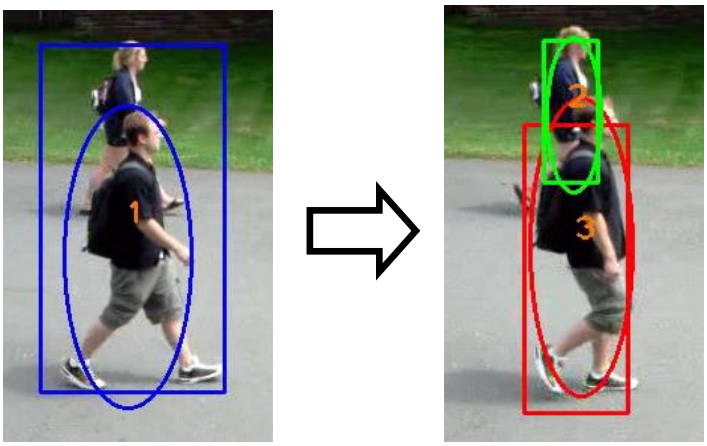


Figure 3.1: Rectangles represent kernels, and ellipse stands for the target object. (a) Single kernel with occlusion. (b) Two kernels with occlusion.

When humans go into the view of the camera, there is the chance that two or more humans are merging together. By doing background subtraction, only one object can be extracted, as shown in Figure 3.2(a). To solve this issue, our work adopts the human

detector C^4 [43] when the size of the extracted object is greater than a pre-defined threshold. The advantage of using background segmentation before human detection is to locate and shrink the search region of humans, so as to reduce the computation. Figure 3.2(b) shows the tracking result after detecting humans within the extracted object in the Figure 3.2(a). As the result shows, two humans, which are regarded as on object originally, are detected and separated properly.



(a) (b)

Figure 3.2: Tracking results when the initial merging issue occurs (a) without human detector, and (b) with human detector.

3.3.b Multiple Static Cameras

A framework that tracks human across the cameras with nonoverlapping views is proposed in [97]. The framework explores the relationship between a particular pair of *entry/exit zones* in two directly-connected cameras; we call it a *camera link model*. Given a camera network consisting of multiple cameras, two pieces of information are required before the camera link model estimation can be performed. (i) The system needs to identify which pairs of cameras have link models between them, i.e., which pairs are directly connected. Wrong links or redundant links decrease the tracking performance easily, due to the increased searching range resulting in reduced recall rate and increased false positives, not to mention the exponentially increased computational complexity. (ii) To our observation, the link actually only connects two entry/exit zones in a pair of

directly-connected cameras; that is, if a person is traveling between two cameras, he/she will likely leave from one particular zone and enter into the other. Hence, the training data used in camera link model estimation (and the subsequent re-identification tracking) should only include the observations happening in these two specific zones in order to avoid too many outliers. Therefore, to identify which specific zones are linked together is another critical issue. Based on the framework in [97], we propose a systematic method that performs the camera link identification by incorporating the information from Google Maps and Google Street View.

Figure 3.3 shows the overview of our proposed system. First of all, the camera link identification, including link existence detection and connected zones identification, is performed based on the incorporation of Google Maps. After that, the system automatically estimate the camera link model based on the training data. Finally, the model is utilized for tracking objects across multiple cameras with nonoverlapping field of views.

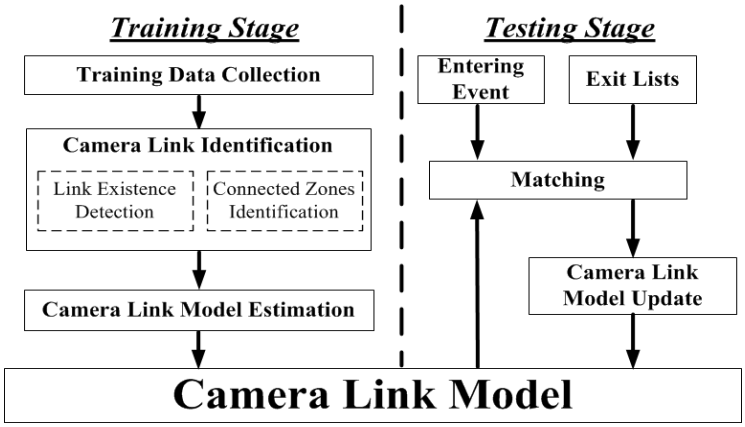


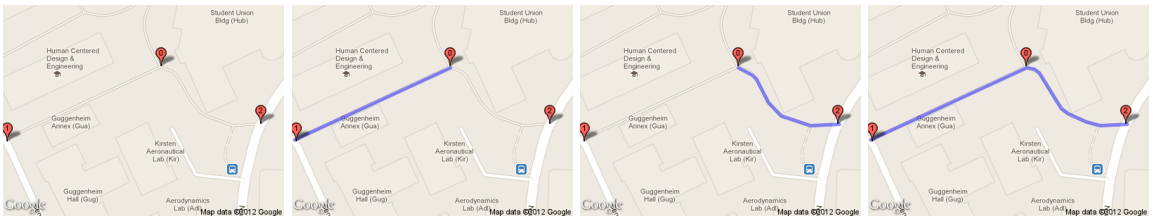
Figure 3.3: System Overview.

The camera link model between each pair of directly-connected cameras has been shown to be effective in tracking human across multiple cameras [96],[97]. Before the camera link model is estimated, the prior knowledge is the identification of which pairs of cameras within the camera network should possess a camera link model. In this section we introduce how our system detects the existence of links given the GPS locations of the

cameras. The camera link identification includes the following two modules: link existence detection and connected zones identification.

Link Existence Detection

Given the locations of the cameras, which are easily obtained when setting up the cameras in the environment, we are able to access the routing information provided by Google Maps. The routing information contains the possible routes between any two locations. If there exists one route that connects two cameras without passing by another camera, we recognize them as directly-connected, and there should be a link between them. If all the routes between two cameras pass by at least one other camera, we recognize the link should not exist between the two cameras. Figure 3.4 shows an example of the routing associated with three cameras denoted by C_0, C_1, and C_2, whose locations are shown in Figure 3.4(a). To our observation, in practice people tend to follow similar paths due to the presence of an available pathway, obstruction, and shortest route, so it is reasonable to utilize the estimated paths from Google Maps as the routing information. Figure 3.4(b), (c) and (d) show the shortest routes between each pair of cameras. Since the route between C_1 and C_2 passes C_0, the system only detects the links between C_0 and C_1 and C_0 and C_2.



(a) (b) (c) (d)

Figure 3.4: An example of the routing associated with camera 0, 1, and 2. (a) The locations of three cameras. The shortest route between (b) camera 0 and 1, (c) camera 0 and 2, (c) camera 1 and 2.

Connected Zones Identification

There may be several entry/exit zones within a camera's view. The link between two directly-connected cameras actually only connects one zone each in these two cameras. So far, we can only know the existence of the link from the link existence detection

without knowing the specific zones that are connected together. If we can know the connected zones, we only need to collect the training data, the exit and entry observations, from the associated zones so as to reduce large number of outliers in the training data during the estimation of the camera link model and also obtain better accuracy when tracking the objects.

First of all, all the zones within each camera are detected in an unsupervised manner by using a Gaussian Mixture Model (GMM) based on the collected entry/exit measurements [96]. Next, we match the camera's view with the panoramic images automatically retrieved from Google Street View to estimate the principal orientation of the camera. The scheme of the street view matching is described as follows: (i) given a GPS location, access the images from Google Street View with different viewing angles θ, pitches φ, and foveation f; (ii) perform feature point matching between the images and the camera's view; (iii) identify the image with the maximum number of matched points, and the corresponding viewing angle θ that gives the principal orientation of the camera. Figure 3.5 shows an illustration of the scheme, where the camera's view highly matches one of the panoramic images. Moreover, the direction of the route is provided by Google Maps according to the GPS locations. Therefore, given the principal orientations, the direction of the route, and the detected entry/exit zones, we can estimate the two zones that are connected together.

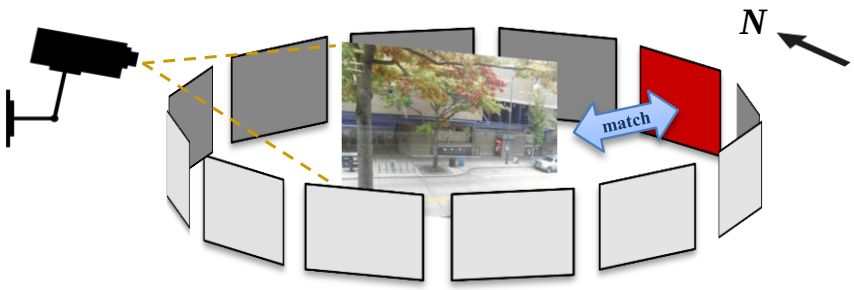
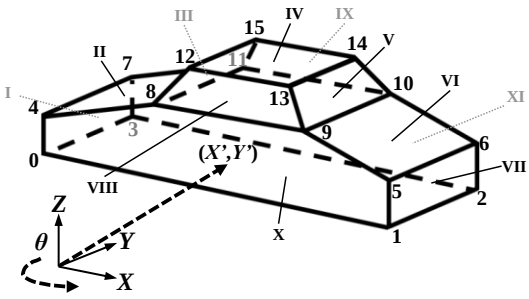


Figure 3.5: The estimation of principal orientation of the camera.

3.4 CMK Based Vehicle Tracking

3.4.a 3-D Vehicle Modeling

3-D vehicle modeling [28] generates an approximate 3-D vehicle model deformed from a 3-D generic model. The basic assumption is that the camera is static and well calibrated (i.e., the 3 × 4 projective matrix $\widehat{\mathbf{P}}$ is known), so as to facilitate the projection from the 3-D model to 2-D geometric primitives. The 3-D deformable model with 16 vertices and 23 arcs, as shown in Figure 3.6(a) [28], is defined by 12 shape parameters, such as vehicle length (L), vehicle widths (W1, W2), vehicle heights (H1, H2, H3, H4), and so on; its pose is determined by three parameters, which are its position (X', Y') on the ground plane and its orientation θ about the vertical axis perpendicular to ground-plane, based on the ground-plane constraints (GPC) [26]. These 15 parameters in total can be estimated by evaluating the fitness between image data and the projection of a 3D deformable model, in an evolutionary computing framework called estimation of multivariate normal algorithm- global ($EMNA_{global}$) [105].



(a)

$K^*\{\bullet\}$	Vertices	Description
I	0, 3, 4, 7	rear-side
II	4, 7, 8, 11	boot cover
III	8, 11, 12, 15	rear window
IV	12, 13, 14, 15	roof
V	9, 10, 13, 14	windshield
VI	5, 6, 9, 10	engine hood
VII	1, 2, 5, 6	front-side
VIII	8, 9, 12, 13	right window
IX	10, 11, 14, 15	left window
X	0, 1, 4, 5, 8, 9	right-side
XI	2, 3, 6, 7, 10, 11	left-side

(b)

Figure 3.6: (a) A generic model for 3-D vehicle modeling [28]. (b) Table of the kernels in 3-D vehicle model.

The basic assumption in 3-D vehicle modeling is that the camera is static and well calibrated (i.e., the 3 × 4 projective matrix $\widehat{\mathbf{P}}$ is known), so as to facilitate the projection from 3-D model to 2-D geometric primitives. The pose initialization of a 3-D model is achieved from image-plane-based vehicle detection. For each object of interest

segmented from the foreground extraction, a bounding box of the vehicle region is assigned. Owing to $\widehat{\mathbf{P}}$, the homography correspondence between the image plane and the ground plane can be easily obtained. Therefore, the center of the bounding box in the image plane, corresponding to the coordinate of the point in the ground plane, is used as the initialized values of (X', Y'). Based on the previous tracking results, which connect the coordinates of vehicles in the current frame with those in the previous one, and the obtained homography correspondence, the location of vehicles in the world coordinate system (WCS) can then be estimated.

A fitness evaluation score (FES) is considered as the measurement of the 3-D vehicle model fitting in an iterative optimization. To evaluate the FES, the wire-frame 3-D vehicle model is projected onto the image plane to construct a series of visible projected line segments. For every visible projected line segment l, a virtual rectangle with size of $L_r \times 2W_r$ is assigned, as shown in Figure 3.7. The idea of the fitness evaluation is to compare gradients along the directions perpendicular to the projected line segments. If a line segment fits the image data well, the gradient directions of pixels with large gradient magnitude values in the rectangle should concentrate on the perpendicular direction of this projected line. The gradient magnitude $m(x, y)$ and orientation $o(x, y)$ are calculated for each pixel S_i within the virtual rectangle. More specifically, the fitness score E_{S_i} for S_i is calculated by the vertical component of its gradient magnitude along the line segment direction with angle ϕ:

$$E_{S_i} = \left| m(x, y) \cdot \sin(o(x, y) - \phi) \right| . \tag{6}$$

Note that E_{S_i} is further weighted by a standard Gaussian distribution $G_{0,w}(d_i)$ with $\mu = 0$ and $\sigma = w$, where d_i denotes the distance between S_i and the projected line segment. Hence, the fitness score of the project line segment l is calculated by the weighted sum E_{S_i}, and the whole FES between the projection of the vehicle model and the image data is accumulated by all the visible lines:

$$E = \sum_{l} [\log(E_l)] = \sum_{l} \log\left(\sum_{S_i} [E_{S_i} \cdot G_{0,w}(d_i)] \right). \quad (7)$$

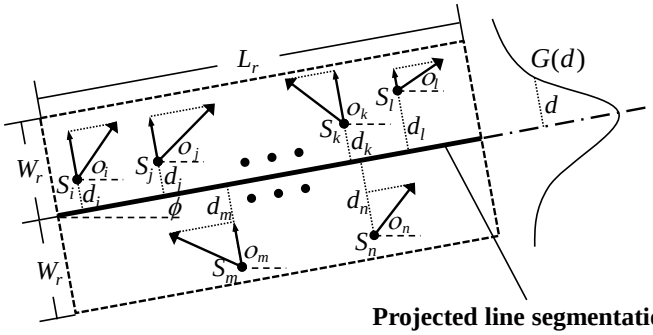


Figure 3.7: The idea of fitness evaluation.

The parameters optimization is to find a better (i.e., higher FES) 3-D vehicle configuration in the 15-dimensional (12 shape + 3 pose) parameter space. In order to simultaneously optimize these 15 parameters, which are independent of each other, an evolutionary computing framework called estimation of multivariate normal algorithm-global ($EMNA_{global}$) is adopted. $EMNA_{global}$ is an extended version of the estimation of distribution algorithm (EDA), which has a high efficiency for optimization of multiple variables [28]. Consequently, the parameters are iteratively optimized to achieve the best fitting 3-D vehicle model.

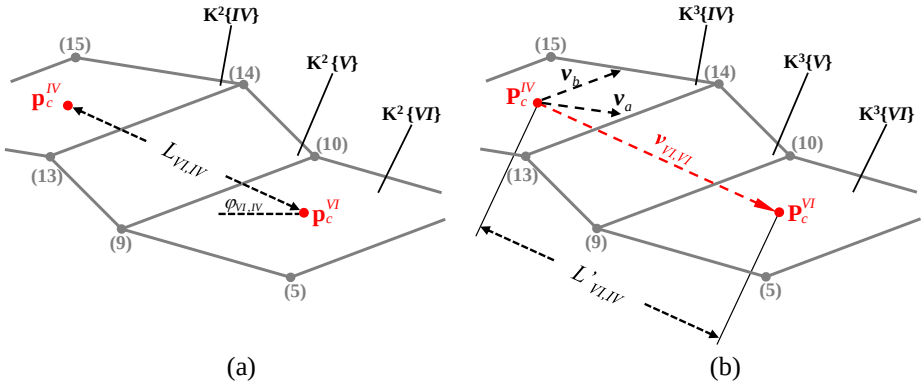


Figure 3.8: An example of the constraints between $K^*\{VI\}$ and $K^*\{IV\}$ in (a) 2-D case and (b) 3-D case.

3.4.b CMK Tracking with 3-D Vehicle Modeling

To combine CMK tracking with 3-D vehicle modeling, we regard each patch (plane) of the 3-D model as a kernel. The corresponding vertices of each kernel, annotated by $K^{*}\{\bullet\}$ (superscript indicates the dimension of the kernel space), are shown in Figure 3.8. Each kernel is a basic component in the tracking procedure.

Multiple-Kernel in 2-D Space

After a 3-D vehicle model is built, it is projected onto the image to form a 2-D wire-frame vehicle model, which includes many kernels ($K^2\{\bullet\}$). Based on the framework in [2], the color histograms of the kernels are used in the measurement of the similarity. Hence, $K^2\{\kappa\}$ (also called the κ^{th} *target kernel*) is described by its probability density function (pdf) q in terms of r-bin histograms, and the u^{th} bin of the κ^{th} kernel is given by:

$$q_u^{\kappa} = \frac{\sum_{i=1}^{n_\kappa} k\left(\left\|\frac{\mathbf{p}^{\kappa} - \mathbf{p}_i^{\kappa}}{h}\right\|^2\right)\delta\left[b(\mathbf{p}_i^{\kappa}) - u\right]}{\sum_{i=1}^{n_\kappa} k\left(\left\|\frac{\mathbf{p}^{\kappa} - \mathbf{p}_i^{\kappa}}{h}\right\|^2\right)}, \quad \sum_{u=1}^{r} q_u^{\kappa} = 1, \tag{8}$$

where $\|\bullet\|$ denotes L2 norm, $\mathbf{p}^{\kappa} \in \mathbb{R}^2$ is the center of $K^2\{\kappa\}$; $\{\mathbf{p}_i^{\kappa}\}_{i=1\ldots n_\kappa}$; $\mathbf{p}_i^{\kappa} \in \mathbb{R}^2$ are the pixel locations inside $K^2\{\kappa\}$; h is the bandwidth of the $K^2\{\kappa\}$; and δ is the Kronecker delta function. The function $b : \mathbb{R}^2 \to \{1...r\}$ associates the pixel at location $\mathbf{p}_i$ with the index $b(\mathbf{p}_i^{\kappa})$ of its bin in the color histograms. $k(\bullet)$ is a kernel function with a convex and monotonic decreasing kernel profile.

During multiple-kernel tracking, all the kernels search the regions which have the most similarity to the target kernel, while maintaining the given constraints. However, either due to the view aspects or occlusions, not all the kernels are reliable. First, according to the view aspect, only completely visible planes projected onto the image frame are used for multiple kernel tracking. In other words, kernels hidden behind are weighted by zero. Second, to keep the inter-relationship of the kernels, several constraints

need to be assigned based on the 2-D geometry between the kernels. Hence, the constraint functions are assigned by:

$$\begin{cases} \left\| \mathbf{p}_c^{\kappa} - \mathbf{p}_c^{IV} \right\|^2 = (L_{\kappa,IV})^2 \\ \dfrac{y^{\kappa} - y^{IV}}{x^{\kappa} - x^{IV}} = \tan^{-1}\left(\varphi_{\kappa,IV}\right) \end{cases}, \text{ for any visible } \mathrm{K}^2\{\kappa \mid \kappa \neq IV\}, \tag{9}$$

where $\mathbf{p}^{IV} = (x^{IV}, y^{IV})$ is the center of $\mathrm{K}^2\{IV\}$ and $\{(x^{\kappa}, y^{\kappa})\}$ are the centers of the visible kernels $\mathrm{K}^2\{\kappa\}$, $L_{\kappa,IV}$ is the initial distance between the centers of $\mathrm{K}^2\{\kappa\}$ and $\mathrm{K}^2\{IV\}$, and $\varphi_{\kappa,IV}$ is the initial angle between the kernel axis and the horizontal axis. These constants need to be adaptively updated when either size and/or orientation of a vehicle are greater than the empirical thresholds. Figure 3.8(a) shows an example of $\kappa = VI$, where (•) indicates the index of the vertices defined in Figure 3.6(b). Notice that both $L_{VI,IV}$ and $\varphi_{VI,IV}$ are measured in 2-D space.

Taking advantage of kernel-based tracking and 3-D vehicle modeling of the tracked vehicle, this approach [102] incorporates 3-D vehicle models into the multiple-kernel based tracking. The system adopts the (2-D) CMK tracking framework [8], facilitated by an automatically built 3-D vehicle model, to efficiently track and localize the vehicles. Even though the approach in [102] performs well in limited scenarios, there are some issues that need to be improved:

1) As the scale or the orientation of vehicles changes greatly, the information of the kernels becomes unreliable and has to be updated.
2) The tracking of the kernels is based on the 2-D (image) space because the color information is extracted from the 2-D image frame, while movement of the 3-D vehicle model is still defined in 3-D space.
3) The constraints for multiple kernels in 2-D space are not suitable to the kernels in 3-D space, since they will change during the movement of a vehicle due to the varying view aspects.

