Ph.D. DISSERTATION

Dynamic Formulation of Maximum Weighted Clique Problem for Online Tracking of Multiple Objects via Multiple Cameras

최대 가중 클릭 문제의 동적 생성법을 이용한
온라인 다중 카메라 다중 물체 추적 기법

BY

Haanju Yoo

August 2016

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Dynamic Formulation of Maximum Weighted Clique Problem for Online Tracking of Multiple Objects via Multiple Cameras

최대 가중 클릭 문제의 동적 생성법을 이용한
온라인 다중 카메라 다중 물체 추적 기법

BY

Haanju Yoo

August 2016

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Dynamic Formulation of Maximum Weighted Clique Problem for Online Tracking of Multiple Objects via Multiple Cameras

최대 가중 클릭 문제의 동적 생성법을 이용한 온라인 다중 카메라 다중 물체 추적 기법

지도교수 최 진 영

이 논문을 공학박사 학위논문으로 제출함

2016 년 6 월

서울대학교 대학원

전기 정보 공학부

유 한 주

유한주의 공학박사 학위논문을 인준함

2016 년 6 월

Abstract

In this dissertation, we propose an online and real-time algorithm for tracking of multiple targets with multiple cameras that have overlapping field of views. Because of its applicability, multiple target tracking with a visual sensor has been studied intensively during the recent decades. Especially, algorithms using multiple overlapping cameras have been proposed to overcome the occlusion and missing problem of target that cannot be resolved by a single camera. Since the multiple camera multiple target tracking (MCMTT) problem is more complicated than the single camera multiple target tracking (SCMTT) problem, most of MCMTT algorithms are based on a batch process which considers a whole sequence at a time. Although the batch-based algorithms have been achieved the robust performance, their usability is limited because many practical applications need an instantaneous result. The objective of this dissertation is to develop an online MCMTT algorithm that has compatible tracking performance compared to the batch-based algorithms, but requires a small amount of computations.

The proposed algorithm generates track hypotheses (or simply called 'track') with all possible data associations between object detections from multiple cameras through frames. Then, it picks a set of tracks that best describes the tracking of targets. To identify a good track, the quality of each track is measured by our score function. The tracking solution is, then, a set of tracks that has the maximum total score. To get the solution, we formulate the problem of finding those track set as the maximum weighted clique problem (MWCP), which is one of the widely adopted formulations for a combinatorial problem that has the pairwise compatibility relationship among the variables. MWCP is well-known NP-complete problem and its worst-case computation time is proportional to the exponent of the number of tracks. Thus, solving MWCP is intractable because the number of candidate tracks exponentially increases when the tracking progresses. To alleviate the huge computational load, we propose an online scheme that dynamically formates multiple MWCPs with small-sized subsets of candidate tracks in every frame. The scheme is motivated by that the tracking solutions from consecutive frames are very similar because the status of each target is not abruptly changed between one frame. When we assume that a specific track set is an actual solution of the previous frame, only a small number of tracks have a possibility to become a solution track of the current frame. Thus, we can narrow down the size of candidate track set with the previous solution. However, propagating only the best solution of each frame can cause irreducible error when a wrong track set is chosen as the solution because of the tracking ambiguity. To hedge the risk of this error, we find multiple good solutions at each frame and propagate the K-best solutions among them to the next frame instead of a single solution. When the candidate tracks are updated and generated with newly obtained detections at the next frame, we generate multiple subsets of the entire candidate tracks with the K-best previous solutions. Each subset consists of candidate solution tracks with respect to each of the previous solutions, and a small-sized MWCP is formated with the subset. Then, our algorithm finds multiple solutions from each MWCP and repeats above procedures until the tracking is terminated. Even the proposed algorithm solves multiple MWCPs, it has lower computational complexity than solving a single MWCP with the entire candidate tracks because the overall computational load is mainly affected by the size of the largest MWCP. Moreover, when an instantaneous result is demanded, our algorithm finds better solution than solving a single large-sized MWCP because it finds more diverse solutions under a limited solving time.

Although our dynamic formulation remarkably moderates the overall computational complexity, it is still challenging to satisfy the real-time capability of the tracking system. Thus, we apply three more strategies to reduce the computation time. First, we generate tracklets, robust fragments of a target's trajectory, at each camera and generate candidate tracks with those tracklets instead of detections. This prevents a generation of many absurd tracks. Second, we adopt a heuristic algorithm called a breakout local search (BLS) to solve each MWCP. With BLS, multiple suboptimal solutions can be found efficiently within a short time. Last, we prune the candidate tracks with a probability that is calculated with the K-best solutions. The probability represents the quality of each track with respect to the overall tracking situation instead of an individual track. Thus, utilizing this probability ensures a proper pruning of candidate tracks.

In the experiments with a public benchmark dataset, our algorithm shows the compatible performance compared to the state-of-the-art batch-based MCMTT algorithms. Moreover, our algorithm shows a real-time capability by achieving a satisfactory performance within a reasonable computation time. We also conduct a self-comparison to verify our dynamic MWCP formation with respect to the tracking performance and solving time. When a sufficient number of solutions are propagated, our algorithm performs better and takes shorter time than solving a single MWCP considering the entire candidate tracks.

Keywords: visual tracking, multiple camera multiple target tracking, multiple hypothesis tracking, the maximum weighted clique problem, online tracking, real-time tracking.

Student Number: 2009-20848

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Visual tracking (also known as video tracking) is the process of locating a moving target over time using a visual sensor, a camera. To acheive its objective, visual tracking algorithm has to identify the same objects from the consecutive video frames. When a target moves or changes its pose, the appearance of the target varies severely. This makes impossible to track the target with only a pixel-based comparison of image patches from candidate locations. Thus, for a successful visual tracking, the algorithm must inference the motion and deformation of target with a semantic understanding about the input video frames, which is challenging work even with a single target. Nowadays, many successful algorithms have been proposed for a sigle target tracking with respect to the tracking performance and runtime [1–5]. The algorithms mainly focus on modeling a target with discriminative features in motion and appearance that help to distinguish the target region from the background.

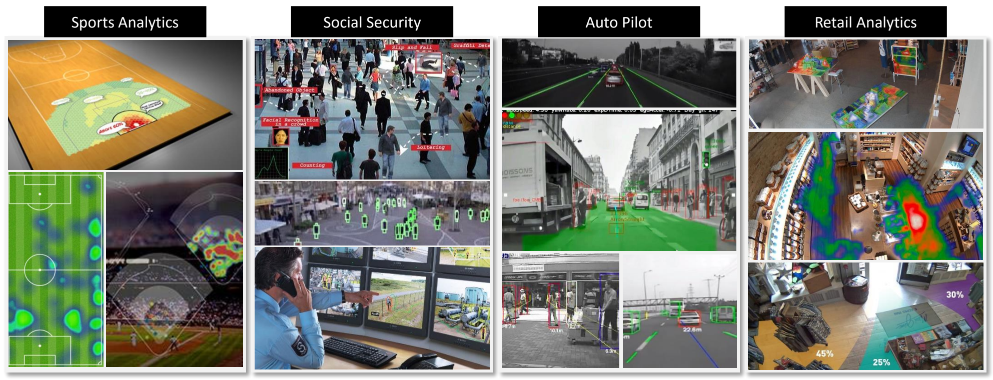


Figure 1.1: Applications of visual tracking. In many industrial and entertainment fields, obtaining the locations of objects in interest by visual tracking is an essential to higher level processings.

Tracking multiple targets can be considered as simultaneously running multiple single target trackers. However, this approach suffers from the tracking ambiguities arising from the targets that are closely distributed. Noramlly, the practical applications using the multiple target tracking (MTT) algorithm have their own objects in interest such as pedstrians, cells, cars, etc. Therefore, they detect those objects with classifier-based object detectors and initiate trackers with each of obtained detections. In this case, detected objects have similar outlooks that makes difficult to determine which detection is obtained from which target. For this reason, visual MTT algorithms have evolved to the tracking-by-detection framework, which tracks the targets with associating the detections from the same target. A number of online and batch methods based on the tracking-by-detection framework have been proposed recently that successfully resolve tracking ambiguities in determining the ownerships of detections by utilizing the target's appearance and motion information [6–25]. However, the state-of-the-art methods mainly suffer from the missed detections of targets that are occluded by obstarcles or other targets. The missed detections make it hard to maintain the consistent labels on the targets, that critically degrades the tracking performance. Beacuse our daily life consists of many crowded environment such as a classroom, retail store, etc, missed detections by occlusions are very common in the practical applications. Thus, resolving the occlusion problem is necessitate for the robust tracking system.

The fundamental way of resolving occlusion problem is to increase the view points of a surveillance area, which gives more chance to detect unoccluded targets. Therefore, the tracking methods utilizing multiple cameras with overlapping fields of views has been studied intensively [26–44]. In the multiple camera multiple target tracking (MCMTT) problem, we have to jointly solve the problems of spatial and temporal association between detections from different camera and different time. In spatial association, the different viewed and concurrent detections from the same target are associated (i.e., reconstruction), while in temporal association, each camera's detections from the same target are associated through frames (i.e., tracking). Thus, MCMTT is more complex and difficult than a single camera multiple target tracking which solves only temporal association. For this reason, most of the recent MCMTT algorithms are based on the batch processing which requires lots of computations (see Section 1.2, Related Works). Many of practical applications need an online and real-time processing. Thus, despite their good performance, the existing MCMTT algorithms have limited applicability. Therefore, the objective of this thesis is to suggest an online MCMTT algorithm that has real-time capability.

1.2 Related Works

In this section, we briefly introduce algorithms in MCMTT. As mentioned at the previous section, MCMTT algorithms are composed of reconstruction and

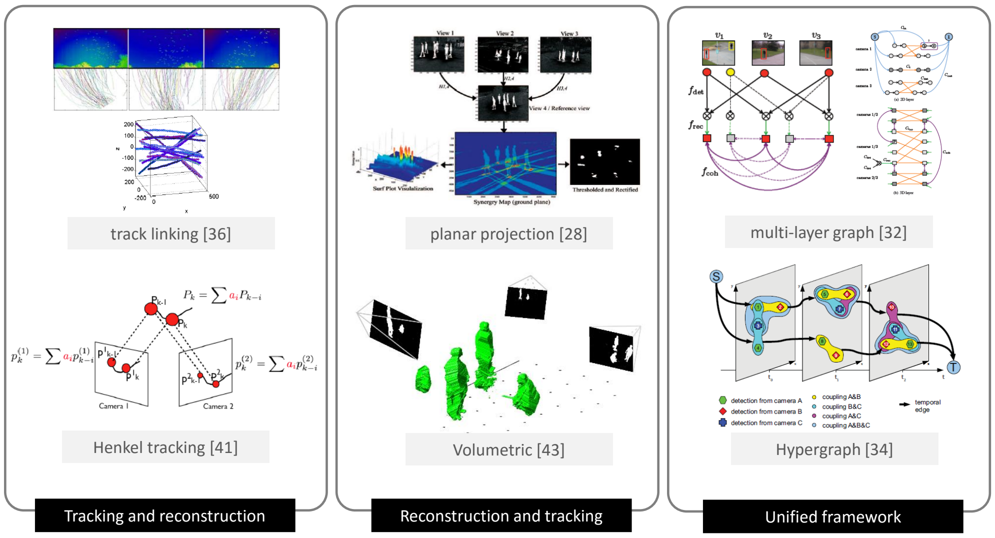


Figure 1.2: Three categories of existing MCMTT algorithms. All graphic resources are from the original papers.

tracking. According to the order of those two steps, the existing algorithms are categorized into three groups as depicted in the Figure 1.2: reconstruction-and-tracking, tracking-and-reconstruction, and unified framework.

1.2.1 Reconstruction-and-tracking methods

The algorithms in this category generate unified measurement in a surveillance space by merging measurements from multiple cameras, to overcome the missing problem of the object detection, caused by occlusion or background clutter. Then, the multiple targets are tracked by a single camera-based tracking method such as Kalman filter [45] and particle filter [46] with those unified measurements.

Mittal et al. [35] generated 3D measurements by foreground region detection with color information and a region-based stereo algorithm. Those 3D measurements were projected onto 3D ground plane and a Kalman filter [45] was applied on the projected 2D measurements to track each target. Because they used a stereo matching based on appearance information such as RGB color, the algorithm only operates when a view angle between cameras is small. Therefore, a sufficient number of cameras are needed to cover the surveillance region, and it is the significant limitation of this work.

Fleuret et al. [26] proposed the probabilistic occupancy map (POM) representing the probability distribution of target's existence on the ground plane. They inferred POM by combining the background subtraction result of each view. POM was used as an input of the tracking method and whole tracking algorithm worked well on the moderate scenarios. Thus, it was adopted by many studies [29–31, 33]. Berclaz et al. [30] formulated tracking problem as a linear programming with the network graph constructed with POM from each frame. Then, they efficiently solved the linear programming with K-shortest path (KSP) algorithm. However, [30] is very sensitive to the false positivie when the prior information about the number of targets does not provided. Moreover, POM based algorithms cannot avoid the quantization of 3D space because of their computational complexity.

The alternative of POM is a synergy map [28], which is the stacked projections of each view's foreground mask onto the planes that are parallel to the ground plane. Unlike POM, a synergy map gives a probability distribution of the target existance in continuous domain. Several studies adopt the synergy map [39, 47], however, they suffered from the ghost (or phantom) problem of projected foreground masks which gives strong ambiguity about the localization of targets.

Possegger et al. [43] proposed an algorithm based on a volumetric density map, which generalizes the ground plane assumption and moderates the above two problems. To produce the volumetric density map, which is called a 3D occupancy map, they segmented foreground regions from input frame images by standard background subtraction method such as [48], and merged those foreground regions into 3D volumes with an efficient method which they proposed. However, including [43], the three methods mentioned are very sensitive to the result of background subtraction, which is used in the generation of the probability maps. Hence, their algorithms suffer from scenes that have dynamic or complex backgrounds.

1.2.2 Tracking-and-reconstruction methods

In this category, the existing algorithms have attempted to associate trajectories from a same target at each view. They first generate trajectories at each view with a single camera-based tracking method, and then formulate a combinatorial problem to associate those trajectories. Wu et al. [36] formulated the trajectory association problem as a multidimensional assignment problem, which is a well-known NP-hard problem. They solved the problem with a heuristic algorithm named greedy randomized adaptive local search procedure [49]. However, the methods in [36] can only handle short-term occlusion because they were originally proposed to track hundreds of flying objects observed as point measurements. Ayazoglu et al. [41] adopted a high order dynamic model in across view association of trajectories. The algorithm has a robust performance without calibration information even though targets have similar appearances. However, the algorithm suffers from a large amount of computations because a comparison between high order dynamic models needs an operation that finds the rank of a huge matrix.

1.2.3 Unified frameworks

Many of the recent studies have proposed a unified framework, which formulates the reconstruction and the tracking problem into one unified global optimization problem, to solve those two problems jointly. The framework achieved good performance in various scenarios with a batch processing over a whole input video sequence. Leal-Taixé et al. [32] tried MCMTT algorithm with a unified framework for the first time. The framework constructs a 2D tracking graph of each camera with detections from the camera, and constructs a 3D reconstruction graph with pairs of detections from different cameras. Then, an optimization problem is solved over two graphs as one unified min-cost flow formulation. However, the proposed graph structure is too complicated, and needs a rough estimate about the number of targets as prior information. To resolve those problems, Hofmann et al. [34] proposed a method based on a hypergraph, which can represent the reconstruction and tracking problem with a single graph. In this approach, all possible reconstructions between simultaneous detections must be enumerated to construct the graph. Furthermore, the approach solves the graph by a binary integer problem formulation, a well-known NP-hard problem, and its exact solver. Thus, the algorithm has a severe computational load. Despite the robust performance on benchmarks, including crowds of more than ten people, the algorithms in [32, 34, 44] are all batch-based algorithms, so they cannot provide an instantaneous tracking result at each frame, and require a huge number of computations. It is a serious limitation for many applications.

1.3 Contents of the Research

As mentioned in the previous section, most of the unified framework algorithms have been proposed in batch schemes, and thus, they have limited applicability. In this dissertation, we propose an online MCMTT algorithm in unified framework. We assume that the object detections from a video sequence are given. Our algorithm associates input object detections to the corresponding targets and estiamtes each target's trajectory in an online manner. An online scheme has a crucial limitation that it has less chances to resolve ambiguities in the tracking than a batch scheme because an online scheme cannot utilize the future detections. To resolve a tracking ambiguity problem arising from densely distributed targets, we adopt the multiple hypothesis tracking (MHT) framework [50] based on a deferred decision. In MHT, track hypotheses (or simply called 'track'), which are the estimated trajectories of targets, are enumerated with all possible data associations between input detections. Among them, a set of tracks that best describes the tracking of targets is selected as the tracking solution. To identify which track is a good track, the quality of each track is measured by our score function considering motion, appearance, and geometric information. The tracking solution is, then, a set of tracks that has the maximum total score. Although Kim et al. [25] showed that MHT performs well the multiple target tracking at the single camera, it is still challenging to moderate the computational complexity of MHT with detections from multiple cameras. In particular, it is an NP-hard combinatorial problem to find a solution in MHT framework for MCMTT.

To reduce the computational load of MHT for multiple cameras, we use tracklets—partial fragments of estimated target trajectories—on the two-dimensional (2D) image coordinates. The tracklets are generated through temporal associations between detections from the consecutive frames by utilizing appearance information. Then, the three-dimensional (3D) candidate tracks are generated by associating tracklets with their motion and appearance information. The use of tracklets instead of detections to generate candidate tracks preventes a generation of many absurd tracks. In the proposed scheme, the tracklets in each view are assembled in 3D space by a back projection based on a ground plane assumption. In a ground plane assumption, all targets are assumed to move on a 3D virtual plane called a ground plane. With this assumption and the camera network calibration information, we can get the 3D location for each tracklet without any triangulation. Thus, our 3D association problem is simplified to a 2D association problem on the ground plane.

To get the solution, we formulate the problem of finding the best track set as the maximum weighted clique problem (MWCP), finding a complete subgraph (or clique) for an arbitrary, undirected graph with the maximum total weights of its edges or vertices. MWCP is one of the widely adopted formulations for a combinatorial problem that has the pairwise compatiblity relationship among the variables. The corresponding graph of our MHT framework consists of vertices corresponding candidate tracks, and edges representing the compatibility of two tracks. Compared to the algorithms for a single camera case [51, 52], our formulation introduces additional compatibility conditions to prevent ID switches between densely distributed targets. Moreover, the algorithm assigns the weights on vertices with our carefully designed score function for 3D candidate tracks. Our score takes into account not only geometric information, but also motion and appearance information, while the scores used in state-of-the-art MCMTT methods [32, 34] only consider geometric information.

The MWCP is a well-known NP-hard problem. As Kim et al. [25] showed, it is hard to find an exact solution during a limited time even when a graph is constructed with a single camera. To alleviate the huge compuational load, we propose an online scheme that dynamically formates multiple MWCPs with small-sized subsets of candidate tracks in every frame. The scheme is motivated by that the tracking solutions from consecutive frames are very similar because the status of each target is not abruptly changed between one frame. When we assume that a specific track set is an actual solution of the previous frame, only a small number of tracks have a possibility to become a solution track of the current frame. Thus, we can narrow down the size of candidate track set with the previous solution. However, propagating only the best solution of each frame can cause irreducible error when a wrong track set is chosen as the solution because of the tracking ambiguity. To hedge the risk of this error, we find multiple good solutions at each frame and propagate the K-best solutions among them to the next frame instead of a single solution. When the candidate tracks are updated and generated with newly obtained detections at the next frame, we generate multiple subsets of the entire candidate tracks with the K-best previous solutions. Each subset consists of candidate solution tracks with respect to each of the previous solutions, and a small-sized MWCP is formated with the subset. Then, our algorithm finds multiple solutions from each MWCP and repeats above procedures until the tracking is terminated. Even the proposed algorithm solves multiple MWCPs, it has lower computational complexity than solving a single MWCP with the entire candidate tracks because the overall computational load is mainly affected by the size of the largest MWCP. Moreover, when an instantaneous result is demanded, our algorithm finds better solution than solving a single large-sized MWCP because it finds more diverse solutions under a limited solving time.

Even if we utilize the dynamic formation of small-sized MWCPs, it is still challenging to solve each problem with an exact algorithm within a reasonable time with respect to the practical applications. For further computation reduction in solving each MWCP, we apply an iterative heuristic algorithm called breakout local search (BSL) [53], which is a state-of-the-art heuristic algorithm for MWCP. BLS not only finds a near-optimal solution rapidly but also generates multiple local optimum solutions for our online scheme when it is slightly modified.

After finding multiple solutions, the resultant solutions are utilized in our pruning scheme to remove unreliable tracks. We calculate an approximated global track probability (AGTP) of each track which represents the quaility of each track with respect to the overall tracking situation instead of an individual track. Thus, utilizing this probability ensures a proper pruning of candidate tracks. We control the total number of candidate tracks according to AGTP. We also restrict the number of candidate tracks for each target with AGTP.

1.4 Thesis Organization

In this section, we provide an organization and overview of subsequent chapters of this thesis. In Chapter 2, as for the preliminaries, we briefly review a MHT framework with its variations. We examine the difference between each MHT framework and solving schemes for them. We also review a MWCP, which is a key formulation of our MHT based MCMTT algorithm. In particular, we introduce BLS, a state-of-the-art heuristic solver for the MCWP. Chapter 3 addresses the proposed MCMTT algorithm based on a MHT framework and a MWCP formulation. We present how the MHT framework can be extended to the multi-camera case. And how an an optimal tracking result can be efficiently found by a MWCP formulation and its heuristic solver, BLS. Chapter 4 presents experimental details including both quantitative and qualitative results of the proposed algorithm. We tested an influence of each component of our tracking algorithm on the performance by self-comparison. We also compared the tracking performance between ours and state-of-the-art methods. In Chapter 5, we conclude by summarizing the contributions of our work and briefly mentioning the direction of our future research.

Chapter 2

Preliminaries

In this chapter, we present the theoretical background of multi-target tracking algorithm including the multiple hypothesis tracking (MHT) framework, which is the baseline of the proposed method. We also present the maximum weighted clique problem (MWCP) that is utilized in our formulation to realize MHT framework for multiple camera multiple target tracking (MCMTT). To provide the concrete concept of our solving procedure, we describe the breakout local search (BLS) algorithm, which is one of the state-of-the-art heuristic solving algorithms for MWCP. If the reader who already familiar with these contents, please skip this chapter and proceed to the next chapter. For details and theoretical proofs that are not written in this chapter, please refer to the cited literatures.

2.1 Bayesian Tracking

In this section, we introduce a Bayesian tracking framework which is most widely adopted to formulate a tracking problem as a probabilistic inference. Tracking a target is to know the position of the target at the specific time. Usually, the information about the target more than a position is needed such as a target's speed, acceleration, scale, etc, for a reliable tracking because that help to predict the target's position at an upcoming time and validate the current tracking result. In a Bayesian tracking, that information is represented by a vector called *state*. In the field of classical mechanics, a *state* is a vector containing all information about what we want to know from the system. This investigated system is usually a target in a tracking problem.

If we have an ideal localization sensor which can exactly report a target's location, we can represent the sequence of target's state as a deterministic process which has only one possible 'reality' at each time step. However, this kind of direct measurement cannot be obtained in an ordinary case. Instead, we have to track the target with indirect and noisy sensor measurements. Thus, we have to estimate the state sequence of the target with a stochastic process, or often random process, which represents the target's state of each time as a random variable. Inferencing the sequence of target's state with this stochastic approach is a classical estimation problem in mechanics and it is called a state estimation problem.

Let $X_t \in \mathbb{R}^{n_x}$ denotes the target's state at time t where n_x indicates the dimension of the state. Then, *state estimation problem* is to estimate X_t through a specific time domain with an assumption that X_t is stochastically generated by a probability density function $p(X_t)$, and X_t is constant during the measurement obtaining process, which is called a scan. A *measurement* is another fundamental vector of a state estimation problem which represents the observations related to a target's state. Let $Z_t \in \mathbb{R}^{n_z}$ denotes the measurement from the target at time t where n_x indicates the dimension of the measurement.