These issues may gradually make the tracking features of the vehicles unreliable, so that the error of the tracking becomes larger and larger. To overcome this problem, in this paper, we propose an innovative 3-D constrained multiple-kernel tracking facilitated by the 3-D vehicle model. Instead of tracking kernels in the 2-D space, the proposed approach directly tracks the vehicles in 3-D space.

Multiple-Kernel in 3-D Space

The basic concept is to consider 3-D planes as kernels and search their candidate kernels in 3-D space instead of 2-D space. In other words, we minimize the cost function $J(\mathbf{x})$ in Eq. (4), where $\mathbf{x} \in \mathbb{R}^3$. To make it clearer, we redefine the formula and the annotations for this problem. Let $\mathbf{p}$ be the 2-D points in the images, and $\mathbf{P}$ be the corresponding 3-D points, where $\mathbf{p} \in \mathbb{R}^2$, $\mathbf{P} \in \mathbb{R}^3$, $\mathbf{p}$ is obtained by projecting $\mathbf{P}$ through the projective matrix $\widehat{\mathbf{P}}$.

To describe a kernel in 3-D space, we need to provide color information to a 3-D kernel ($\mathrm{K}^3\{\bullet\}$), because the color information from the images is only defined in 2-D space. Thus, for each visible kernel $\mathrm{K}^3\{\kappa\}$, we associate the color information by back-projecting the 2-D points within $\mathrm{K}^2\{\kappa\}$ to the 3-D points within $\mathrm{K}^3\{\kappa\}$. Let $\overleftarrow{\mathbf{P}}^{\kappa}$ denote the 3-D points back-projected from the 2-D point $\mathbf{p}^{\kappa}$ in the image to the 3-D kernel (or plane) $\mathrm{K}^3\{\kappa\}$, i.e., the intersection between a viewing ray passing through $\mathbf{p}^{\kappa}$ and the plane $\mathrm{K}^3\{\kappa\}$. Therefore, $\mathrm{K}^3\{\kappa\}$ can now be described by its pdf q in terms of the r-bin histograms, and the u^{th} bin of the κ^{th} kernel is given by:

$$q_u^{\kappa} = \frac{\sum_{i=1}^{n_{\kappa}} k\left(\left\|\frac{\mathbf{P}_c^{\kappa} - \overleftarrow{\mathbf{P}}_i^{\kappa}}{h'}\right\|^2\right)\delta\left[b(\mathbf{p}_i^{\kappa}) - u\right]}{\sum_{i=1}^{n_{\kappa}} k\left(\left\|\frac{\mathbf{P}_c^{\kappa} - \overleftarrow{\mathbf{P}}_i^{\kappa}}{h'}\right\|^2\right)}, \quad \sum_{u=1}^{r} q_u^{\kappa} = 1, \tag{10}$$

where $\{\overleftarrow{\mathbf{P}}_i^{\kappa}\}_{i=1\ldots n_{\kappa}}, \overleftarrow{\mathbf{P}}_i^{\kappa} \in \mathbb{R}^3$ are the 3-D points back-projected from $\{\mathbf{p}_i^{\kappa}\}$ (inside $\mathrm{K}^2\{\kappa\}$) to $\mathrm{K}^3\{\kappa\}$, $\mathbf{P}_c^{\kappa}$ is the center and h' is the bandwidth of $\mathrm{K}^3\{\kappa\}$.

A reference kernel (annotated by $K^3\{\kappa^*\}$) which has the maximum visible area is selected within $K^3\{I, II,\ldots, IX\}$. To properly set the constraints, we consider the 3-D geometry of the vehicle model. First, the distance between $K^3\{\kappa\}$ and $K^3\{\kappa^*\}$should be the same, which implies

$$\left\|\mathbf{P}_c^{\kappa} - \mathbf{P}_c^{\kappa^*}\right\|^2 = (L'_{\kappa,\kappa^*})^2 \text{, for any visible } K^3\{\kappa \mid \kappa \neq \kappa^*\}, \tag{11}$$

where L'_{κ,κ^*} is the initial distance between $K^3\{\kappa\}$ and $K^3\{\kappa^*\}$. Second, the pitch and yaw between $K^3\{\kappa^*\}$ and $K^3\{\kappa\}$ should be the same. We start to calculate the two vectors v_a and v_b which are orthogonal to each other and cross $\mathbf{P}_c^{\kappa^*}$,

$$v_a = \frac{\mathbf{P}_a^{\kappa} + \mathbf{P}_o^{\kappa}}{2} - \mathbf{P}_c^{\kappa^*}, \quad v_b = \frac{\mathbf{P}_b^{\kappa} + \mathbf{P}_o^{\kappa}}{2} - \mathbf{P}_c^{\kappa^*}, \tag{12}$$

where $\mathbf{P}_o^{\kappa}$ are the intersection of two adjacent lines selected from the $K^3\{\kappa\}$, and $\mathbf{P}_a^{\kappa}$, $\mathbf{P}_b^{\kappa}$ are the end points of both wires, respectively. Let us define $v_{\kappa,\kappa^*} = \mathbf{P}_c^{\kappa} - \mathbf{P}_c^{\kappa^*}$, we can obtain the constraints for the pitch and yaw,

$$\begin{cases} \dfrac{v_a \cdot v_{\kappa,\kappa^*}}{\| v_a \| \| v_{\kappa,\kappa^*} \|} = \cos\left(\phi_{\kappa,\kappa^*}\right) \\ \dfrac{v_b \cdot v_{\kappa,\kappa^*}}{\| v_b \| \| v_{\kappa,\kappa^*} \|} = \cos\left(\varsigma_{\kappa,\kappa^*}\right) \end{cases} \text{, for any visible } K^3\{\kappa \mid \kappa \neq \kappa^*\}. \tag{13}$$

Figure 3.8(b) shows an example of the constraints of $K^3\{VI\}$ with $\kappa^* = IV$, with v_a, v_b, and $v_{VI,IV}$ being indicated in the figure.

Similarity and Fitness Terms

Although the vehicles can be approximately tracked by the CMK tracking based on the color information, the 3-D vehicle models cannot be properly fitted. To overcome this problem, we take the fitness of the 3-D vehicle model into account when solving the optimization; in other words, the cost function in Eq. (4) becomes

$$J(\mathbf{x}) = \sum_{\kappa=1}^{N_k} w_\kappa \left(J_\kappa^s(\mathbf{x}) + J_\kappa^f(\mathbf{x}) \right), \tag{14}$$

where $J_{\kappa}^{s}(\mathbf{x})$ is the *similarity term* associated with the color similarity of the kernel, and $J_{\kappa}^{f}(\mathbf{x})$ is the *fitness term* associated with the fitness of the kernel, $\mathbf{x} \in \mathbb{R}^3$. Hence, the cost function in Eq. (5) can be rewritten:

$$\begin{aligned}\delta_{\mathbf{x}} &= \alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{W}(\mathbf{J}_{\mathbf{x}}^{s}+\mathbf{J}_{\mathbf{x}}^{f})+(-\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}(\mathbf{x})) \\ &= \alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{W}\mathbf{J}_{\mathbf{x}}^{s}+(-\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}(\mathbf{x}))+\alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{W}\mathbf{J}_{\mathbf{x}}^{f} \quad (15) \\ &= \delta_{\mathbf{x}}^{A} \quad + \quad \delta_{\mathbf{x}}^{B} \quad + \quad \delta_{\mathbf{x}}^{C},\end{aligned}$$

where $\delta_{\mathbf{x}}^{C}$ is the movement vector associated with the fitness. Again it can be shown $\delta_{\mathbf{x}}^{C}$ is orthogonal to both $\delta_{\mathbf{x}}^{A}$ and $\delta_{\mathbf{x}}^{B}$ (see Appendix A), which implies moving along $\delta_{\mathbf{x}}^{C}$ is able to achieve better 3-D model fitting while maintaining the similarity and constraints of the kernels.

Given a 3-D kernel $\mathrm{K}^3\{\kappa\}$, the similarity term is determined by the similarity measurement, for instance, the distance between the target and the candidate kernels. As for the fitness term, the purpose is to further fit the 3-D vehicle model onto the image. Since higher FES implies a better fitness of the model, the FESs of the kernels are considered as the measurement. Thus, we design a kernel density estimator by taking the FES into account. For a pixel $\mathbf{p}^{\kappa}$ belonging to $\mathrm{K}^2\{\kappa\}$, we calculate the FESs of the lines, associated with $\mathrm{K}^2\{\kappa\}$ centered at $\mathbf{p}^{\kappa}$ (annotated by $\mathrm{K}_{\mathbf{p}}^{2}\{\kappa\}$). In other words, $\mathrm{K}_{\mathbf{p}}^{2}\{\kappa\}$ is projected from $\mathrm{K}^3\{\kappa\}$ which is shifted by $\bar{\mathbf{P}}^{\kappa}-\bar{\mathbf{P}}_{c}^{\kappa}$. Therefore, for each $\mathbf{p}^{\kappa}$, the FES $E_{\kappa}(\mathbf{p})$ is defined by:

$$E_{\kappa}(\mathbf{p}) = -\sum_{l \in \mathrm{K}_{\mathbf{p}}^{2}\{\kappa\}} [\log(E_l)]. \quad (16)$$

Note that $E_{\kappa}(\mathbf{p})$ is then applied to a kernel density estimator with kernel profile $k(\bullet)$. More specifically, $J_{\kappa}^{f}(\mathbf{x})$ is calculated as follows:

$$J_{\kappa}^{f}(\mathbf{x}) = \frac{\sum_{i=1}^{n_\kappa} k\left(\left\|\frac{\mathbf{P}^{\kappa} - \bar{\mathbf{P}}_i^{\kappa}}{h'}\right\|^2\right) E_{\kappa}(\mathbf{p}_i^{\kappa})}{\sum_{i=1}^{n_\kappa} k\left(\left\|\frac{\mathbf{P}^{\kappa} - \bar{\mathbf{P}}_i^{\kappa}}{h'}\right\|^2\right)}. \tag{17}$$

The purpose of the fitness term is to densely estimate the fitness of the kernel, such that the optimization tends to better fit the kernels. However, it is not necessary to evaluate all of $\{E_{\kappa}(\mathbf{p})\}$ for a kernel $\mathrm{K}^3\{\kappa\}$, since the effect of the fitness is dominated by the points around the center of $\mathrm{K}^3\{\kappa\}$. Thus, we uniformly sample the (dense) points $\{\mathbf{p}_i^{\kappa}\}_{i=1\ldots n_d}$, $1 \le n_d \le n_\kappa$, whose normalized distances between their back-projected points and the center of the kernel are smaller than a given threshold, i.e., $\left|(\mathbf{P}^{\kappa} - \bar{\mathbf{P}}_i^{\kappa})/h'\right| < \rho, 0 < \rho \le 1$.

3.4.c A Complete Tracking System

We propose an automatic system which combines the Kalman-filtering framework with our 3-D vehicle model based CMK tracking. Figure 3.9 shows the schematics of the proposed vehicle tracking system. The first step is to segment the foreground objects, by using background subtraction technique. Second, the Kalman prediction is applied to the segmented objects. Then, if the 3-D vehicle model is not yet built or needs to be updated, the predicted pose of the vehicle is used for building the 3-D vehicle model, which is then applied to the CMK tracking; otherwise, the CMK tracking is facilitated by the pre-built 3-D vehicle model to track and locate the vehicles. Finally, the tracking results are used to update the states of the Kalman filter for further prediction.

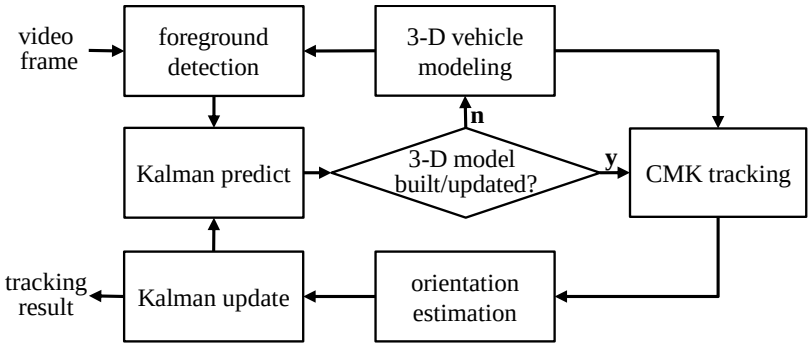


Figure 3.9: Overall proposed system framework.

After the CMK tracking, the kernels are local-optimally shifted based on the color similarity and shape fitness. However, when a vehicle turns left or right, the 3-D vehicle model cannot be fitted perfectly due to the orientation change. To improve the fitness of the 3-D vehicle model during the orientation change, we evaluate the FES of the 3-D vehicle models within a possible range of θ, and then select the 3-D vehicle model candidate with the highest FES. More specifically, for a tracked vehicle, we first predict the orientation (annotated by θ_{predict}) by Kalman-filtering. Then, the 3-D vehicle models with different θ are regarded as the candidates, where θ is set as $\theta_{\text{predict}} \pm \Delta\theta \cdot d$, $1 \le d \le 10$. A typical value of $\Delta\theta$ is 0.5. Finally, the candidate with the highest FES is determined as the solution.

When occlusion occurs, the system requires different strategies to deal with partial and/or total occlusions. The challenge of partial occlusion is that the features of the tracked vehicle become unreliable in some partitions, resulting in mismatch of the tracked objects. In the CMK tracking, a less reliable partition (kernel) will be assigned a smaller weight; thus, the tracking tends to be more confident on a partition with larger weights [8]. Thanks to this adaptive weight assignment, the system has the ability to overcome partial occlusions during tracking. As for total occlusion, where the tracked object totally disappears, a possible way is to predict the traces of the vehicle and then resume the tracking when the vehicle appears again. Some approaches [13], [17] are proposed to deal with the total occlusion with reasonable performance. In our system, we

believe that Kalman-filtering has the ability to approximately predict the location of the totally occluded vehicles, because the vehicles usually go straight forward in a short time duration under the surveillance video. If the vehicle appears again after the total occlusion, Kalman-filtering is adopted to predict the location of the vehicle in the subsequent frame, and resume CMK tracking combined with the pre-built 3-D vehicle model.

3.4.d Experimental Results

Several experiments and corresponding discussions are provided in this section to demonstrate the performance of the proposed method. All the experiments are processed on a personal computer with a P4 2.67GHz CPU and 2G DDR. The implementation is constructed by C/C++, and the experimental settings are described as follows. In the CMK tracking, K-L distance is used for all similarity measures, and the histogram of the object is constructed based on the HSV color space with a roof kernel. In the 3-D vehicle modeling, the parameters used in $\text{EMNA}_{\text{global}}$ are $N = 2000$, $R = 100$, and the stopping threshold for the gradient magnitude is 2. For improving the robustness of the system, we rebuild the 3-D vehicle and update the kernels every 5 frames. The surveillance camera is supposed to be well calibrated, which implies intrinsic and extrinsic parameters are known. For comparison, we manually labeled the ground truth of the 3-D vehicle models and measure the performance in terms of average errors (per frame), which is defined by:

$$err_{\text{ave}}(t) = \frac{1}{N_t} \sum_{j=0}^{N_t} \sum_{i=0}^{15} \left\| x_i^j(t) - g_i^j(t) \right\|, \tag{18}$$

where $x_i^j(t)$ is the 3-D location, $g_i^j(t)$ is the ground truth of vertex i of vehicle j, and N_t is the number of tracked vehicles in Frame t. The testing datasets include PETS 2000 Dataset [106], portion of AVSS PV Easy [107], and our own recorded six datasets (Dataset 1–6). The content of the datasets are described in the left partition of Table 3.2. We use Dataset 4 for the discussion of subsections A and B, and test all the videos under

certain scenarios in subsection C. All the videos associated with the simulations reported in this paper can be viewed on our website[1].

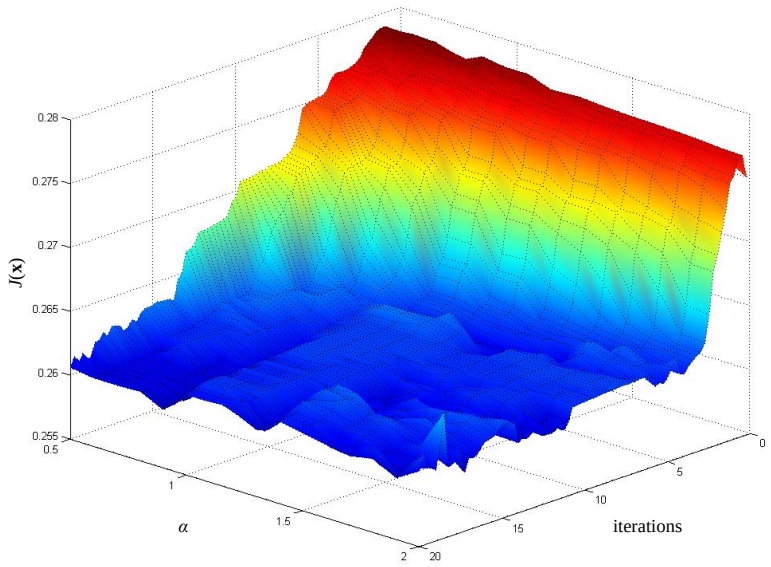


Figure 3.10: An example that shows the values of $J(x)$ with different step sizes (α) vs. the number of iterations, in case of 3-D kernels.

2-D Kernel vs. 3-D Kernel

The 2-D kernels ($K^2\{\bullet\}$) is the projection (onto image) of the 3-D kernels ($K^3\{\bullet\}$). Although they have different dimensions, the 3-D kernels still have some characteristics of the 2-D kernels:

1) The objects (patches in the 3-D model) are described by color histograms with a monotonically decreasing kernel profile $k(\bullet)$. The color information of $K^3\{\bullet\}$ is provided from the pixels within the corresponding $K^2\{\bullet\}$.
2) The mode of the kernel density in the local neighborhood can be found by applying the mean shift procedure [3]. Thus, we apply the mean shift vector with the opposite direction as our $\mathbf{J}_{\mathbf{x}}^{s}$ and $\mathbf{J}_{\mathbf{x}}^{f}$ in Eq. (15).
3) The K3{•} are bound together under certain constraints, which are defined by the

[1] website: http://allison.ee.washington.edu/kuanhuilee/3dvt

inter-relationship of the kernels in 3-D space, so that these constraints can be consistent during the tracking.

Owing to these characteristics, the candidate kernels tend to be more similar to the target kernels, and the optimization in Eq. (2) can gradually converge. To speed up the convergence of the optimization in the 3-D CMK tracking, one possible way is to increase the step size α in Eq. (5). However, enlarging α will increase the probability of oscillation. Figure 3.10 shows an example of how $J(\mathbf{x})$ changes with different α versus the number of iterations. From the figure, when α is smaller, $J(\mathbf{x})$ decreases slower, taking more iterations to converge. In contrast, when α is larger, $J(\mathbf{x})$ decreases more rapidly to relatively low cost values. Nevertheless, $J(\mathbf{x})$ oscillates more before convergence. For example, with $\alpha = 2$, $J(\mathbf{x})$ quickly decreases its value after 4 iterations, but still cannot converge at the 20^{th} iteration. For $1.0 < \alpha < 1.3$, the optimization converges in around 15 iterations, and there are no severe oscillations before convergence. Thus, to achieve better performance, we empirically choose $\alpha = 1.2$ for the step size, and $\varepsilon_j = 0.26$, $T = 20$ for the stopping criterion. Also, if the difference between the previous and the current ε_j is less than 0.001, the optimization is regarded as converged. Table 3.1 shows the average number of iterations required for the convergence in 2-D and 3-D CMK tracking, where 3DCMK– and 3DCMK denote the optimization without and with the fitness term, separately. As shown in the table, the optimization in 2-D CMK tracking takes less than 10 iterations to converge, while the 3-D CMK tracking takes about 12 to 15 iterations. The reason why the 3-D CMK tracking requires more iterations to converge is that the movement vectors need to shift along three directions ($\delta_{\mathbf{x}}^{A}$, $\delta_{\mathbf{x}}^{B}$ and $\delta_{\mathbf{x}}^{C}$). The 3-D CMK tracking with the fitness term ($J^{f}(\mathbf{x})$) needs more iterations than that without the fitness term, which implies shifting along the $\delta_{\mathbf{x}}^{C}$ need more iterations to converge. Although including $J^{f}(\mathbf{x})$ takes more time, it is still acceptable and worthwhile with the improved tracking performance.

Table 3.1: Convergence Comparison

Scheme	2DCMK	3DCMK–	3DCMK
Average iterations	9.561	11.744	15.915

Table 3.2: Average Error (Meter) Experiments

Dataset	Situations			Edge	# vehicles	Approach				
	s	*o*	*c*	Density	in total	Zhang [28]	Zheng [18]	2DCMK	3DCMK–	3DCMK
Dataset 1	√			1.4137	24	0.7180	0.8058	0.6868	0.5824	**0.5668**
Dataset 2		√		0.9135	5	0.8386	0.8985	1.8892	0.7545	**0.7148**
Dataset 3			√	1.0022	2	0.5075	0.5838	0.4125	0.3996	**0.2209**
Dataset 4	√	√	√	1.3510	11	0.4136	0.4203	0.3553	0.3414	**0.2607**
Dataset 5	√	√		1.4604	12	0.6338	0.7109	0.5886	0.4963	**0.4575**
Dataset 6	√		√	5.6056	5	1.9771	2.3334	0.8174	0.7233	**0.6245**
PETS 2000	√	√		1.8940	3	0.8512	1.0732	0.7144	0.6976	**0.4616**
AVSS 2007	√	√		2.4921	7	0.6554	0.7277	0.7995	0.7786	**0.5043**

Fitness Term

In Eq. (14), the similarity term tends to retrieve a candidate kernel which has the highest similarity to the target kernel, but, only based on the color histogram. The fitness term plays an important role in achieving better fitness of the 3-D vehicle model on the image, because it tends to refine the location of the kernel for obtaining the best fitness level. As mentioned in Section 3.4.b, a kernel density estimator is designed for fitting the vehicle 3-D model, and the dense points $\{\mathbf{p}_i^{\kappa}\}_{i\in 1,\ldots n_d}$ are uniformly sampled from the surrounding region $\left|(\mathbf{P}^{\kappa}-\tilde{\mathbf{P}}_i^{\kappa})/h'\right|<\rho$. If ρ is too small, the moving range for refinement of the fitness will be limited; if ρ is too large, although the moving range becomes wide, the optimization may be stuck in a local minimum which is far away from the ground truth. Besides, sampling too many dense points for the pdf evaluation is not efficient, since many dense points do not have color models highly similar to the target model. Figure 3.11 shows the average errors of the 3-D CMK tracking with the fitness term for different ρ values. The solid line in Figure 3.11 denotes the case of 3-D CMK with the fitness term, and the dotted line denotes the case of 3-D CMK without the fitness term. In the simulations without the fitness term, we disregard the fitness term $\delta_{\mathbf{x}}^{C}$ in Eq. (15),

with other elements being kept intact. When ρ becomes larger, the improvement of the fitness term becomes smaller, because many dense points far away from the ground truth dominate the movement to the other local minima. The improvement also becomes smaller when ρ is smaller than 0.3. This is because there are too few reliable dense points to dominate the movement toward better fitness. Owing to these observations, we choose $\rho = 0.4$ in the experiments to achieve better results.

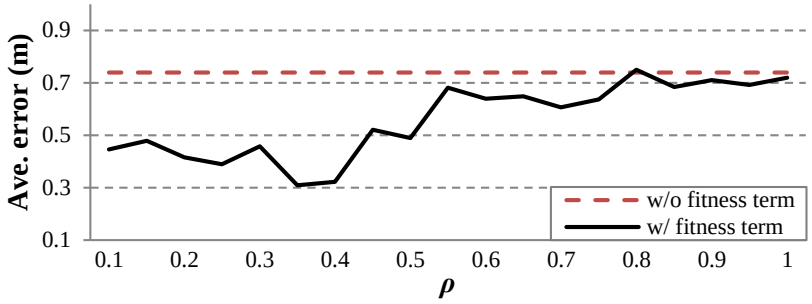


Figure 3.11: The average errors of the 3-D CMK tracking with fitness term under different ρ.

Systematic Tracking Testing

In these experiments, we test our proposed approach under certain scenarios, including scale change, orientation change and occlusion. Dataset 1 is used for testing scale change, Dataset 2 is for testing orientation change, the Dataset 3 is for testing occlusion, and the rest are for testing the situations of mixed scenarios. The influence of surrounding complexity is discussed in subsection 4. The performance of the proposed 3-D CMK tracking approach (with and without the fitness term) is also compared with other state-of-the-art intensity-based approaches, such as Zhang's [28] and Zheng's [18] approaches. All three approaches adopt the 3-D vehicle modeling in [28], but with different pose estimation schemes. Our implementations of their approaches do not include additional optimization, such as parallel computing mentioned in [28], to the 3-D vehicle modeling, since such optimization cannot change the accuracy of the pose estimation.

i. Scale Change

Scale change is one of the major issues during tracking. In [2], an adaptive scale change scheme for the kernel-based tracking is proposed. The basic idea is to adaptively change the bandwidth *h* of the kernel when the scale of the target varies, according to the scale invariance property of the similarity function of the kernel. The CMK tracking framework in [8] proposes a density estimator derived from [2], and performs well in human tracking. Thanks to the use of 3-D kernels, the movement vectors shifting in 3-D space are able to systematically adjust the scales of the kernels during the tracking. To demonstrate the efficiency of the 3DCMK, we tested the vehicles in Dataset 1, where the scale change happens to all vehicles (24 vehicles in total). Figure 3.12 shows the visual results of the 3DCMK in different frames. Table 3.2 shows the average errors of the tracking results with different approaches tested with Dataset 1. Figure 3.15 shows the frame-by-frame average errors of a tracked vehicle with different approaches. The 3DCMKs can achieve better results because the color information of the vehicle is effectively utilized.

To further investigate the performance difference between 2-D and 3-D CMK tracking, we compare them without updating the 3-D vehicle model. Figure 3.16 shows the average similarity of the kernels of an example of Dataset 1, in 2-D and 3-D CMK tracking. In Frame 56, the 3-D vehicle model is built, so that the average similarity is 1. In the 2-D CMK tracking, the average similarity continuously decreases due to the scale change. However, in the 3-D CMK tracking, the average similarity decreases until Frame 67, and then maintains values within a range. This is because the candidate kernels get modified when the scale is changing. In our previous work [102], as well as the 2DCMK, we overcome the issue by updating the $K^2\{\bullet\}$ in a period of time to maintain the reliability of the $K^2\{\bullet\}$. However, such a scheme may not succeed in some cases, for example, when the vehicle is occluded with a simultaneous scale change, or the 3-D vehicle model is not well fitted when the $K^2\{\bullet\}$ is rebuilt. These cases will cause inaccurate results, and the fitting errors further propagate during tracking.

Figure 3.12: The results of 3-D CMK tracking in Dataset 1, where a scale change of the vehicle happens.

Figure 3.13: The results of 3-D CMK tracking in Dataset 2, where an orientation change of the vehicle happens.

Figure 3.14: The results of 3-D CMK tracking in Dataset 3, where a vehicle is occluded by another one.

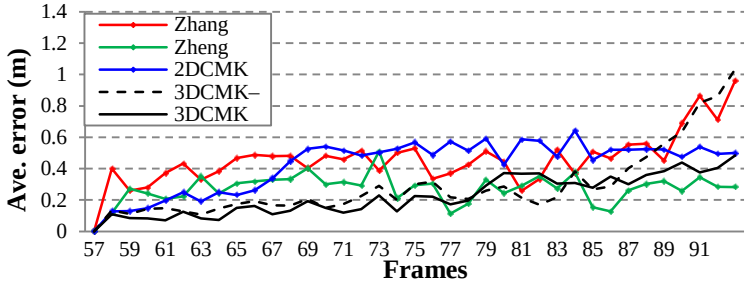


Figure 3.15: The average errors of the tracking results with different approaches during scale change.

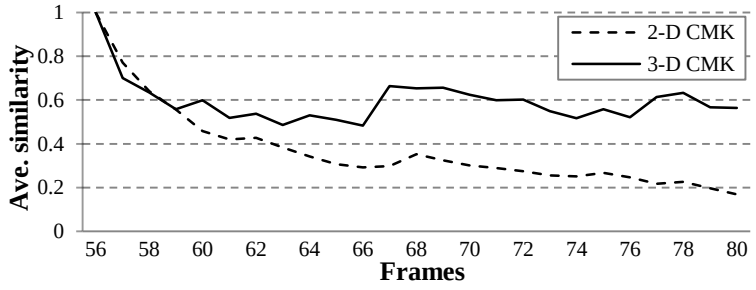


Figure 3.16: The average similarity of the kernels of the example in Figure 3.13.

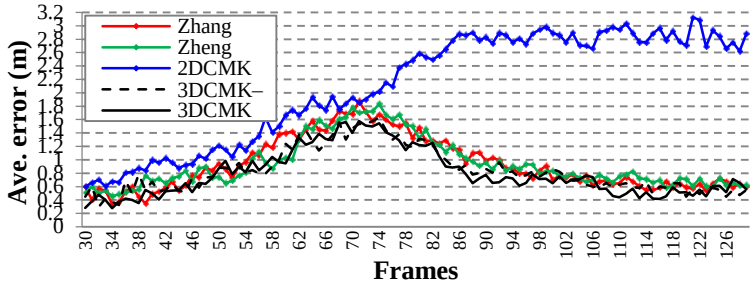


Figure 3.17: The average errors of the tracking results with different approaches during orientation change.

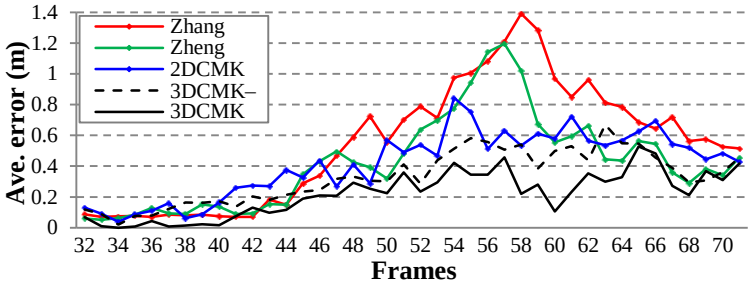


Figure 3.18: The average errors of the tracking results with different approaches under (partial) occlusion.

ii. Orientation Change

Orientation change occurs when a vehicle turns left or right. To test the performance during the orientation change, we test with Dataset 2 in which several sedans turn left at a corner. Figure 3.13 shows the visual results of the 3DCMK approach. Each figure consists of two half frames: the left half is the first frame when a vehicle enters into the region of interest, and the right half shows the tracked vehicle several frames after the first frame. The 3DCMK performs well while the vehicles continuously change their

orientations. Table 3.2 shows the average errors of the tracking results of different approaches, where the 3DCMK approaches perform as well as Zhang's and Zheng's approaches. This is because the performance for orientation change is dominated by the fitness evaluation. If the initial tracking contains adequate fitness in the iterative FES optimization, CMK tracking does not have an obvious advantage. Figure 3.17 shows the frame-by-frame average errors of a tracked vehicle with different approaches. The performance of the 3DCMKs is as good as the other approaches. On the other hand, the results of 2DCMK tracking are much worse than the other approaches, because it does not have ability to handle the orientation changes.

iii. Occlusion

Occlusion is another major issue during tracking, because occlusion will break the matching of the object template previously described, especially in terms of 2-D geometry features. Owing to the adaptive weighting mechanism associated with the similarity, the 3-D CMK tracking is able to track the vehicles facilitated by the more reliable kernels. In this section, we only consider the partial occlusion scenario. The total occlusion scenarios, where the whole vehicle is occluded such that it is impossible to accurately locate the vehicle, will require some additional dynamic state-space modeling (e.g., Kalman filtering) to facilitate the tracking as discussed in Section 3.4.c. Figure 3.14 shows the results of the proposed approach while one vehicle is partially occluded by another vehicle. Note that the vehicles can be located and tracked well without occlusion (see the leftmost and rightmost figures in Figure 3.14) or with partial occlusion (see the middle three figures in Figure 3.14). Since the similarity of the occluded kernels becomes smaller, a smaller w_k in Eq. (4) is assigned. Figure 3.18 shows the frame-by-frame average errors of the tracking results for different approaches. From the figure, 3DCMK again performs better than the other approaches. When occlusion occurs (from Frame 43 to Frame 60), Zhang's and Zheng's approaches are not able to locate or track properly, because the gradients of the occluded vehicle are adversely affected by the occlusion. As

for the CMK-based tracking, kernels mainly rely on the non-occluded partitions, so as to maintain proper tracking under occlusion.

iv. *Automatic Tracking System*

In order to show the performance of the proposed approach, we test the complete tracking system which combines the 3-D CMK tracking with the Kalman filtering framework, as described in Section 3.4.c. Figure 3.19 (a)–(e) shows some of the results in the different datasets respectively, showing the localization of different vehicles in different colors. We further compare the performance, in terms of the average errors, of several competing techniques, as shown in Table 3.2. In the testing of PETS 2000, the proposed approach has obviously better performance because Zhang's and Zheng's approaches evaluate the localization of the 3-D vehicles by only using the FES scheme, which is sensitive to the intensity changes of the surroundings. The 2DCMK cannot achieve bigger improvement in the presence of scale and orientation changes. In AVSS 2007, most cases have scale changes and orientation changes; thus the 3DCMK can achieve better performance than the others. The results of the 2DCMK and 3DCMK– are worse than Zhang's approach, because both only rely on color information, resulting in the increased errors when orientation changes occur. In Dataset 1, Dataset 5, and Dataset 6, in which the scenarios are mostly scale changes, the CMK based approaches perform slightly better than Zhang's and Zheng's approaches. Dataset 2 is an orientation change scenario, as shown in Figure 3.13, and Dataset 3 is an occlusion scenario, as shown in Figure 3.14, where 3DCMK performs better than the other approaches. In Dataset 4, the performance of 3DCMK is slightly improved because few scale changes and occlusions occur. In Dataset 6, total occlusion occurs when two vehicles intersect (see the leftmost figure of Figure 3.19(e)). Since the background is complicated, Zhang's and Zheng's approaches are adversely influenced by the initial misleading of the shape fitting.

From Table 3.2, the 2DCMK performs as well as the 3DCMK– since they are mainly based on color information. Therefore, the CMK based approaches can achieve good

results especially when there is image clutter caused by the surroundings; the improvement resulting from the fitness term is obvious, since it efficiently enhances the shape fitting of the 3-D vehicle models. Especially in the cases of the orientation changes, the improvement is more obvious than the other cases, since the projected 3-D vehicle models adapt more effectively.

v. Surrounding Complexity

Surrounding complexity has a large impact on the fitness evaluation. Since higher complexity may produce a higher FES, there is more possibility to lead the FES evaluation to an unexpected local optimum. In order to measure the surrounding complexity, we use edge density in the region of interest (ROI) as the metric:

$$d_{edge} = \frac{1}{A_{roi}} \sum_{\mathbf{p} \in \mathrm{ROI}} e(\mathbf{p}), \tag{19}$$

where $e(\mathbf{p})$ is the magnitude of the Sobel 3×3 operation at $\mathbf{p}$, and A_{roi} is the area of the ROI.

Higher d_{edge} normally indicates higher scene variability, which can produce possible local maxima of the FES, including other objects than the vehicle. If the FES evaluation is trapped in an unexpected local optimum, the error of the shape fitting gradually increases, and then the tracking prediction is further degraded. On the other hand, the 3DCMKs track the object by first considering color information, thus providing better initialization for the FES optimization. As shown in Table 3.2, 3DCMK tends to have more obvious improvement than the other approaches when testing the datasets with higher d_{edge}, such as Dataset 5, 6, and PETS2000. As for AVSS2007, the performance is only slightly improved. This is because the vehicles in that video are more discriminative when compared to the surroundings. In the occlusion case, such as Dataset 2, the fitting confusion is caused by occlusion. If the edge density of the occlusion is high, the proposed approach can achieve better performance; otherwise, if the edge density of the

occlusion is low, such as the traffic lights in Dataset 4, the improvement is not as impressive.

Figure 3.19: A vehicle in PETS 2000 changes its orientation continuously. The results of 3-D CMK tracking in (a) PETS 2000, (b) AVSS 2007 , (c) Dataset 4, (d) Dataset 5, and (e) Dataset 6.

vi. Computation

Thanks to the high efficiency of the kernel-based tracking, the proposed approach is quite efficient in computations. To demonstrate the efficiency, we compare the CPU computation time of the pose estimation, i.e., locating the pose parameters (X', Y', θ), in different approaches. Table 3.3 shows the results of the computation time in different approaches, in terms of seconds per vehicle. Zhang's approach has the highest computation due to the large number of generations of individuals in the $EMNA_{global}$ framework. Instead of generating the individuals randomly, Zheng's approach efficiently narrows the searching range, so that the computation is greatly reduced. Within the CMK based approaches, the 2DCMK has the best results because the optimization converges along the 2-D mean-shift vector. As for the 3DCMK– and 3DCMK, the optimization converges along the 3-D mean-shift vector, which implies that more iterations are needed. As mentioned above, when compared with 2DCMK, 3DCMK– needs about 2 more iterations on average, and 3DCMK needs about 6 more iterations on average. From Table 3.3, the 3DCMK– spends one second more on average, and the 3DCMK is about 2 seconds more on average; which is still relatively more efficient than Zhang's and Zheng's approaches.

Table 3.3: Computational Complexity

Ave. time (second)	Approach				
	Zhang [28]	Zheng [18]	2DCMK	3DCMK–	3DCMK
per vehicle	47.268	5.277	1.197	1.283	1.425

vii. Limitations

Although performing well in many scenarios mentioned above, the proposed approach has some limitations. First, the 3-D vehicle modeling plays an important role in initially describing the kernels, as well as the target kernels. The tracking procedure is to find the candidate kernels which have the highest similarity to the corresponding target kernel. If the 3-D vehicle modeling cannot provide reliable target kernels, tracking will be based on incorrect kernels, and cause error propagation. Second, the color of the

vehicle also impacts the tracking results, since 3-D CMK tracking is highly dependent on the color histogram. For example, if a vehicle is occluded by an obstacle which has very similar color to the occluded vehicle, the CMK tracking will be confused and create the wrong tracking results.

viii. Discussion

In Zhang's approach, the 3-D vehicle models are located by iteratively optimizing its parameters, according to the measurement of the fitness in terms of intensity and gradients. Zheng's approach, also based on intensity, efficiently measures the fitness of the 3-D vehicle models in certain ranges, so as to reduce the searching computation. The difference between these intensity-based approaches and our proposed approach is that we further take the color information into account. In our proposed framework, the optimization of the cost function includes not only the color histogram but the 3-D shape fitness score. Therefore, the vehicles are tracked based on the color, which is a distinctive characteristic of the vehicle, and their shapes are fitted with 3-D models at the same time. Another difference between the competing approaches and our proposed approach is the optimization of the cost function. The other approaches search for the location of the vehicles with a relatively better fitness level, which implies that high computation is necessary in order to achieve good performance. However, the proposed approach optimizes the cost function along the mean-shift vectors, which is able to quickly converge.

Most importantly, we propose a 3-D model-based tracking framework for vehicle tracking. Theoretically, the framework can also be applied to a highly complicated 3-D vehicle model [29], or any object with a 3-D model, to efficiently track and locate the objects.

Chapter 4 – Tracking under a Single Moving Camera

4.1 Overview

In this chapter, our proposed human tracking approach under a single moving camera is described in detail [108]. The proposed approach effectively integrates Visual Simultaneous Localization And Mapping (V-SLAM), human detection, ground plane estimation, and kernel-based tracking techniques, so as to successively track humans in 3-D space. The system starts with human detection and V-SLAM for camera calibration. Then, the ground planes are estimated based on the camera motions, so that the 3-D locations (relative to the cameras) of the humans can be inferred. By taking 3-D information into account, we reformulate the tracking problem based on the Constrained Multiple-Kernel (CMK) approach [8], which can effectively resolve the occlusions during tracking, to globally optimize the data association between consecutive frames. Hence, the proposed system can not only track the humans effectively, but can also robustly handle occlusion during tracking, especially for ground moving platforms such as unmanned ground vehicles.

Figure 4.1 shows the overview of the system. First, the system calibrates the camera motion by the classic Structure from Motion (SfM) pipeline within f_s video frames from a moving platform, where we assume the height of the camera is known. Meanwhile, a pre-trained human detector is adopted to detect humans in the video frames. Then, according to the calibrated camera motion, we can thus estimate the ground plane, which is used by the pose estimation step to back-project the humans' 2-D locations to 3-D locations. Based on the humans' 3-D locations relative to the camera motion, a depth map is constructed to represent the relative depth of the detected humans. Finally, the proposed Depth CMK tracking technique which combines the CMK tracking and the depth information, is used to globally associate the detections frame by frame. If a detection is

missing during the tracking, a hypothesized detection is inserted based on the tracking results, so that the detected humans can be tracked continuously.

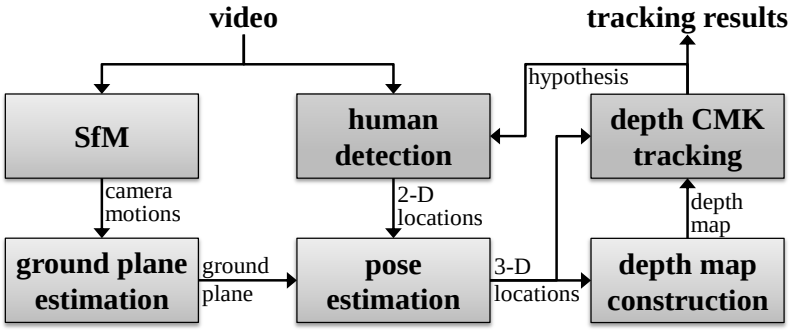


Figure 4.1: Overview of the proposed system.