Without any measurements, we can make a joint probability density function of the states $p(X_1, ..., X_T)$ with prior knowledge. For example, in case of a ship tracking, we know that the ship cannot sail on the ground and it usually on sea routes. We also narrow down the interval of available velocity of the ship. In many practical tracking problem, the prior probability $p(X_1, ..., X_T)$ is carefully modeled on manual, or by the machine learning techniques with a number of data. Naturally, the prior probability is not sufficient for a reliable tracking.

When a set of measurement $\mathbf{Z} = \{Z_1, Z_2, ..., Z_T\}$ is available, we can define a conditional probability of the sequence of target's state given the measurements. This conditional probability is called a *posterior* of a state sequence, and it can be obtained with a Bayes rule as

$$p(X_1, ..., X_T | Z_1, ..., Z_T) = \frac{p(Z_1, ..., Z_T | X_1, ..., X_T) \times p(X_1, ..., X_T)}{p(Z_1, ..., Z_T)}, \tag{2.1}$$

where $p(Z_1, ..., Z_T | X_1, ..., X_T)$ is the likelihood function measuring how "likely" the states are given the measurements that have been made. $p(X_1, ..., X_T)$ is the prior joint probability density function of states which is aforementioned. $p(Z_1, ..., Z_T)$ is the prior joint probability density function of measurement, often called 'evidence'. The evidence does not depend on the states and it is a constant value when the measurements are fixed. Thus, in many cases, it is regarded as a normalizing constant. For convenience, we use the subscript notation $X_{1:T}$ to indicate $(X_1, ..., X_T)$. Then, the equation 2.1 can be rewritten

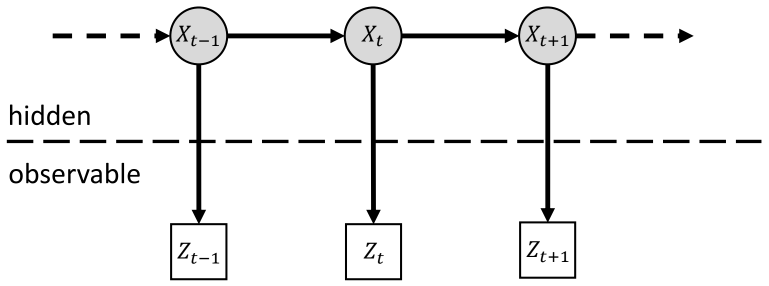


Figure 2.1: Hidden Markov model for the dynamic system. The states are shaded to represent that they are not observable. Each state depends only on the just prior state.

as

$$p(X_{1:T}|Z_{1:T}) = \frac{p(Z_{1:T}|X_{1:T}) \times p(X_{1:T})}{p(Z_{1:T})}. \tag{2.2}$$

The tracking is, then, to find the most probable state sequence of the target based on the equation 2.2. It is a batch-based tracking that estimating the whole sequence of target's state at once with the measurements from the whole time domain. However, in case of an online tracking, the state of the target has to be estimated whenever the new measurement arrives only with the measurements obtained up to the current time. That is, the measurements from later time cannot be available in an online tracking. In the following section, we will introduce a recursive Bayesian tracking with a hidden Markov model which is employed by a number of recent online tracking algorithms.

2.1.1 Recursive Bayesian Tracking

A recursive Bayesian tracking is the tracking via a recursive Bayesian estimation of a target's state. A recursive Bayesian estimation is to estimate an unknown probability density function recursively over time using incoming measurements and a mathematical process model such as a markov model. A Markov model is a stochastic model about a state transition with a Markov assumption. A Markov assumption is the assumption that the probability distribution over the possible next states depends on only the current state. That is, the next state is conditionally independent from all the states at the previous time steps when the current state is determined. This property is one of the “memorylessness” property, and called the Markov property.

A *hidden Markov model* (HMM) is a Markov model with the hidden states, which cannot be observed directly and only can be estimated by the stochastically generated measurements that are dependent on the hidden states. As mentioned in the previous section, direct measurements of a target’s state are not available in the realworld tracking problem. Therefore, an HMM is most widely adopted by the recursive Bayesian tracking algorithms. The Figure 2.1 depicts the graphical model of HMM. Compared to a simpler Markov model, an HMM additionally has the conditional probability distribution over the possible outputs given the hidden states. All random variables in the model are dependent on each other. However, when a specific hidden state is determined, its measurement and the next state are conditionally independent from the all preceding states and measurements. That is, when the previous state is determined, then the current state can be estimated with the current measurement which is observable and the previous state which is already determined. Thus, we can recursively estimate the sequence of target’s state with an HMM as a Kalman filter [45] which tracks the target in the linear system in an online manner.

2.1.2 Bayesian Tracking for Multiple Targets

In the case of tracking multiple targets, directly applying aforementioned recursive Bayesian tracking is suffered from the uncertainty in the ownerships of measurements caused by the clutters or other targets. The term ‘clutter’ refers to the detections produced by weather, electromagnetic interference, acoustic anomalies, nearby objects, etc, that are generally random in number, location, and intensity [54]. The clutter induces false positives in the measurements, which degrades the tracking performance. Using the measurements from other targets to estimate the current target’s state also drops the tracking performance down. Thus, recognizing that which measurement from which target is essential for the multiple target tracking when we track an individual target with a recursive Bayesian tracking.

A *track* is a hypothetical trajectory estimated from a set of measurements that is supposed to be from the same target. To determine the ownership of measurements in the multiple target tracking is equivalent to solve the measurement-track association problem. When the measurements are newly obtained, they are associated to each track and tracks are updated with associated measurements. There are many robust single target tracking algorithms [45,46] and they operate well with proper measurements. However, when a wrong measurement is given, their performance is severely degraded. Thus, the measurement-track association is the most critical phase in the multiple target tracking with respect to the overall tracking performance.

There are two different approaches to the measurement-track association problem. One is a stochastical approach called the joint probabilistic data association filter (JPDAF) [55]. In JPDAF, the conditional probability of a measurement given a track is calculated with respect to the possibility that the measurement is from the target. Then, each measurement contributes to the updating of the track according to its probability. JPDAF performs well in the moderate tracking scenario, but it is easily degraded when the targets are densely distributed. In JPDAF, a track is affected by the proximate measurements that are from other targets. Thus, tracks of JPDAF in the crowded scene tend to be merged.

The second one is the multiple hypothesis tracking (MHT) [50] framework which deterministically associates measurements to tracks but in many possible scenarios. The term 'hypothesis' refers to one of the possible measurement-track association. In each hypothesis, each measurement can be associated with at most one track and each track can be associated with at most one measurement. MHT enumerates all possible hypotheses and propagates them to hedge the risk of the wrong association. At each frame, the MHT selects the best hypothesis that best describes the tracking of targets. Because MHT considers all possible measurement-track associations, it performs better than JPDAF in the crowded scene. Thus, many practical tracking applications have adopted MHT instead of JPDAF. However, MHT has huge computational complexity. In the next section, we will present more details of MHT and how the previous works have reduced the computational complexity of MHT.

2.1.3 Multiple Hypothesis Tracking (MHT)

The multiple hypothesis tracking (MHT) framework was proposed to avoid the performance degradation caused by a wrong measurement-track association which instantly occurs when there is an ambiguity in the association. MHT has a huge computational load but promising practical results and the simple concept. Moreover, because of the recent dramatic increases in computational capabilities, MHT is the most preferable data association method for modern multiple target tracking systems. MHT is based on two major assumptions. First, it assumes that the measurement-track association is the matching problem, that is, each track can have at most one measurement and each measurement can be associated with at most one track. Second, MHT assumes that the ambiguity

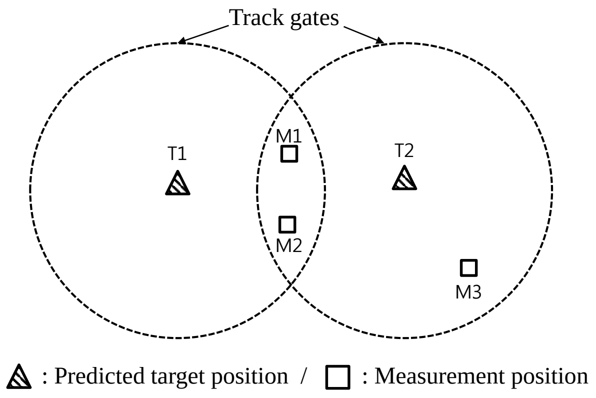


Figure 2.2: Example of typical data association conflict situation [56]. Track gate indicates the possible range of measurement-track association with respect to each target. When measurements are obtained from the intersection region of different track gates, MHT enumerates all possible measurement-track associations and propagates them until the uncertainty of the association is resolved.

in the measurement-track association at the specific time will be resolved with the future measurements within a small temporal domain. The term 'hypothesis' in MHT representing the hypothesis on the measurement-track association as mentioned in the previous section. Whenever measurement-to-track conflict situations occur, such as shown in Fig. 2.2, MHT hedges the risk of a wrong association by generating and propagating alternative measurement-track hypotheses, and decides the current tracking result at the future when the uncertainty is resolved. Because MHT defers to decide the tracking result for a few time steps, it is called a deferred decision logic. It can be compared to the semi-batch processes, but in ordinary cases, the processing window's size of the semi-batch processes are much larger than the deferred time of MHT.

In HMT framework, there exist two main terminologies which are 'global hypothesis' and 'track'. A *track* is a hypothetical estimation of the sequences of target' state, as mentioned in the previous section. Tracks are defined to be compatible if they do not have any common measurements. A *global hypothesis* is a set of compatible tracks that are induced by a measurement-track association. There are two alternative frameworks in MHT approach according to the way of generating global hypotheses: 'hypothesis-oriented' MHT (HO-MHT) and 'track-oriented' MHT (TO-MHT).

The first development of a complete algorithmic approach proposed by Reid et al. [50] is HO-MHT, which generates global hypotheses of the current scan by enumerating all feasible associations between measurements of the current scan and tracks in a hypothesis of the previous scan. The generation of global hypotheses in this manner can be formulated as the matching problem between tracks in a global hypothesis and newly obtained measurements. In HO-MHT, the probability of an individual track is obtained through marginalization of probabilities of hypotheses which contain the track. The probability of hypothesis is defined as different formulations in each methods. But the most favorable one is based on a log likelihood ratio (LLR) of each track, proposed in the pioneering paper by Sittler [57]. A likelihood ratio (LR) of the formation of a given set of measurements $\mathbf{Z}$ into a track can be defined by the recursive form:

$$LR = \frac{P(\mathbf{Z}|H_1)P_0(H_1)}{P(\mathbf{Z}|H_0)P_0(H_0)} \triangleq \frac{P_T}{P_F}, \tag{2.3}$$

where H_1 and H_0 are hypotheses for each the true target and false positive, respectively. That is, H_1 assumes that the measurements are from an actual target while H_0 assumes that the measurements are all false positives. P_T and P_F indicate the probabilities of H_1 and H_0, respectively. Then, $P(\mathbf{Z}|H_i)$ can be defined as a probability density function evaluated with the received measurements under the assumption that H_i is correct, and $P_0(H_i)$ can be defined as a priori probability of H_i, such as expected density of true targets in a given area.

In here, false positive does not mean a measurement from other targets (also called false target) but from non-persistent clutter. Because the equation 2.3 has a priori probabilities, LR is not a likelihood ratio but a probability ratio. However, we will refer it as a likelihood ratio for following the original formulation of [57]. For a convenience, a log likelihood ratio (LLR) or a track score defined as below have been mostly used instead of LR:

$$LLR = \log\left(\frac{P_T}{P_F}\right). \tag{2.4}$$

Then, the probability of a true target can be directly obtained from LLR

$$\begin{aligned} \frac{P_T}{P_F} &= \frac{P_T}{1-P_T} = \exp(LLR) \\ P_T &= \frac{\exp(LLR)}{1+\exp(LLR)}. \end{aligned} \tag{2.5}$$

The probability of a global hypothesis $\mathcal{H}_i$ is calculated with the sum of LLR of all tracks contained in the hypothesis:

$$P(\mathcal{H}_i) = \frac{\sum_{\mathcal{T}_j \in \mathcal{H}_i} LLR(\mathcal{T}_j)}{\sum_{\mathcal{H}_j \in \mathbb{H}} P(\mathcal{H}_j)} \tag{2.6}$$

where $\mathcal{T}_j$ represents the $\mathrm{j_{th}}$ track and $\mathbb{H}$ indicates the union set of global hypotheses. With the probabilities, the tracking result of HO-MHT can be found by the best match between tracks and measurements which induces the global hypothesis having the maximum probability. Finding a single best match, or the limited number of best matches have the linear time complexity with a matching algorithms such as Hungarian method [58] and the K-best Hungarian method [59]. However, finding all matches is an NP-hard problem and has the exponential time complexity.

The alternative framework to HO-MHT is TO-MHT which is proposed by

Kurien et al. [60]. It is proposed to resolve the memory efficiency problem of HO-MHT. In HO-MHT, a number of identical tracks are generated by multiple global hypotheses. To avoid this, TO-MHT discards all previous hypotheses and instead maintains tracks in tree structures (i.e., track trees) and updating those trees at the every receipt of new measurements. Then, TO-MHT formates global hypotheses by finding compatible track sets from the entire track trees. It is certain that TO-MHT consumes less memory than HO-MHT. However, the combinatorial problem to find compatible tracks from the entire track tree is an NP-hard problem.

In both frameworks, there is a potential combination explosion in the number of (global or track) hypotheses [60]. Thus, whichever the framework is used, the reduction of all the unlikely hypotheses at every scan is essential for the practical applications. It can be done by screening, which filters out the unlikely hypotheses before the generation such as gating, clustering, classification. Otherwise, pruning that removes the unlikely hypotheses after the generation such as an N-scan back approximation can enforce the reduction of hypotheses. Hypotheses reduction techniques are varied by each of MHT based tracking algorithms according to their target environment. Please refer to the literature for more details.

MHT has been adopted by many visual tracking algorithms [25, 52, 61, 62] Cox et al. [61] proposed HO-MHT algorithm to track feature points in the video sequence. In their paper, they introduced Murty's K-best Hungarian algorithm [59] to approximate the ideal global hypotheses generation. Papageorgiou et al. [52] proposed a visual tracking algorithm based on TO-MHT. They employed the maximum weighted independent set problem, which will be discussed in the next section, to formulate the problem finding compatible tracks from the track trees. Because of their formulation, they cannot exactly solve their tracking problem in the linear time. However, they did not provide any approximation method. Ren et al. [62] proposed to solve TO-MHT with a greedy randomized adaptive search procedure (GRASP) [63]. All of the above methods are designed to track the targets with a single camera. Up to our knowledge, there is no MHT for multiple camera cases. When we extend MHT to the multiple camera case, totally new formulation and track-instantiation method are required. For instance, finding the best global hypothesis in HO-MHT is no longer linearly solvable when the measurements are obtained from multiple cameras. Although the classical tracking area considered MHT with multiple sensors, they mainly concerned about track-to-track fusion, not the unified framework, which considers the multi-sensing issue at the measurement level.

2.2 Maximum Weighted Clique Problem (MWCP)

In this section, we introduce a combinatorial optimization problem of which the variables have the compatibility between them.

2.2.1 Clique Problems

When an arbitrary undirected graph $\mathcal{G} = V, E$ is given with its vertex set V and edge set E, a clique C is a subset of V which has edges between every pair of vertices in it. That is,

$$\forall u, v \in C, \{u, v\} \in E. \tag{2.7}$$

A clique C is also called a complete subgraph of $\mathcal{G}$ because a complete graph is a graph consisting of vertices that are all adjacent. There are two important types of cliques: the maximum clique and the maximal clique. Among the possible cliques in $\mathcal{G}$, a maximum clique is a clique has the largest number of vertices. The number of vertices in the maximum clique is called a clique number of $\mathcal{G}$. Another important clique, a maximal clique is a clique that cannot be enlarged, meaning it is not a subset of a larger clique. A maximum clique is always a maximal clique, but the converse does not hold.

The maximum clique problem (MCP) is to find the maximum clique in a given graph $\mathcal{G}$. Finding the maximum clique is very difficult because MCP is a well-known NP-complete problem. In contrary, a maximal clique can be found easier than the maximum clique. It can be found in the linear time, that is, the computational time of finding the maximal clique is linearly proportion to the graph size. However, listing all possible maximal cliques in a given graph is also NP-complete. Listing all possible maximal cliques and MCP are the famous instances of the clique problem.

Because of its applicability, the MCP is utilized in many practical applications such as a social network. Let's assume that we want to make a social marketing only with people who know each other because of maximizing the viral impact. We can define an acquaintance with vertices representing our customers and edges indicating whether two customers are acquaintances or not. Then, the largest targeting group can be found by the maximum clique in the acquaintance graph. MCP is also adopted in the coding theory. It was utilized to make the robust codes to an incomplete message transfer.

Given a graph having positive weights on its vertices (or edges), a maximum weighted clique problem (MWCP) is to find a clique that has the maximum total weight. It can be regarded as a generalized problem of MCP, which solves MWCP with a graph having unit weights on its vertices. Back to the social marketing example, we can score each person with the marketing effect on him or her. Then, the best marketing group can be found by the acquaintance graph with weights, which are defined by the marketing effect. Because MWCP is the more generalized problem than MCP, its applicability is larger than MCP.

2.2.2 Solving MWCP

There are many formulations for MWCP, but the most famous one is an integer programming formulation called *edge formulation*:

$$
\begin{aligned}
&\max_{x} \sum_{i \in V} w_i x_i, \\
&s.t.\ x_k + x_l \leq 1, \forall \{k, l\} \in \bar{E}, \\
&x_i \in \{0, 1\}, i = 1, ..., |V|,
\end{aligned}
\tag{2.8}
$$

where x indicates the vector that contains decision variables determining the inclusion of each vertex in the solution. The complement edge set $\bar{E}$ is the set of all vertex pairs that are not adjacent in the original graph. Although MWCP has a huge applicability, it is hard to apply MWCP to the practical applications because solving the equation 2.8 is not that easy. It is a well-known NP-complete problem, which means that the computational complexity exponentially increases when the graph enlarges. Despite the fact that many exact algorithms have improved the worst-case time complexity of solving MWCP [64–66], they are easily degraded by the large graphs. For this reason, many powerful heuristic algorithms have been proposed [53, 67–70] and applied to many practical issues. A breakout local search (BLS) [53] is one of the state-of-the-art heuristic algorithms for MWCP which outperforms the other competitive algorithms on the benchmark dataset with respect to the exactness of the solution and the solving time. We will introduce BLS in the following section because we adopt BLS to solve our MCMTT problem which is formulated as MWCP.

There is an important mathematical concept called an independent set, which is closely related with a clique. An independent set is a set of vertices in V, no two of which are adjacent. It is an inverse of a clique's definition. A maximum independent set problem (MISP) is to find an independent set with the largest cardinality. Similar to MWCP, there is a generalization of MISP called a maximum weighted independent set problem (MWISP). Let's define a complement graph of G by $\bar{G} = (V, \bar{E})$. By definition, a clique in G is an independent set in $\bar{G}$ and then, MISP and MCP are mathematically equivalent. When a graph has a large number of edges, solving MWISP with the complement graph is an efficient alternative from the viewpoint of the practical implementation. Thus, many applications vary their formulation depending on their graph's topology.

Many multi-target tracking algorithms utilzes MWCP or MWISP for their formulations [51, 52, 71, 72]. In particular, Papageorgiou et al. [52] proposed the solving scheme of MHT with MWISP formulation. However, Papageorgiou et al. [52] did not give any practical solving techniques which are necessary for the practical implementation. In contrary, other methods utilizing MWCP [51, 71, 72] proposed heuristic solving techniques, but their tracking graphs are different from ours. In this thesis, we construct a tracking graph with tracks while they build a graph with measurements. Thus, a clique indicates the overall tracking result in our formulation, but, is an individual track in their formulations.

2.3 Breakout Local Search (BLS)

The breakout local search (BLS) [53] is a heuristic solving algorithm for both MCP and MWCP. Given an undirected graph $G = (V, E)$, when a positive weight w_v is associated to each vertex $v \in V$, the goal of BLS is to find a clique C having the largest total weight $W(C) = \sum_{v \in C} w_v$. Because the maximum clique can be considered as a special case of MWCP when a graph has unit weights on its vertices, our description in this section focuses on how BLS solves MWCP. In this section, we refer to a subset of V that is currently held by BLS as the current solution. We say that the solution is feasible when the set is a clique, that is a complete subgraph of V.

BLS is an iterative algorithm which consists of two phases: local search procedure and adaptive solution perturbation. The key concept of BLS is an adaptive perturbation which utilizes the trace of solution searching. When BLS has an arbitrary feasible solution, it finds a local optimum solution, the maximal weighted clique, in the vicinity of the holding solution by its local search procedure. In the local search procedure, the algorithm explores solutions by adding or replacing vertex whenever the resulting vertex set is a feasible solution. Then, BLS perturbs the current local optimum solution in an attempt to discover a better solution through the local search with that perturbed solution. The term 'adaptive' represents that the perturbation strategy varies according to the solution searching status. When the local search visits an identical solution repeatedly, BLS gradually increases the strength of its perturbation until it escapes from the basin of the current local optimum solution. The perturbation strength is reset when BLS finds another local optimum solution. More details will be discussed in the following sections.

2.3.1 Solution exploration

There are four movements in BLS to explore the solution space: 'insert', 'switch', 'remove', and 'perturb'. For these movements, Benlic et al. [53] defined four vertex sets and one ordered vertex pair set as follows:

C : the current solution.

$N(v)$: $\{i | i \in V, \{i, v\} \in E\}$, neighbors of a vertex v.

PA : $\{v | v \notin C$ s.t. $\forall u \in C, \{v, u\} \in E\}$, vertices that are adjacent to all vertices in C.

OC : $\{v | v \notin V \backslash C\}$, all vertices that are not included in C. '$\backslash$' indicates the subtract operation.

OM : $\{(v, u) | v \notin C$ and $u \in C, |N(v) \cap C| = |C| - 1, \{v, u\} \in E\}$. When u is removed from C, the insertion of v into C yields a feasible solution.

In Benlic et al. [53], the four movement are named as M_1, M_2, M_3, and M_4, respectively, and defined as follows:

M_1 : (insertion) Select a vertex $v \in PA$ and insert into C. The total weight of C increases by w_v.

M_2 : (switching) Select a vertex pair $(v, u) \in OM$. Then, remove u from C and insert v into C. The total weight of C increases by $w_v - w_v$.

M_3 : (removing) Select a vertex $v \in C$ and remove it from C. The total weight of C decreases by w_v.

M_4 : (perturbinb) Select a vertex $v \in OC$ such that $(w_v + \sum_{\{v,u\} \in E, u \in C} w_u) \geq \alpha \times f(C)$, where $f(C)$ indicates the total weight of C and $0 < \alpha < 1$. Then, repair the resulting C to be a clique by removing all vertices $x \in C$ such that $\{v, x\} \notin E$.

In Benlic et al. [53], $C \oplus m$ indicates the new solution induced by the move m on the current solution C.

During the local search phase, among $M_1 \cup M_2$, the best move, which induces the maximum increment of $f(C)$, is taken. When there is no possible movement increasing $f(C)$, the local search is terminated and returns C as the local optimum solution. Then, BLS compares the best solution C_{best} with C and updates C_{best} to C when $f(C)$ is bigger than $f_{best} = f(C_{best})$. In the adaptive perturbation phase, there are two types of perturbations called ‘directed’ and ‘random’ perturbation. The directed perturbation randomly takes a move among $M_1 \cup M_2 \cup M_3$. The random perturbation randomly takes a move among from M_4. Details about the perturbation strategies will be discussed in the next section.

2.3.2 Perturbation Strategies

BLS employs random and directed perturbations to escape from the local optimum that the algorithm is stagnated. The *directed perturbation* is based on the tabu search [73]. The tabu search leverages the exploration of the search by prohibiting the selection of moves that are recently taken. See [73] for more details about the tabu search. In the directed perturbation, BLS selects moves that minimize the total weight decrement, under the constraint that they are not prohibited by the tabu list TL, which records the prohibition time of each vertex. BLS manages its tabu list as the following way. Removing a vertex from C is always allowed. However, when a vertex is removed from C, insert that vertex into C is prohibited for γ iterations which is determined by

$$\gamma = \phi + rancom(|OM|), \tag{2.9}$$

where ϕ is a coefficient and *random* indicates the function that generates the random number between 1 to $|OM|$. The remaining iteration number to release the prohibition of each vertex is recorded in TL. The move prohibition is ignored only if the move leads to a new solution better than the best solution found so far. Benlic et al. [53] defined move set A for the directed perturbation as below

$$A = \{m | m \in M_1 \cup M_2 \cup M_3, prohibited(m) = false \text{ or } f(C \oplus m) > f_{best}\}, \tag{2.10}$$

where $prohibited(m)$ indicates whether m is prohibited by TL or not.