4.1.a Structure-from-Motion (*SfM*)

The monocular V-SLAM framework follows a standard bundle adjustment formulation, where an interest point operator is first applied to extract feature points that are matched between consecutive frames. Then, according to the matched feature points, a 5-point based RANSAC algorithm is used to estimate the initial epipolar geometry. Finally, the camera motions are determined by camera resection. The set of 3-D points and the corresponding feature points are used in the bundle adjustment process to iteratively minimize the reprojection error:

$$X_p = \arg\min_{X,\hat{\mathbf{P}}} \sum_{\forall p,q} d(\hat{\mathbf{P}}_q \cdot X_{pq}, x_{pq}), \tag{20}$$

where X_p is the 3-D location of the p^{th} feature point, x_{pq} is the observed 2-D location corresponding to X_p from the q^{th} video frame (the original formulation was referred to the q^{th} camera), $\hat{\mathbf{P}}_q$ is the projective matrix of the q^{th} frame, $\hat{\mathbf{P}}_q \cdot X_{pq}$ is the reprojection of X_p onto the q^{th} frame, and $d(\bullet)$ is the distance measurement between the reprojected locations and the observed locations in the image. This nonlinear least-square problem is solved by the Levenberg-Marquardt algorithm.

4.1.b Human Detection

For human detection, we try to use some existing pre-trained human detectors, for comparative study, to detect humans in the video frames, i.e., the HOG-based human detector [45], the ISM human detector [48], the DPM human detector [46], and the C^4 human detector [43]. These human detectors can be embedded independently in the proposed system, so as to functionally perform human detection. To efficiently find the targets, we start to track a target when it has been detected in three consecutive frames; otherwise, the detections are regarded as false alarm. Furthermore, saliency detection is first processed by the detected human blobs [109], and then morphological operations (closing + opening + closing with 3×3 structuring elements) are executed to accurately obtain the segmentations. However, if the executed saliency detection is not sufficient, we then create an ellipse mask as the segmentation. To determine the saliency detection is sufficient, we calculate the ratio of the saliency area over the whole area of the human blob. If the ratio is larger than a threshold (0.7 in our work), this saliency detection is regarded as sufficient.

4.1.c Ground Plane Estimation

Due to unpredictability of road conditions, a ground plane estimated in the beginning may not be applicable for the entire video sequence. Therefore, the ground plane needs to be continuously re-estimated based on the updated camera motions from the SfM. Since the noise produced by camera calibration usually has an adverse impact on the ground plane estimation, we re-estimate the ground planes in every f_g frames, $f_g \leq f_s$. Therefore, we collect f_g ground planes, each calculated by a pair of consecutive camera motions, to form a set of ground planes $\{(\mathbf{g}_q, \psi_q)\}$, where $\mathbf{g}_q \in \mathbb{R}^3$ is the normal vector of the ground plane and $\psi_q \in \mathbb{R}$ is the offset of the plane. Finally, we combine them into a single 4-by-f_g matrix $\mathbf{D}$,

$$\mathbf{D} = \left[(\mathbf{g}_q, \psi_q)^{\mathrm{T}} \quad \cdots \quad (\mathbf{g}_{q+f_g}, \psi_{q+f_g})^{\mathrm{T}} \right]. \tag{21}$$

Note that some $\{(\mathbf{g}_q, \psi_q)\}$ may be unreliable due to varying road conditions and noisy camera calibrations.

To recover the uncorrupted version from the corrupted matrix **D**, we employ Robust Principle Component Analysis (RPCA) [110] to extract a low-rank 4-by-f_g matrix **A** from **D**. Thanks to the characteristics of RPCA, the low rank matrix **A** consists of the uncorrupted data which represent more reliable ground planes for the application. The mean vector of the matrix **A**, annotated by $(\mathbf{g}_m, \psi_m)$, is considered to be our final ground plane for those f_g consecutive frames and is more resilient to the existence of noise in the system. Figure 4.2 shows an example of using a ground plane set $\{(\mathbf{g}_q, \psi_q) | q = 1, \ldots, f_g\}$ to estimate the final ground plane $(\mathbf{g}_m, \psi_m)$.

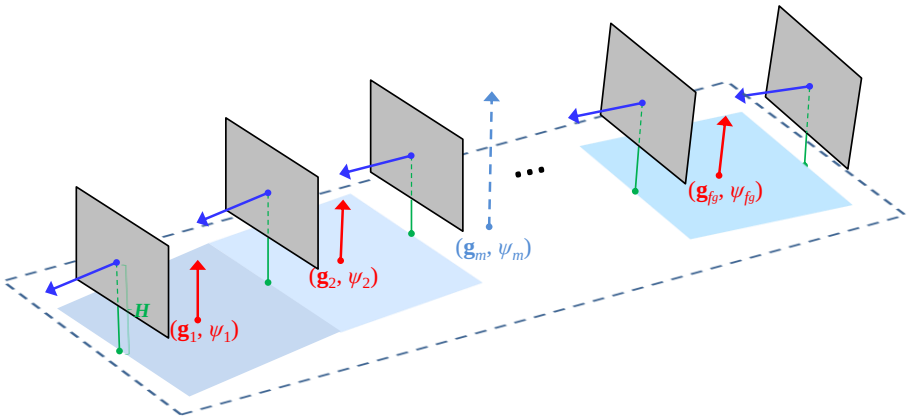


Figure 4.2: An example of ground plane estimation. Gray planes are the video frames, and H is the height of the camera. The final ground plane for f_g ground planes (dot-line plane) is obtained by a set of ground planes (solid planes).

4.2 Depth CMK Human Tracking

Once we have obtained the 3-D locations of the humans from the pose estimation stage (see Figure 4.1), we apply the CMK tracking technique to track them. In other words, we associate the targets in the current frame with the detections in the next frame. First, for each target, the 3-D location of its candidate is predicted by Kalman filtering. Then, CMK tracking is used to relocate the candidate's 3-D location effectively by achieving minimum color distance. On the other hand, the depth information helps us to

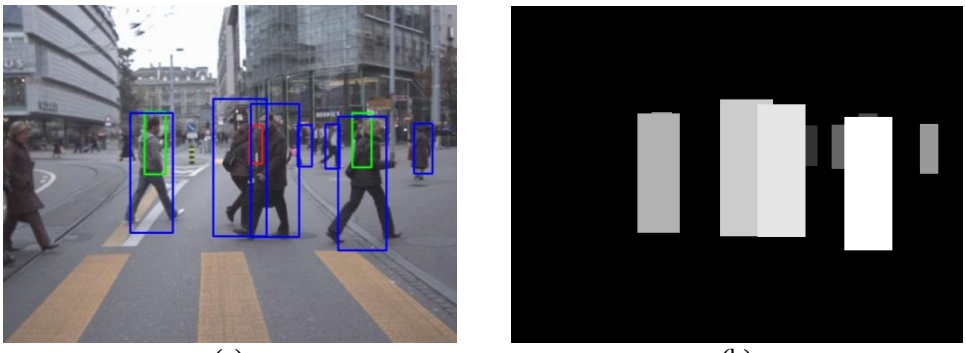
understand the relative 3-D locations between the targets, so that we are able to handle occlusion issues in the tracking. By efficiently combining depth information into the CMK tracking, the proposed system not only effectively tracks the humans, but also handles occlusions well.

4.2.a Depth Map Construction

Based on the 3-D locations of the humans, we can construct a depth map to describe the relative 3-D locations of all the tracked targets. Figure 4.3 shows an example of the depth map, where Figure 4.3(a) shows the result of human detection and Figure 4.3(b) is the corresponding depth map. The depth map depicts the relative distance between the camera and the detected humans; higher (brighter) intensity means the detected humans are closer to the camera. As shown in Figure 4.3(a), two green-boxed targets are severely occluded by the other targets in front of them, while a red-boxed target is totally occluded by other targets. Thanks to the depth map, we can approximately evaluate if the i^{th} target is occluded or not, in terms of the visibility $v_i \in [0,1]$:

$$v_i = \frac{\text{visible area of } i^{th} \text{ target}}{\text{total area of } i^{th} \text{ target}}. \tag{22}$$

If $v_i = 1$, it implies the i^{th} target is totally visible without being occluded by other targets; if $0 < v_i < 1$, it means the i^{th} target is partially occluded; otherwise, it is totally occluded.



(a) (b)

Figure 4.3: An example of the depth map, showing (a) detections and (b) depth map, where higher intensity indicates detected humans are closer to the camera.

4.2.b Problem Formulation

In [8], the CMK tracking scheme tracks video objects in 2-D space (image), i.e., $\mathbf{x} \in \mathbb{R}^{2 \cdot N_k}$ in Eq. (4). To efficiently integrate the depth information into the CMK framework, we need to reformulate the problem. First we extend Eq. (4) from 2-D to 3-D space,

$$J^i(\mathbf{X}) = \sum_{\kappa=1}^{N_\kappa} w_\kappa^i \cdot J_\kappa^i(\mathbf{X}), \quad \mathbf{X} \in \mathbb{R}^{3 \cdot N_\kappa}. \tag{23}$$

This equation is regarded as the local optimization for each individual target i with multiple kernels. Second, considering the depth information, we assign the visibility of each target as a weight to deal with the global optimization. In other words, the total cost function becomes:

$$J(\mathbf{X}) = \sum_{i=1}^{N_q} v_i \cdot J^i(\mathbf{X}) = \sum_{i=1}^{N_q} v_i \cdot \left(\sum_{\kappa=1}^{N_\kappa} w_\kappa^i \cdot J_\kappa^i(\mathbf{X}) \right), \tag{24}$$

where N_q is the number of the targets in the q^{th} video frame, $\mathbf{X} \in \mathbb{R}^{3 \cdot N_q \cdot N_\kappa}$, and $\mathbf{X} = \left[(\mathbf{X}_1^1)^T \quad \cdots \quad (\mathbf{X}_\kappa^i)^T \right]^T$ for the i^{th} target and the κ^{th} kernel.

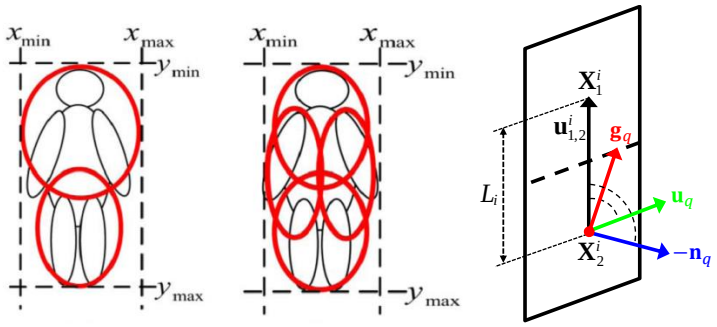


Figure 4.4: Layout for (a) 2-kernel and (b) 4-kernel, both in 2-D space. (c) Illustration of the 3-D based constraints in case of 2-kernel layout.

Necessarily, the constraint functions $\mathbf{C}(\mathbf{X}) = \mathbf{0}$ must be considered to maintain the relative locations of the kernels. In [8], 2-kernel and 4-kernel layouts are proposed to describe a human, as shown in Figure 4.4(a) ,(b). Unlike the constraints used in [8],

which are mainly based on 2-D geometry, we set the constraints based on 3-D geometry. Without loss of generality, here we only discuss the 2-kernel case as shown in Figure 4.4(c), but the idea can be easily extended to the 4-kernel case. To represent a target in 3-D space, we define a *target plane* $(-\mathbf{n}_q, \pi_q^i)$ for the i^{th} target in the q^{th} frame; where $\mathbf{n}_q$ is the normal vector of the q^{th} frame, and π_q^i is the offset of the target plane. To properly set the constraints, we start to calculate two auxiliary vectors, $\mathbf{u}_q = -\mathbf{n}_q \times \mathbf{g}_q$ and $\mathbf{u}_{1,2}^i = \mathbf{X}_1^i - \mathbf{X}_2^i$. First, the distance between two kernel centers should be the same, which implies

$$\left\| \mathbf{u}_{1,2}^i \right\|^2 = (L_i)^2 \text{, for the } i^{\text{th}} \text{ target,} \tag{25}$$

where L_i is the initial distance between $\mathbf{X}_1^i$ and $\mathbf{X}_2^i$. Second, both the angle between $\mathbf{u}_q$ and $\mathbf{u}_{1,2}^i$, and the angle between $-\mathbf{n}_q$ and $\mathbf{u}_{1,2}^i$, should be consistent. The constraint with angle $\phi_{q,i}$ limits right-left movement of the kernels, as shown in Figure 4.5(a); while the constraint with angle $\varsigma_{q,i}$ limits forward-backward movement of the kernels, as shown in Figure 4.5(b). Therefore, we can have

$$\begin{cases} \dfrac{\mathbf{u}_q \cdot \mathbf{u}_{1,2}^i}{\| \mathbf{u}_q \| \| \mathbf{u}_{1,2}^i \|} = \cos\left(\phi_{q,i}\right) \\ \dfrac{-\mathbf{n}_q \cdot \mathbf{u}_i^{1,2}}{\| -\mathbf{n}_q \| \| \mathbf{u}_{1,2}^i \|} = \cos\left(\varsigma_{q,i}\right) \end{cases} \text{, for the } i^{\text{th}} \text{ target in the } q^{\text{th}} \text{ frame.} \tag{26}$$

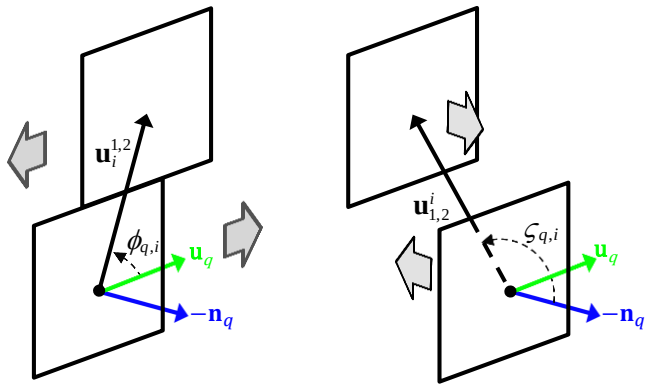
Those constraints can bind the kernels with each other in 3-D space during tracking.

Hence, for each target *i*, the movement vector $\delta_{\mathbf{X}}$ can be iteratively solved by using the projected gradient method:

$$\delta_{\mathbf{X}} = \alpha(-\mathbf{I} + \mathbf{C}_{\mathbf{X}}(\mathbf{C}_{\mathbf{X}}^T\mathbf{C}_{\mathbf{X}})^{-1}\mathbf{C}_{\mathbf{X}}^T)\mathbf{V}\mathbf{W}\mathbf{J}_{\mathbf{X}} + (-\mathbf{C}_{\mathbf{X}}(\mathbf{C}_{\mathbf{X}}^T\mathbf{C}_{\mathbf{X}})^{-1}\mathbf{C}(\mathbf{X})), \tag{27}$$

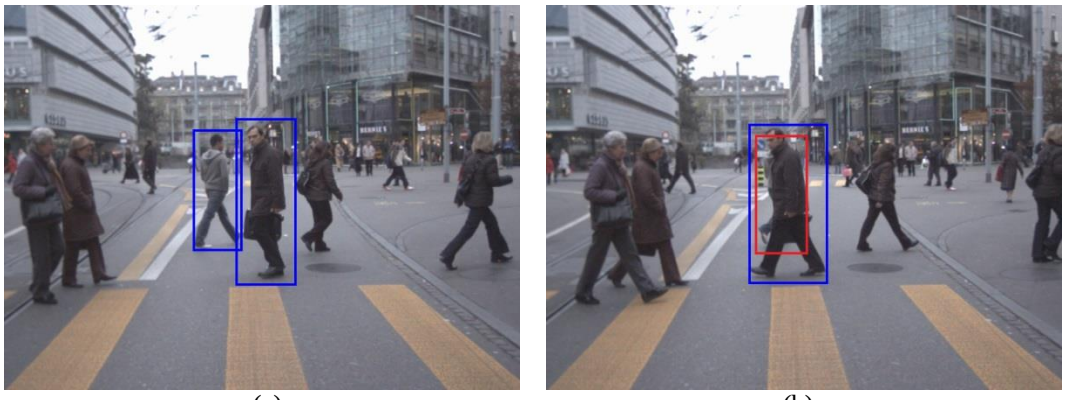
where $\mathbf{C}(\mathbf{X}) = \left[c_1(\mathbf{X}) \ \cdots \ c_m(\mathbf{X}) \right]^T$ is the matrix including m constraint functions, and $c_j(\mathbf{X}) : \mathbb{R}^{3 \cdot N_q \cdot N_k} \to \mathbb{R}$ is the j^{th} constraint function, $\mathbf{V} = \begin{bmatrix} v_1\mathbf{I}_v & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & v_{N_q}\mathbf{I}_v \end{bmatrix}$, $\mathbf{I}_v$ is an $3N_k \times 3N_k$ identity

matrix, $\mathbf{W} = \begin{bmatrix} w_1\mathbf{I}_w & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & w_{N_k}\mathbf{I}_w \end{bmatrix}$, $\mathbf{I}_w$ is an $3N_q \times 3N_q$ identity matrix, and $\mathbf{J_X}$ is the gradient vector of the total cost function with respect to $\mathbf{X}$.



(a) (b)

Figure 4.5: Constraints for binding 2 kernels in 3-D space along (a) left-right direction, and (b) forward-backward direction.



(a) (b)

Figure 4.6: An example of the hypothesized association in (a) frame 230 and (b) frame 233; the blue boxes represent the detections, and the red box is the inserted hypothesized association.

4.2.c Hypothesized Association

However, due to unreliable detection or occlusion, humans may not be detected for several frames. Hence, some tracked targets cannot be successfully associated with the detections in the subsequent frame. To consistently track a non-associated target, we insert a *hypothesized association* which has been located by the CMK tracking with the

best color similarity. Inserting a hypothesized association not only improves the detection rate but also helps to continuously track the targets. Figure 4.6 shows an example of the hypothesized association in case of occlusion. The person wearing gray clothes is detected in frame 230, but is then fully occluded by another person 3 frames later. By 3-D information, we can predict his 3-D location and understand that he is occluded. Hence, a hypothesized association is used here to represent a possible detection, as shown by the red box in Figure 4.6(b).

4.3 Experimental Results

In this section, we show experimental results of the proposed system on the ETH Mobile Scene (ETHMS) dataset [56], which is very challenging and difficult because of the heavy occlusions and busy crowd in the camera views. Since the proposed system is developed for the application of a monocular camera, we only use the left view sequences of the dataset. In our experiments, the proposed system is evaluated by its detection performance and tracking performance. Furthermore, we compare our results with three state-of-the-art methods [56],[63],[64]. In order to have a fair comparison of our proposed scheme with previous work, we use the same video sequences as used in their simulations. All the experiments are processed on a personal computer with a P4 2.67GHz CPU and 2G DDR. The implementation is constructed in C/C++, and the experimental settings are described as follows. In the SfM framework, the proposed system adopts Harris corners as the features, which are tracked by a KLT tracker. Here we assume the height of the camera is known. In the human detection, the pre-trained detectors [43]–[45] are functionally applied to the proposed system to detect humans within 30 meters. In the CMK tracking, K-L distance is used for all distance measures, and an 8-bin histogram of the object is constructed based on the HSV color space with a roof kernel. The proposed approach is compared to the following three approaches. The approach in [56] is a stereo algorithm based on a graphical model. The approach in [64] is a dynamic programming algorithm based on flow network framework, with and

without the Non-Maxima Suppression (NMS) in it. The C2K is the proposed approach with 2 kernels and C4K is for 4 kernels.

Table 4.1: Comparison of Detection Rate and FPPI

Method	Detector	Detection rate (%)	False Positive Per Image (FPPI)
[56]	ISM	47	1.5
[56]	HOG	67.5	1.0
[64]	DPM	49.53	0.93
[64] + NMS	DPM	49.94	0.93
C4K	HOG	53.28	1.32
C4K	DPM	59.74	1.24
C2K	HOG	70.61	0.97
C2K	C^4	62.36	1.14
C2K	DPM	75.58	0.89

4.3.a Detection Performance

The competing approaches are evaluated with different human detectors, in terms of the detection rate and false positives per image (FPPI). The results of the ETHMS dataset are shown in Table 4.1. As we can see, the approaches with DPM achieve better performance than that with other detectors, indicating that DPM provides more robust detections to facilitate better tracking. The results show that both the proposed approach (C2K, C4K) and the approach in [56] are superior to the approach in [64]. Since both approaches further utilize the 3-D information, instead of 2-D information only in [102], they can effectively handle occlusion issues. When comparing C2K with C4K, C2K performs much better than C4K because C2K has better tracking performance, which results in increasing the detection rate and decreasing the FPPI. The detection rate of the proposed approach can achieve about 75%, owing to proper insertion of hypothesized associations and successive tracking. This implies that missed detections can be improved by the tracking techniques, and also better detection results benefit the tracking performance.

4.3.b Tracking Performance

To evaluate the tracking performance, we consider the following metrics which are widely used in the previous work [56],[63]:

- Most Tracked trajectories (MT): the number of trajectories that successfully tracked more than 80% of frames in a video sequence.
- Partially Tracked trajectories (PT): the number of trajectories that successfully tracked between 20% and 80%.
- Most Lost trajectories (ML): the number of trajectories that successfully tracked less than 20%.
- FragMentation (FM): the number of times a trajectory is interrupted.
- ID Switches (IDS): the number of times two trajectories switch their IDs.

The results of several sequences in dataset are shown in Table 4.2, where GT denotes the number of trajectories in the ground truth. Higher MT, lower FM, and lower IDS imply better performance. From the MT results, the approach in [64] is not as good as other approaches. Due to the limitation of 2-D information, the approach in [64] cannot solve occlusion issues well, thus resulting in lower MT and higher FM. On the other hand, the 3-D based approaches are able to improve the performance significantly by efficiently handling occlusions. C2K performs better than C4K, because the detected humans are too small to be clearly described by 4 kernels, which also include some additional background regions so that the tracking is often impacted by the background. Furthermore, there are more constraints for binding 4 kernels, making the optimization of the cost function more divergent. When comparing C2K with the approach in [56], although the MT/PT/ML results are comparable, the FM results of the CMK are much better than that of [56]. This implies that the proposed 3-D based CMK framework can effectively associate the targets, so as to perform well for successively tracking the targets. Several visual tracking results are shown in Figure 4.7, where (a), (b), (c), (d) are the ETHMS dataset, and (e), (f), (g) are the datasets recorded by ourselves. The results

show favorable performance of the proposed system, not only successively tracking humans but also handling occlusion in the tracking. Moreover, since relative 3-D locations of the humans are obtained with GPS information, we can also construct 3-D visualization of the dynamic scenes. Figure 4.8 shows the 3-D visualization of the dataset in Figure 4.7(f) and (g), showing the dynamic scenes in different aspects of views. All the videos associated with the simulations reported in this paper can be viewed on our website[2].

4.3.c Tracking in Unmanned Ground Vehicle

The moving platform used in our experiments is an automocous ground robot programmed under a Robot Operating System (ROS). The robot is equipped with a 1.6GHz CPU and WiFi connection. To achieve better real-time performance, the video streams are wirelessly sent to a nearby laptop computer with a 3.0GHz CPU to remotely compute the tracking algorithm, and then feedback the commands to the robot. The scenario is that the robot tracks a single person who is first detected and then continuously tracked/followed by the robot. Figure 4.9 shows the visual results from the perspective views of the robot. When the tracked person is partially occluded by the other person, the robot can still continuously track the same person.

4.3.d Limitations and Discussion

First, the proposed approach adopts the tracking-by-detection scheme to detect and then track humans; this implies that the approach highly relies on the detection results. However, if the quality of video sequences is not sufficient for the human detector(s), the proposed approach is not able to perform well on the poor detection results. More specifically, the tracking of a specific human is always triggered by the positive detection of a target. In other words, the proposed approach may not work well at night or in cases with insufficient lighting. Second, the proposed approach effectively estimates ground

[2] website: http://allison.ee.washington.edu/kuanhuilee/mpht

planes based on certain video frames when the platform moves on flat roads/fields, but will produce less reliable estimation if the roads/fields are severely bumpy, resulting in larger error of the object back-projection and impacting accuracy of the reprojected 3-D information. Hence, the proposed approach is not reliable for unmanned aerial vehicles, because the height dynamically changes, which then yields unreliable 3-D information of humans.

Table 4.2: Tracking Performance

Dataset	Method / Detector	GT	MT	PT	ML	FM	IDS
Seq#1	[56] / ISM	73	**66**	**5**	2	**8**	1
	[64] / DPM	73	54	13	6	19	8
	[64] + NMS / DPM	73	55	12	6	19	8
	C4K / DPM	73	58	10	5	7	2
	C2K / DPM	73	**64**	**7**	2	**3**	3
Seq#4	[56] / ISM	88	**74**	**8**	6	**20**	3
	[64] / DPM	88	52	16	20	29	14
	[64] + NMS / DPM	88	55	14	19	29	14
	C4K / DPM	88	64	14	10	18	6
	C2K / DPM	88	**71**	**9**	8	**11**	6
Downtown#1	[64] / DPM	5	**4**	**1**	0	**1**	0
	[64] + NMS / DPM	5	4	1	0	1	0
	C4K / DPM	5	5	0	0	0	0
	C2K / DPM	5	**5**	**0**	0	**0**	0
Downtown#2	[64] / DPM	38	10	10	15	2	1
	[64] + NMS / DPM	38	**10**	10	15	**2**	1
	C4K / DPM	38	12	9	16	1	0
	C2K / DPM	38	**13**	8	16	**1**	0
UWcamp#1	[64] / DPM	13	9	3	0	1	0
	[64] + NMS / DPM	13	**9**	3	0	**1**	0
	C4K / DPM	13	12	1	0	0	0
	C2K / DPM	13	**12**	1	0	**0**	0

GT: Ground Truth, **MT**: Most Tracked trajectories, **PT**: Partially Tracked trajectories, **ML**: Most Tracked trajectories, **FM**: FragMentation, **IDS**: ID Switches.

Figure 4.7: Visual tracking results, from the top to the bottom: (a) ETHMS Seq#1, (b) ETHMS Seq#2, (c) ETHMS Seq#3, (d) ETHMS Seq#4, (e) Downtown Seq#1, (f) Downtown Seq#2, and (g) UWcamp Seq#1.

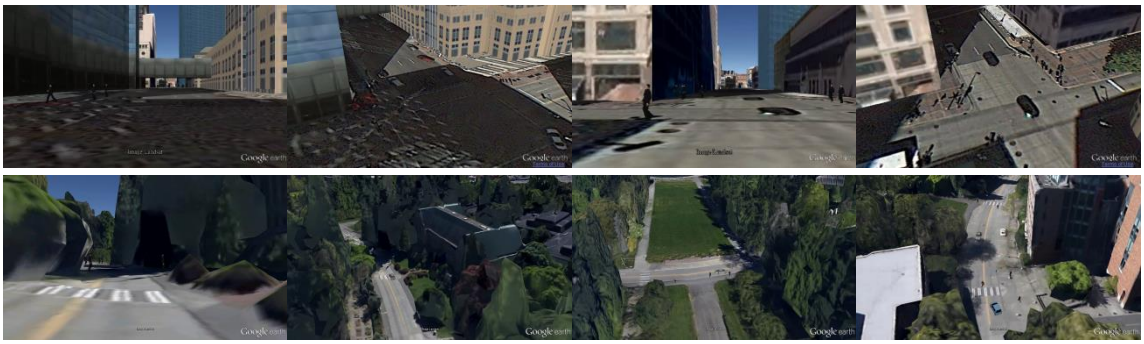


Figure 4.8: 3-D visualization reconstructed from the video sequences, showing different view aspects; top: Downtown Seq#2, bottom: UWcamp Seq#1.

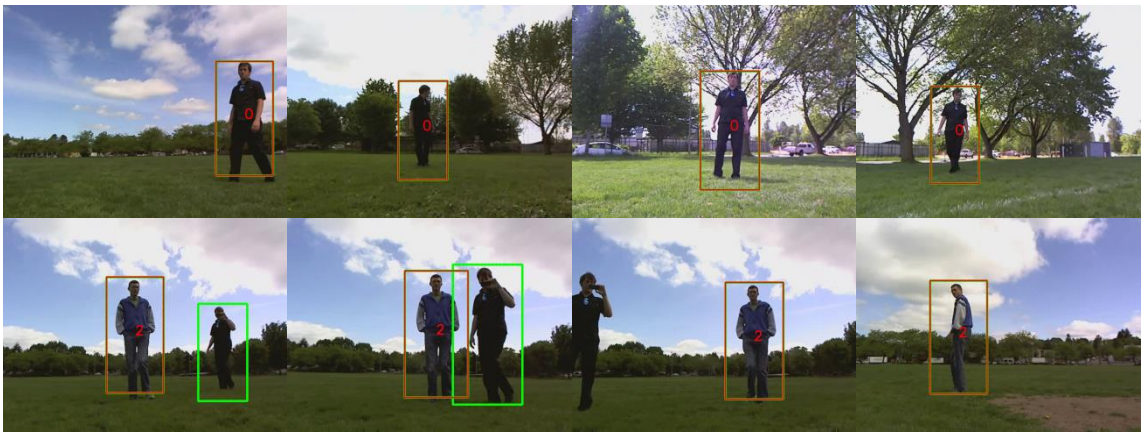


Figure 4.9: Visual tracking results from the perspective views from an autonomous ground vehicle.

Chapter 5 – Tracking across Multiple Moving Cameras

5.1 Overview

In this chapter, we deal with tracking across multiple driving recorders, which can also be extended to other types of moving cameras (such as mobile robots or flying drones), and propose a framework to track on-road pedestrians recorded in the videos [111]. We assume a cloud server is used to collect the driving information of the vehicles via a mobile surveillance network. First, pedestrian tracking in a single moving camera is applied to each video. Based on the single-camera-tracking results, we treat the problem of tracking across cameras as a multi-label classification task, which determines each target belonging to one or several cameras' FOVs by considering the association likelihood of the target as calculated based on the targets' motion cues and appearance features. When a target is out of the camera's FOVs, we predict the target's locations facilitated by an open map service such as *Google Maps*. Moreover, by using the *Google Earth* service, a 3-D visualization of a dynamic scene can be reconstructed for users to see a holistic view or different viewing perspectives of the 3-D scenes reconstructed by the multiple videos, as shown in Figure 5.11.

5.2 Framework Overview

In the proposed approach, we assume a mobile surveillance network where a server collects the driving information within a local area and a period of time $t = 1,\ldots,T$. The driving information includes (intrinsic) camera parameters, GPS, global timestamp and videos, which are synchronized by the global timestamps. Figure 5.1 shows the overview of the proposed framework. First, pedestrian tracking, which produces the moving trajectory and associated features of the tracked person (tracklet) in 3-D space, in a single camera is applied to each video. The videos are then used to build Brightness Transfer Functions (BTFs) for compensating color diversity of the cameras. After estimating the

pedestrians' 3-D tracklets in each camera, the pedestrian tracking across multiple cameras is further applied, facilitated with the BTFs and map prior. Finally, the pedestrians' 3-D tracklets are summarized and 3-D visualized in the 3-D real-world environment, based on an open map service, such as Google Earth.

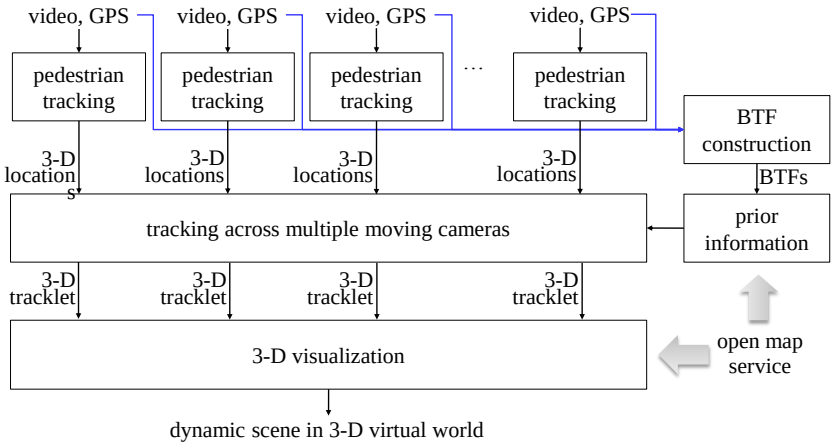


Figure 5.1: Overview of the proposed framework.

The pedestrian algorithm under a moving camera is based on the tracking-by-detection method in Chapter 3, which separately tracks pedestrians in a single moving camera. Once the pedestrians are successfully tracked in each moving camera, the profile of the tracked targets, including appearance features and motion cues, can be obtained.

5.3 Tracking across Multiple Moving Cameras

First, we describe the notations used in the proposed algorithm. Given N tracked targets in total by M moving cameras, the actual number of the distinct tracked pedestrians N' should be smaller than or equal to the total number tracked in the cameras, i.e., $N' \leq N = \Sigma_{j=1}^{M} N_j$, where there are N_j targets tracked in the j^{th} moving camera, since one pedestrian can appear in more than one camera's FOV. For the i^{th} tracked target, i = 1,…,N, the target profile O_i^t at timestamp t is composed of its appearance model and motion model. For the j^{th} moving camera, j = 1,…,M, the camera profile C_j^t at t is

composed of calibrated camera parameters $\mathbf{P}(C_j^t)$, GPS locations $gps(C_j^t)$, forwarding directions $dir(C_j^t)$, and a set of target profiles $\mathbf{O}(C_j^t)$, which are specifically derived profiles for all the targets appearing in the j^{th} camera's FOV. A tracklet l_i is a set which has the i^{th} target profiles for all possible t. $\mathbf{C}_\times$ is a set of camera profiles, containing the predicted intersections of the specific target and the cameras' FOVs at later t_j. Finally, p_{map} is a prior of map topology which describes routing information.

For each t (i.e., current timestamp), we check whether each tracked target is leaving the camera's FOV or not. If the i^{th} target leaves the j^{th} camera's FOV, implying the target either disappears forever or later enters into other cameras' FOVs, we then apply three steps to determine its later locations. 1) *Prediction*: By utilizing p_{map}, we can predict when the i^{th} target intersects with the j^{th} camera's FOV at later t_j, as shown in Figure 5.2(a). These intersections in terms of camera profiles are put into $\mathbf{C}_\times$. 2) *Classification*: Based on the appearance features and motion cues, we associate the leaving target O_i^t with every target which appears in the possible intersections (i.e., the elements of $\mathbf{C}_\times$), as shown in Figure 5.2(b). 3) *Interpolation*: If the leaving target is determined to appear in some cameras' FOVs, we uniformly interpolate the hypothesized target profiles into the tracklet of the associated target(s), as shown in Figure 5.2(c). More specifically, if the i^{th} target appears in the j^{th} camera's FOV, we associate O_i^t with the targets in one or multiple cameras by a classification operation, according to the previous target profile O_i^{t-1}. If the target appears in multiple cameras simultaneously, we apply the *Overlapping* operation which bundle-adjusts the targets 3-D locations from multiple views, as shown in Figure 5.2(d). Figure 5.3 shows an example of the proposed framework, where each row represents a tracklet and the horizontal axis is the time line. In the example, O_1^4 is classified to camera 1 and 3 at $t = 4$, so the overlapping operation is applied to O_1^4 and O_9^4; when $t = 8$, O_2^8 is predicted to appear in C_2^{10} and C_3^{21}, i.e., $\mathbf{C}_\times = \{C_2^{10}, C_3^{21}\}$. When $t = 10$, there is no possible candidate for O_2^8. At $t = 21$, the targets in camera 3 are classified

based on O_2^8, and O_8^{21} is regarded as a candidate. Hence, the hypothesized target profiles uniformly interpolated from O_2^8 to O_8^{21} are inserted into l_8.

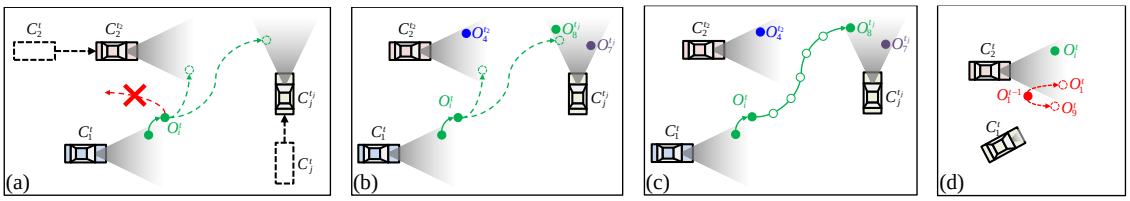


Figure 5.2: Four operations in the proposed framework. (a) prediction: O_i^t is predicted to appear in the camera 2 at t_2 and the camera j and t_j. (b) classification: only O_8^{tj} is associated with O_i^t. (c) interpolation: insert hypothesized target profiles (non-solid circles) from O_i^t to O_8^{tj} along the route provided by p_{map}. (d) overlapping: If a target associate with targets in more than one camera's FOV, we apply bundle adjustment to the targets.

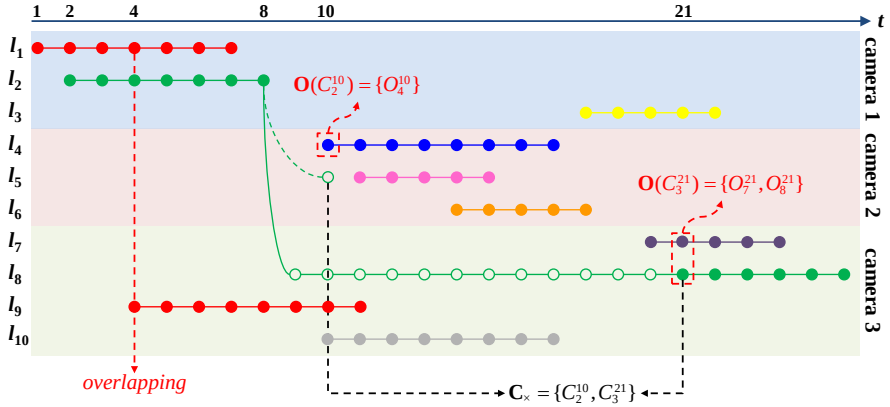


Figure 5.3: An example of the proposed framework in case of M = 3 and N = 10 (N' = 8 in this case), where each point is a target profile, l_1 and l_9 are identical, l_2 and l_8 are identical. The non-solid circles are the hypothesized target profiles

5.3.a Prediction

In the prediction, p_{map} is used to denote the topology information (i.e., routing path, transition time, and direction) with respect to GPS location. Based on p_{map}, a shortest route from one GPS location (gps_1) to another (gps_2) can be estimated, so that the transition time $t_s(gps_1, gps_2)$, and the forwarding direction of the route at gps_1, denoted by $dir_{map}(gps_1 \mid gps_2) \in \mathbb{R}^2$, can be obtained. In this paper, we adopt Google Maps service to generate p_{map}, by querying routing information between two GPS locations.

The goal of prediction is to find the possible intersections of the i^{th} target and the j^{th} camera's FOV in the near future. Given an O_i^t, in order to locate an intersection with the j^{th} camera, we try to find a later timestamp t_j, such that the transition time from $gps(O_i^t)$ to the FOV of $C_j^{t_j}$, denoted by t_o, is similar to the transition time from the $gps(C_j^t)$ to the $gps(C_j^{t_j})$, denoted by t_c. Consider that $gps(C_j^{t_j}, r_{fov} | O_i^t)$ is a GPS location which has the shortest route from $gps(O_i^t)$ to $C_j^{t_j}$'s FOV with the visible range r_{fov}, then t_o and t_c can be defined as

$$\begin{cases} t_o = t_s\left(gps(O_i^t), gps(C_j^{t_j}, r_{fov} | O_i^t)\right), \\ t_c = t_s\left(gps(C_j^t), gps(C_j^{t_j})\right). \end{cases} \tag{28}$$

Moreover, the estimated target's forwarding direction $dir(\bullet)$, as obtained from the results of single-camera-tracking, and the measured target's forwarding direction $dir_{map}(\bullet)$, as obtained from p_{map}, should be consistent, as shown by the red-dotted line in Figure 5.2(a). This results in a constraint to the selection of $C_j^{t_j}$:

$$\begin{gathered} \mathbf{C}_{\times} = \left\{ C_j^{t_j} \,\middle|\, |t_o - t_c| < \tau_t \right\} \\ \text{s.t. } dir(O_i^t) \cdot dir_{map}\left(gps(O_i^t) \,|\, gps(C_j^{t_j}, r_{fov} | O_i^t)\right) < \tau_{dir}, \end{gathered} \tag{29}$$

where τ_{dir} and τ_t are the thresholds to restrict the directions and transition time, respectively.