The *random perturbation* is performing moves randomly selected from M_4, and it is significantly stronger than the directed perturbation. The degree of perturbation can be adjusted by chaging the value of $\alpha(0 < \alpha < 1)$ because it affects M_4. When α is set to a small number, the random perturbation is very strong as a random restart. In contrary, α close to 1 induces insignificant perturbation.

As soon as the stagnation is detected in the search, BLS performs the random perturbation to escape the local basin. BLS determines the type of perturbation with the probability defined by ω, which counts the consecutive non-improving iterations. When ω increases, the probability of using the directed perturbation progressively decreases while that of applying the random perturbation increases. However, it was empirically found that the minimum of applications of the directed perturbation guarantees a good performance. Therefore, the probability of the directed perturbation is defined by

$$P = \begin{cases} e^{-\omega/T} & \text{if} e^{-\omega/T} > P_0, \\ e^{-\omega/T} & \text{otherwise}, \end{cases} \tag{2.11}$$

where T is the maximum allowable number of non-improving local optima visited before triggering a stronger perturbation. The overall procedure of BLS is summarized in the Algorithm 1. Also, the perturbation procedure and the perturbation operator are summarized in the Algorithm 2 and Algorithm 3, respectively.

2.3.3 Initial Solution and Termination Condition

The initial solution of BLS is generated in the following way. Fisrt, select uniformly at random a vertex $v \in V$ and insert it to C. Then, insert all of its neighbors that are also the neighbors of $\forall u \in C$. Repeat above two actions until there is no more vertices can be inserted into C, giving a valid clique.

The termination condition of BLS in the original works is the hitting of the maximum iteration number. When BLS reaches the maximum iteration number, all procedures are terminated and the best solution found so far is returned as an output. In [53], the maximum iteration number was set to a huge number such as a billion.

Algorithm 1 Breakout Local Search for MWCP [53].

```
Require: Graph $G = (V, E)$, weights $\{w_v | v \in V\}$, initial and maximal jump
    magnitud $L_0$ and $L_{Max}$, number $T$ of non-improving attractors visited be-
    fore strong perturbation, coefficients $\alpha_r$ and $\alpha_s$ for random and strong ran-
    dom perturbations.
Ensure: The maximum weighted clique.
 1: $C \leftarrow generate\_initial\_solution(G)$
 2: Creat initial $PM, OM$ and $OC$ vertex sets
 3: $f_c \leftarrow f(C)$                ▷ $f_c$ records the objective value of the solution
 4: $C_{best} \leftarrow C$               ▷ $C_{best}$ records the best solution found so far
 5: $f_{best} \leftarrow f_c$            ▷ $f_{best}$ records the best objective value reached so far
 6: $C_p \leftarrow C$                   ▷ $C_p$ records the last local optimum
 7: $\omega \leftarrow 0$                ▷ Set counter for consecutive non-improving local optima
 8: while stopping condition not reached do
 9:     Select the best move $m \in M_1 \cup M_2$
10:     while $f(C \oplus m) > f_c$ do
11:         $C \leftarrow C \oplus m$                 ▷ Perform the best-improving move
12:         $f_c \leftarrow f(C \oplus m)$
13:         Update $PM, OM$ and $OC$
14:         $TL \leftarrow update\_tabu\_list(m, Iter)$
15:         $Iter \leftarrow Iter + 1$
16:         Select the best move $m \in M_1 \cup M_2$
17:     end while
18:     if $f_c > f_{best}$ then
19:         $C_{best} \leftarrow C; f_{best} \leftarrow f_c$     ▷ Update the best solution found so far
20:         $\omega \leftarrow 0$   ▷ Reset counter for consecutive non-improving local optima
21:     else
22:         $\omega \leftarrow \omega + 1$
23:     end if
    /*Determine the perturbation strenght L to be applied to C */
24:     if P then $\omega > T$
    /*Search seems to be stagnating, strong perturbation required */
25:         $L \leftarrow L_{Max}$
26:     else
    /*Search escaped from the previous local optimum, reinitialize perturbation
    strength */
27:         $L \leftarrow L_0$
28:     end if
    /*Perturb the current local optimum C with perturbation strenght L */
29:     $C_p \leftarrow C$
30:     $C \leftarrow Perturbation(C, L, TL, Iter, \omega, \alpha_r, \alpha_s)$
31: end while
```

Algorithm 2 Perturbation procedure $Perturbation(C, L, TL, Iter, \omega, \alpha_r, \alpha_s)$ [53].

```
Require: Local optimum $C$, perturbation strength $L$, tabu list $TL$, global iteration counter $Iter$, number of consecutive non-improving local optima visited $\omega$, coefficients $\alpha_r$ and $\alpha_s$ for random and strong perturbations.
Ensure: A perturbed solution $C$.
1: if $\omega = 0$ then
   /*Best solution is not improved after a certain number of visited local optimum */
2:     $C \leftarrow Perturb(C, L, M_4)$        ▷ Strong random perturbation with moves from $M_4$ when $\alpha = \alpha_s$
3: else
4:     Determin probability $P$ according to the equation 2.11
5:     With probability $P$, $C \leftarrow Perturb(C, L, A)$
   /*Directed perturbation with moves from set $A$ */
6:     With probability $(1 - P)$, $C \leftarrow Perturb(C, L, M_4)$
   /*Random perturbation with moves from set $M_4$ when $\alpha = \alpha_s$ */
7: end if
8: return $C$
```

Algorithm 3 Perturbation operator $Perturb(C, L, M)$ [53].

```
Require: Local optimum $C$, perturbation strength $L$, tabu list $TL$, global iteration counter $Iter$, the set of perturbation moves $M$.
Ensure: A perturbed solution $C$.
1: for $i := 1 \text{to} L$ do
2:     Take move $m \in M$
3:     $C \leftarrow C \oplus m$        ▷ Apply move $m$ to $C$
4:     $TL \leftarrow update\_tabu\_list(m, Iter)$
5:     Update $PM, OM$ and $OC$
6:     $Iter \leftarrow Iter + 1$
7: end for
8: return $C$
```

Chapter 3

Proposed Approach

3.1 Problem Statements

The goal of our algorithm is to estimate trajectories of multiple targets from the given object detections in an online manner. A set of detections from all cameras is denoted by $\mathbf{D} = \{d_i | d_i = (l_i, s_i, c_i, t_i), i = 1, ..., N_{\mathbf{D}}\}$ where l_i, s_i indicate image coordinate location and scale, respectively, $c_i \in \{1, ..., N_C\}$ is a camera index, and t_i represents the time stamp when d_i is detected. $N_{\mathbf{D}}$ is the total number of input detections. A 2D tracklet is defined by a set of detections which are regarded as the successive measurements from the same target by the same camera. We define $\mathcal{Y}_j \in \mathbf{Y}$ as a j_{th} 2D tracklet by

$$\mathcal{Y}_j = \{d_i | i \in \mathbf{I}_{\mathcal{Y}_j}\}, \tag{3.1}$$

where $\mathbf{I}_{\mathcal{Y}_j}$ is an index set of the detections which belong to $\mathcal{Y}_j$. We assume that each detection cannot be shared by more than one target, which means that

$\mathbf{I}_{\mathcal{Y}_i} \cap \mathbf{I}_{\mathcal{Y}_j} = \phi$ is always satisfied for $i \neq j$. Details on the generation of 2D tracklets from the given detections will be discussed in Section 3.2.

A track is an estimated trajectory of a target in 3D world coordinates. It is generated by associating tracklets presumed to be generated from the same target. When we define $\mathbf{I}_{\mathcal{T}_k}$ as an index set of the tracklets associated to a track $\mathcal{T}_k$, the detection set of $\mathcal{T}_k$, $\mathbf{Z}_k \subset \mathbf{D}$, can be defined with the 2D tracklets in $\mathbf{I}_{\mathcal{T}_k}$ as

$$\mathbf{Z}_k = \bigcup_{j \in \mathbf{I}_{\mathcal{T}_k}} \mathcal{Y}_j. \tag{3.2}$$

Let us define $\mathbf{Z}_k^t = \{d_i | d_i \in \mathbf{Z}_k, t_i = t\}$ as a set of the track's detections observed at time t from all cameras. Letting x_k^t be the estimated 3D location of target at time t, a track $\mathcal{T}_k \in \mathbf{T}$ is defined as the sequence of these estimated locations:

$$\mathcal{T}_k = (x_k^{t_k^s}, x_k^{t_k^s+1}, ..., x_k^{t_k^e}), \tag{3.3}$$

where $t_k^s = \min(\{t_i | d_i \in \mathbf{Z}_k\})$ and $t_k^e = \max(\{t_i | d_i \in \mathbf{Z}_k\})$ are the initiating and the terminating time of $\mathcal{T}_k$, respectively. In Section 3.3, we will describe the details on the estimation of x_k^t from $\mathbf{Z}_k^t$ and the design of track $\mathcal{T}_k$'s score $S_{\mathcal{T}_k}$.

A global hypothesis $\mathcal{H}_n \in \mathbf{H}$ is a set of estimated trajectories of multiple targets, that is, a subset of $\mathbf{T}$. When we define $\mathbf{I}_{\mathcal{H}_n}$ as an index set of the tracks belong to $\mathcal{H}_n$, then $\mathcal{H}_n$ is defined by

$$\mathcal{H}_n = \{\mathcal{T}_k | k \in \mathbf{I}_{\mathcal{H}_n}\}. \tag{3.4}$$

For feasible global hypotheses, any two different tracks $\mathcal{T}_k$, $\mathcal{T}_l$ belonging to the same global hypothesis must satisfy the compatibility conditions given by:

1. no common tracklet in any two tracks:

$$\mathbf{I}_{\mathcal{T}_k} \cap \mathbf{I}_{\mathcal{T}_l} = \phi, \tag{3.5}$$

2. collision avoidance:

$$|x_k^t - x_l^t| \geq \theta_s, \ \ \forall t \in [\max(t_k^s, t_l^s), \min(t_k^e, t_l^e)], \tag{3.6}$$

where θ_s means the minimum distance required in order to avoid a collision between targets. Here, we define the compatibility set $\mathbb{C}$ which consists of unordered index pairs of compatible tracks, that is, $\{k, l\} \in \mathbb{C}$ for all k, l satisfying the above conditions. Multiple target tracking is to find the best global hypothesis $\mathcal{H}_*$ which has the maximum total score among feasible global hypotheses satisfying the compatibility conditions:

$$\begin{aligned} \mathcal{H}_* = \arg\max_{\mathcal{H}_n} \sum_{\mathcal{T}_k \in \mathcal{H}_n} S_{\mathcal{T}_k} \\ s.t. \ \{k, l\} \in \mathbb{C}, \quad \forall k, l \in \mathbf{I}_{\mathcal{H}_n}. \end{aligned} \tag{3.7}$$

Since the problem in the equation (3.7) is an NP-hard problem, in this paper, we aim to propose a novel online scheme to rapidly find a near-optimal solution of the equation (3.7) at every frame by utilizing the past frame's solutions. The Figure 3.1 depicts an overall scheme of the proposed method. It consists of four parts: tracklet, track, global hypothesis, and pruning.

At each camera, the tracklet part generates tracklets by associating detections through frames. To associate detections in an online manner, we formulate the association problem as a detection-to-tracklet matching problem. Then, we generate new tracklets or update established tracklets with the matching result. The resulting tracklets are passed to the track part. Details on the matching and the tracklet management will be described in Section 3.2.

The track part generates new candidate tracks with associations between

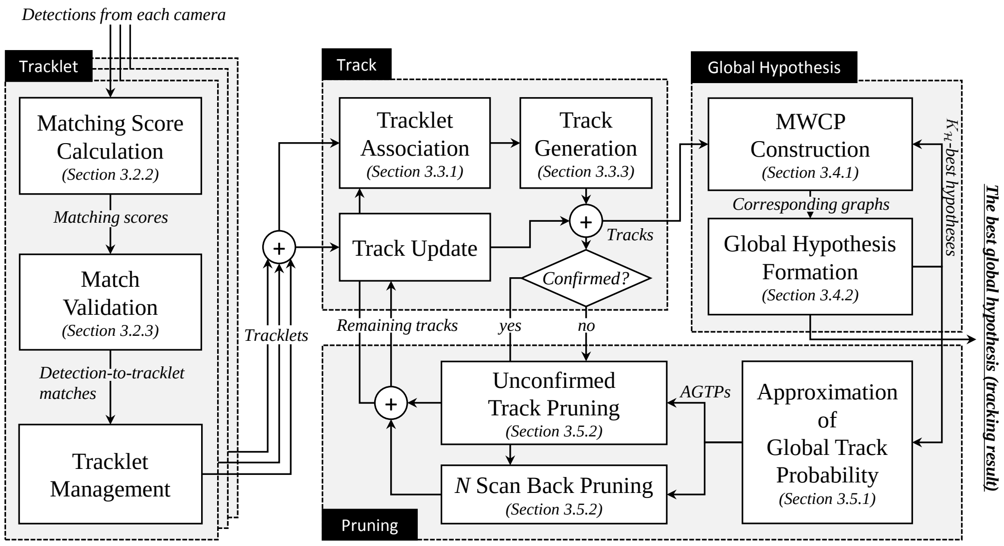

Detections from each camera
Tracklet
Matching Score Calculation (Section 3.2.2)
Matching scores
Match Validation (Section 3.2.3)
Detection-to-tracklet matches
Tracklet Management
Tracklets
Track
Tracklet Association (Section 3.3.1)
Track Generation (Section 3.3.3)
Track Update
Tracks
Confirmed?
Remaining tracks
yes
no
Global Hypothesis
MWCP Construction (Section 3.4.1)
Corresponding graphs
Global Hypothesis Formation (Section 3.4.2)
$K_{\mathcal{H}}$-best hypotheses
The best global hypothesis (tracking result)
Unconfirmed Track Pruning (Section 3.5.2)
AGTPs
Approximation of Global Track Probability (Section 3.5.1)
N Scan Back Pruning (Section 3.5.2)
Pruning

Figure 3.1: Overall scheme of the proposed method. The arrows indicate the information flows. The proposed online scheme utilizes the past tracking results in finding the current frame's tracking result.

tracklets and manages established tracks in an online manner. To reduce the number of candidate tracks, we check the proposed validation conditions for the spatial and temporal association between tracklets. Then, a score of each track is computed with a carefully designed score function. Details on the track generation procedure and the score function will be described in Section 3.3.

In the global hypothesis part, we solve the equation (3.7) for $\mathbf{T}^t$, the set of entire tracks which exist in the current frame. To reduce the computation, we generate subproblems of the equation (3.7) by referring $\mathbf{H}^{t-1} = \{\mathcal{H}_1^{t-1}, , ..., \mathcal{H}_{K_\mathcal{H}}^{t-1}\}$ which is the set containing the $K_\mathcal{H}$ best global hypotheses in the previous frame according to their total score. Then, we solve the subproblems instead of the original problem. Each subproblem is MWCP for $\mathbf{T}_n^t \subset \mathbf{T}^t$, a set of the tracks, which are candidates of the current best global hypothesis, with an assumption that the previous best global hypothesis was $\mathcal{H}_n^{t-1}$. To resolve the NP-hard issue in solving each MWCP, we adopt BLS with proposing a good initial solution and a proper iteration number. We also modify BLS to generate multiple near-optimal solutions. After gathering all global hypotheses found by solving subproblems, we pick $K_{\mathcal{H}}$ best global hypotheses into $\mathbf{H}^t$. $\mathbf{H}^t$ is stored in the global hypothesis part for the next frame and it is conveyed to the pruning part. Details on the construction of MWCPs and for solving each of them will be described in Section 3.4.

Table 3.1: Notable notations.

Symbol	**Description**
d_i	i_{th} detection at image coordinates l_i of camera c_i at time t_i with a scale s_i
$\mathbf{D}, N_D = \lvert\mathbf{D}\rvert$	set of all detections
$\mathbf{D}^t$	set of all detections which are detected at time t
$\mathcal{Y}_j \in \mathbf{Y}$	j_{th} tracklet
$\mathcal{Y}_j^t$	detection of $\mathcal{Y}_j$ at time t, i.e., $\mathcal{Y}_j \cap \mathbf{D}^t$
$\mathbf{Y}^t$	set of all tracklets continuing until time t
$\mathbf{I}_{\mathcal{Y}_j}$	index set of detections in $\mathcal{Y}_j$
$\mathcal{T}_k \in \mathbf{T}$	k_{th} track hypothesis
$\mathbf{T}^t$	set of all existing tracks at time t
x_k^t	estimated 3D location of $\mathcal{T}_k$ at time t
t_k^s, t_k^e	initiating and terminating time of $\mathcal{T}_k$, respectively
$S_{\mathcal{T}_k}$	score of $\mathcal{T}_k$
$\mathbf{Z}_k \subset \mathbf{D}$	set of detections associated to $\mathcal{T}_k$
$\mathbf{Z}_k^t$	set of detections at time t in $\mathbf{Z}_k$, i.e., $\mathbf{Z}_k \cap \mathbf{D}^t$
$\mathbf{I}_{\mathcal{T}_k}$	index set of tracklets associated to $\mathcal{T}_k$
$\mathbb{C}$	compatibility set containing unordered index pairs of all compatible tracks in $\mathbf{T}$
$\mathbb{C}^t$	compatibility set of $\mathbf{T}^t$ instead of $\mathbf{T}$
$\mathcal{H}_n \in \mathbf{H}$	n_{th} global hypothesis
$\mathcal{H}_*$	the best global hypothesis
$K_{\mathcal{H}}$	the maximum number of global hypotheses for the subproblem generation and track pruning
$\mathbf{H}^t$	set of $K_{\mathcal{H}}$ best global hypotheses of time t
$\mathbf{I}_{\mathcal{H}_n}$	index set of tracks in $\mathcal{H}_n$

In the pruning part, two pruning techniques are applied to tracks, depending on whether a track is confirmed or not. A track is confirmed when its duration is longer than a certain length. We compute an approximated global track probability (AGTP) of each track with $\mathbf{H}^t$ and use it in each pruning technique as a criterion. Then, the pruning information is passed to the track part for reducing the number of tracks in the next frame. The definition of AGTP and details on pruning techniques will be described in Section 3.5.

Before proceeding to the following sections, we summarize our notable notations with the Table 3.1. In the table, we also present notations for an online scheme, which are essential for the rest of this paper. We separate the notations into four groups related with inputs, tracklets, tracks, and global hypotheses, respectively.

3.2 Tracklet Generation

A tracklet was widely used as an intermediate solution or a mid-level input in many previous works [14–16, 22, 51, 74–77]. Ge and Collins [74] generated tracklets with the single target tracking algorithms such as a mean-shift [78] tracker and a particle filtering [46] within a small time interval, e.g., 30 frames. Each tracklet is initialized with an object detection result that is produced by an edge based head detector or a background subtraction method which is similar to the manner of Zhao et al. [79]. In the case of tracklets from the same target, the tracklets are not only temporally overlapped, but also spatially overlapped. Based on this fact, the final estimation of target's trajectory is made by associating tracklets with Monte-Carlo Markov chain data association.

Benfold at el. [75] generated tracklets by data association between consecutive detections. At each frame, the matching scores between detections are calculated by the forward and backward feature point tracking with the KLT feature tracking algorithm [80]. The maximum length of each tracklet is restricted under four seconds to ensure the reliability of them. In their paper, they do now explicitly describe how the matching is done with the feature point tracking result. They just revealed that more than four points have to be successively tracked through the detections in the same tracklet.

Roth et al. [76] proposed a tracklet generation in two states with detection inputs. Tracklets in the first stage are produced by matching detections from consecutive frames according to their size, location and pose affinities. Thus, they are short but very reliable. The resulting tracklets are then fed into the next stage as an input. In the second stage, tracklets are associated to each other according to three cues. The first cue is the discrete cosine transform (DCT) feature which is robust against illumination changes and occlusions. The second cue is an online learned discriminative appearance model (OLDAM) [81] that incrementally learns the appearance of a target. To train an OLDAM for each tracklet, pairs of detections in the tracklet are selected as the positive samples while pairs of detections that one is from the target tracklet and the other is from a co-occurring, but spatially distinct tracklet are collected as the negative samples. Then, the trained classifier determines whether newly obtained detection (or new tracklet) can be associated to the tracklet or not. The last cue is the smoothness in the motion and the pose variation. With matching similarities considering above cues, tracklets are associated with each other with Hungarian method [58] to generate longer trajectories.

Zamir et al. [51] hierarchically generated trajectories of targets and they called their intermediate results tracklets. In their work, an input sequence is segmented into a small number of consecutive frames and detections from each frame segments are partitioned into sets that all detections in each of them are from the same target. This partitioning problem is realized by a generalized minimum clique problem for those detections. In the clique problem, each edge has a cost between detections based on appearance and motion similarity. A color histogram and a constant velocity model were used for appearance and motion similarity, respectively. Because the association was done by successively finding minimal cliques in the graph, it cannot be guaranteed that the resulting tracklets are optimal. However, the algorithm was reliable on moderate scenarios.

Hofmann et al. [22] conservatively grouped a set of detections from consecutive frames to generate tracklets with three conditions. First, a tracklet has at most one detection at each frame. Second, a discontinuity in a tracklet is not allowed. In other words, each tracklet must have detection at every frame within its duration. Last, detections in the same tracklet must have the similar appearance. The appearance similarity between detections is measured by RGB histograms and Bhattacharyya distance. When detections from different tracklets are severely overlapped, the tracklets including those detections are terminated and new tracklets are started at the next frame.

Wen et al. [14] hierarchically generated tracklets by associating tracklets with solving a hypergraph-based combinatorial problem. Thus, the algorithm associates tracklets with considering a global relationship between whole tracklets while the most of the previous works have considered only the pairwise relationship between tracklets. From a single detection to a target's whole trajectory, the algorithm iteratively generates the higher level tracklets with affinities based on appearance, motion, and the smoothness that are similar with the affinities defined in [77]. A 36 dimensional HoG and 8 dimensional intensity histogram of each RGB channel are used in an appearance affinity. The motion affinity between tracklets is defined based on an assumption about a constant velocity of a target within a small time interval. For the trajectory smoothness affinity, the smoothed trajectory is estimated by the moving average of positions of detections in the trajectory over a small temporal domain. The trajectory smoothness affinity gets a high value when the difference between the smoothed and original trajectories is small.

In our framework, we also use tracklets to reduce the number of overall computations. Because our association algorithm does not have any strategy to recover from the wrong tracklets, the robustness of tracklets is crucial to the tracking performance of our algorithm. In this chapter, we present how we generate tracklets robustly with associating detections through successive frames at each camera, as defined in Section 3.1. Our tracklet generation is similar to the method proposed by Benfold at el. [75] that utilizes the KLT feature tracking algorithm [80] in forward and backward direction. However, to generate tracklets in an online manner, we reformulate the inter-frame association problem to a detection-to-tracklet matching problem with a newly defined matching score. We also apply matching validations on the matches between detections and tracklets to enhance the robustness of tracklets.