5.3.b Classification

A tracked target may appear in one or several cameras' FOVs either at the same time or at subsequent timestamps. Without loss of generality, we assume the task is to associate one tracked target with the other at a (later) different timestamp. Therefore, we treat this scenario as a multi-label classification task. To measure the likelihood of the presence of the i^{th} previously tracked target in the j^{th} camera's FOV at t, we calculate the association likelihood based on the targets' appearance and motion information,

respectively. Given an O_i^t to be classified and a reference profile O_i^{ref}, the association likelihood of the i^{th} target in the j^{th} camera's FOV is defined as

$$\ell_{i,j}^t = \max\left\{\ell_{app}(O_i^{ref}, O_k^t)\cdot\ell_{mo}(O_i^t, O_k^t) \mid \forall O_k^t \in \mathbf{O}(C_j^t)\right\}, \tag{30}$$

where ℓ_{app} is the likelihood of the targets' matching of appearance (see Section 5.4); ℓ_{mo} is the likelihood of the i^{th} target appearing in the j^{th} camera's FOV, and is defined as a Gaussian weighted function of the distance between two targets:

$$\ell_{mo}(O_i^t, O_k^t) = \rho\cdot G_{0,\sigma_{mo}}\left(X(O_i^t) - X(O_k^t)\right), \tag{31}$$

where $G_{0,\sigma_{mo}}(\bullet)$ is the a standard Gaussian function with μ = 0 and $\sigma = \sigma_{mo}$, ρ is a normalization value, and $X(\bullet)$ is the 3-D location of the assigned target profile.

Thus, regarding each camera's FOV as a class, we tend to classify the i^{th} target at each t, by formulating the problem as a Quadratic Boolean Problem (QBP):

$$\max_{\mathbf{v}_i^t}(\mathbf{v}_i^t)^{\mathrm{T}}\mathbf{L}_i^t(\mathbf{v}_i^t),\ \mathbf{L}_i^t = \begin{bmatrix} l_{11}^t & \cdots & l_{1M}^t \\ \vdots & \ddots & \vdots \\ l_{M1}^t & \cdots & l_{MM}^t \end{bmatrix}, \tag{32}$$

where $\mathbf{v}_i^t = [v_1,\ldots,v_M]^{\mathrm{T}}$ is a vector of indicator variables such that $v_j = 1$ if the target appears in the j^{th} camera's FOV, and $v_j = 0$ otherwise. The diagonal elements of $\mathbf{L}_i^t$ are $\ell_{i,p}^t$, as defined in Eq. (30), i.e., $l_{pp}^t = \ell_{i,p}^t$, denoting the association likelihood of the i^{th} target appearing in the p^{th} camera's FOV, while the rest of the elements are the association likelihood of the targets which correspond to the i^{th} target in the p^{th} and the q^{th} cameras' FOVs separately, i.e., $l_{pq}^t = \ell_{i,p}^t\cdot\ell_{i,q}^t - \eta, p \neq q$; η is the penalty for non-correspondence of two targets. Such a problem can be solved by standard optimization methods [112].

In general, the dimension of $\mathbf{v}_i^t$ is M (the total number of the cameras). However, larger M may make $\mathbf{L}_i^t$ a sparse matrix, resulting in divergence of the problem and higher

computational cost. To lower the dimension of $\mathbf{v}_i^t$, we only select the cameras whose 3-D locations are within the visible range r_{fov}:

$$\forall C_k^t \text{ s.t. } \left\| X(O_i^t) - X(C_k^t) \right\|_2 < r_{fov}, \tag{33}$$

where $X(\bullet)$ is the 3-D locations of the specific profile.

5.3.c Interpolation

Once the prediction of the i^{th} tracked target intersecting with the j^{th} camera at a later timestamp t_j is confirmed, we insert the hypothesized target profiles $O_i^{t'}$ into l_i, from t to t_j. All $O_i^{t'}$ are constructed by the appearance models of O_i^t, and their 3-D locations (as well as GPS locations) are uniformly interpolated along the routes, as provided by Google Maps service. If the predicted target intersects with multiple cameras' FOVs, then the disappeared (exited) target will later enter into two or more cameras' FOVs at different t_j. Therefore, multiple hypothesized tracklets are inserted into the corresponding l_i.

5.3.d Overlapping

If O_i^t appears in multiple cameras' FOVs at t, we apply a standard bundle adjustment formulation to optimize the 3-D locations of the O_i^t in multiple cameras' FOVs. The set of 3-D points and the corresponding 2-D points are used in the bundle adjustment [69] process to iteratively minimize the total reprojection error:

$$\hat{X}(O_i^t) = \underset{X(O_i^t)}{\arg\min} \sum_{C_j^t \in \mathbf{C}_*} d_{proj}\left(\mathbf{P}(C_j^t) \cdot X(O_i^t \mid C_j^t), x_{i,j}\right), \tag{34}$$

where $x_{i,j}$ is the observed 2-D location (i.e., middle of bottom of the pedestrian blob) corresponding to $X(O_i^t \mid C_j^t)$ from the j^{th} camera's FOV, $\mathbf{P}(C_j^t)$ is the projective matrix of the j^{th} camera at t, $\mathbf{C}_*$ is a set of C_j^t which all include $O_i^t \in \mathbf{O}(C_j^t)$ for all j, and $d_{proj}(\bullet)$ is the distance measurement between the reprojected locations and the observed locations in the image. Such a nonlinear least-square problem is solved by the Levenberg-

Marquardt algorithm. Figure 5.4 simply illustrates the idea of the bundle adjustment in the overlapping operation.

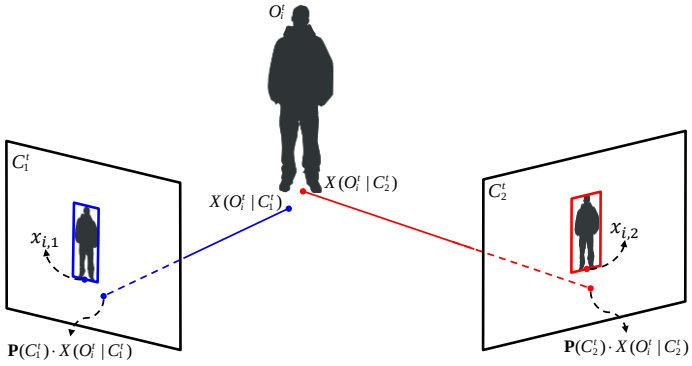


Figure 5.4: An illustration of the bundle adjustment in the overlapping operation, where the estimated 3-D location will be optimized by minimizing the reprojection error.

5.4 Pedestrian Association Likelihood

In this section, we describe the details of $\ell_{app}(\bullet,\bullet)$, i.e., the association likelihood of two targets according to their appearance features. The appearance features, such as color and texture, are commonly used to represent the targets. However, due to viewing perspectives and diversity of camera devices, a target's color-channel image intensities extracted from one camera are normally different from those of the other camera. Therefore, the Brightness Transfer Function (BTF) [83] between two cameras must be applied before the feature matching.

5.4.a Brightness Transfer Functions

Generally, the BTFs can be estimated through overlapping FOVs of cameras. Our assumption is that the lighting conditions of cameras are roughly consistent within a short period of time in the same vicinity. Based on this assumption, we utilize "spatially" overlapping cameras' FOVs, i.e., a camera's FOV overlaps with the other cameras' FOVs, but not necessarily simultaneously. To calculate the BTF between two cameras, we first group the camera views (i.e., video frames) by applying mean-shift clustering to the GPS locations of the cameras. If a cluster includes more than one camera, the cluster is

regarded as overlapping. Hence, for any two cameras within an overlapping cluster G_g, we can have a set $\mathbf{F}_g^{p,q}$ containing pairs of (C_p^t , $C_q^{t'}$), where $gps(C_p^t)$ is the nearest location to $gps(C_q^{t'})$:

$$\mathbf{F}_g^{p,q} = \left\{ (C_p^t, C_q^{t'}) \,|\, \min\left(d_{phy}\left(gps(C_p^t), gps(C_q^{t'}) \right) \right) \right\}, \tag{35}$$

where $d_{phy}(\bullet)$ is the physical distance between two GPS locations. SIFT-feature matching is used to obtain all pairs of matching points, which are then used to estimate the BTF from the p^{th} camera to the q^{th} camera within the g^{th} cluster, denoted by $f_{BT,g}^{p\to q}$. Based on the color histograms calculated by the matching points, we then apply the RANSAC algorithm to obtain the optimal $f_{BT,g}^{p\to q}$.

On the other hand, if there are no spatially overlapping cameras' FOVs, we use an identity matrix as the BTF, i.e., $f_{BT,g}^{p\to q} = \mathbf{I}$. This issue will be further discussed in Section 5.5.d.

5.4.b Appearance Features

To evaluate the association likelihood of the pedestrians, the proposed framework adopts appearance (low-level) features. Many appearance features have been developed and have shown good performance in different applications. Color information is widely used and is considered to generate high-impact features in general cases ([75]–[77],[80],[114]). Some approaches additionally consider shape ([80],[81]) and texture ([77]–[79],[115],[116]), to improve the performance in case of large color variations. Recently, many patch-based/local features ([75]–[80]) have attracted much attention for the effective use of spatial information. From pedestrian images, lots of local patches are extracted to describe regional features, which have the advantage of invariance to misalignment, pose variation, and the change in viewpoint.

Assume the appearance features of a pedestrian are A_κ, κ = 1,2,…,K, and the κ^{th} appearance feature can be extracted from the object profile, which is first transformed by

the corresponding BTF, i.e., $A_\kappa\left(f_{BT,g}^{p\to q}(O_i^t)\right)$, where $f_{BT,g}^{p\to q}$ is the BTF from the p^{th} camera to the q^{th} camera and O_i^t is within the g^{th} cluster. If the g^{th} cluster is not overlapping (i.e., no overlapping FOVs are grouped), the BTF from the p^{th} camera to the q^{th} camera closest to the g^{th} cluster is selected. Hence, the association likelihood of the κ^{th} appearance features, denoted by $\ell_{app,\kappa}(A_\kappa, A'_\kappa)$, is defined as the similarity between A_κ and A'_κ. By taking multiple appearance features into account, a total likelihood for the feature matching is defined by the linear combination of the selected appearance features:

$$\begin{aligned}\ell_{app} &= w_1\cdot\ell_{app,1} + w_2\cdot\ell_{app,2} + \cdots + w_K\cdot\ell_{app,\kappa} + b \\ &= \mathbf{w}\cdot\mathbf{l}_{app} + b\,,\end{aligned} \tag{36}$$

where $\mathbf{w}\in\mathbb{R}^\kappa$ consists of the weights for the appearance features and represents the impact factor of the corresponding features; $\mathbf{l}_{app}\in\mathbb{R}^\kappa$ is a vector of the association likelihoods; and b is the offset of hyperplane.

By integrating some well-developed features, the proposed framework can take advantage of several appearance features' properties, to achieve good performance when tracking across multiple moving cameras. However, some features are mutually complementary and some are correlated with each other. Hence, how to select useful features is quite important for the proposed framework. The appearance feature selection is discussed in Section 5.5.a.

5.4.c Interior Training

To determine the corresponding weights for the selected appearance features (i.e., **w** in Eq. (36)), a training stage is necessary. However, since the FOV dynamically changes when a camera is moving, there is no chance to label the ground truth of the corresponding targets. To solve this problem, we apply an interior training scheme to determine **w**. The idea is to label positive and negative training data sets within the target profiles, based on their spatial-temporal relationship. More specifically, two target profiles belonging to the same tracklet are labeled as positive because they have been

successively tracked in a single moving camera. In contrast, if two targets appear in one camera's view simultaneously, these two target profiles can not be the same target, i.e., ℓ_{mo} is smaller than a threshold. Hence, these two target profiles are labeled as negative. Next, we use a standard linear Support Vector Machine (SVM) to obtain the **w**, to be used in Eq. (36). Finally, to represent the output of the SVM as a likelihood, we apply a sigmoid function to normalize ℓ_{app} [113].

5.5 Experimental Results

Several experiments are conducted to demonstrate the performance of our proposed framework. To the best of our knowledge, there is no public dataset specific for the case of multiple moving cameras based on driving recorders. Accordingly, we recorded videos and built test datasets ourselves. The configurations of the devices and datasets are shown in Table 5.1, and the recorded videos are pre-synchronized. To evaluate the tracking performance, we consider the following metrics which are widely used in the previous work [56], [63], [108]:

- Most Tracked tracklets (MT): the number of tracklets that successfully tracked more than 80% of frames across all video sequences.
- Partially Tracked tracklets (PT): the number of tracklets that successfully tracked between 20% and 80%.
- Most Lost trajectories (ML): the number of tracklets that successfully tracked less than 20%.
- FragMentation (FM): the number of times a tracklet is interrupted. Furthermore, we evaluate the FM in each single camera (sFM) and across multiple cameras (mFM), respectively.
- ID Switches (IDS): the number of times two tracklets switch their IDs, in a single camera (sIDS) and across multiple cameras (mIDS).

Table 5.1: Configurations of the Devices and Datasets

Device	Type	Resolution	fps
1	PAPAGO P2	1280×720	30
2	Pro V DV-2021	640×480	30
3	DOD F500LHD	1280×720	30
4	PAPAGO P2	1280×720	30

Dataset	Devices	*T*	Detection rate (%)	False Positive Per Image (FPPI)
A	1,2	4015	64.78	0.244
B	1,2	2042	79.33	0.127
C	1,2,3,4	4169	75.11	0.103

5.5.a Appearance Feature Selection

The proposed framework can use multiple appearance features for calculating the pedestrian association likelihood. To select the proper appearance features, we tested many combinations of different appearance features to determine the most effective features for the proposed framework. We tested the selected appearance features with three datasets, and measured the performance in terms of average accuracy of the pedestrian matching. Moreover, since background subtraction cannot be applied in the moving camera scenarios, no explicit segmentation masks can be created; we thus created an ellipse bounding mask for each tracked target when calculating the appearance features.

The features selected (shown in Table 5.2) include color information such as the weighted color histogram (WCH) and maximally stable color regions (MSCR) [114], texture information such as scale invariant feature transform (SIFT) [115] and local binary pattern (LBP) [116], and the patch-based/local descriptors such as recurrent high-structured patches (RHSP) [75]. First, we select color information since it is widely used and is considered to be a high-impact feature. From Figure 5.5, the average accuracy of using WCH is 68%, and the accuracy of using MSCR is 66%; which implies color information performs sufficiently well for the matching in our scenarios. Then, we further add texture features by testing the selections no.3 (WCH+LBP) and no.4 (WCH+SIFT),

and the average accuracy increased to 80% and 79%, respectively. This implies that texture can further improve the matching performance when combined with the WCH. Next, we consider the patch-based features by testing the RHSP combined with the color and texture features. As shown in the figure, the average accuracy of using selection no.5 (WCH+RHSP) is 72%. The accuracy of using selection no.7 (WCH+LBP+RHSP) is 77%, which is worse than that of using selection no.3 (WCH+LBP). This means that the RHSP cannot improve the matching performance in our scenarios. We also tested selection no.9 (WCH+MSCR+RHSP), which is known as the symmetry-driven accumulation of local features (SDALF) [75]; however, the result (82%) is still not impressive. This is because such patch-based descriptors highly rely on accurate segmentations of the pedestrian blobs, which cannot be easily achieved in the moving cameras. Finally, we recall the MSCR, which contains both color information and regional description. As shown in the results, selection no.11 (WCH+MSCR+SIFT) can achieve average accuracy of 89% and selection no.12 (WCH+MSCR+LBP) can reach 93%; both can achieve much better performance than other selections. This means that global features (WCH and LBP) mainly contribute to the results, and regional features (MSCR and SIFT) are necessary to enhance the performance. Finally, to better understand the influence of segmentation performance, given the segmentations of the pedestrian blobs which are manually refined, we tested the selections which have RHSP. The results show that the performance of the RHSP has significant improvement when the manual segmentations are used. The average accuracy of selection no.6 (WCH+RHSP) is 83%, selection no.8 (WCH+LBP +RHSP) is 87%, and the SDALF is up to 94%. This demonstrates that the well-developed features can be applied to the proposed framework under proper configurations to obtain better performance.

In summary, the selected features should include global features such as color and texture information. Moreover, regional features are also necessary for improvement of the tracking performance. The patch-based features can be taken into account if accurate segmentations of the pedestrians' blobs are available.

Table 5.2: Combinations of feature selections

No.	Feature Selections	Segmentations
1	WCH	NO
2	MSCR	NO
3	WCH+LBP	NO
4	WCH+SIFT	NO
5	WCH+RHSP	NO
6	WCH+RHSP	YES
7	WCH+LBP+RHSP	NO
8	WCH+LBP+RHSP	YES
9	WCH+MSCR+RHSP (**SDALF**)	NO
10	WCH+MSCR+RHSP (**SDALF**)	YES
11	WCH+MSCR+SIFT	NO
12	WCH+MSCR+LBP	NO

WCH: Weighted Color Histogram, **MSCR:** Maximally Stable Color Regions [114], **LBP:** Local Binary Pattern [116], **SIFT:** Scale-Invariant Feature Transform [115], **RHSP:** Recurrent High-Structured Patches [75], **SDALF:** Symmetry-Driven Accumulation of Local Features [75].

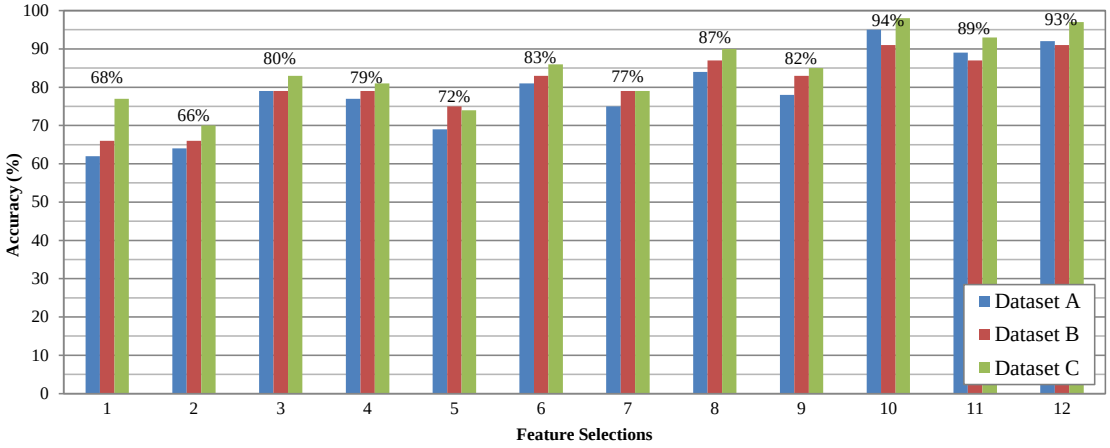


Figure 5.5: Comparison of matching accuracy of different selections.

5.5.b Tracking Results

We test three datasets (A, B, and C), with different scenarios, to demonstrate the performance of the proposed method. In dataset A, a pedestrian in one camera's FOV will leave for a while, and then enter into another (or the same) camera's FOV. Figure 5.6 shows that the pedestrians appearing in camera 1's FOV will later appear in camera 3's FOV. For example, pedestrian no.46 at t = 1128 in Figure 5.6(a) will appear at t = 2477 in Figure 5.6(b). In dataset B, a pedestrian in one camera's FOV, will also enter into

another camera's FOV at the same time. As shown in Figure 5.7, pedestrian no.3, no.7, and no.18 appear in both camera 1's and camera 3's FOVs almost at the same time. In dataset C, we simultaneously recorded four videos with four driving recorders; this complicated scene includes all the scenarios mentioned in the previous section. For example, Figure 5.8(a) shows a 2-camera overlapping case, while Figure 5.8(b) shows a non-overlapping case. For single-camera-tracking in the proposed framework, we adopt the constrained 2-kernel based method in [108], and compare with the flow network based association method in [64][3]. As for tracking across multiple cameras, we compare the proposed approach with the *M*vs*M* scheme, which is widely used for person re-identification [74],[75].

The tracking results are shown in Table 5.3, where GT denotes the number of trajectories in the ground truth. The lower FM and IDS values represent the better ability to track pedestrians successively. From the table, the methods with C2K perform better than those with FN, even with different feature selections (no.10, no.11, and no.12 in our experiments). This implies that better single-camera-tracking methods provide better tracking results for the multiple-cameras-tracking framework. Next, we compare the proposed approach with *M*vs*M* scheme, based on several feature selections. As shown in the results, the proposed approach has lower mFM and mIDS than *M*vs*M* by using feature selections no.10 and no.12. This is because *M*vs*M* identifies the pedestrians by only considering appearance features, but the proposed method further utilizes the relative 3-D locations predicted by the map prior. Moreover, with the same feature selections, the results of the proposed approach are better than that of *M*vs*M*. This indicates that the proposed approach can improve the performance by adopting well-developed features. These results clearly show favorable performance of the proposed framework, not only in overlapping and non-overlapping scenarios, but also in complicatedly mixed scenarios.

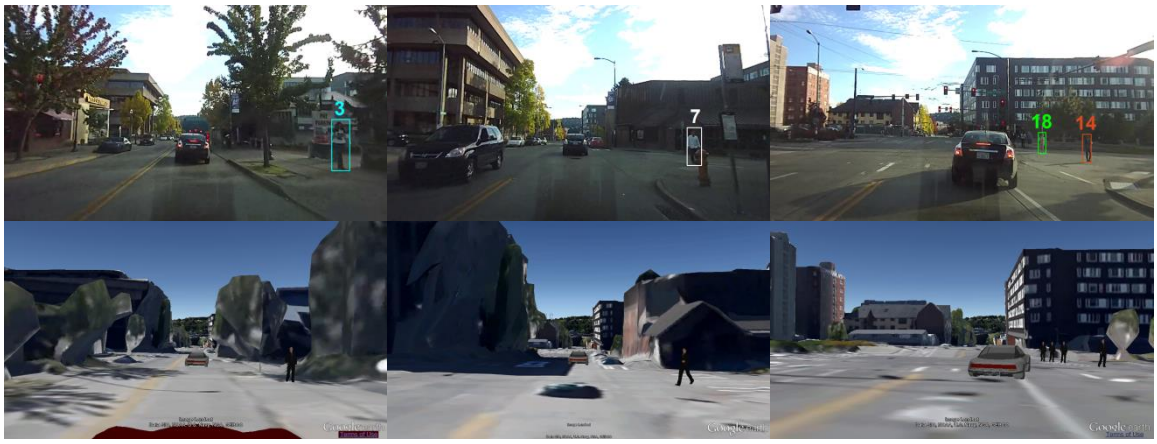
[3] available: http://people.csail.mit.edu/hpirsiav/

(a)

(b)

Figure 5.6: Visual tracking results in Dataset A, where the top rows are the recorded frames, and bottom rows are the corresponding 3-D visualization. (a) Frames from device 1 at t = 1128 (left), t = 2154 (middle), and t = 2984 (right). (b) Frames from device 3, at t = 2477 (left), t = 2977 (middle), and t = 3266 (right).

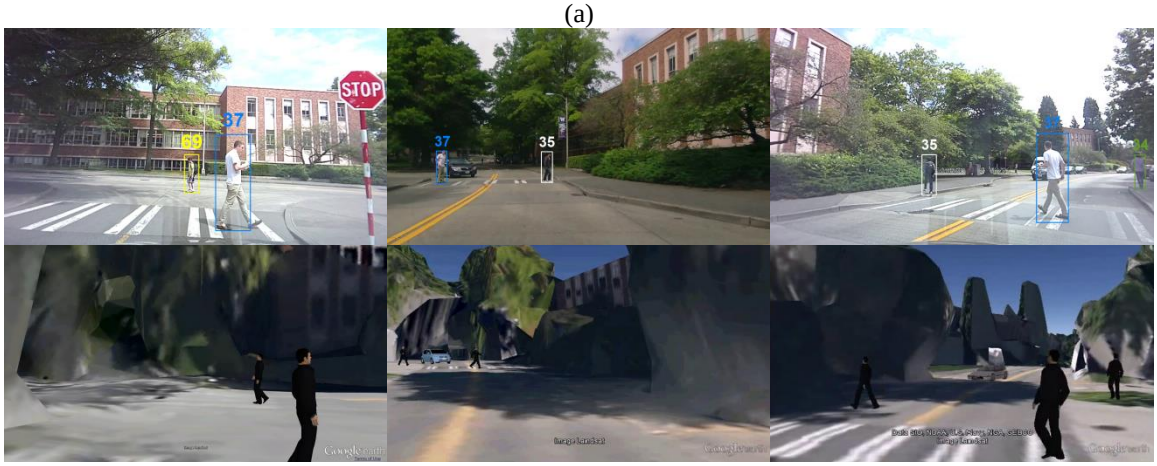
(a)



(b)

Figure 5.7: Visual tracking results in Dataset B, where the top rows are the recorded frames, and bottom rows are the corresponding 3-D visualization. (a) Frames from device 1 at t = 89 (left), t = 1070 (middle), and t = 1497 (right). (b) Frames from device 3, at t = 662 (left), t = 1220, and t = 1497 (right).

(a)



(b)

Figure 5.8 (previous page): Visual tracking results in Dataset C, where the top rows are the recorded frames, and bottom rows are the corresponding 3-D visualization. (a) Non-overlapping case, tracking from device 1 at t = 348 (left), followed by device 3 at t = 846 (middle), and device 4 at t = 3231 (right). (b) Overlapping case, tracking from device 3 at t = 902 (left), and device 1's view (middle) overlapped with device 3 (right), both at t = 1175.

Table 5.3: Comparison of the Tracking Results

Dataset A										
Single	**Multiple**	**Features**	**GT**	**MT**	**PT**	**ML**	**sFM**	**sIDS**	**mFM**	**mIDS**
FN	Proposed	no.11	71	27	13	2	16	3	7	5
FN	Proposed	no.12	71	27	13	2	16	3	7	5
FN	Proposed	no.10	71	27	13	2	16	3	5	3
C2K	Proposed	no.11	71	34	8	4	9	1	6	3
C2K	*M*vs*M*	no.9	71	34	8	4	9	1	7	9
C2K	Proposed	no.12	71	34	8	4	9	1	**5**	**2**
C2K	*M*vs*M*	no.12	71	34	8	4	9	1	7	6
C2K	Proposed	no.10	71	34	8	4	9	1	**3**	**2**
C2K	*M*vs*M*	no.10	71	34	8	4	9	1	5	4

Dataset B										
Single	**Multiple**	**Features**	**GT**	**MT**	**PT**	**ML**	**sFM**	**sIDS**	**mFM**	**mIDS**
FN	Proposed	no.11	12	12	9	1	1	1	3	2
FN	Proposed	no.12	12	12	9	1	1	1	2	2
FN	Proposed	no.10	12	12	9	1	1	1	1	2
C2K	Proposed	no.11	12	10	1	0	1	0	2	2
C2K	*M*vs*M*	no.9	12	10	1	0	1	0	2	3
C2K	Proposed	no.12	12	10	1	0	1	0	**1**	**0**
C2K	*M*vs*M*	no.12	12	10	1	0	1	0	2	3
C2K	Proposed	no.10	12	10	1	0	1	0	**1**	**0**
C2K	*M*vs*M*	no.10	12	10	1	0	1	0	2	2

Dataset C										
Single	**Multiple**	**Features**	**GT**	**MT**	**PT**	**ML**	**sFM**	**sIDS**	**mFM**	**mIDS**
FN	Proposed	no.11	43	23	12	6	4	2	8	5
FN	Proposed	no.12	43	23	12	6	4	2	7	5
FN	Proposed	no.10	43	23	12	6	4	2	3	4
C2K	Proposed	no.11	43	27	10	4	0	0	4	1
C2K	*M*vs*M*	no.9	43	27	10	4	0	0	4	13
C2K	Proposed	no.12	43	27	10	4	0	0	**1**	**1**
C2K	*M*vs*M*	no.12	43	27	10	4	0	0	4	6
C2K	Proposed	no.10	43	27	10	4	0	0	**1**	**0**
C2K	*M*vs*M*	no.10	43	27	10	4	0	0	4	6

FN: flow network based [64], **C2K**: constrained 2-kernel based [108], **GT**: ground truth, **MT**: Most Tracked trajectories, **PT**: Partially Tracked trajectories, **ML**: Most Tracked trajectories, s**FM**: FragMentation in single moving camera, s**IDS**: ID Switches in single moving camera, **mFM**: FragMentation in multiple moving cameras, **mIDS**: ID Switches in multiple moving cameras.

Based on their relative 3-D (GPS) locations, we create a 3-D visualization of the dynamic scene, so as to observe what happens to the roads/streets from different aspects, as shown in Figures 5.6–5.8. The 3-D real-world environment is built upon Google Earth, where the pedestrians are replaced by an avatar-like 3-D human model, and the vehicles equipped with cameras are also represented by the default 3-D vehicle models. The videos associated with the simulations and the demo of the 3-D visualization can be viewed in our website[4].

5.5.c Impact of σ_{mo}

The parameter σ_{mo} in Eq. (31) plays an important role in our proposed framework, since it not only determines the valid range of the cameras' FOVs, but also filters out the targets whose locations are improper. Figure 5.9 shows the results of mFM and mIDS versus different σ_{mo} in terms of meters. As we can observe, mFM and mIDS become larger when σ_{mo} is larger. This is because a larger σ_{mo} relaxes the restriction provided by the motion cues, resulting in more failure of the pedestrians' association, as well as mFM. In this case, the scenario is similar to the *M*vs*M*; that is, the association only considers appearance features without taking into account the motion cues (obtained from p_{map}, i.e., open map service), resulting in a typical re-identification task. However, if σ_{mo} is smaller, mFM becomes larger since the restriction to a pedestrian's presence is too strong to preserve the candidates. This implies that the performance is highly improved when the motion cues are adequately incorporated. Hence, we empirically choose $\sigma_{mo} = 15$ in our experiments.

5.5.d Discussion and Limitations

In the proposed framework, the BTF is calculated by using spatially overlapping FOVs, as shown in Figure 5.10(a). If there is no spatially overlapping FOV, as with FOV1 and FOV2 in Figure 5.10(b), we use an identity matrix for the BTF, and expect

[4] website: http://allison.ee.washington.edu/kuanhuilee/mmcv

degraded performance. However, thanks to the mobility of the moving cameras, there is still an opportunity to estimate the BTF between two non-overlapped cameras' FOVs. As shown in Figure 5.10(b), if FOV1 overlaps with FOV3 in the region B, and FOV3 overlaps with FOV2 in the region C, we can estimate the BTF between FOV1 and FOV2 via FOV3. In the future, we intend to build BTF connectivity between all the collected videos, so that we can explore the relationship between the cameras, and further estimate the BTFs with the connectivity.

Furthermore, the proposed framework can be scaled up to larger systems with larger numbers of moving cameras. Each local region within a time section can be regarded as a processing unit, and the proposed framework is adopted in the unit. Eventually, facilitated by the cloud computing, all the units can be processed in parallel, so as to efficiently construct larger and wider mobile surveillance.

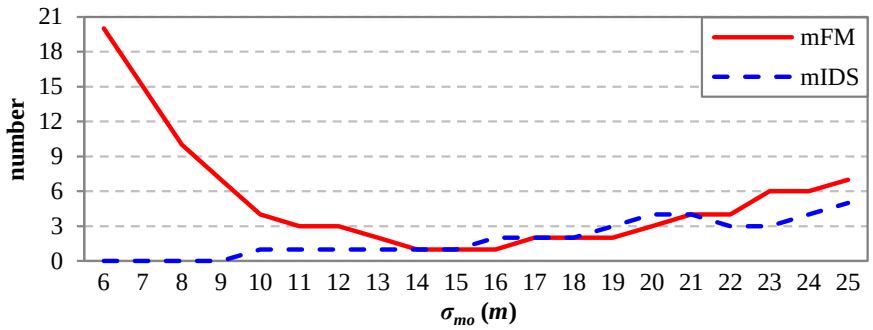


Figure 5.9: Results of mFM and mIDS with different σ_{mo}.

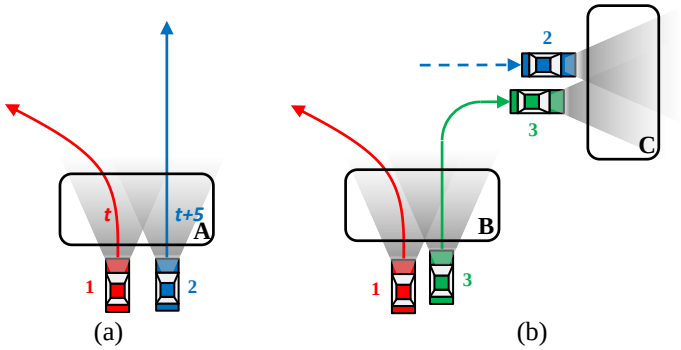


Figure 5.10: Examples of spatially overlapping FOVs. (a) FOV1 and FOV2 are spatially overlapping in the region A at different timestamp. (b) FOV1 and FOV2 are non-overlapping; but FOV1 and FOV3 are spatially overlapping in the region B, FOV2 and FOV3 are spatially overlapping in the region C.

Figure 5.11: 3-D visualization of the scene recorded by four driving recorders. Each row belongs to one driving recorder; the leftmost is the video frames, the middle is the corresponding view of 3-D visualization on Google Earth, and the right is the scene visualized from a different aspect.

Chapter 6 – Conclusion and Future Work

6.1 Conclusion

In this dissertation, we propose an automatic system which dynamically tracks video objects (human and vehicle) and creates their 3-D visualization from big visual data. The system includes human/vehicle tracking under a single static camera, human tracking in single moving camera, and human tracking across multiple moving cameras.

In static camera tracking, our proposed work is mainly based on the constrained multiple-kernel tracking framework. For human tracking under a single static camera, a pre-trained human detector is additionally applied to solve initial merging issues. Moreover, a self-organized system that tracks human across the cameras with nonoverlapping views is proposed by utilizing information from Google Maps, so as to be scaled up efficiently when more cameras are added into the network. For vehicle tracking, our proposed approach regards each patch of the 3-D vehicle model as a kernel, and track the kernels under certain constraints facilitated with the 3-D geometry of the vehicle model. By elegant application of constrained multiple-kernel tracking facilitated with the 3-D vehicle model, the vehicles are able to be tracked efficiently and located precisely.

For the case of tracking under a single moving camera, a robust human tracking system in a moving camera is proposed. The proposed system effectively integrates the human detectors and V-SLAM framework to relocate the humans in 3-D space, followed by an innovative 3-D based CMK tracking, which not only locally associates the targets but also globally optimizes the associations according to the 3-D information. A novel tracking framework, combining CMK tracking and the estimated 3-D information is proposed to globally optimize the data association between consecutive frames. By taking advantage of the appearance model and 3-D information, the proposed system not only achieves high effectiveness but also handles occlusion in the tracking.

Finally, based on the results of tracking under a single moving camera, we propose a new framework for tracking pedestrians across multiple moving cameras, by treating the problem as multi-label classification at each timestamp. Based on the appearance and motion cues, facilitated by the Google Maps, the proposed framework evaluates the association likelihood of the targets, so as to associate with the targets across the multiple moving cameras. Moreover, based on their relative 3-D locations, a 3-D visualization of dynamic scene can be reconstructed for users to see a holistic view or different viewing perspectives of the 3-D scenes reconstructed by the multiple videos.

6.2 Future Work

In this dissertation, the video object tracking under different scenarios are proposed and developed, including human/vehicle tracking under a single static camera, human tracking under a single moving camera, and human tracking across multiple moving cameras. However, the proposed approaches still have limitations as discussed in Section3.4.d-vi, Section 4.3.d, and Section 5.5.d.

One extension to this work is to explore the feature connectivity within the surveillance mobile network. As discussed in Section 5.5.d, the BTF is calculated by using spatially overlapping FOVs; if there is no spatially overlapping FOV, we only can apply identity matrix to the BTF, and expect degraded performance. By utilizing possible spatially overlapping FOVs within the mobile network, a graph called Brightness Feature Transfer Connectivity (BFTC) can be built to describe the cameras' connectivity. Each camera is regarded as one node, if two cameras have spatially overlapping FOVs, an edge is assigned to connect two nodes. Therefore, to find the BTF between two cameras, we tend to find a shortest path between two corresponding nodes within the graph. Once the shortest path is determined, the BTF between two cameras can be obtained by iteratively mapping BTFs one by one, along the shortest path.

In addition, our work starts from an ideal and small scaled mobile surveillance network, proposing an innovative framework to track objects within the network. In the future, the work can be extended to a large scaled application when the mobile

surveillance network is scaled up to larger systems with larger amounts of moving cameras. This also involves many applications in other area, such as video transmission and communication [117]−[128]. Each local region within a time section can be regarded as a processing unit, and the proposed framework is adopted in the unit. Eventually, facilitated by the cloud computing, all the units can be processed in parallel, so as to efficiently construct larger and wider mobile surveillance.

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surveys*, vol. 38, no.4, 2006.

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, May 2003.

[3] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[4] R. T. Collins, "Mean-shift blob tracking through scale space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2003, pp. 234–240.

[5] A. Yilmaz, "Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007.

[6] B. Martinez, L. Ferraz, X. Binefa, and J. Diaz-Caro, "Multiple kernel two-step tracking," in *Proc. IEEE Int.. Conf. Image Process.*, 2006, pp. 2785–2788.

[7] Z. Fan, Y. Wu, and M. Yang, "Multiple collaborative kernel tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 502–509.

[8] C.-T. Chu, J.-N. Hwang, H.-I Pai, and K.-M. Lan, "Tracking human under occlusion based on adaptive multiple kernels with projected gradients," *IEEE Trans. Multimedia*, vol.5, no.7, pp. 1602–1615, Nov. 2013.

[9] N. Buch, S. A. Velastin and J. Orwell, "A review of computer vision techniques for the analysis of urban traffic", *IEEE Trans. Intell. Transp. Syst.*, vol.12, no.3, pp.920–939, Sep. 2011.

[10] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Proc. Eur. Conf. Comput. Vis.*, May 1994, pp. 189–196.

[11] P. L. M. Bouttefroy, A. Bouzerdoum, S. -L. Phung, and A. Beghdadi, "Vehicle tracking by non-drifting mean-shift using projective Kalman filter," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Oct. 2008, pp. 61–66.

[12] D. M. Ha, J. M. Lee, and Y. D. Kim, "Neural-edge-based vehicle detection and traffic parameter extraction," *J. Image and Vis. Comput.*, vol. 22, no. 11, pp. 899–907, 2004.

[13] W. Zhang, Q. M. J. Wu, X. Yang, and X. Fang, "Multilevel framework to detect and handle vehicle occlusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 161–174, Mar. 2008.

[14] J. W. Hsieh, S. H. Yu, Y. S. Chen, and W. F. Hu, "Automatic traffic surveillance system for vehicle tracking and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 175–187, Jun. 2006.

[15] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proc. Eur. Conf. Comput. Vis.*, May 2002, pp. 661–675.

[16] T. Xiong and C. Debrunner, "Stochastic car tracking with line- and color-based features," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 324–328, Dec. 2004.

[17] J. Scharcanski, A.B. Oliveira, P. G. Cavalcanti, and Y. Yari, "A particle-filtering approach for vehicular tracking adaptive to occlusions," *IEEE Trans. Veh. Technol.*, vol. 60, no.2, pp. 381–389, Feb. 2011.

[18] Y. Zheng and S. Peng, "Model based vehicle localization for urban traffic surveillance using image gradient based matching," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Sep. 2012, pp. 945–950.

[19] D. Koller, K. Daniilidis, and H. H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *Int. J. Comput. Vis.*, vol. 10, no.3, pp. 257–281, Jun. 1993.

[20] M. Haag and H.-H. Nagel, "Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences" *Int. J. Comput. Vis.*, vol.35, no. 3, pp. 295–319, Dec. 1999.

[21] J. Lou, T. Tan, W. Hu, H. Yang, and S. J. Maybank, "3-D model-based vehicle tracking," *IEEE Trans. Image Process.*, vol. 14, no. 10, pp. 1561–1569, Oct. 2005.

[22] J. Liebelt, C. Schmid, and K. Schertler, “Viewpoint-independent object class detection using 3D feature maps,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, June 2008.

[23] C. Reinbacher, M. Ruther and H. Bischof, “Pose estimation of known objects by efficient silhouette matching”, P*roc. IEEE Int. Conf. Pattern Recognit.*, Aug. 2010, pp.1080–1083.

[24] N. Buch, J. Orwell and S. A. Velastin, “Urban road user detection and classification using 3D wire frame models”, *IET Comput. Vis.*, vol.4, no. 2 pp.105–116, Jun. 2010.

[25] J. Ferryman, A. Worrall, G. Sullivan, and K. Backer, “A generic deformable model for vehicle recognition,” in *Proc. British Mach. Vis. Conf.*, Sep. 1995, pp. 127–136.

[26] T. N. Tan, G. D. Sullivan, and K. D. Baker, “Model-based localisation and recognition of road vehicles,” *Int. J. Comput. Vis.*, vol. 27, no.1, pp. 5–25, Mar. 1998.

[27] T. N. Tan and K. D. Baker, “Efficient image gradient based vehicle localization,” *IEEE Trans. Image Process.*, vol. 9, no.8, pp. 1343–1356, Aug. 2000.

[28] Z. Zhang, T. Tan, K. Huang, and Y. Wang, “Three-dimensional deformable-model-based localization and recognition of road vehicles,” *IEEE Trans. Image Process.*, vol.21, no.1, pp.1–13, Jan., 2012.

[29] M. J. Leotta and J. L. Mundy, “Vehicle surveillance with a generic, adaptive, 3D vehicle model,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1457–1469, Jul. 2011.

[30] D. Gerónimo, A. M. López, A. D. Sappa and T. Graf, “Survey of pedestrian detection for advanced driver assistance systems,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1239–1258, Jul. 2010.