3.2.1 Detection-to-tracklet Matching

In our method, a tracklet is the set of consecutive detections obtained from the same camera as aforementioned in Section 3.1 in Chapter 1. Thus, making tracklets is equivalent to solving an association problem between detections from the same camera. In the case of a batch processing, it can be formulated as a single optimization problem with entire detections obtained from a whole input sequence. However, generating tracklets in an online manner has to incrementally associate input detections whenever new detections arrive as depicted in the Figure 3.2. Thus, we formulate a tracklet generation as a bi-partite

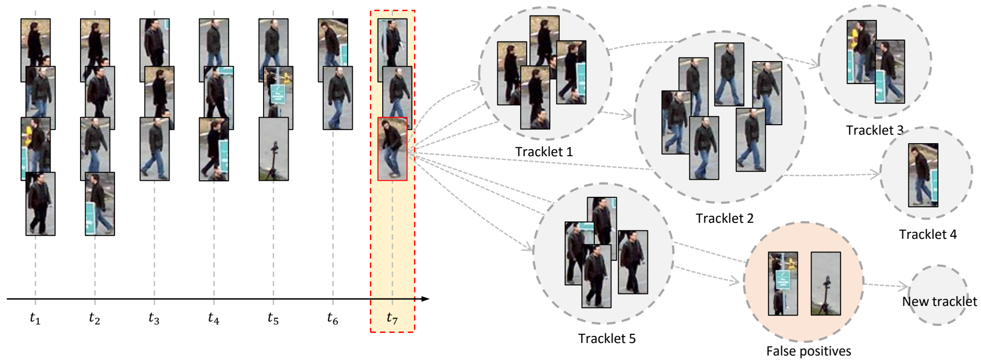


Figure 3.2: Incremental association for the online generation of tracklets. When detections are newly obtained, each of them is associated with one of the established tracklets or regarded as false positive or a new tracklet.

matching problem between established tracklets and newly obtained detections. A semi-batch approach, which associates detections in a small temporal domain from scratch, can be an alternative of our formulation. A semi-batch approach normally performs better than ours from the viewpoint of ordinary tracking measures such as durability and reliability. However, we discard a semi-batch approach because of following two reasons.

First, a semi-batch approach would change the previous portion of tracklets which ruins the efficiency of our whole tracking framework. When tracklets constituting a track are changed, the track has to be updated with the changed tracklets. Perhaps the track would not be valid anymore. In such a case, global hypotheses consisting of that track are also not valid anymore. Therefore, changing established tracklets yields high computational overhead.

Second, there is no need to consider a matching between detections more than one frame because our tracking framework needs extremely robust tracklets. Our tracking framework has no strategy recovering from wrong tracklets, so, if there is any uncertainty in a match between consecutive detections, those detections must not be associated into the same tracklet. That is, we only

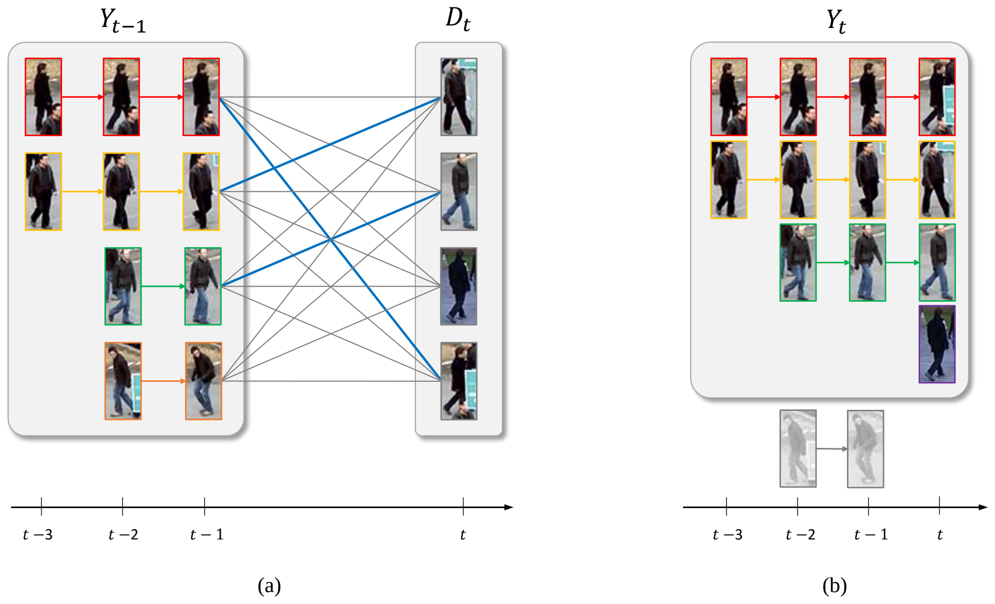


Figure 3.3: Example of bi-partite matching for tracklet generation. (a) A bi-partite graph at time t and its matching solution. The bi-partite graph having established tracks $\mathbf{Y}^{t-1}$ and new detections $\mathbf{D}^t$ its partite. Each edge of the graph has a matching score defined in the following section as its weight. As an example, we depict an optimal matching with the bold lines colored by blue. (b) Result of tracklet management with the bi-partite matching of the graph in (a). Tracklets having a matched detection are extended by a new detection while tracklets which do not have any matched detection are terminated. Each remaining detection initializes a new tracklet.

continue a tracklet when a match between the tracklet and a newly obtained detection is highly reliable. Under this extremely conservative association rule, the results of a semi-batch processing and our online matching are very similar.

In our tracklet generation scheme, we formulate a bi-partite matching problem with established tracklets and new detections as illustrated in the Figure 3.3(a). In the figure, $\mathbf{D}^t = \{d_i | t_i = t\}$ indicates the set of detections newly detected at time t and $\mathbf{Y}^{t-1}$ indicates the set of all tracklets of which the last detection has a time index $t-1$, that is,

$$\mathbf{Y}^{t-1} = \{\mathcal{Y}_j | \max(\{t_i | i \in \mathbf{I}_{\mathcal{Y}_j}\}) = t-1\}. \tag{3.8}$$

We construct a fully connected bi-partite graph that has $\mathbf{Y}^{t-1}$ and $\mathbf{D}^t$ as its partites. Each weight on the graph is defined by the matching score will be presented in Section 3.2.2. Then, we solve the matching problem of the bi-partite graph by Hungarian method [58] which finds an optimal matching in polynomial time. To ensure the robustness, we additionally validate the resulting matched between tracklets and detections as described in Section 3.2.3. After the validation, we update tracklets with valid matches as shown in the Figure 3.3(b). When a tracklet $\mathcal{Y}_j \in \mathbf{Y}^{t-1}$ is matched with a detection $d_i \in \mathbf{D}^t$, $\mathcal{Y}_j$ is updated as below

$$\mathcal{Y}_j \leftarrow \mathcal{Y}_j \cup \{d_i\}. \tag{3.9}$$

If there is no matched detection for a tracklet, the tracklet is terminated. When there is no tracklet matched to a current detection d_l, a new tracklet is generated with the detection:

$$\mathcal{Y}_n \leftarrow \{d_l\}. \tag{3.10}$$

3.2.2 Matching Score with Motion Estimation

In this section, we propose a matching score between a newly obtained detection and an established tracklet that is used in a bi-partite matching for the tracklet management as mentioned in the previous section. The matching score between a tracklet $\mathcal{Y}_j \in \mathbf{Y}^{t-1}$ and a detection $d_i \in \mathbf{D}^t$ is defined by

$$S_{j,i} = \frac{1}{L_c}\left(S_{box}(\hat{\mathcal{Y}}_j^t, d_i) + \sum_{n=1}^{L_c-1} S_{box}(\mathcal{Y}_j^{t-n}, \hat{d}_i^{t-n})\right), \tag{3.11}$$

$$S_{box}(d_p, d_q) = -\log\left(\frac{\|l_p - l_q\|_2}{(s_p + s_q)/2}\right), \tag{3.12}$$

where $\mathcal{Y}_j^t = \mathcal{Y}_j \cap \mathbf{D}^t$ is the detection at time t which is included by tracklet $\mathcal{Y}_j$. L_c is the length of the comparison interval. $\hat{d}_i^t$ and $\hat{\mathcal{Y}}_j^t$ are results of the bi-directional (forward and backward) tracking which is based on the motion estimation technique described in the following. $\hat{d}_i^{t'}$ is the estimation of d_i at time $t' = t_i \pm 1$ ($+1$: forward, -1: backward). We assume that the size of a target does not change abruptly between consecutive frames. Thus, $\hat{d}_i^{t'}$ is defined by

$$\hat{d}_i^{t'} = (l_i + \tilde{\delta}_i^{t'}, s_i, c_i, t'), \tag{3.13}$$

where $\tilde{\delta}_i^{t'}$ indicates the major disparity between t_i and t' that are estimated by the motion estimation described in the following. In the motion estimation, we extract feature points from d_i and track them with the KLT feature tracking algorithm [80] on the frame at time t'. Let us define $\delta_{i,j}^{t'}, j = 1, ..., q_i^{t'}$ as the disparity between the j_{th} successfully tracked feature point in d_i at time t_i and its tracking result at time t'. Then, the disparity set $\mathbf{\Delta}_i^{t'}$ is defined by

$$\mathbf{\Delta}_i^{t'} = \{\delta_{i,j}^{t'} | \left\|\delta_{i,j}^{t'}\right\|_2 > \delta_{min}\}, \tag{3.14}$$

where δ_{min} is a design parameter to reject the disparities from static feature points. When the cardinality of $\mathbf{\Delta}_i^{t'}$ is smaller than the half of the number of all tracked feature points, we determine that d_i does not move, so $\tilde{\delta}_i^{t'} = 0$. Otherwise, we find the major disparity which has the largest neighbor set

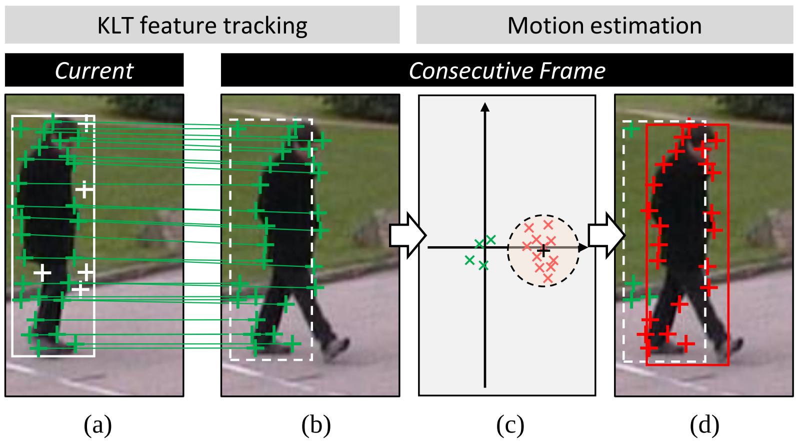


Figure 3.4: Motion estimation between consecutive frames. The goal of the motion estimation is to find a major disparity between feature points in consecutive frames. (a) Extraction of feature points from the detection. (b) Feature points tracking between the current and the consecutive frame. Green and white points represent successfully tracked points and lost points, respectively. A dashed box represents the original detection at the current frame. (c) Disparities between feature points in consecutive frames. The black cross is the major disparity and red ones are its neighbors defined by the equation (3.15). (d) The result of the motion estimation is depicted by the red box.

defined by

$$\mathcal{N}_{i,j}^{t'} = \{\delta_{i,k}^{t'} | \forall \delta_{i,k}^{t'} \in \mathbf{\Delta}_i^{t'} \ s.t. \ ||\delta_{i,j}^{t'} - \delta_{i,k}^{t'}||_2 < w_\delta(s_i)\}, \tag{3.15}$$

where $w_\delta(s_i)$ indicates the neighbor window size which is proportioned to s_i. Then, the major disparity is defined by

$$\tilde{\delta}_i^{t'} = \underset{\delta_{i,j}^{t'} \in \mathbf{\Delta}_i^{t'}}{\arg\max} |\mathcal{N}_{i,j}^{t'}|. \tag{3.16}$$

By using forward tracking in the above, $\hat{\mathcal{Y}}_j^t$ can be obtained from $\mathcal{Y}_j^{t-1}$, i.e., the last detection of the tracklet in the previous frame. We summarize our

Algorithm 4 Motion estimation

```
Require: Previous location $l_p$, previous scale $s_p$, disparities of feature points
    $\mathbf{\Delta}$, the minimum movement distance $\delta_{min}$, neighbor window size ratio $\gamma_w$
Ensure: New location $l_n$
 1: $w \leftarrow s_p \times \gamma_w$                          ▷ size of neighbor window
 2: $\Delta_m \leftarrow \phi$                                  ▷ set of non-static disparities
 3: $n_\mu \leftarrow 0$                                        ▷ number of major dispariies
 4: $\delta_\mu \leftarrow (0, 0)$                              ▷ major disparity
 5: for $\delta_i$ in $\mathbf{\Delta}$ do                      ▷ non-static disparities to $\Delta_m$
 6:     if $\|\delta_i\|_2 \geq \delta_{min}$ then
 7:         $\Delta_m \leftarrow \Delta_m \cup \{\delta_i\}$
 8:     end if
 9: end for
10: if $|\Delta_m| \geq 0.5 \times |\Delta|$ then
11:     for $\delta_i$ in $\mathbf{\Delta}_m$ do
12:         $n_w \leftarrow 0$                                  ▷ number of current neighbors
13:         for $\delta_j$ in $\mathbf{\Delta}_m$ do
14:             if $\|\delta_i - \delta_j\|_2 < w$ then
15:                 $n_w \leftarrow n_w + 1$
16:             end if
17:         end for
18:         if $n_w > n_\mu$ then                               ▷ update major disparity
19:             $\delta_\mu \leftarrow \delta_i$
20:             $n_\mu \leftarrow n_w$
21:         end if
22:     end for
23: end if
24: $l_n \leftarrow l_n + \delta_\mu$                           ▷ Update the new location
25: return $l_n$
```

motion estimation in the Algorithm 4 and depict an example of the motion estimation in the Figure 3.4.

3.2.3 Matching Validation

At each frame, we match detections and tracklets by the Hungarian method [58], with scores defined in the equation (3.11). To enhance the robustness of track-

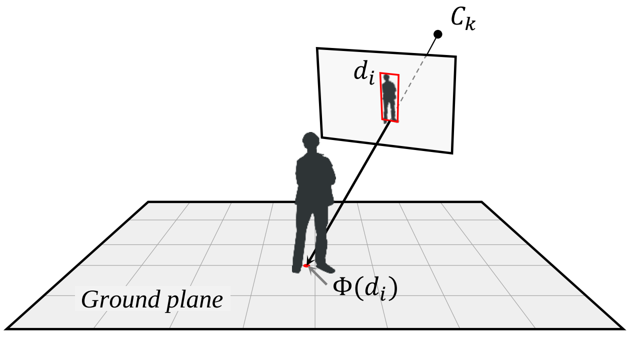


Figure 3.5: Back projection of a detecion d_i onto the 3D ground plane. Detection's image coordinate l_i indicates the bottom center of a bounding box of the detection. Then, a 3D location of the detection is obtained by a back projection function $\Phi(d_i)$.

lets, we validate each match with two 3D geometric conditions in the following.

The first condition is about the distance in 3D space between the matched detection and the matched tracklet's last detection. We assume that the distance must be close enough when the match is valid. To measure the 3D distance between detections, we have to know the 3D position of each detected pedestrian with a single detection. We resolve the depth ambiguity arises from a single detection by assuming that all pedestrian move on a specific 3D plane as mentioned in Chapter 1 and depicted in the Figure 3.5. With the assumption and a Tsai camera calibration model [82], we can define a back projection function $\Phi(d_i)$ transferring the image coordinates l_i of camera c_i to the coordinates on the 3D ground plane. If the match between the detection d_i and the tracklet $\mathcal{Y}_j$ is valid, d_i and $d_k = \mathcal{Y}_j^{t-1}$, the last detection of $\mathcal{Y}_j$, must satisfy the condition below

$$|\Phi(d_k) - \Phi(d_i)| \leq \varepsilon_\Phi, \tag{3.17}$$

where ε_Φ is an allowable maximum 3D distance between consecutive detections in the same tracklet and is a design parameter related with the frame rate of an input video and the average moving speed of targets.

The second condition is about the estimated height of a detected object. We assume that the height of target does not change abruptly, thus d_i and $d_k = \mathcal{Y}_j^{t-1}$ are likely to have similar heights when the match between them is valid. Let us define $\hbar$ as the function that estimates the height of a detected object. Then, d_i and d_k must satisfy the condition below to ensure the validity of the match between d_i and $\mathcal{Y}_j$

$$|\hbar(d_k) - \hbar(d_i)| \leq \varepsilon_{\hbar}, \tag{3.18}$$

where $\epsilon_{\hbar}$ is a design parameter about the maximum allowable variation in target's height between consecutive frames.

3.3 Track Hypothesis

As mentioned in Section 3.1, a track (hypothesis) is an estimated trajectory by combining tracklets from the same target. However, it is very challenging to determine ownerships of tracklets without any target information, including the exact number of targets. To resolve this, we generate all possible tracks through spatial-temporal association of tracklets until the current frame, and find the optimal tracks among them by solving the optimization problem in Section 3.4. In this section, we describe an online generation scheme of tracks and propose a track score representing the quality of each track. Furthermore, we also define a track tree which represents a hierarchical relationship between tracks.

3.3.1 Tracklet Association

The data association between tracklets is to determine which tracklets are generated from the same target. Through the associations, the entire tracklets are

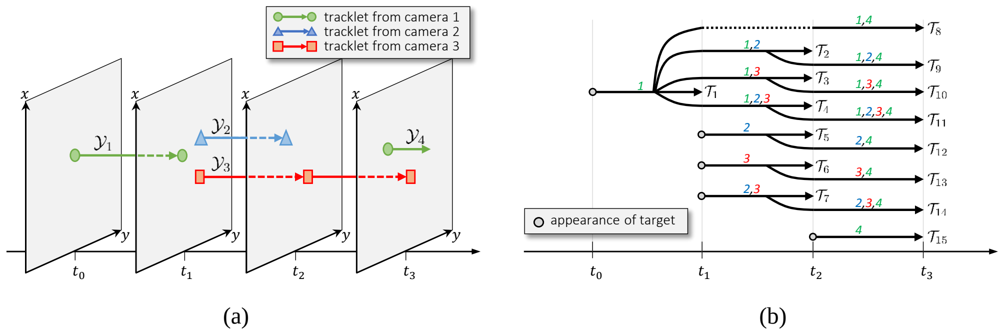


Figure 3.6: Example showing the track generation. (a) (Back-projected) Tracklets from three cameras during four frames. The gray colored planes represent the ground plane in the world coordinates at each time. (b) Four track trees that are generated by associating tracklets in (a). The colored numbers on each track indicate the tracklets that are assigned to the track (i.e., *association set*). The dashed line in $\mathcal{T}_8$ represents the time gap in a temporal association, i.e., the missing of detections.

partitioned into a multiple number of subsets. Each subset is assumed to be related to a target or a false alarm. A false alarm is the tracklet which is generated by non-target clutter. Let $\{\mathbf{\Omega}^1(\mathbf{Y}), \mathbf{\Omega}^2(\mathbf{Y}), ...\}$ be the collection of all possible partitions of an entire tracklet set $\mathbf{Y} = \{\mathcal{Y}_1, ..., \mathcal{Y}_q\}$. Then, the goal of our MCMTT problem is to find a partition $\mathbf{\Omega}^i(\mathbf{Y})$ which best describes the tracking of targets. The i_{th} partition is defined by $\mathbf{\Omega}^i(\mathbf{Y}) = \{\omega^i_\phi(\mathbf{Y}), \omega^i_1(\mathbf{Y}), ..., \omega^i_{n_i}(\mathbf{Y})\}$. $\omega^i_\phi(\mathbf{Y})$ is the set of false alarms which we call the *false alarm set*. $\omega^i_k(\mathbf{Y}), k = 1, ..., n_i$ is the set of tracklets supposed to be from the k_{th} target in the i_{th} partition, which we call the *association set*. Note that a track can be generated in a deterministic way when the corresponding association set is given. Thus, enumerating all possible association sets is equal to enumerating all possible tracks.

We define the universe set of all possible association sets without partition index as below

$$\begin{aligned}\mathbf{\Omega}(\mathbf{Y}) &= \bigcup_{i=1,2,...} \mathbf{\Omega}^i(\mathbf{Y})\backslash\{\omega^i_\phi(\mathbf{Y})\} \\ &= \{\omega^1_1, ..., \omega^1_{n_1}, \omega^2_1 ...\} \\ &:= \{\omega_1, ..., \omega_{n_1}, \omega_{n_1+1}, \omega_{n_1+2}, ...\}.\end{aligned} \tag{3.19}$$

Here, we omit the argument '$(\mathbf{Y})$' for convenience. We notate the corresponding track of ω_k as $\mathcal{T}_k$ and the set of detections associated with ω_k as $\mathbf{Z}_k$. To describe our online generation scheme of association sets, we define three operations: *spatial association*, *temporal association* and *merge*.

Spatial Association

As $\mathcal{T}_2$ and $\mathcal{T}_7$ in the Figure 3.6(b), spatial association is defined by an association between temporally overlapping tracklets, which are supposed to be from the same target with different cameras. To determine whether tracklet $\mathcal{Y}_l$ and $\mathcal{Y}_m$ are from the same target or not, we propose the spatial association condition that checks if the distance between simultaneous detections in tracklets is bounded by ε_{3D} with a back projection function Φ described in Section 3.2.3, as below

$$\begin{aligned}&||\Phi(\mathcal{Y}^t_l) - \Phi(\mathcal{Y}^t_m)||_2 \leq \varepsilon_{3D}, \\ &\qquad \forall t \in \{t_i | \forall d_i \in \mathcal{Y}_l\} \cap \{t_j | \forall d_j \in \mathcal{Y}_m\}.\end{aligned} \tag{3.20}$$

Temporal Association

Temporal association is defined by an association between any temporally not overlapped tracklets in the same camera or from different cameras that are supposed to be from the same target. For temporal association, the preceding tracklet $\mathcal{Y}_l$ and the succeeding tracklet $\mathcal{Y}_m$ must satisfy following two conditions.

Condition 1: According to the fact that each target can generate at most one detection in each view, $\mathcal{Y}_l$ and $\mathcal{Y}_m$ must be temporally non-overlapped. That is,

$$\{t_i | d_i \in \mathcal{Y}_l\} \cap \{t_j | d_j \in \mathcal{Y}_m\} = \phi. \tag{3.21}$$

Condition 2: A target cannot change its location abruptly, therefore d_p, the last detection of $\mathcal{Y}_l$, and d_n, the first detection of $\mathcal{Y}_m$, are close enough in 3D space. When v_{max} is defined as the maximum distance that a target can move during one frame, the condition is given by

$$\|\Phi(d_p) - \Phi(d_n)\|_2 \leq v_{max} \times |t_p - t_n|. \tag{3.22}$$

Merge

According to the fact that each target is assumped to generate at most one detection in each view at each time, an association set does not allowed to have temporally overlapped tracklets when they are from the same camera. That is, if the tracklets $\mathcal{Y}_l, \mathcal{Y}_m \in \omega_i$ are generated from the same camera, they must be temporally non-overlapped. If the union set of two association sets ω_i and ω_j satisfies all of spatial and temporal association conditions, and the condition above, the union set is also an association set. Those two association sets are called as *mergeable* sets. Based on this, the merging operation '$\oplus$' between them is defined by

$$\omega_i \oplus \omega_j = \begin{cases} \omega_i \cup \omega_j, & \omega_i \text{ and } \omega_j \text{ are mergeable,} \\ \phi, & \text{otherwise.} \end{cases} \tag{3.23}$$

Since tracklets in each ω_i and ω_j already satisfy all association conditions, we only have to check the satisfaction of the conditions between a tracklet from

ω_i and a tracklet from ω_j to determine whether the two association sets are mergeable or not. Note that, the meaning of '$\oplus$' is different from that in the preliminary about BLS (Section 2.3).

3.3.2 Online Generation of Association Sets

In this section, we describe how to generate $\mathbf{\Omega}^t$ with utilizing the set of association sets established until the previous frame $\mathbf{\Omega}^{t-1}$ and the set of newly generated tracklets at the current frame $\mathbf{Y}^t_{new} = \{\mathcal{Y}_i | \min\{t_j | d_j \in \mathcal{Y}_i\} = t\}$. At first, we generate $\mathbf{\Omega}^t_{new}$, the set of all possible association sets generated only with $\mathbf{Y}^t_{new}$. In $\mathbf{\Omega}^t_{new}$, there are association sets with a single tracklet in $\mathbf{Y}^t_{new}$ and association sets with at least two tracklets which satisfy the spatial association condition in the equation (3.20). After generating $\mathbf{\Omega}^t_{new}$, we generate $\mathbf{\Omega}^t_{\oplus}$ by merging association sets in $\mathbf{\Omega}^{t-1}$ with mergeable association sets in $\mathbf{\Omega}^t_{new}$. But to reduce the computational complexity, we do not merge association sets if the frame gap between them is larger than δ_a. That is,

$$\begin{aligned} \mathbf{\Omega}^t_{\oplus} = \{\omega_i \oplus \omega_j | \forall \omega_i \in \mathbf{\Omega}^{t-1}, \forall \omega_j \in \mathbf{\Omega}^t_{new} \\ s.t.\ \omega_i \oplus \omega_j \neq \phi \text{ and } |t^s_j - t^e_i| \leq \delta_a\}. \end{aligned} \tag{3.24}$$

Then, $\mathbf{\Omega}^t$, the set of association sets established up to the current frame, is defined by

$$\mathbf{\Omega}^t = \mathbf{\Omega}^{t-1} \cup \mathbf{\Omega}^t_{new} \cup \mathbf{\Omega}^t_{\oplus}. \tag{3.25}$$

With $\mathbf{\Omega}^t$, we generate tracks by the method will be described in in the next section. We summarize our online scheme for track generation in the Algorithm 5.