[31] Z. Sun, G. Bebis, and R. Miller, “On-road vehicle detection: A review,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.

[32] T. Tsuji, H. Hattori, M. Watanabe, and N. Nagaoka, “Development of night-vision system,” *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 3, pp. 203–209, Sept. 2002.

[33] N. Srinivasa, “A vision-based vehicle detection and tracking method for forward collision warning,” in *Proc. IEEE Intell. Veh. Symp.*, 2002, pp. 626–631.

[34] T. Bucher, C. Curio, J. Edelbrunner, C. Igel, D. Kastrup, I. Leefken, G. Lorenz, A. Steinhage, and W. von Seelen, “Image processing and behavior planning for intelligent vehicles,” *IEEE Trans. Ind. Electron.*, vol. 50, no. 1, pp. 62–75, 2003.

[35] H. Elzein, S. Lakshmanan, and P. Watta, “A motion and shape-based pedestrian detection algorithm,” in *Proc. IEEE Intell.Veh. Symp.*, 2003, pp. 500–504.

[36] U. Franke and S. Heinrich, “Fast obstacle detection for urban traffic situations,” *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 3, pp. 173–181, Sept. 2002.

[37] A. Jazayeri, H. Cai, J.-Y. Zheng, and M. Tuceryan, “Vehicle detection and tracking in car video based on motion model,” *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 583–595, Jun. 2011.

[38] A. Kundu, K. M. Krishna and C. V. Jawahar, “Realtime multibody visual SLAM with a smoothly moving monocular camera,” in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2080–2087.

[39] H. Nanda and L. Davis, “Probabilistic template based pedestrian detection in infrared videos,” in *Proc. IEEE Intell. Veh. Symp.*, 2002, pp. 15–20.

[40] D. Gavrila, J. Giebel, and S. Munder, “Vision-based pedestrian detection: The PROTECTOR system,” in *Proc. IEEE Intell. Veh. Symp.*, 2004, pp. 13–18.

[41] D. Gavrila and S. Munder, “Multi-cue pedestrian detection and tracking from a moving vehicle,” *Int. J. Comput. Vis.*, vol. 73, no. 1, pp. 41–59, 2007.

[42] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Moulminet, “A cooperative approach to vision-based vehicle detection,” in Proc. *IEEE Intell. Transp. Syst.*, 2001, pp. 209–214.

[43] J. Wu, N. Liu, C. Geyer, and J. M. Rehg, "C^4: A real-time object detection framework," *IEEE Trans. Image Process.*, vol. 22, no.10 , pp. 4096–4106, Oct. 2013

[44] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Proc. Int. Conf. Comput. Vis.*, vol. 2, 2003, pp. 734–741.

[45] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 1, 2005, pp. 886–893.

[46] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no.9, pp. 1627–1645, Sep. 2010.

[47] L. C. León and R. Hirata, "Car detection in sequences of images of urban environments using mixture of deformable part models," *Pattern Recognit. Lett.*, Nov. 2013.

[48] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *Int. J. Comput. Vis.*, vol. 77, no. 1–3, pp. 259–289, May 2008.

[49] M. Bertozzi, A. Broggi, A. Fascioli, A. Tibaldi, R. Chapuis, and F. Chausse, "Pedestrian localization and tracking system with Kalman filtering," in *Proc. IEEE Intell. Veh. Symp.*, 2004, pp. 584–589.

[50] V. Philomin, R. Duraiswami, and L. Davis, "Pedestrian tracking from a moving vehicle," *Proc. IEEE Intell. Veh. Symp.*, 2000, pp. 350–355.

[51] J. Giebel, D. Gavrila, and C. Schnör, "A Bayesian framework for multi-cue 3D object tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2004, pp. 241–252.

[52] R. Arndt, R. Schweiger, W. Ritter, D. Paulus, and O. Lóhlein, "Detection and tracking of multiple pedestrians in automotive applications," in *Proc. IEEE Intell. Veh. Symp.*, 2007, pp. 13–18.

[53] I.J. Cox, "A review of statistical data association techniques for motion correspondence," *Int. J. Comput. Vis.*, vol. 10, no. 1, pp. 53–66, 1993.

[54] D. B. Reid, “An algorithm for tracking multiple targets,” *IEEE Trans. Autom. Control*, vol. 24, no. 6, pp. 843–854, Dec. 1979.

[55] T. E. Fortmann, Y. Bar Shalom, and M. Scheffe, “Sonar tracking of multiple targets using joint probabilistic data association,” *IEEE J. Ocean. Eng.*, vol. 8, no. 3, pp. 173–184, Jul. 1983.

[56] A. Ess, B. Leibe, K. Schindler, and L. VanGool, “Robust multi-person tracking from a mobile platform,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1831–1846, Oct. 2009.

[57] M. Adnriluka, S. Roth, and B. Schiele, “People-tracking-by-detection and people-detection-by-tracking,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008.

[58] H. T. Niknejad, A. Takeuchi, S. Mita and d McAllester, “On-road multivehicle tracking using deformable object model and particle filter with improved likelihood estimation,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, Jun. 2012, pp. 748–758.

[59] A. Jazayeri, H. Cai, J. Y. Zheng, and M. Tuceryan, “Vehicle detection and tracking in car video based on motion model,” *IEEE Trans. Intell. Transp. Syst*., vol. 12, no. 2, pp. 583–595, Jun. 2011.

[60] B. Leibe, N. Cornelis, K. Cornelis, and L. VanGool, “Dynamic 3D scene analysis from a moving vehicle,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007.

[61] B. Leibe, K. Schindler, N. Cornelis, and L. VanGool, “Coupled object detection and tracking from static cameras and moving vehicles,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1683–1698, Oct. 2008.

[62] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, “3D traffic scene understanding from movable platforms,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol.36, no.35, pp. 1012–1025, May 2013.

[63] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008.

[64] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011.

[65] Z. Wu, A. Thangali, S. Sclaroff and M. Betke, "Coupling detection and data association for multiple object tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp.1948–1955.

[66] A. A. Butt and R. T. Collins, "Multi-target tracking by Lagrangian relaxation to min-cost network flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013.

[67] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: Hybridboosted multi-target tracker for crowded scene," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun 2009, pp. 2953–2960.

[68] B. Yang and R. Nevatia, "Multi-target tracking by online learning a CRF Model of appearance and motion patterns," *Int. J. Comput. Vis.*, vol.107, no.2, pp. 203–217, Apr. 2013

[69] R. Hartley and A. Zisserman, "*Multiple View Geometry in Computer Vision*," Cambridge, U.K.: Cambridge University Press, 2004.

[70] N. Snavely, S. M. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *Int. J. Comput. Vis.*, vol. 80, no. 2, pp. 189–210, Jul. 2007.

[71] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real time localization and 3D reconstruction" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 363–370.

[72] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G.Welch and H. Towles, "Detailed real-time

urban 3D reconstruction from video," *Int. J. Comput. Vis.*, vol. 78, no. 2–3, pp. 143–167, Jul. 2008.

[73] S. Song and M. Chandraker, "Robust scale estimation in real-time monocular SfM for autonomous driving", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014.

[74] S. Gong, M. Cristani, S. Yan, and C. C. Loy, *Person Re-Identification*. New York, NY, USA: Springer, 2014.

[75] M. Farenzenz, L. Bazzani, A. Perina, V. Murino, and M. Cristani. "Person re-identification by symmetry-driven accumulation of local features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2360–2367.

[76] W. Li and X. Wang, "Locally aligned feature transforms across views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3594–3601.

[77] F. Jurie, and A.Mignon, "PCCA: A new approach for distance learning from sparse pairwise constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2666–2672.

[78] D. Gray and H. Tao, "Viewpoint invariant pedestrian recognition with an ensemble of localized features," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp.262–275.

[79] M. Hirzer, P. M. Roth, and M. Kostinger, "Relaxed pairwise learned metric for person re-identification," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp.780–793.

[80] W. Zheng, S. Gong, and T. Xiang, "Person re-identification by probabilistic relative distance comparison," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 649–656.

[81] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, P. Tu, "Shape and appearance context modeling," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2007.

[82] A. Alahi, P. Vandergheynst, M. Bierlaire, and M. Kunt, "Cascade of descriptors to detect and track objects across any network of cameras," *Comput. Vis. Image Und.*, vol.114, no.6, 2010 pp.624–640.

[83] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, “Tracking across multiple cameras with disjoint views,” in *Proc. IEEE Conf. Comput. Vis.*, 2003, pp. 952–957.

[84] B. Moller, T. Plotz, and G. A. Flink, “Calibration-free camera hand-over for fast and reliable person tracking in multi-camera setups,” in *Proc. IEEE Int. Conf. Pattern Recognit.*, 2008.

[85] C.-T. Chu, J.-N. Hwang, K.-M. Lan and S.-Z. Wang, “Tracking across multiple cameras with overlapping views based on brightness and tangent transfer functions,” in *Proc. ACM/IEEE Int.. Conf. Distrib. Smart Cam.*, 2011.

[86] C.-T. Chu, K.-H. Lee, and J.-N. Hwang, “Self-organized and scalable camera networks for systematic human tracking across nonoverlapping cameras,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, pp. 2322－2326, May 2013.

[87] K.-H. Lee, C.-T. Chu, Y. Lee, Z. Fang, and J.-N. Hwang, “Consistent human tracking over self-organized and scalable multiple-camera networks,” *Distributed Embedded Smart Cameras*, New York, NY, USA: Springer, pp. 189－209, 2014.

[88] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua, “Multicamera people tracking with a probabilistic occupancy map,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 267–282, Feb. 2008.

[89] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua, “Multiple object tracking using K-shortest paths optimization,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no.9, pp. 1806–1819, Sep. 2011.

[90] T. D’Orazio, P. Mazzeo, and P. Spagnolo, “Color brightness transfer function evaluation for non-overlapping multi camera tracking,” in *Proc. ACM/IEEE Intl. Conf. Distrib. Smart Cam.*, 2009.

[91] S.-I Yu, Y. Yang and A. Hauptmann, “Harry Potter’s Marauder’s map localizing and tracking multiple persons-of-interest by nonnegative discretization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013.

[92] K. Tieu, G. Dalley, W. Grimson, “Inference of non-overlapping camera network topology by measuring statistical dependence,” in *Proc. IEEE Conf. Comput. Vis.*, 2005.

[93] A. Gilbert and R. Bowden, “Tracking objects across cameras by incrementally learning inter-camera colour calibration and patterns of activity,” in *Proc. Eur. Conf. Comput. Vis.*, pp. 125–136, 2006.

[94] B. C. Matei, H. S. Sawhney, and S. Samarasekera, “Vehicle tracking across nonoverlapping cameras using joint kinematic and appearance features,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 3465–3472.

[95] C. Ding, B. Song, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury, “Collaborative sensing in a distributed PTZ camera network,” *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3282–3295, Jul. 2011.

[96] K. Chen, C. Lai, P. Lee, C. Chen and Y. Hung, “Adaptive learning for target tracking and true linking discovering across multiple non-overlapping cameras,” *IEEE Trans. Multimedia*, vol. 13, no.4 pp. 625–638, Mar. 2011

[97] C.-T. Chu and J.-N. Hwang, “Fully unsupervised learning of camera link models for tracking humans across non-overlapping cameras,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no.6, pp.979–994, Jan. 2014.

[98] D. Zou and P. Tan, “CoSLAM: Collaborative visual SLAM in dynamic environments,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 354–366, Feb. 2013.

[99] I. O. Sebe, J. Hu, S. You and U. Neumann, “3D video surveillance with augmented virtual environments,” in *Proc. 1st ACM SIGMM Int. Workshop Video Surveillance*, 2003, pp. 107–112.

[100] K. Kim, S. Oh, J. Lee and I. Essa, “Augmenting aerial earth maps with dynamic information”, in *Proc. IEEE Int. Symp. Mixed Augment. Reality*, 2009, pp. 35–38.

[101] E. R. Corral-Soto, T. Tal, L. Wang, R. Persad, L. Chao, C. Solomon, B. Hou, G. Sohn and J. H. Elder, “3DTown: The automatic urban awareness project,” in *Proc. IEEE Conf. Comput. Robot Vis.*, May 2012, pp. 433–440.

[102] K.-H. Lee, J.-N. Hwang, J. Yu, and K.-Z. Lee, "Vehicle tracking iterative by Kalman-based constrained multiple-kernel and 3-D model-based localization," in *Proc. IEEE Int. Symp. CircuitsSyst.*, May 2013, pp. 2396–2399.

[103] K.-H. Lee, Y.-J. Lee, and J.-N. Hwang, "Multiple-kernel based vehicle tracking using 3-D deformable model and license plate self-similarity," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, pp. 1793－1797, May 2013

[104] K.-H. Lee, J.-N. Hwang and S.-I Chen, "Model-based vehicle localization based on three-dimensional constrained multiple-kernel tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol.25, no.1, pp. 38–50, Jun. 2014.

[105] P. Larrauaga and J. A. Lozano, "Estimation of distribution algorithms: A new tool for evolutionary computation," Norwell, MA: Kluwer, 2001.

[106] PETS 2000 database [Online]. Available: ftp://ftp.pets.rdg.ac.uk/pub/PETS2000/.

[107] Advanced Video and Signal based Surveillance (AVSS) 2007 [Online]. Available: http://www.eecs.qmul. ac.uk/~andrea/avss2007_d.html.

[108] K.-H. Lee, J.-N. Hwang, G. Okopal, and J. Pitton, "Driving recorder based on-road pedestrian tracking using visual SLAM and constrained multiple-kernel," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Oct. 2014, pp.2629–2635.

[109] C. Yang, L. Zhang, H. Lu, M.-H. Yang, and X. Ruan, "Saliency detection via graph-based manifold ranking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3166–3173.

[110] J. Wright, Y. Peng, Y. Ma, A. Ganesh, and S. Rao, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization," in *Proc. Neural Inf. Process. Syst.*, Dec. 2009.

[111] K.-H. Lee and J.-N. Hwang, "On-road pedestrian tracking across multiple driving recorders, " *IEEE Trans. Multimedia, Special Issue Multimedia: The Biggest Big Data*, vol.17, no.9, pp. 1429－1438, Jul. 2015.

[112] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2012. Available: http://www.gurobi.com

[113] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*. Cambridge, MA, USA: MIT Press, 2000.

[114] P.-E. Forssén, “Maximally stable colour regions for recognition and matching,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2007

[115] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[116] T. Ojala, M. Peitikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture with local binary patterns,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 1683–1698, Jul. 2002.

[117] X. Chen, J.-N. Hwang, C.-N. Lee, and S.-I Chen, “A Near optimal QoE-driven power allocation scheme for scalable video transmissions over MIMO Systems,” *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 1, pp. 76-88, 2015.

[118] X. Chen, J.-N. Hwang, K.-H. Lee, R. L. de Queiroz, “Quality-of-Content (QoC)-driven rate allocation for video analysis in mobile surveillance networks,” in *Proc. IEEE Int. Workshop Multimedia Signal Process*., Oct. 2015.

[119] J. Park, X. Chen, J.-N. Hwang, “Optimum power allocation and rate adaptation for scalable video streaming over multi-user MIMO networks”, in *Proc. IEEE Conf. Global Commun.* Dec. 2015.

[120] X. Chen, J.-N. Hwang, C.-J. Wu, S.-R. Yang, and C.-N. Lee, “A QoE-based APP layer scheduling scheme for scalable video transmissions over multi-RAT systems,” in *Proc. IEEE Int. Conf. Commun.*, Jun. 2015.

[121] X. Chen, H. Du, J.-N. Hwang, J. A. Ritcey, and C.-N. Lee, “A QoE-driven FEC rate adaptation scheme for scalable video transmissions over MIMO systems,” in *Proc. IEEE Int. Conf. Commun.*, Jun. 2015.

[122] X. Chen, J.-N. Hwang, C.-Y. Wang, and C.-N. Lee, “A near optimal QoE-driven power allocation scheme for SVC-based video transmissions over MIMO systems,” in *Proc. IEEE Int. Conf. Commun.*, Jun. 2014.

[123] X. Chen, J.-N. Hwang, C.-N. Lee, C.-W. Hwang, “An efficient CQI feedback resource allocation scheme for wireless video multicast services,” in *Proc IEEE Conf. Global Commun.*, Dec. 2013.

[124] X. Chen, J.-N. Hwang, P.-H. Wu, H.-J. Su, and C.-N. Lee, “Adaptive mode and modulation coding switching scheme in MIMO multicasting system,” in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2013.

[125] F. Qu, X. Chen, P. Deng, and L. Yang, “Iterative MLSE for MIMO underwater acoustic channel,” in *Proc. MTS/IEEE Oceans Conf.*, Sep. 2010.

[126] K.-H. Lee and P.-C. Chung, “An attention emphasized bit arrangement in 3-D SPIHT video coding for human vision,” *J. Visual Commun. Image Represent.*, vol.24, no.3, pp. 255－269, Apr. 2013.

[127] L. Hou, W. Wan, K.-H. Lee, J.-N. Hwang, G. Okopal, and J. Pitton, “Deformable multiple-kernel based human tracking using a moving camera,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2015.

[128] K.-H. Lee, J.-N. Hwang, J. Yoo, and K. Choi “Effective car video retrieval using feature matching in a mobile video cloud,” in *Proc. 6th ACM/IEEE Int. Conf. Distrib. Smart Cam.*, Nov. 2012.

Appendix

A. Orthogonality of $\delta_{\mathbf{x}}^{C}$ to $\delta_{\mathbf{x}}^{A}$, $\delta_{\mathbf{x}}^{B}$

In [8], the fact that $\delta_{\mathbf{x}}^{A}$ and $\delta_{\mathbf{x}}^{B}$ are orthogonal to each other is proved. The following proves $\delta_{\mathbf{x}}^{C}$ is orthogonal to both $\delta_{\mathbf{x}}^{A}$ and $\delta_{\mathbf{x}}^{B}$:

$$
\begin{aligned}
&\left(\delta_{\mathbf{x}}^{B}\right)^{T}\left(\delta_{\mathbf{x}}^{C}\right)\\
&=(-\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}(\mathbf{x}))^{T}(\alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{J}_{\mathbf{x}}^{s})\\
&=\alpha\left(-\mathbf{C}(\mathbf{x})^{T}((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}\mathbf{C}_{\mathbf{x}}^{T}\right)\left(-\mathbf{J}_{\mathbf{x}}^{s}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{s}\right)\\
&=\alpha\left(\mathbf{C}(\mathbf{x})^{T}\mathbf{C}_{\mathbf{x}}^{-1}\mathbf{J}_{\mathbf{x}}^{s}-\mathbf{C}(\mathbf{x})^{T}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{s}\right)\\
&=\alpha\mathbf{C}(\mathbf{x})^{T}\left(((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}-(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\right)\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{s}\\
&=0
\end{aligned}
$$

Thus, the $\delta_{\mathbf{x}}^{B}$ and the $\delta_{\mathbf{x}}^{C}$ are orthogonal to each other.

$$
\begin{aligned}
&\left(\delta_{\mathbf{x}}^{A}\right)^{T}\left(\delta_{\mathbf{x}}^{C}\right)\\
&=(\alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{J}_{\mathbf{x}}^{s})^{T}(\alpha(-\mathbf{I}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T})\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}(-(\mathbf{J}_{\mathbf{x}}^{s})^{T}+(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}\mathbf{C}_{\mathbf{x}}^{T})(-\mathbf{J}_{\mathbf{x}}^{f}+\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}((\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f})-(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f}-(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f}\\
&\qquad+(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}((\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f})-(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}\left(((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}+(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}-((\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1})^{T}\right)\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}((\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f})-(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}}^{T}\mathbf{C}_{\mathbf{x}})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}((\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f})-(\mathbf{J}_{\mathbf{x}}^{s})^{T}\mathbf{C}_{\mathbf{x}}(\mathbf{C}_{\mathbf{x}})^{-1}(\mathbf{C}_{\mathbf{x}}^{T})^{-1}\mathbf{C}_{\mathbf{x}}^{T}\mathbf{J}_{\mathbf{x}}^{f})\\
&=\alpha^{2}((\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f})-(\mathbf{J}_{\mathbf{x}}^{s})^{T}(\mathbf{J}_{\mathbf{x}}^{f}))\\
&=0
\end{aligned}
$$

Therefore, the $\delta_{\mathbf{x}}^{A}$ and the $\delta_{\mathbf{x}}^{C}$ are orthogonal to each other.

Vita

Kuan-Hui Lee received the Bachelor of Science degree in the Department of Electrical Engineering from National Taiwan Ocean University in 2003, and the Master of Science degree in the Institute of Computer and Communication Engineering from National Cheng Kung University in 2005. He was in HTC Corporation for developing multi-media applications on smart phone from 2007 to 2009. In 2015, he received a Doctor of Philosophy degree from the Department of Electrical Engineering, University of Washington. His research interests are in image processing, computer vision, and machine learning.