If an association set $\omega_k \in \mathbf{\Omega}^t_{\oplus}$ is the result of a merge operation with $\omega_i \in \mathbf{\Omega}^{t-1}$ and $\omega_j \in \mathbf{\Omega}^t_{new}$, then ω_k and ω_i are two association sets of the same target but differ in the current measurement association. Thus, their corresponding

Algorithm 5 Online generation of track hypotheses

```
Require: Previous track set $\mathbf{T}^{t-1}$, previous association set $\mathbf{\Omega}^{t-1}$, newly generated tracklet set $\mathbf{Y}^t_{new}$, the maximum distance for the spatial association $\varepsilon_{3D}$, the maximum velocity of pedestrian $v_{max}$.
Ensure: current track set $\mathbf{T}^t$
    /∗spatial association with new tracklets ∗/
 1: $\mathbf{\Omega}^t_{new} \leftarrow \phi$
 2: for $\mathcal{Y}_i \in \mathbf{Y}^t_{new}$ do
 3:     $\mathbf{\Omega}^t_{new} \leftarrow \mathbf{\Omega}^t_{new}\{\mathcal{Y}_i\}$
 4: end for
 5: $\mathbf{\Omega}_1 \leftarrow \phi$
 6: $\mathbf{\Omega}_2 \leftarrow \phi$
 7: for $c \in \{1, ..., C\}$ do
 8:     for $\omega_i \in \mathbf{\Omega}^t_{new}$ do
 9:         for $\omega_j \in \mathbf{\Omega}_1$ do
10:             if $\omega_i \oplus \omega_j \neq \phi$ then
11:                 $\mathbf{\Omega}_2 \leftarrow \mathbf{\Omega}_2 \cup \{\omega_i \oplus \omega_j\}$
12:             end if
13:         end for
14:     end for
15:     $\mathbf{\Omega}_1 \leftarrow \mathbf{\Omega}_2$
16: end for
17: $\mathbf{\Omega}^t_{new} \leftarrow \mathbf{\Omega}_1$
    /∗spatial-temporal association for track continuation ∗/
18: $\mathbf{\Omega}^t_{\oplus} \leftarrow \phi$
19: for $\omega_i \in \mathbf{\Omega}^{t-1}$ do
20:     for $\omega_j \in \mathbf{\Omega}^t_{new}$ do
21:         if $\omega_i \oplus \omega_j \neq \phi$ then
22:             $\mathbf{\Omega}^t_{\oplus} \leftarrow \mathbf{\Omega}^t_{\oplus} \cup \{\omega_i \oplus \omega_j\}$
23:         end if
24:     end for
25: end for
    /∗track estimation ∗/
26: $\mathbf{T}^t \leftarrow \mathbf{T}^{t-1}$
27: for $\omega_k \in \mathbf{\Omega}^t_{new} \cup \mathbf{\Omega}^t_{\oplus}$ do
28:     $\mathcal{T}_k \leftarrow estimate\_track(\omega_k)$                ▷ see Section 3.3.3
29:     $\mathbf{T}^t \leftarrow \mathbf{T}^t \cup \{\mathcal{T}_k\}$
30: end for
31: return $\mathbf{T}^t$
```

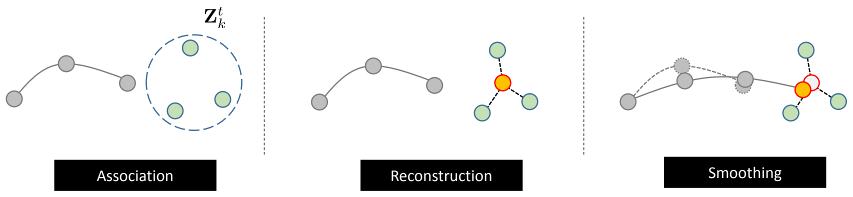


Figure 3.7: Steps for track generation. When an association set ω_k is given, $\hat{x}_k^t$ is calculated by reconstruction and x_k^t is estimated by smoothing on the reconstructed locations.

tracks $\mathcal{T}_k$ and $\mathcal{T}_i$ are incompatible. That is, they cannot become optimal tracks at the same time. $\mathcal{T}_i$ is called the *parent track* of $\mathcal{T}_k$ whereas $\mathcal{T}_k$ becomes the *child track* of $\mathcal{T}_i$. A track is incompatible with not only its parent but also all of its ancestor tracks. Those incompatibilities between tracks are essential for the global hypotheses formation at Section 3.4 and for the track pruning at Section 3.5. We depict an example of those relationship with a hierarchical structure referred to as a *track tree* in the Figure 3.6(b).

3.3.3 Track Generation

When an association set ω_k is given, we can determine the detection set of $\mathcal{T}_k$ at time t, $\mathbf{Z}_k^t = \{d_i | t_i = t \text{ and } d_i \in \mathcal{Y}_j \text{ where } \mathcal{Y}_j \in \omega_k\}$. With $\mathbf{Z}_k^t$, the location of track $\mathcal{T}_k$ at time t, x_k^t, is estimated by two steps: *reconstruction* and *smoothing*. Those two steps are depiceted in the Figure 3.7. In this section, the definition of *reconstruction* is to generate the estimated 3D location $\hat{x}_k^t$ of a track $\mathcal{T}_k$ at time $t \in [t_k^s, t_k^e]$. When $\mathbf{Z}_k^t$ is a non-empty set, $\hat{x}_k^t$ is defined by the geometric center point of $\mathbf{Z}_k^t$ as below

$$\hat{x}_k^t = \frac{1}{|\mathbf{Z}_k^t|} \sum_{d_i \in \mathbf{Z}_k^t} \Phi(d_i). \tag{3.26}$$

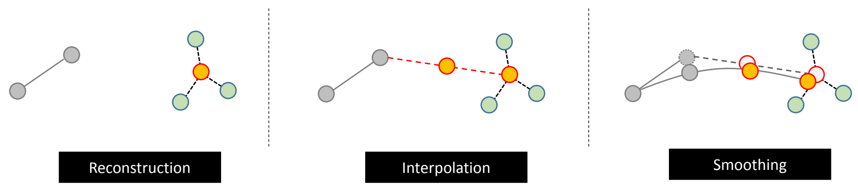


Figure 3.8: Track generation with a time gap. When there is a time gap in the track, unobserved locations are estimated by interpolation and smoothing.

If the target is not detected by any camera at time $t \in (t_k^s, t_k^e)$, $|\mathbf{Z}_k^t| = 0$. In this case, we estimate $\hat{x}_k^t$ by interpolation of the adjacent two reconstructed locations as depicted in the Figure 3.8. Let us t_p and t_n denote the closest preceding and following time from t, which have a non-empty detection set, respectively. Then, the reconstructed 3D location at time t is defined by linear interpolation:

$$\hat{x}_k^t = \hat{x}_k^{t_p} + \frac{t - t_p}{|t_n - t_p|} \left(\hat{x}_k^{t_n} - \hat{x}_k^{t_p} \right). \tag{3.27}$$

$\hat{x}_k^t$ found in the reconstruction step is independent from other 3D locations of $\mathcal{T}_k$ at different time. However, adjacent locations are highly correlated with each other because the target moves under a specific motion. We do *smoothing* on reconstructed 3D locations to consider those dependencies. When all individual reconstructed 3D locations of $\mathcal{T}_k$ are found, x_k^t, the final 3D location of $\mathcal{T}_k$ at time t, is obtained by

$$x_k^t = \mathcal{F}((\hat{x}_k^{t_k^s}, ..., \hat{x}_k^{t_k^e}), t), \quad t = t_k^s, ..., t_k^e, \tag{3.28}$$

where '$\mathcal{F}(\cdot, t)$' is a function which returns the smoothed location at time t. We used Savitzky–Golay filter [83] for this smoothing in our experiments.

3.3.4 Track Score

We propose a score for each track while considering five factors. The first one is a reconstruction score $S_{\mathcal{R}}(\cdot)$ representing how the track's locations are identical to detections. The second one is a linking score $S_{\mathcal{L}}(\cdot)$ which considers the geometrical suitability of the consecutive locations of the track. The third and fourth ones are an initiation score $S_{\mathcal{I}}(\cdot)$ and a termination score $S_{\mathcal{T}}(\cdot)$. Each of them evaluates the suitability of the starting or the ending location of the track. When the track starts or ends far from boundaries of the visible area or entrances, the track has a low initiation or termination score. The last one is a visual score $S_{\mathcal{V}}(\cdot)$ representing the visual similarity between detections associated to the track. Then, the track score is defined by those factors as

$$\begin{aligned} S_{\mathcal{T}_k} &= S\left(\mathcal{T}_k, \omega_k\right) \\ &= \sum_{t=t_k^s}^{t_k^e} S_{\mathcal{R}}\left(x_k^t, Z_k^t\right) + \sum_{t=t_k^s}^{t_k^e-1} S_{\mathcal{L}}\left(x_k^t, x_k^{t+1}\right) \\ &+ S_{\mathcal{I}}\left(x_k^{t_k^s}\right) + S_{\mathcal{T}}\left(x_k^{t_k^e}\right) + S_{\mathcal{V}}(\omega_k). \end{aligned} \tag{3.29}$$

Reconstruction Score $S_{\mathcal{R}}$

The proposed reconstruction score is based on $P_Z(\cdot)$, a likelihood of detection set on the target at a specific time. Letting X_k^t be the random variable standing for the location of the target tracked by $\mathcal{T}_k$ at time t, the reconstruction score is defined by

$$\begin{aligned} & S_{\mathcal{R}}\left(x_k^t, Z_k^t\right) \\ & \quad = \log\left(P_Z(Z_k^t | X_k^t = x_k^t)\right) - \log\left(P_Z(Z_k^t | X_k^t \neq x_k^t)\right). \end{aligned} \tag{3.30}$$

We give a penalty to the score with the second term when the target does not exist on Z_k^t at time t. This penalty term helps us to exclude false positive tracks in solving the MWCP in Section 3.4.

The likelihood is defined as:

$$P_Z(Z_k^t|X_k^t) = P_{vis}(Z_k^t|X_k^t) \times P_{rec}(Z_k^t|X_k^t), \tag{3.31}$$

where the first term is a visibility term representing the detection probability of Z_k^t, and the second term is a reconstruction term representing the error between Z_k^t and x_k^t in 3D space. Here, we assume that the two terms are independent of each other. The visibility term is defined in two circumstances depending on the existence of the target on x_k^t at time t. If the target exists on x_k^t, detections in Z_k^t are all true positives. By contrast, they are all false positives when the target does not exist on x_k^t. Letting γ_{fp} and γ_{fn} be the false positive ratio and the false negative ratio of the object detector, respectively, the visibility term of Z_k^t is defined by

$$P_{vis}(Z_k^t|X_k^t = x_k^t) = (1-\gamma_{fp})^{|Z_k^t|}(\gamma_{fn})^{n(x_k^t)-|Z_k^t|}, \tag{3.32}$$

$$P_{vis}(Z_k^t|X_k^t \neq x_k^t) = (\gamma_{fp})^{|Z_k^t|}(1-\gamma_{fn})^{n(x_k^t)-|Z_k^t|}, \tag{3.33}$$

where $n\left(x_k^t\right)$ indicates the number of cameras covering x_k^t by their field of views. When $|Z_k^t| > n\left(x_k^t\right)$, the track $\mathcal{T}_k$ is regarded as an invalid track. The reconstruction term $P_{rec}(Z_k^t|X_k^t)$ is based on the reconstruction error, which is defined by

$$\varepsilon_{rec}(x_k^t, Z_k^t) = \frac{1}{|Z_k^t|} \sum_{d_i \in Z_k^t} \left\| \Phi(d_i) - x_k^t \right\|_2 . \tag{3.34}$$

To determine how large the allowable reconstruction error is, we borrow the maximum allowable reconstruction error from [22] which considers a calibration error and an object detection error as below

$$\varepsilon_{rec}^{max}(Z_k^t) = \varepsilon_{det} \cdot \sum_{d_i \in Z_k^t} \|\Theta(d_i)\| + \varepsilon_{cal}, \tag{3.35}$$

where ε_{det} represents the maximum allowable pixel error between the bottom center of the detection box and the actual grounding location of the target. ε_{cal} is a parameter about calibration error and it represents the maximum allowable distance between the back projections of each camera's image coordinates, which indicate the common 3D location, onto the 3D ground plane. The projection sensitivity function $\Theta(d_i)$ [22] indicates the variation of coordinates on the 3D ground plane, which is induced by one pixel variation around an image coordinates l_i of the camera c_i. Using the equation (3.34) and (3.35), the reconstruction term of the likelihood is defined by

$$\begin{aligned} P_{\mathcal{R}_k^t} :=& P_{rec}(Z_k^t | X_k^t = x_k^t) \\ =& \begin{cases} \frac{1}{2}\, erfc\left(4\frac{\varepsilon_{rec}(x_k^t, Z_k^t)}{\varepsilon_{rec}^{max}(Z_k^t)} - 2\right), & |Z_k^t| > 1, \\ \frac{1}{2}, & \text{otherwise,} \end{cases} \end{aligned} \tag{3.36}$$

$$P_{rec}(Z_k^t | X_k^t \neq x_k^t) = 1 - P_{\mathcal{R}_k^t}. \tag{3.37}$$

where '$erfc(\cdot)$' is the complementary error function which is known as one minus the (Gauss) error function [84]. '$erfc(\cdot)$' returns a large value on a small input, so the reconstruction term becomes larger when the reconstruction error of x_k^t is small. Here, we give 0.5 for the case of a single detection because it is impossible to get the reconstruction error, so we do not have any information about it. Using the equation (3.30) and (3.32)-(3.36), the reconstruction score can be rewritten as

$$\begin{aligned} S_{\mathcal{R}}\left(x_k^t, Z_k^t\right) =& \log\left(\frac{P_Z(Z_k^t|X_k^t = x_k^t)}{P_Z(Z_k^t|X_k^t \neq x_k^t)}\right) \\ =& \left(n\left(Z_k^t\right) - |Z_k^t|\right) \times \log\left(\frac{\gamma_{fn}}{1-\gamma_{fn}}\right) \\ &+ |Z_k^t| \log\left(\frac{1-\gamma_{fp}}{\gamma_{fp}}\right) + \log\left(\frac{P_{\mathcal{R}_k^t}}{1-P_{\mathcal{R}_k^t}}\right). \end{aligned} \tag{3.38}$$

Linking Score $S_{\mathcal{L}}$

The motion of a pedestrian is hard to predict because it is usually non-linear. Therefore, we consider only the proximity of consecutive locations in the track to determine whether linking those locations is proper or not. The proposed linking score is defined to become bigger as the distance between consecutive locations in the track becomes smaller. That is,

$$S_{\mathcal{L}}\left(x_k^t, x_k^{t+1}\right) = \log\left(\frac{1}{2} erfc\left(4\frac{\left\|x_k^t - x_k^{t+1}\right\|_2}{v_{max}} - 2\right)\right), \tag{3.39}$$

where v_{max} is a design parameter modeling the maximum distance that a pedestrian can move during one frame. When $\left\|x_k^t - x_k^{t+1}\right\|_2$ is bigger than v_{max}, the linking is regarded as an invalid linking, so we discard the track.

Initiation/Termination Score $S_{\mathcal{I}}$, $S_{\mathcal{T}}$

As previously mentioned, a track suddenly initiating or terminating in the middle of the visible area is less probable. To reflect this tendency, the initiation/termination scores are defined by

$$S_{\mathcal{I}}\left(x_k^t\right) = \log\left(P_s\right) - \tau_s \times \max\left(0, \mathcal{B}\left(x_k^t\right) - m_{\mathcal{B}}\right), \tag{3.40}$$

$$\begin{aligned} S_{\mathcal{T}}\left(x_k^t\right) = & \log\left(P_e\right) - \tau_e \times \max\left(0, \mathcal{B}\left(x_k^t\right) - m_{\mathcal{B}}\right) \\ & - \tau_l \times \left(t_k^e - t_k^s\right), \end{aligned} \tag{3.41}$$

where P_s and P_e are the probability of appearance and disappearance of a target in the visible area, respectively. τ_s and τ_e are coefficients of penalties with respect to the distance between the target and boundaries of the visible area. τ_l is a coefficient of a penalty that prevents the termination of long tracks. $\mathcal{B}\left(x_k^t\right)$ is the distance between x_k^t and boundaries of the visible area. $m_{\mathcal{B}}$ is the margin of boundary which is fixed to one meter in our experiments. A track which starts near the beginning frame is exempt from the initiation penalty, thus it has $\log\left(P_s\right)$ as its initiation score. With those initiation/termination scores, we can avoid the excessive number of fragmentations in the trajectories of targets.

Visual Similarity Score $S_{\mathcal{V}}$

We assume that a target does not change its appearance abruptly, so adjacent detections of the same track in each view should have similar appearances. To ensure this, we check the visual similarity between successive tracklets which are associated by a temporal association. We assign the scores at every temporal association in ω_k of an arbitrary track $\mathcal{T}_k$. Letting $\omega_k^c = \{\mathcal{Y}_{k_1}^c, \mathcal{Y}_{k_2}^c, ..., \mathcal{Y}_{k_{n_{kc}}}^c\}$ be the ordered set of camera c's tracklets in ω_k according to their starting time,

Ψ_k^c, the set of ordered pairs of detections is defined by

$$\Psi_k^c = \{(d_i, d_j)|\ d_i = \arg\max_{d_q \in \mathcal{Y}_{k_l}^c} t_q,\ d_j = \arg\min_{d_p \in \mathcal{Y}_{k_{l+1}}^c} t_p, \\ s.t.\ \mathcal{Y}_{k_l}^c, \mathcal{Y}_{k_{l+1}}^c \in \omega_k^c \text{ for } l = 1, ..., n_{kc} - 1\}. \tag{3.42}$$

With Ψ_k^c and the function $\mathcal{V}(\cdot)$ extracting the visual feature from a detection, the visual similarity score is defined by

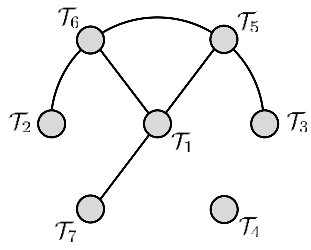
$$S_{\mathcal{V}}(\omega_k) = -\sum_{c=1}^{C} \sum_{(d_i,d_j)\in\Psi_k^c} \alpha_v \times e^{\tau_v |t_j - t_i - 1|} \times \|\mathcal{V}(d_i) - \mathcal{V}(d_j)\|_2, \tag{3.43}$$

where α_v and τ_v are parameters of modeling the declining confidence of visual similarity as the time gap between the detections increases. Since the goal of the score is to give a penalty to temporal associations between tracklets having inconsistent visual features, the visual similarity score is always less than or equal to zero.

3.4 Global Hypothesis

A global hypothesis is the set of tracks generated by the partition of tracklets described in Section 3.2. However, as mentioned in Section 3.1, a global hypothesis also can be defined as the set of compatible tracks. Thus, from a specific set of tracks, several global hypotheses can be generated by following compatibilities between the tracks. The goal of our tracking method is to find the best global hypothesis $\mathcal{H}_*$ among those global hypotheses, according to the track score defined in the previous section.

In this section, we present the online scheme which finds $\mathcal{H}_*$ rapidly. At every frame, we formulate MWCP as the optimization problem finding $\mathcal{H}_*^t$ among all



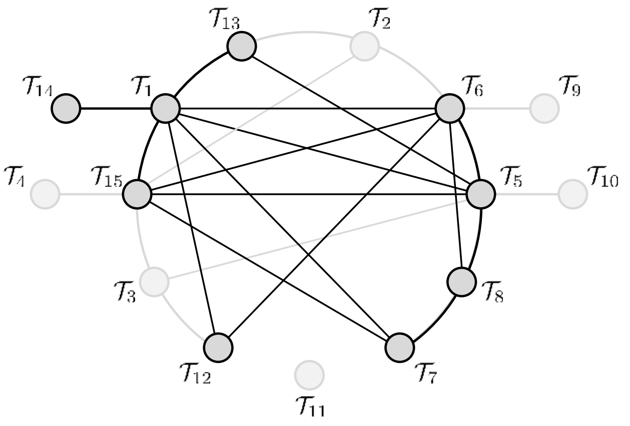
$$\mathcal{H}_1^{t_2} = \{\mathcal{T}_1\}, \quad \mathcal{H}_2^{t_2} = \{\mathcal{T}_2\}, \quad \mathcal{H}_3^{t_2} = \{\mathcal{T}_3\},$$
$$\mathcal{H}_4^{t_2} = \{\mathcal{T}_4\}, \quad \mathcal{H}_5^{t_2} = \{\mathcal{T}_5\}, \quad \mathcal{H}_6^{t_2} = \{\mathcal{T}_6\},$$
$$\mathcal{H}_7^{t_2} = \{\mathcal{T}_7\},$$
$$\mathcal{H}_8^{t_2} = \{\mathcal{T}_1, \mathcal{T}_5\}, \ \mathcal{H}_9^{t_2} = \{\mathcal{T}_1, \mathcal{T}_6\}, \ \mathcal{H}_{10}^{t_2} = \{\mathcal{T}_1, \mathcal{T}_7\},$$
$$\mathcal{H}_{11}^{t_2} = \{\mathcal{T}_1, \mathcal{T}_8\}, \ \mathcal{H}_{12}^{t_2} = \{\mathcal{T}_2, \mathcal{T}_6\}, \ \mathcal{H}_{13}^{t_2} = \{\mathcal{T}_3, \mathcal{T}_5\},$$
$$\mathcal{H}_{14}^{t_2} = \{\mathcal{T}_5, \mathcal{T}_6\},$$
$$\mathcal{H}_{15}^{t_2} = \{\mathcal{T}_1, \mathcal{T}_5, \mathcal{T}_6\}$$

Figure 3.9: The corresponding graph of tracks in the Figure 3.6 at the frame of t_2 and all possible global hypotheses from the graph. Each vertex corresponds to a track and edges represent compatibilities between tracks. If there is no edge, two tracks are incompatible to each other. A global hypothesis is a set of tracks that are compatible to each other, and a clique is a set of fully connected vertices. Thus, global hypotheses can be generated by finding the cliques in the corresponding graph.

possible global hypotheses from $\mathbf{T}^t$, the entire track set of the t_{th} frame. To reduce computation and enhance performance, we utilize $\mathcal{H}_*^{t-1}$, the optimal solution from the previous frame. However, it is easy to be trapped in a local optimum when we propagate only the best solution up to the current frame. To resolve this problem, we find not only the best solution but also the $K_\mathcal{H}$ best solutions $\mathbf{H}^t = \{\mathcal{H}_1^t, ..., \mathcal{H}_{K_\mathcal{H}}^t\}$ at each frame and use them to construct multiple MWCPs in the next frame. We also describe how we modify BLS, a state-of-the-art heuristic of solving MWCP, and apply it to our online scheme for the rapid generation of multiple solutions.

3.4.1 MWCP for MCMTT

At t_{th} frame, we construct $K_\mathcal{H}$ MWCPs that find global hypotheses which consist of high scored and compatible tracks. Each MWCP is constructed with $\mathbf{T}_n^t, n = 1, ..., K_\mathcal{H}$, a set of tracks which are candidates of the current best global hypothesis with an assumption that the previous best global hypothesis



$$\mathbf{T}_{uc}^{t_3} = \{\mathcal{T}_5, \mathcal{T}_6, \mathcal{T}_7, \mathcal{T}_{12}, \mathcal{T}_{13}, \mathcal{T}_{14}, \mathcal{T}_{15}\}$$

$$\mathcal{H}_1^{t_2} \rightarrow \mathbf{T}_1^{t_3} = \{\mathcal{T}_1, \mathcal{T}_5, \mathcal{T}_6, \mathcal{T}_7, \mathcal{T}_8, \mathcal{T}_{12}, \mathcal{T}_{13}, \mathcal{T}_{14}, \mathcal{T}_{15}\}$$

Figure 3.10: Example of a related track set and an unconfirmed track set. Let's define $\mathbf{T}^{t_3}$ as a set of all all tracks in the Figure 3.6 at t_3. Then, the depicted graph represents the corresponding graph of $\mathbf{T}^{t_3}$. By the equation (3.44), the unconfirmed track set $\mathbf{T}_{uc}^{t_3}$ is defined as written in the below the graph. Then, the related track set of $\mathcal{H}_1^{t_2}$, $\mathbf{T}_1^{t_3}$ can be found by the equation (3.45). We depict a subgraph defined by $\mathbf{T}_1^{t_3}$ on the corresponding graph with black boundaries.

was $\mathcal{H}_n^{t-1} \in \mathbf{H}^{t-1}$. We call $\mathbf{T}_n^t$ a *related track set* of $\mathcal{H}_n^{t-1}$. An example of the related track set is depicted in the Figure 3.10. It contains three types of tracks: (i) tracks in $\mathcal{H}_n^{t-1}$, (ii) tracks newly generated at the current frame among the children of (ii), i.e., $\mathbf{T}_{ch}^t(\mathcal{H}_n^{t-1})$, and (iii) unconfirmed tracks defined by

$$\mathbf{T}_{uc}^t = \{\mathcal{T}_i | \mathcal{T}_i \in \mathbf{T}^\mathbf{t} \text{ and } |t - t_i^s| < N_{conf}\}. \tag{3.44}$$

An unconfirmed track is a track shorter than N_{conf} frames. An unconfirmed track is too short to determine whether it is a false positive or not. Thus, we constantly insert unconfirmed tracks into related track sets, even if those unconfirmed tracks are not in the previous solution. Then, $\mathbf{T}_n^t$ is given by

$$\mathbf{T}_n^t = \mathcal{H}_n^{t-1} \cup \mathbf{T}_{ch}^t(\mathcal{H}_n^{t-1}) \cup \mathbf{T}_{uc}^t. \tag{3.45}$$

MWCP for each $\mathbf{T}_n^t$ for $n = 1, ..., K_{\mathcal{H}}$ is formulated as

$$\begin{aligned} \boldsymbol{\tau}^* = \arg\max_{\boldsymbol{\tau}} \quad & \sum_{\mathcal{T}_i \in \mathbf{T}_n^t} S_{\mathcal{T}_i} \cdot \tau_i \\ s.t. \quad & \tau_i + \tau_j \leq 1, \quad \forall \{i,j\} \notin \mathbb{C}_n^t, \\ & \tau_i = 0, \quad \forall \mathcal{T}_i \notin \mathbf{T}_n^t, \\ & \tau_i \in \{0, 1\}, \end{aligned} \tag{3.46}$$

where $\mathbb{C}_n^t = \{\{i,j\} | \mathcal{T}_i, \mathcal{T}_j \in \mathbf{T}_n^t, \{i,j\} \in \mathbb{C},\}$ is the compatibility set of $\mathbf{T}_n^t$. The solution of the problem can be represented by the selection variable τ_i as $\mathcal{H}_{*,n}^t = \{\mathcal{T}_i | \tau_i^* = 1\}$. We solve each MWCP with the extended BLS described in Section 3.4.2, which quickly generates multiple locally optimal solutions from a single MWCP. Letting $\mathbf{H}_n^t$ be the locally optimal solutions found during solving MWCP for $\mathbf{T}_n^t$. Then, the entire feasible solutions found from all MWCPs in the current frame can be obtained by $\hat{\mathbf{H}}^t = \bigcup_{n=1,...,K_{\mathcal{H}}} \mathbf{H}_n^t$. We pick the $K_{\mathcal{H}}$ best solutions $\mathbf{H}^t$ from $\hat{\mathbf{H}}^t$ according to their total track scores. Then, the tracking solution of the current frame is the best one in $\mathbf{H}^t$. That is,

$$\mathcal{H}_*^t = \arg\max_{\mathcal{H}_n^t \in \mathbf{H}^t} \sum_{\mathcal{T}_k \in \mathcal{H}_n^t} S_{\mathcal{T}_k}. \tag{3.47}$$

When $\mathbf{H}^{t-1} = \phi$ as in the starting frame of an input video sequence, we construct and solve a single MWCP with an entire track set $\mathbf{T}^t$.

The computation of solving MWCP is exponentially proportional to the number of tracks when we solve it exactly. In ordinary cases, $|\mathbf{T}_n^t|$ is much smaller than $|\mathbf{T}^t|$. When the number of target is η, on average η detections are obtained by each camera. Thus, the number of possible spatial associations between detections from C cameras is η^C at the maximum, and then the number of possible candidate tracks that can be generated within N frames is η^{CN} at the maximum. The fastest exact sovling algorithm for MWCP with arbitrary undirected graph up to now is proposed by Robinson and John M. [85] which has $O(2^{n/4}) = O(1.8888^n)$ as its computation time. Therefore, the time for exactly solving MWCP that is constructed with an entire candidate track $|\mathbf{T}^t|$ is $O(1.8888^{\eta^{CN}})$. In case of our subproblems, the maximum size of $|\mathbf{T}^t_n|$ is determined by the size of $\mathcal{H}^{t-1}_n \cup \mathbf{T}^t_{ch}(\mathcal{H}^{t-1}_n)$ and $\mathbf{T}^t_{uc}$. If the solution $\mathcal{H}^{t-1}_n$ is proper solution, the size of it will be η. Then, the size of $\mathbf{T}^t_{ch}(\mathcal{H}^{t-1}_n)$ can be η^{C+1} at the maximum. Thus, $|\mathcal{H}^{t-1}_n \cup \mathbf{T}^t_{ch}(\mathcal{H}^{t-1}_n)| \leq \eta + \eta^{C+1} \approx \eta^{C+1}$. In our pruning procedure that will be represented at Section 3.5, the number of tracks in each unconfirmed track tree is limited below N_{uc}. Since the tracks are unconfirmed when their durations are less than N_{conf}, the number of possible unconfirmed tracks is

$$|\mathbf{T}^t_{uc}| \leq \{(N_{conf} - 1) \times N_{uc} + 1\} \times \eta^C. \tag{3.48}$$

Thus, the maximum size of $\mathbf{T}^t_{uc}$ is

$$||\mathbf{T}^t_n|| \leq \{(N_{conf} - 1) \times N_{uc} + 1\} \times \eta^C + \eta^{C+1} \approx \eta^{C+1}. \tag{3.49}$$

Hence, the computation time of our subproblem is $O(K_{\mathcal{H}} \times 1.8888^{\eta^{C+1}}) = O(1.8888^{\eta^{C+1}})$. Compared to $O(1.8888^{\eta^{CN}})$, it is much smaller compuation time. Thus, our online scheme has less computational order than solving the original problem.

3.4.2 BLS for MCMTT

BLS [53] is a state-of-the-art heuristic algorithm, finding the maximum weighted clique in an undirected graph. BLS is based on an iteration which consists of a local search and a random perturbation. When it gets a locally optimal solution from the local search, it randomly perturbs the current solution to find another locally optimal solution. If the local search consecutively ends up with a same solution, BLS gradually increases its perturbation strength to escape from the local basin. This is called an adaptive perturbation, and is the key concept of BLS. In our online scheme, BLS is applied to solve each MWCP, with following three variants:

Multiple Solutions

Originally, BLS only keeps the best solution found at the moment. In contrast, we keep all the locally optimal solutions found until the algorithm meets the termination condition. Then, we pack those solutions into $\mathbf{H}_n^t$ and return it as the solving result of n_{th} MWCP.

Termination Condition

BLS uses only the maximum number of iterations as its termination condition because there is no way to guarantee the global optimality of a found solution. Thus, in [53], the maximum number of iterations was set to a huge constant to have more chance to find a better solution. However, it is intractable to the practical algorithms requiring real time requirements. We assume that the proper iteration number is in proportion to the complexity of the graph. And we assume that $|\mathbb{C}_n^t|$, a size of compatibility set, reflects the complexity of the graph. So, we propose the maximum iteration number i_{bls} as

$$i_{bls} = \alpha_{bls} \times |\mathbb{C}_n^t|, \tag{3.50}$$

where α_{bls} is a predetermined parameter and we set it to ten during all of our experiments. However, the equation (3.50) has to be bounded for practical applications. Therefore, we saturate the maximum number of iterations with a predetermined parameter i_{bls}^{max}.

Initial Solution

In [53], BLS generates an initial solution $\mathcal{H}_{n0}^t$ by random selection of compatible tracks. It is natural when there is no prior information. But in an online MCMTT, the solution from the previous frame can be a strong prior information to the current MWCP because targets move smoothly between consecutive frames. Therefore, we set $\mathcal{H}_{n0}^t$ to $\mathcal{H}_n^{t-1}$ and perform a local search to refine an initial solution. However, the compatibilities between tracks can be changed as the tracking goes on, so we have to repair $\mathcal{H}_{n0}^t$ before the local search when it is infeasible. The repairing of $\mathcal{H}_{n0}^t$ is done in the following way: First, we set $\mathbf{T}_{cand}$, the candidate tracks for an initial solution, to $\mathcal{H}_n^{t-1}$. Then, insert a track with the highest track score in $\mathbf{T}_{cand}$ into $\mathcal{H}_{n0}^t$ and update $\mathbf{T}_{cand}$ to $\{\mathcal{T}_i | \mathcal{T}_i \in \mathbf{T}_{cand} \setminus \mathcal{H}_{n0}^t \ s.t. \ \forall \mathcal{T}_j \in \mathcal{H}_{n0}^t, \{i,j\} \in \mathbb{C}_n^t\}$. Repeat those insertion and update until $\mathbf{T}_{cand}$ becomes an empty set.

3.5 Pruning

In this section, we introduce our track pruning scheme to moderate the computational cost of our tracking algorithm. First, we compute each track's *approximated global track probability* (AGTP) which is proposed in the following subsection. After that, we apply two different pruning techniques depending on the confirmation of each track tree. For each unconfirmed track tree, the K-best method with AGTP is applied. Meanwhile, we adopt a classical pruning technique called N scan back approach [86] to pruning in each confirmed track tree.

3.5.1 Approximated Global Track Probability

To score a track in a global view, [56] proposed a global track probability (GTP), defined with all global hypotheses that includes the track. When we define $\mathbb{H}^t$ as the set of all possible global hypotheses from $\mathbf{T}^t$, the GTP of track $\mathcal{T}_i \in \mathbf{T}^t$ is defined by

$$P_{\mathcal{T}}^t(\mathcal{T}_i) = \sum_{\forall \mathcal{H}_j \in \mathbb{H}^t, \mathcal{H}_j \ni \mathcal{T}_i} P_{\mathcal{H}}^t(\mathcal{H}_j), \tag{3.51}$$

where $P_{\mathcal{H}}^t(\mathcal{H}_j)$ represents the probability of global hypothesis $\mathcal{H}_j \in \mathbb{H}^t$, which is defined by

$$P_{\mathcal{H}}^t(\mathcal{H}_j) = \frac{\sum_{\mathcal{T}_k \in \mathcal{H}_j} S_{\mathcal{T}_k}}{\sum_{\mathcal{H}_l \in \mathbb{H}^t} \sum_{\mathcal{T}_k \in \mathcal{H}_l} S_{\mathcal{T}_k}}. \tag{3.52}$$

Since finding $\mathbb{H}^t$ is intractable in most cases, it is also intractable to calculate an exact GTP. Therefore, we approximate GTP with $\mathbf{H}^t$, the $K_{\mathcal{H}}$ best global hypotheses mentioned in Section 3.4, as below

$$\hat{P}_{\mathcal{T}}^t(\mathcal{T}_i) = \sum_{\forall \mathcal{H}_j \in \mathbf{H}^t, \mathcal{H}_j \ni \mathcal{T}_i} \hat{P}_{\mathcal{H}}^t(\mathcal{H}_j), \tag{3.53}$$

$$\hat{P}_{\mathcal{H}}^t(\mathcal{H}_j) = \frac{\sum_{\mathcal{T}_k \in \mathcal{H}_j} S_{\mathcal{T}_k}}{\sum_{\mathcal{H}_l \in \mathbf{H}^t} \sum_{\mathcal{T}_k \in \mathcal{H}_l} S_{\mathcal{T}_k}}. \tag{3.54}$$

We prune tracks having zero AGTPs as the first step of our track pruning. Since AGTP is defined by the $K_{\mathcal{H}}$ best global hypotheses, a zero AGTP means that the track does not belong to any of the $K_{\mathcal{H}}$ best global hypotheses.

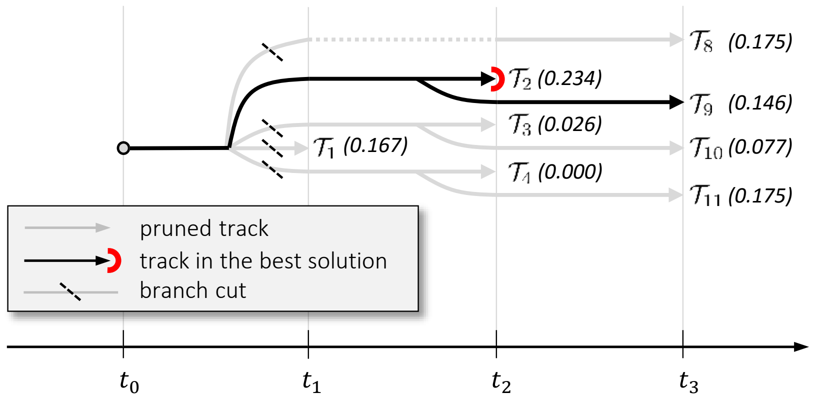


Figure 3.11: Example of N scan back approach with confirmed track tree in the Figure 3.6(b) at t_3. In this example, we set $N = 3$. Parenthesized numbers are AGTPs. All branches are discarded except the branch containing a track in the best global hypothesis.

3.5.2 Track Pruning Scheme

Unconfirmed tracks in a same track tree are similar because their measurements are almost the same. Therefore, it is inefficient to keep all tracks in an unconfirmed track tree. Thus, we discard all tracks from an unconfirmed track tree, except the K_{uc} best tracks in each tree according to their AGTPs. But to maintain the best global hypothesis, we also keep unconfirmed tracks in the best global hypothesis, no matter that they are not included in the K_{uc} best tracks of their trees.

For the case of confirmed track trees, it is intractable to keep all tracks in each tree because the number of tracks in each tree increases exponentially. Thus, we apply N scan back approach [86] to confirmed track trees. It is the pruning technique based on a deferred decision. After finding the best global hypothesis, it scans N frames back and fixes locations of targets based on the current best global hypothesis. It then prunes all tracks incompatible with the fixed locations of targets. In view of a tree structure, it prunes branches of each track tree at N frames before, except the branch containing a track in the current best global hypothesis. The Figure 3.11 depicts an example of N scan back approach, with one of the track trees in the Figure 3.6.

We summarize our online scheme by the Algorithm 6.

Algorithm 6 Online scheme of MCMTT

```
Require: detections from the dataset $\mathbf{D}$
Ensure: tracking result $\mathbb{H}_* = \{\mathbf{H}_*^1, ..., \mathbf{H}_*^T\}$    ▷ $T$ is the number of frames
 1: $\mathbf{H}^0 \leftarrow \phi$
 2: $\mathbf{Y}_c^0 \leftarrow \phi$ for $c = 1, ..., C$    ▷ $C$ is the number of cameras
 3: for $t$ in $\{1, ..., T\}$ do
 4:     for $c$ in $\{1, ..., C\}$ do
 5:         $\mathbf{D}_c^t \leftarrow \{d_i | c_i = c, t_i = t\}$
 6:         $\mathbf{Y}_c^t \leftarrow \textit{generate\_tracklet}(\mathbf{Y}_c^{t-1}, \mathbf{D}_c^t)$    ▷ see Section 3.2
 7:     end for
 8:     $\mathbf{Y}^t \leftarrow \bigcup_{c=1}^{C} \mathbf{Y}_c^t$
 9:     $\mathbf{\Omega}^t \leftarrow \textit{tracklet\_association}(\mathbf{\Omega}^{t-1}, \mathbf{Y}^t)$    ▷ see Section 3.3.1
10:     $\mathbf{T}^t \leftarrow \textit{track\_generation}(\mathbf{T}^{t-1}, \mathbf{\Omega}^t)$    ▷ see Section 3.3.3
11:     if $\mathbf{H}^{t-1} = \phi$ then
12:         $\hat{\mathbf{H}}^t \leftarrow \textit{extended\_BLS}(\mathbf{T}^t)$    ▷ see Section 3.4.2
13:     else
14:         for $n$ in $\{1, ..., |\mathbf{H}^{t-1}|\}$ do
15:             $\mathbf{H}_n^t \leftarrow \textit{extended\_BLS}(\mathbf{T}_n^t)$    ▷ see Section 3.4.2
16:         end for
17:         $\hat{\mathbf{H}}^t \leftarrow \bigcup_{n=1}^{|\mathbf{H}^{t-1}|} \mathbf{H}_n^t$
18:     end if
19:     $\mathbf{H}^t \leftarrow$ the $K_{\mathcal{H}}$ best global hypotheses in $\hat{\mathbf{H}}^t$
20:     $\mathbf{H}_*^t \leftarrow$ the best global hypothesis in $\mathbf{H}^t$
21:     $\mathbf{P}_{\mathcal{T}}^t \leftarrow \textit{calculate\_AGTP}(\mathbf{T}^t, \mathbf{H}^t)$    ▷ see Section 3.5.1
22:     $\mathbf{T}^t \leftarrow \textit{track\_pruning}(\mathbf{T}^t, \mathbf{P}_{\mathcal{T}}^t)$    ▷ see Section 3.5.2
23:     $\mathbf{\Omega}^t \leftarrow \textit{update\_association\_set}(\mathbf{T}^t)$    ▷ collect association sets of
    remaining tracks
24: end for
25: $\mathbb{H}_* \leftarrow \bigcup_{t=1}^{T} \{\mathbf{H}_*^t\}$
26: return $\mathbb{H}_*$
```

Chapter 4

Experiments

The proposed method was compared with the state-of-the-art MCMTT algorithms on the PETS 2009 dataset [87], the most widely used public benchmark dataset having multiple views from overlapping cameras. Also, the proposed method's design parameters were examined to see how they affect the trade-off between computation time and accuracy performance with a newly constructed PILSNU dataset[1]. Finally, the influence of the proposed track score's each term and the influence of the feedback information in the online scheme were analyzed with the PILSNU dataset.

Dataset For the performance comparison, we used three sets in the second scenario of the PETS 2009 dataset. The PETS 2009 dataset is the most widely used benchmark dataset by the multi-camera based applications such as tracking, people counting, and behavior understanding. In particular, the second scenario (S2) has videos for MCMTT problem. It has three video sequences:

[1]available on `https://sites.google.com/site/neohanju/mcmtt`

S2.L1, S2.L2, and S2.L3. These sets comprise four to seven different views from overlapping outdoor cameras capturing 10 to 74 pedestrians at 7 fps. S2.L1 is the easiest one with capturing 10 pedestrians in low density. S2.L2 and S2.L3 has more number of pedestrians densely distributed. Althought S2.L2 has more number of pedestrians than S2.L3, S2.L3 is the most hardest one becuase the pedestrians in it move in the same direction with very high density. With test sequences, the PETS 2009 benchmark dataset also provides intrinsic and extrinsic parameters of each camera with Tsai's camera model [82]. However, the dataset does not contain the ground truth trajectories. In our evaluation with PETS 2009, we used the ground truth provided in [88], which gives locations of targets in the region of interest in each frame.

To examine how the parameters and the terms of cost function affect the overall tracking performance, we conducted additional experiments with another dataset, the PILSNU dataset. The PILSNU dataset contains 332 frames from each of four overlapping cameras capturing ten pedestrians whose are densely distributed in a small indoor environment. Its frame rate is 6 fps. The dataset also provides Tsai's camera model for each camera, and a ground truth which is generated by hand labeled tracking result. It also provides pedestrian detections obtained by the detector proposed by Nam at el. (LDCF detector) [89] for each frame of each camera. Since it is believed that Hofmann's algorithm [34] has the best performance among MCMTT batch algorithms, we also present the performance of Hofmann's on PILSNU to give a reference for the readers.

Evaluation metrics As the metrics of a quantitative evaluation, we used the multiple object tracking accuracy (MOTA) and the multiple object tracking precision (MOTP) in the classification of events, activities, and relationships (CLEAR-MOT) metrics [90], the most popular metrics for MCMTT. We also used identity switches (IDS), fragments (FM), mostly tracked (MT), and mostly lost (ML), proposed by [91]. For these metrics, a ground truth location is matched to the closest one among the estimated locations placed within one meter. When the ground truth location is not matched to any estimated locations, it is counted as a false negative. On the other hand, an estimated location that is not matched with any ground truth locations is regarded as a false positive. A MOTA is defined by

$$\text{MOTA} = \left(1 - \frac{\sum_{t=1}^{T} (M_t + FP_t + \log_{10}(IDS_t))}{\sum_{t=1}^{T} G_t}\right) \times 100, \tag{4.1}$$

where FP_t indicates the number of false positives at time t and M_t indicates the number of missings at time t. IDS_t is the number of identity switches at time t defined in [91]. The MOTP measures the accuracy of estimated trajectories from the view point of a distance between a ground truth and an estimation. The original MOTP proposed in [90] measures the alignment accuracy of the estimated tracking box with respect to the ground truth bounding boxes. Since we track the targets in 3D space, we use a MOTP defined in Milan et al. [21] instead of the original one. In [21], the MOTP is defined by an average distance between the ground truth locations and the estimated locations, which are matched to each other. That is,

$$\text{MOTP} = \left(1 - \frac{\sum_{t=1}^{T} \sum_{i=1}^{N_t} d_t^i}{\sum_{t=1}^{T} N_t}\right) \times 100, \tag{4.2}$$

where N_t refers to the number of matched ground truth locations at time t and d_t^i indicates the distance between i_{th} pair of matched ground truth and estimated locations at time t. Here, the unit of distance is meters. When the average distance is zero, that means we perfectly track the targets, then MOTP becomes 100. IDS is total identity switches over time, i.e., $IDS = \sum_{t=1}^{T} IDS_t$.

FM counts how many trajectory fragements are matched to the ground truth trajectories. FM increases whenever any severance of trajectory occurs because of a identity switch or a false negative. MT is the percentage of ground truth trajectories which are covered by result tracks for more than 80% in length. ML is the percentage of ground truth trajectories which are covered by result tracks for less than 20% in length. Except IDS and FM, all metrics are represented by percentage values.

Parameter settings In our experiments, the degree and span size of the Savitzky–Golay filter for smoothing in the eqaution (3.28) were set to one and nine, respectively. The other parameters of the proposed method were set as shown in the Table 4.1. Although some parameters were empirically set, there are intuitions in using these parameters. When pedestrian density increases, the false negative ratio of a pedestrian detector goes up. In this case, a track's initiation or termination far from the boundary must be permitted because of occlusion. The initiation or termination controlled with P_s, P_e, τ_s and τ_e as shown in the Table 4.1. When a scene is sparse, the confidence of each track is higher than in a crowded scene. In this case, a smaller value of P_e than P_s is needed to prevent the impetuous termination of a well continuing track, which would be better than starting a new track. In the Table 4.2, we listed the parameters of which the tuned values have the biggest impact on the tracking performance. Those parameters directly affect the number of false positives and false negatives that are also related with the number of ID switches. In the table, we also presents the adjustment strategies for the parameter tuning. Except them, most of our parameters do not need any tuning for the specific surveillance scene. Since tuning the parameters is very tedious work even with a small number of parameters, we present a simple automatic parameter tuning

Algorithm 7 Automatic parameter tuning

```
Require: detections from the dataset $\mathbf{D}$, algorithm $\mathcal{A}$, parameter domain $\Theta$,
    initial parameter $\theta_0 \in \Theta$, minimum number of run for each parameter $N_{min}$,
    random seed array $\Gamma = \{\gamma_1, \gamma_2, ...\}$
Ensure: best parameter for $\mathcal{A}$, $\theta_{best}$
 1: $\theta_c \leftarrow \theta_0$                                        ▷ current parameter
 2: $\theta_{best} \leftarrow \theta_c$
 3: $f_{best} \leftarrow 0$
 4: $f_{best}^{prev} \leftarrow 0$
 5: for $\theta \in \Theta$ do
 6:     $\mathbf{f}_\theta = (f_\theta^1, f_\theta^2, ..., f_\theta^{N_{min}}) \leftarrow (0.0, 0.0, ..., 0.0)$    ▷ evaluation cache for each
    parameter
 7:     $\hat{f}_\theta \leftarrow 0$
 8:     $N_{theta} \leftarrow 0$          ▷ number of excuted evaluation for each parameter
 9: end for
    /*initial parameter evaluation */
10: for $i = 1$ to $N_{min}$ do
11:     $f_{\theta_0}^i \leftarrow evaluation(\mathcal{A}, \theta_c, \gamma_1)$
12: end for
13: $\hat{f}_{\theta_0} \leftarrow performance\_estimation(\mathbf{f}_{\theta_c}, N_{min})$
14: $f_{best} \leftarrow \hat{f}_{\theta_0}$
15: while until meet the termination condition do
    /*local search */
16:     while $f_{best} > f_{best}^{prev}$ do
17:         $f_{best}^{prev} \leftarrow f_{best}$
18:         $\Theta_{neighbor} \leftarrow get\_neighbor\_parameters(\theta_c)$
19:         for $\theta_i \in \Theta_{neighbor}$ do
20:             while $j = N_{theta_i}$ to $N_{min}$ do
21:                 $\hat{f}_{\theta_i} \leftarrow performance\_estimation(\mathbf{f}_{\theta_i}, j)$
22:                 if $\hat{f}_{\theta_i} < f_{best}$ then
23:                     $\hat{f}_{\theta_i} \leftarrow 0$; break
24:                 end if
25:                 $f_{\theta_i}^j \leftarrow evaluation(\mathcal{A}, \theta_i, \gamma_j)$
26:             end while
27:             if $\hat{f}_{\theta_i} > f_{best}$ then
28:                 $f_{best} \leftarrow \hat{f}_{\theta_i}$; $\theta_{best} \leftarrow \theta_i$
29:             end if
30:         end for
31:         $\theta_c \leftarrow \theta_{best}$
32:     end while                                  ▷ continued to the next page
```

```
    /*perturbation */
33:     $\theta_c \leftarrow random\_perturbation(\theta_c)$
34: end while
35: return $\theta_{best}$
```

procedure in the Algorithm 7. The algorithm is motivated by Hutter et al. [92] which optimizes a number of parameters with iterated local search (ILS) procedure [93]. The main differences between our tuning algorithm and ParamILS, which proposed by Hutter et al. [92], are the definition of local search and evaluation procedure. We define the local search as the exploration on parameters through ordered discrete variables while ParamILS considers the parameters as the categorical variables. We evaluate each parameter setting with the median value of MOTA while ParamILS evaluates the parameters with the runtime of a target algorithm. In the Algorithm 7, *evaluation* procedure evaluates MOTA of algorithm $\mathcal{A}$ with its parameter θ and the random seed γ. Here, the random seed ensures to get a statistical result of an algorithm based on the random procedures. Because BLS in our tracking algorithm randomly operates its perturbation step, our algorithm also needs various random seeds for a proper evaluation. $performance\_estimation$ procedure calculates the median value of MOTA. When the number of evaluation of the parameter setting is less than the required number of evaluation N_{min}, $performance\_estimation$ assumes that the parameter setting will get 100.0 MOTA at the remaining evaluations. Then, the result of $performance\_estimation$ is now the best possible performance of the parameter in the future. At $get\_neighbor\_parameters(\theta_c)$, the parameter θ_c is modified to generate its neighbors by changing one of its element to the values in the parameter domain Θ which are right next to the current value. To escape from the local optimum, we perturb the parameter with a random selection at *random_perturbation* procedure. In *random_perturbation*, the number of parameters to be changed and the values for those parameters are randomly selected according to the uniform distribution.

Implementation details We ran the experiments with a single threaded C++ implementation of our tracking algorithm on an i7 CPU, with 3.4 GHz, and 32 GB RAM. For the detection input of our algorithm, we used the deformable part model (DPM) [94] for PETS 2009 because many exsiting works have been used DPM as their input. However, we used LDCF for PILSNU since it is much more faster than DPM while its performance is comparable to DPM. As the visual feature in the equation (3.43), we used color histograms for RGB color space with 16 bins for each channel. We concatenated those histograms into a one vector and used that vector as a visual feature. In the tracklet generation, feature points were detected by FAST algorithm [95] combined with a grid adaptation technique to uniformly extract feature points from a target image.

4.1 Comparison with the State-of-the-art Methods

The proposed method was compared with the state-of-the-art MCMTT methods on the PEST 2009 dataset and we show the results in The Table 4.3. The performance of existing works in the table have been cited from their papers. Since the performance may depend on the detection performance, we have conducted an additional experiment on our own PILSNU dataset including the detection algorithm for fair comparison. In the experiment on the PILSNU, our method has been compared with Hofmann's algorithm [34], which has shown the best performance on the PETS 2009. For this experiment, we have implemented and tuned Hofmann's algorithm to perform comparably as it did in his paper. The Table 4.4 compares Hofmann's algorithm and ours on the PILSNU. The as-

Table 4.1: Parameter Settings

	Eq. no.	Symbol	Definition	Basis	Values (S2.L1, S2.L2, S2.L3, PILSNU)
Tracklet	(3.11)	L_c	length of the comparison interval	empirically	4 frames
	(3.14)	δ_{min}	threshold for static feature point	feature tracking error	0.1
	(3.17)	ε_{Φ}	maximum allowable 3D distance between consecutive detections	object moving speed	0.6 meters
	(3.18)	$\varepsilon_{\hbar}$	maximum allowable height variation of target during one frame	sufficient value	0.4 meters
	(3.15)	$w_{\delta}(s_i)$	neighbor window size for disparities	empirically	$0.2 \times s_i$
Track	(3.6)	θ_s	minimum distance for collision avoidance	size of half body	0.2 meters
	(3.20)	ε_{3D}	maximum distance between simultaneous detections	object moving speed & calibration error	2.0 meters
	(3.22), (3.39)	v_{max}	maximum movable distance of target during one frame	normal speed of human	0.9 m/s
	(3.24)	δ_a	maximum allowable frame gap between consecutive tracklets	sufficient number	9 frames
	(3.32)	γ_{fp}	false positive ratio of the object detector	from detection input	0.01
	(3.32)	γ_{fn}	false negative ratio of the object detector	empirically	(0.1, 0.4, 0.4, 0.2)
	(3.35)	ε_{det}	maximum error bound of detected location	from detection input	4 pixels
	(3.35)	ε_{cal}	maximum error bound of camera calibration	measured by reprojection	0.5 meters
	(3.40)	P_s	probability of target appearance	empirically	(1.0^{-3}, 1.0^{-1}, 1.0^{-1}, 1.0^{-1})
	(3.40)	τ_s	decaying coefficient of P_s w.r.t. a distance from boundary	empirically	(1.0^{-3}, 1.0^{-4}, 1.0^{-4}, 1.0^{-3})
	(3.41)	P_e	probability of target disappearance	empirically	(1.0^{-6}, 1.0^{-1}, 1.0^{-1}, 1.0^{-2})
	(3.41)	τ_e	decaying coefficient of P_e w.r.t. a distance from boundary	empirically	(1.0^{-3}, 1.0^{-4}, 1.0^{-4}, 1.0^{-3})
	(3.41)	τ_l	decaying coefficient of P_e w.r.t. a track length	empirically	(1.0^{-2}, 1.0^{-2}, 1.0^{-2}, 1.0^{-1})
Etc.	(3.44)	N_{conf}	minimum number of frames for track confirmation	empirically	3 frames
	-	i_{bls}^{max}	maximum iteration number of BLS	empirically	2,000
	-	K_{uc}	maximum number of tracks in unconfirmed track tree	empirically	4
	-	N	scan-back size	empirically	10 frames
	-	$K_{\mathcal{H}}$	number of global hypotheses	-	See Table 4.3 and 4.4

Table 4.2: Important parameters

Symbol	Range	Tuning strategy	Remarks
γ_{fn}	0.1 to 0.5	add or subtract 0.1	Higher value makes the algorithm tend to generate each reconstruction with a small number of detections
P_s	1.0^{-6} to 1.0^{-1}	multiply or divide by 10	Modification with order of 10 is used beacause the costs
P_e	1.0^{-6} to 1.0^{-1}	multiply or divide by 10	are proportional to the log-scale of these probabilities
τ_s	1.0^{-6} to 1.0^{-3}	multiply or divide by 10	Those parameters determine the distance (in mm unit)
τ_e	1.0^{-6} to 1.0^{-3}	multiply or divide by 10	from boundary which makes the half value of probabilities
τ_l	1.0^{-2} to 1.0^{-1}	multiply or divide by 10	When the targets are detected robustly, there are few missed detections, decrease this value to ensure the consistend labeling

Table 4.3: Quantitative Results for the PETS 2009 Dataset (Scenario 2)

Sequence	Method		Camera IDs	MOTA [%]	MOTP [%]	MT [%]	ML [%]	FM	IDS
PETS S2.L1	Batch	Berclaz et al. [29]	1+3+5+6+8	82.0	56.0	-	-	-	-
		Leal-Taixé et al. [32]	1+5	76.0	60.0	-	-	-	-
		Leal-Taixé et al. [32]	1+5+6	71.4	53.4	-	-	-	-
		Hofmann et al. [34]	1+5	99.4	82.9	**100.0**	**0.0**	1	1
		Hofmann et al. [34]	1+5+7	99.4	**83.0**	**100.0**	**0.0**	1	2
		Byeon et al. [44]	1+5+6+7+8	99.4	**83.0**	**100.0**	**0.0**	1	2
	Online	Ours (instant, $K_{\mathcal{H}} = 25$)	1+5+7	98.9	72.9	**100.0**	**0.0**	5	1
		Ours (deferred, $K_{\mathcal{H}} = 25$)	1+5+7	**99.5**	78.1	**100.0**	**0.0**	**0**	**0**
PETS S2.L2	Batch	Hofmann et al. [34]	1+2	**87.6**	73.5	**86.0**	**0.0**	128	**111**
		Hofmann et al. [34]	1+2+3	79.7	**74.2**	69.8	2.3	129	132
	Online	Ours (instant, $K_{\mathcal{H}} = 5$)	1+2	78.0	62.7	74.3	2.7	198	249
		Ours (deferred, $K_{\mathcal{H}} = 5$)	1+2	81.1	64.9	77.0	2.7	**99**	163
		Ours (instant, $K_{\mathcal{H}} = 10$)	1+2+3	69.5	61.0	75.7	2.7	220	357
		Ours (deferred, $K_{\mathcal{H}} = 10$)	1+2+3	72.9	63.1	73.0	2.7	132	246
PETS S2.L3	Batch	Hofmann et al. [34]	1+2	**68.5**	72.3	**54.5**	20.5	149	156
		Hofmann et al. [34]	1+2+4	65.4	**73.9**	40.9	25.0	88	116
	Online	Ours (instant, $K_{\mathcal{H}} = 20$)	1+2	63.6	59.1	43.2	20.5	87	119
		Ours (deferred, $K_{\mathcal{H}} = 20$)	1+2	64.4	59.9	40.9	18.2	**44**	**61**
		Ours (instant, $K_{\mathcal{H}} = 20$)	1+2+4	53.9	55.6	31.8	**9.1**	143	197
		Ours (deferred, $K_{\mathcal{H}} = 20$)	1+2+4	54.5	57.0	34.1	**9.1**	78	101

terisks next to method names signify that we did the implementation ourselves and details would be different from the original implementation. Since our algorithm has randomness due to BLS, we ran 30 experiments for each sequence to get statistical results and present each metric with its median value. In regard to our results, "deferred" represents ten frame deferred result, and "instant" represents the instant result without delayed decision. In the parentheses, we wrote the value of $K_{\mathcal{H}}$ used in each sequence.

As shown in the Table 4.4, our deferred result is comparable to Hofmann's algorithm. Notably, our deferred result is superior to any other state-of-the-art batch methods in all metrics on the PETS 2009 S2.L1 except MOTP, which is significantly affected by a post processing. On S2.L1, our deferred result

Table 4.4: Quantitative Results for the PILSNU Dataset

Sequence	Method		Camera IDs	MOTA [%]	MOTP [%]	MT [%]	ML [%]	FM	IDS
PILSNU	Batch	Hofmann et al.* [34]	1+2	61.3	77.6	60.0	**0.0**	8	12
		Hofmann et al.* [34]	1+2+3+4	**88.2**	**80.0**	80.0	**0.0**	**1**	**1**
	Online	Ours (instant, $K = 15$)	1+2	66.9	54.6	60.0	**0.0**	62	80
		Ours (deferred, $K_{\mathcal{H}} = 15$)	1+2	72.6	61.3	60.0	**0.0**	28	35
		Ours (instant, $K_{\mathcal{H}} = 10$)	1+2+3+4	80.0	64.4	**90.0**	**0.0**	32	44
		Ours (deferred, $K_{\mathcal{H}} = 10$)	1+2+3+4	85.7	72.5	**90.0**	**0.0**	12	18

achieves 100% MT with zero IDS, which means a qualitatively perfect tracking result. Despite that our deferred result have much more IDS than Hofmann's algorithm on S2.L2, on which a pedestrian detector has low recall, the proposed framework achieves a comparable performance to the state-of-the-art batch algorithm although the online scheme has less chances to recover the missing detections than the batch scheme.

4.2 Influence of Parameters

On the PILSNU dataset, the proposed method was ran with $K_{\mathcal{H}} = 1$, 5, 10, 15, 20, 25, 30 and $i_{bls}^{max} = 500$, 1,000, 2,000 to examine the effect of $K_{\mathcal{H}}$ and i_{bls}^{max} on accuracy and computation time. Due to the randomness of the proposed method, 30 experiments were done at each parameter setting, as mentioned in the previous section. The Figure 4.1 shows the tendencies of the processing time, MOTA, and IDS affected by $K_{\mathcal{H}}$ in the proposed algorithm. The processing time depending on $K_{\mathcal{H}}$ increased linearly. Thus, a desired processing time can be achieved by adjusting $K_{\mathcal{H}}$. MOTA increased with $K_{\mathcal{H}}$ and converged around 85%. IDS decreased until it touched 18 at $K_{\mathcal{H}} = 10$. Clearly, large values of $K_{\mathcal{H}}$ boost performance. Therefore, $K_{\mathcal{H}}$ is a conclusive control variable of the trade-off between performances in accuracy and computation times. Note that the PILSNU has 333 frames captured at 6 fps. Thus, processing the whole dataset within 55.5 seconds is the condition of real-time processing. Since our result at $K_{\mathcal{H}} = 1$ had a processing time of less than 50 seconds and shows a

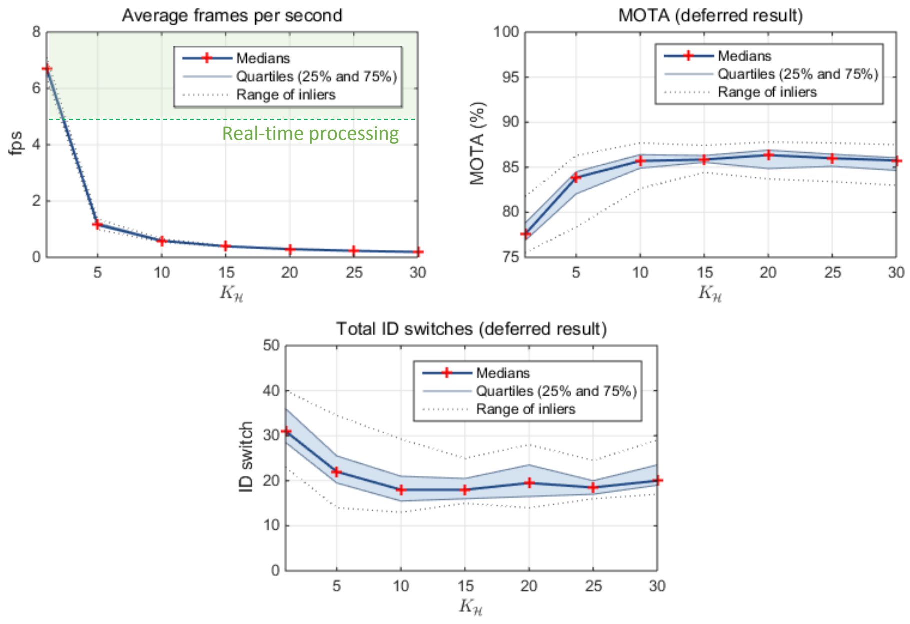


Figure 4.1: Evaluation results with various values of $K_{\mathcal{H}}$ on PILSNU dataset. Regions between lower quartile and upper quartile are colored sky blue. Dashed lines represent the range of inlier defined by the maximum Whisker length, which is about ± 2.7 std. deviations from the mean value. In the left figure, we indicate a range of real-time processing by green.

performance comparable with the state-of-the-art batch algorithms, we believe that the proposed algorithm has a real-time capability.

The Figure 4.2 shows the tendencies of the processing time, MOTA, variation at MOTA, and IDS affected by i_{bls}^{max}. i_{bls}^{max} is also crucial to the performance of our algorithm because the equation (3.50) usually hits i_{bls}^{max}. As shown in the figure, the processing time also increased linearly depending on i_{bls}^{max}. However, the performance was not increased without a sufficiently large value of i_{bls}^{max}. The last graph in the Figure 4.2 represents the gap between the maximum and minimum MOTA in each setting. Following that graph, a small i_{bls}^{max} caused an unstable performance while MOTA with a sufficiently large i_{bls}^{max} was gradually

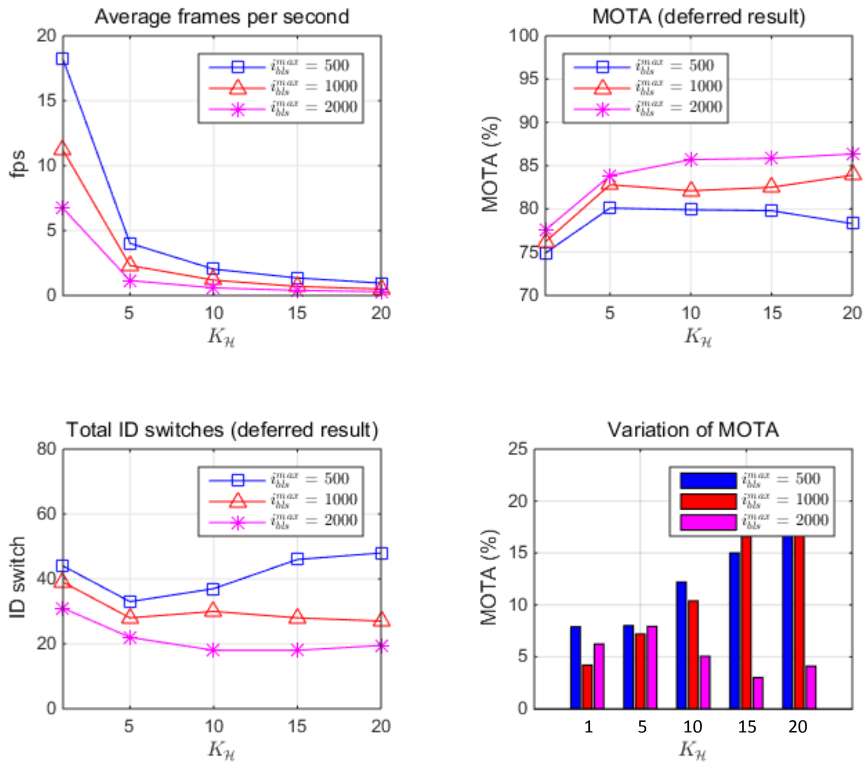


Figure 4.2: Evaluation results with various i_{bls}^{max} and $K_{\mathcal{H}}$ on the PILSNU dataset. The last graph represents a gap between the maximum and minimum MOTA in each setting.

stabilized according to $K_{\mathcal{H}}$.

We also conducted additional experiments with various values of $K_{\mathcal{H}}$ on PETS2009 S2.L1. The Figure 4.3 shows the results. As the results of PILSNU dataset, a large value of $K_{\mathcal{H}}$ has better performance and the higher computational complexity than a small value of $K_{\mathcal{H}}$.

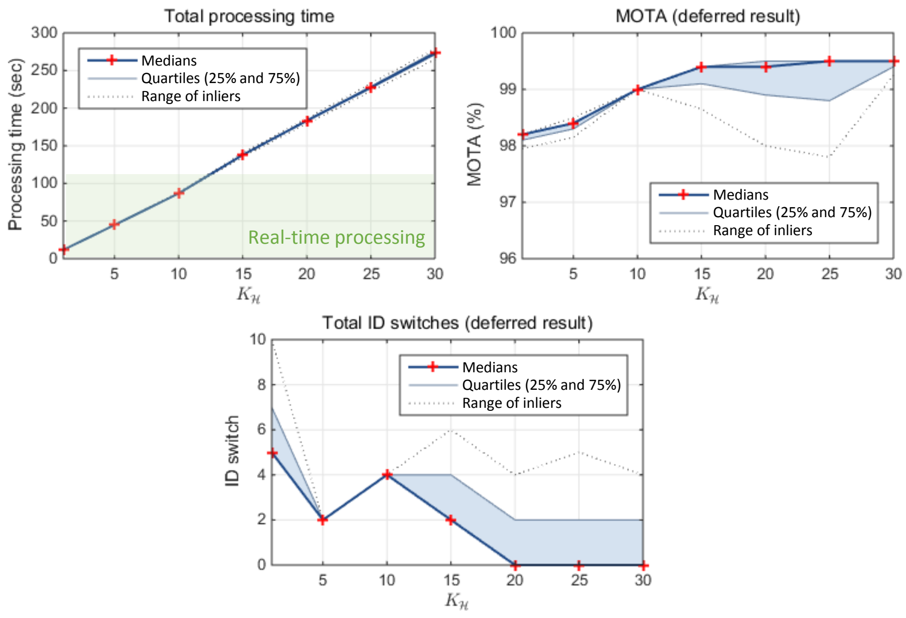


Figure 4.3: Evaluation results with various values of $K_{\mathcal{H}}$ on PETS2009 S2.L1. Regions between lower quartile and upper quartile are colored sky blue. Dashed lines represent the range of inlier defined by the maximum Whisker length, which is about ± 2.7 std. deviations from the mean value. In the left figure, we indicate a range of real-time processing by green.

4.3 Score Function Analysis

To examine the influence of each term in the proposed score function upon the performance of our algorithm, score function was evaluated with term variations: (i) without the visual similarity score in the equation (3.43), (ii) without the visibility term in the equation (3.31), (iii) without the reconstruction term in the equation (3.31), (iv) with a constant score instead of the equation (3.40) and (3.41), (v) with a constant score instead of the equation (3.39). The Table 4.5 provides the results and shows that the most crucial term was the linking score. With a constant linking score, only the visual similarity score measures the quality of a linkage between tracklets. However, the visual similarity score

Table 4.5: Quantitative Results of Score Function Variations

Description	MOTA [%]	MOTP [%]	MT [%]	ML [%]	FM	IDS
proposed	**85.7**	72.5	**90.0**	**0.0**	**12**	18
w/o visual similarity	85.4	72.2	**90.0**	**0.0**	15	19
w/o visibility term	84.9	**72.9**	**90.0**	**0.0**	15	**17**
w/o reconstruction term	81.4	71.2	**90.0**	**0.0**	19	29
constant init/term score	75.7	69.9	80.0	**0.0**	21	45
constant linking score	74.1	71.8	80.0	**0.0**	35	52

does not work with tracklets from different cameras; hence, the result is trivial. Except the linking score, the initiation and termination scores were the most crucial terms. These scores are deeply connected to the false positives and false negatives of target tracking. Thus, constant initiation and termination scores dropped down MT as constant linking scores did. The most uninfluential term was the visual similarity score; it only improved IDS and FM slightly. The influence of visibility on the performance was also negligible.

4.4 Solving Scheme Analysis

To verify the effectiveness of our proposed solving scheme, we compared our scheme to its variation and its baseline scheme. The Figure 4.4 shows the quantitative result of each solving schemes. “MHT+BLS” indicates the baseline solving scheme that used only one MWCP for solving the equation (3.46) without feedback information on the previous global hypotheses. Therefore, we plot the result of “MHT+BLS” with a single point at $K_{\mathcal{H}} = 1$. “Ours (w/o initial solution)” indicates the solving scheme that constructed multiple MWCPs with the feedback information, but used a randomly generated initial solution as the original BLS. Except the baseline scheme, the maximum iteration number i_{bls}^{max} was set to 2,000. For a fair comparison, i_{bls}^{max} in the baseline scheme was set to $5 \times 2,000 = 10,000$. With i_{bls}^{max} = 10,000, the computation time of the baseline

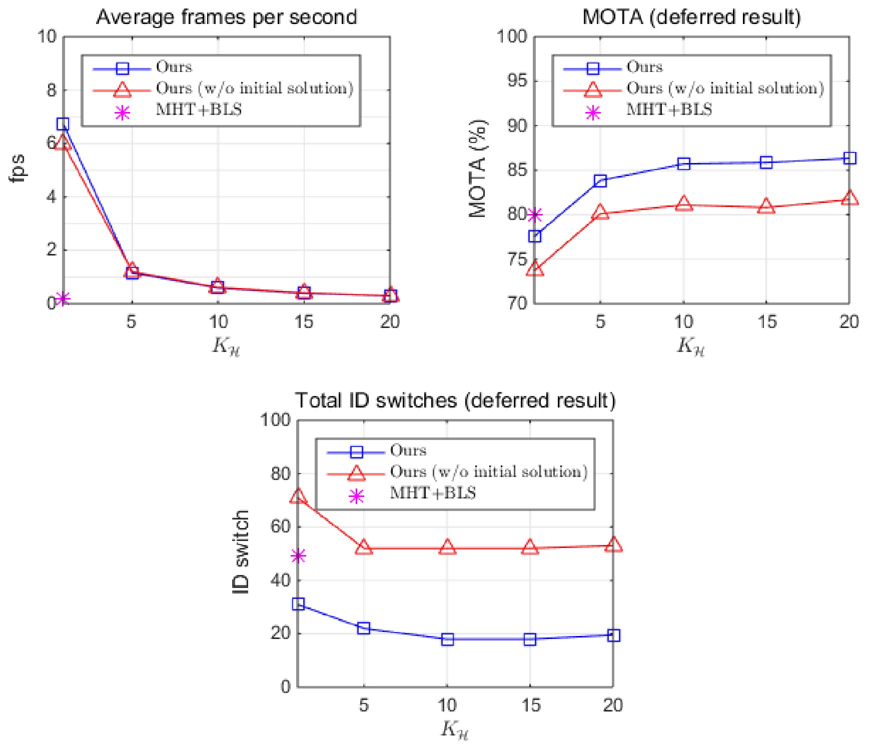


Figure 4.4: Quantitative results with various solving schemes. “MHT+BLS” indicates the baseline scheme that solves the equation (3.46), one MWCP with entire candidate tracks, without utilizing any feedback information on the previous global hypothesis. “Ours (w/o initial solution)” indicates the proposed scheme, but without the initial solution decribed in Section 3.4.2. Instead, it uses a random initial solution as the original BLS.

exceeded that of the other schemes.

As shown in the Figure 4.4, the baseline scheme was superior to the other schemes when $K_{\mathcal{H}} = 1$. This, however, is trivial because the other schemes solve smaller graphs than the baseline scheme, signifying that they cannot escape from the local optimum solution near or around the previous solution. However, despite solving smaller graph rather than the baseline scheme, performance increased when $K_{\mathcal{H}}$ had a large value. The proposed scheme performed better than the baseline scheme when $K_{\mathcal{H}} = 5$ or greater. “Ours (w/o initial solution)” also achieved a comparable performance to the baseline scheme when $K_{\mathcal{H}} = 5$ or greater. Since the computation time of the proposed scheme is much smaller than the baseline scheme, it is certain that our problem dividing strategy is beneficial to performance when the computation time is limited. Compared to “Ours (w/o initial solution)”, the performance of the proposed solving scheme improved MOTA by 5.5% and IDS by 61.6% on average. Thus, the effectiveness of the proposed initial solution is verified. Note that this result shows that the solutions for consecutive frames are closely related.

4.5 Qualitative Results

In this section, we illustrate qualitative results of our algorithm and discuss those results. The Figure 4.5, 4.6, and 4.7 show the result of our algorithm on PETS2009 S2.L1, S2.L2, and S2.L3, respectively. The Figure 4.8 shows our result on PILSNU dataset. In all figures, each column represents each camera view, and each row is the input frames with the same frame index. The tracking result of each pedestrian is depicted by a colored line that ends up at the current position of the pedestrian. To avoid a complicated visualization, we only draw the recent 30 locations of each trajectory instead of an entire trajectory. The numbers at the center of each box indicate a tracking label, which has to be constant for the same pedestrian. A number at the left top corner of each box indicates a tracklet ID.

PETS2009 : As shown in the quantitative result, our algorithm produced a nearly perfect tracking result of S2.L1. In particular, closely standing pedestrians at frame 168 and pedestrians that were missed in a specific view at frame 315 were successfully tracked by our algorithm. Moreover, the pedestrian at

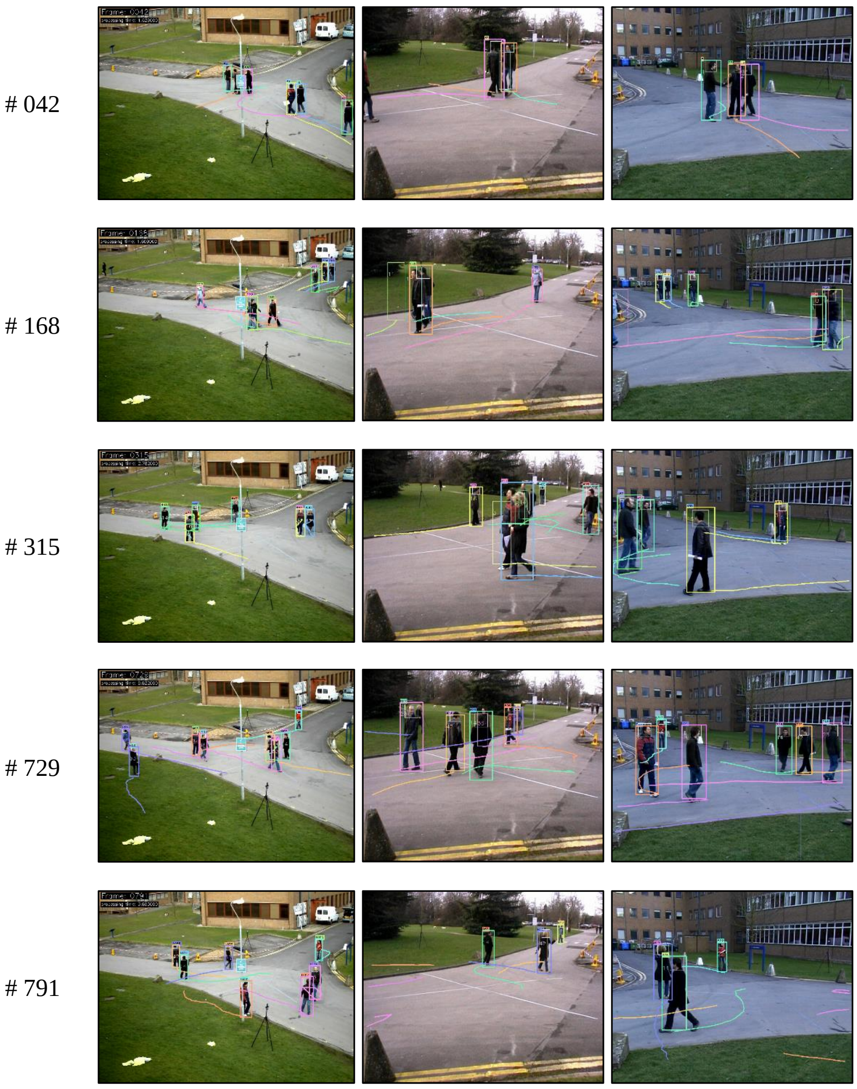


Figure 4.5: Qualitative result of the proposed algorith on PEST2009 S2.L1.

frame 729 who works backward was also robustly tracked despite his abrupt trajectory. The pedestrian labeled with 31 at frame 791 had not been detected several frames before that frame because he was occluded by the telegraph pole in the middle of the first view. Although there exists a time gap, our algorithm

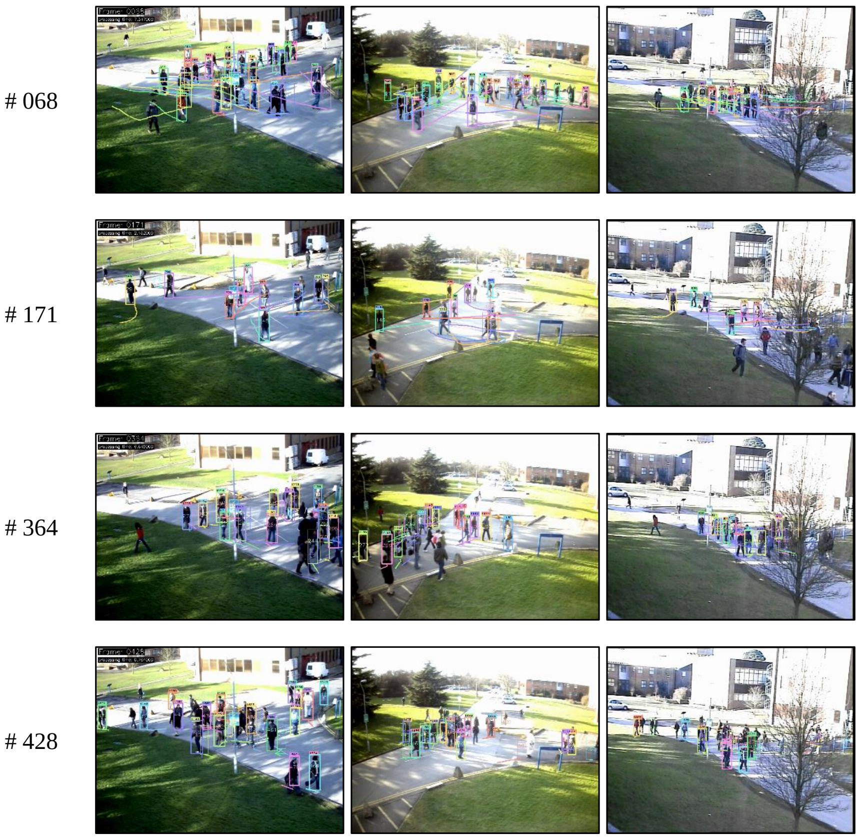


Figure 4.6: Qualitative result of the proposed algorith on PEST2009 S2.L2.

successfully associated his tracklets and reconstructed his trajectory by the interpolation and the smoothing procedure. In the case of S2.L2, our algorithm missed some pedestrians because they were not detected by a pedestrian detector. When the density of pedestrian increases, the miss rate of a pedestrian detector also increases. Although our algorithm can handle a short-term occlusion, the performance of our algorithm drops down when there are the pedestrians have not been detected for a long period. The performance degradation is most severe in the result on S2.L3. In S2.L3, many pedestrians were merely detected during the whole sequence because they stuck together and moved in

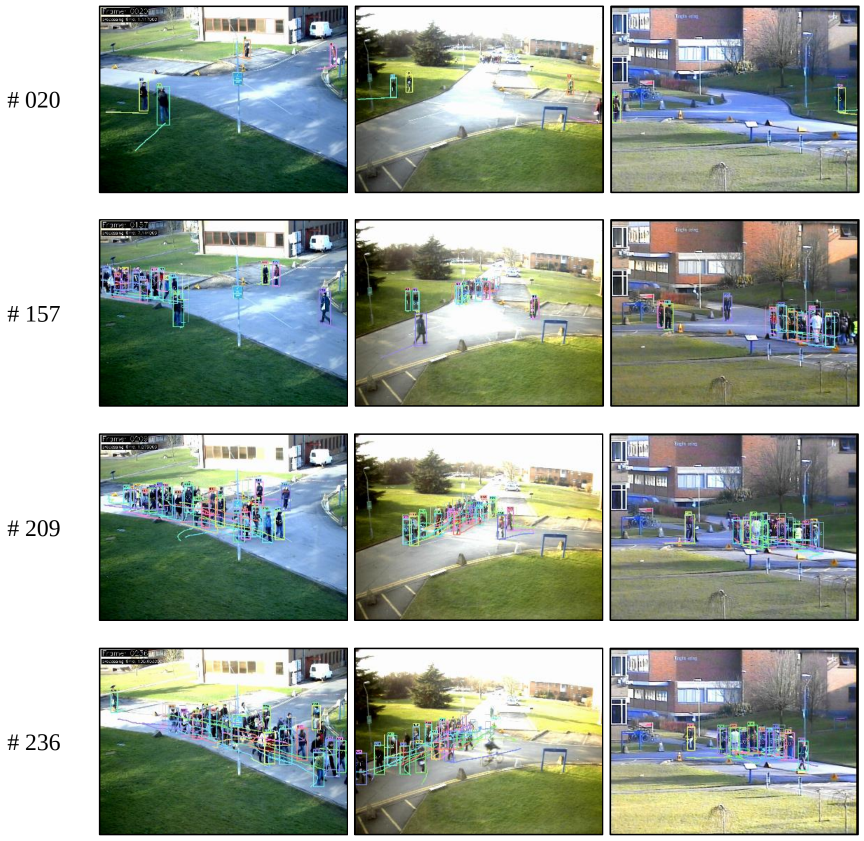


Figure 4.7: Qualitative result of the proposed algorith on PEST2009 S2.L3.

the same direction. Notably, the pedestrians in the middle of the crowd were not detected since they were occluded by other pedestrians in all views. Thus, tracks for those pedestrians were not initialized and it caused missing of many pedestrians in our tracking result. Although our tracking result heavily depends on the detection input, our algorithm successfully associated tracklets for the same target through frames and cameras.

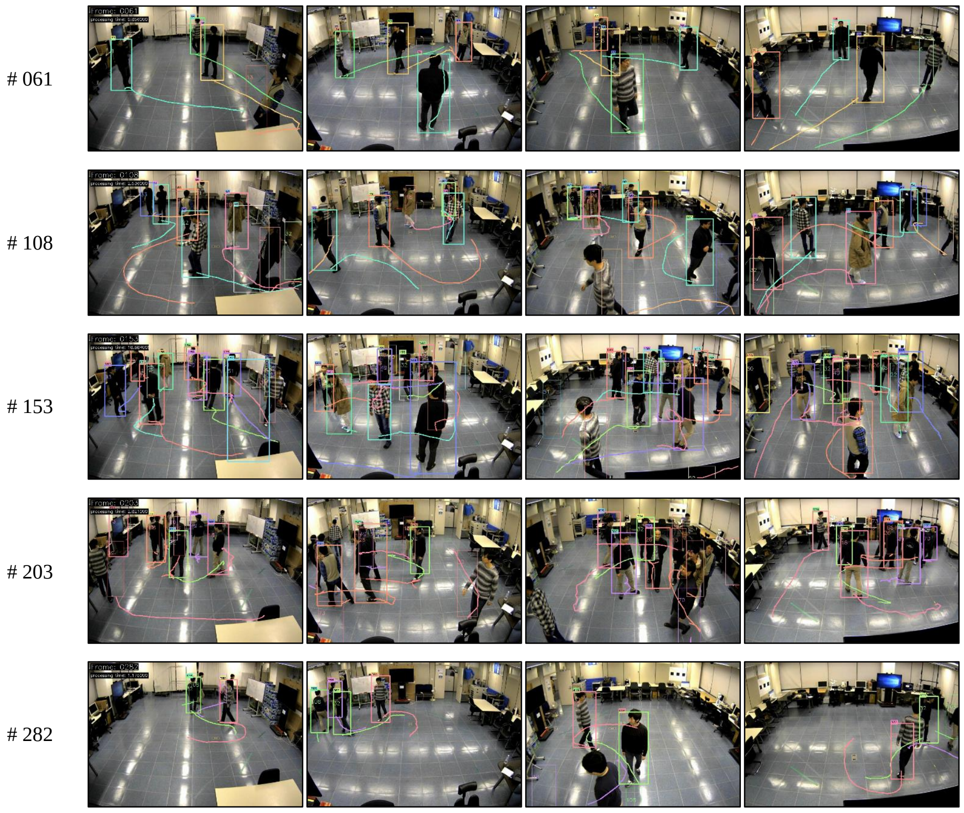


Figure 4.8: Qualitative result of the proposed algorith on PILSNU dataset.

PILSNU : The density of pedestrians in PILSNU is higher than S2.L1 and S2.L2, but lower than S2.L3. The real challenging point of this dataset is that the pedestrians stand aslant at the verge of views. When the pedestrian stands aslant, the bottom centers of detection box bounding the pedestrian badly align with the real grounding points of the pedestrian. This bad alignment problem induces the performance degradation since it increases the ambiguity of data association. However, our algorithm properly associated the tracklets and generated the satisfying results. As shown in the Table 4.4, our algorithm found more trajectories than the state-of-the-art method. Although our algorithm produced more IDS than the comparing method, ours is more proper to the practical applications since the lost of the whole trajectory costs more than the occurring of IDS.

Chapter 5

Concluding Remarks

5.1 Conclusions

In this dissertation, we presented an online multiple camera multiple target tracking (MCMTT) algorithm based on the multiple hypothesis tracking (MHT) framework. Our MHT framework was realized by MWCP for the association of input detections through frames and cameras to find 3D trajectories of multiple objects until the current frame. Through the dissertation, our goal was to achieve a robust tracking performance with a small amount of computations. The first trial of reducing the computational load was generating and using a tracklet as a unit of associations. The use of tracklets instead of detections to generate candidate tracks prevents the generation of many absurd tracks while minimizing the performance drop down. We introduced our own tracklet generation scheme that rapidly produces highly reliable tracklets.

To efficiently solve MWCP in an online manner, we proposed an online scheme which dynamically formates multiple small-sized MWCPs by utilizing the previous tracking result. The solution spaces of those MWCPs are much smaller than the original MWCP considering the entire tracks at a time. Thus, the proposed scheme significantly reduced the time to find reliable tracking solutions. Moreover, the proposed scheme also found better solutions than solving MWCP with entire tracks because it can explore more various solutions under a limited solving time. To resolve the NP-hard issue in solving each MWCP, we applied a heuristic solver named BLS to solving our MWCP with a proper adaptation. Owing to our adaptation, BLS found near-optimal solutions within a few iterations.

We also proposed the track pruning scheme utilizing the previous solutions. In our pruning scheme, an approximated global track probability (AGTP) of each track is calculated and tracks are pruned according to their AGTP. Because AGTP represents the quality of each track with respect to the overall tracking situation instead of an individual track, we could prevent the performance degradation from a hard track pruning.

As shown in the experiments, our online MCMTT algorithm showed the state-of-the-art performance on the public benchmark dataset, and also showed the capability of real-time processing. With our own dataset, we carefully examined the effect of our solving scheme, an adaptation of BLS, and each term in our score function on the tracking performance. As a result, they all contributed to the performance of our method.

5.2 Future Works

Even though we proposed the online MCMTT algorithm showing a state-of-the-art performance, there are still many future works to improve the performance of our algorithm. The most conspicuous weakness of the proposed algorithm is on the tedious parameter tuning. Our algorithm has a number of parameters and some of them severely affect the overall performance. However, all parameters were manually set by intuitions and tuned with the dataset. Recently, the opening researches to learn of data association in the tracking problem has been proposed [96, 97]. They adopt deep learning techniques such as a convolutional neural network and a recurrent neural network. The important issue of the deep learning technique is obtaining a sufficient number of training dataset. For this issue, an approach that makes the synthetic dataset with 3D CAD program [98] would be a good participant. Another future work issue is related with a visual cue for temporal association for which we only utilized a color histogram. Nowadays, many robust and efficient appearance modeling techniques have been proposed ant they can be better alternatives of our color histogram-based visual feature. Owing to deep learning, there is much of possibility about utilizing appearance information into inter-camera association (i.e., spatial association). Our most urgent future work might be enhancing the utility of advanced appearance modeling techniques in our algorithm. Another future work can be an issue that is related to the diversity of global hypotheses. We empirically showed that our solving scheme rapidly finds better solutions than existing works. That means our algorithm searches more various solutions than other algorithms. However, we have not examined the diversity among founded solutions. It might be beneficial for finding a better solving strategy to explicitly measure the diversity of resulting solutions. It can be also a good direction for reducing the computation time to monitor the population of pedestrians at the processing time. Recently, many reliable people counting methods have been proposed. Instead of generating and maintaining all possible tracks from input measurements, selecting tracks in a small number, which is decided by the people counting method, surely reduce the computational load. Applying parallel processing to our algorithm is an another issue for our future work. Nowadays, many excellent hardwares such as GPU have been invented for parallel processing. When an algorithm is properly implemented for those hardwares, a computation time is remarkably smaller than that of serial processing. Our global hypothesis generation scheme has a suitable structure for parallel processing because it solves multiple MWCP independently. Thus, to increase the practical value of our algorithm, it is trivial to provide an implement of our algorithm with parallel processing techniques such as multi-threading. As shown in the experiments, the performance of an object detector severely affects the overall tracking performance. In particular, pedestrians are frequently missed by an object detector when a scene is crowded. That is, low recall of an object detector is a bottleneck of overall tracking performance. Therefore, studying a pedestrian detection in crowded scenes is also a good future work even it is beyond the tracking algorithm.

Bibliography

[1] Mario Edoardo Maresca and Alfredo Petrosino, "Clustering local motion estimates for robust and efficient object tracking," in *Computer Vision-ECCV 2014 Workshops*. Springer, 2014, pp. 244–253.

[2] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "High-speed tracking with kernelized correlation filters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 3, pp. 583–596, 2015.

[3] Hyeonseob Nam and Bohyung Han, "Learning multi-domain convolutional neural networks for visual tracking," *arXiv preprint arXiv:1510.07945*, 2015.

[4] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[5] Gao Zhu, Fatih Porikli, and Hongdong Li, "Tracking randomly moving objects on edge box proposals," *arXiv preprint arXiv:1507.08085*, 2015.

[6] Jun Yang, Patricio A Vela, Zhongke Shi, and Jochen Teizer, "Probabilistic multiple people tracking through complex situations," in *11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.

[7] Anton Andriyenko and Konrad Schindler, "Globally optimal multi-target tracking on a hexagonal lattice," in *Computer Vision–ECCV 2010*, pp. 466–479. Springer, 2010.

[8] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 9, pp. 1820–1833, 2011.

[9] Anton Andriyenko and Konrad Schindler, "Multi-target tracking by continuous energy minimization," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1265–1272.

[10] Zheng Wu, Ashwin Thangali, Stan Sclaroff, and Margrit Betke, "Coupling detection and data association for multiple object tracking," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1948–1955.

[11] Anton Andriyenko, Konrad Schindler, and Stefan Roth, "Discrete-continuous optimization for multi-target tracking," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1926–1933.

[12] Zheng Wu, Jianming Zhang, and Margrit Betke, "Online motion agreement tracking," in *Proc. BMVC*, 2013.

[13] Menglong Yang, Yiguang Liu, Longyin Wen, Zhisheng You, and Stan Z Li, "A probabilistic framework for multitarget tracking with mutual occlusions," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 1298–1305.

[14] Longyin Wen, Wenbo Li, Junjie Yan, Zhen Lei, Dong Yi, and Stan Z Li, "Multiple target tracking based on undirected hierarchical relation hypergraph," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 1282–1289.

[15] Seung-Hwan Bae and Kuk-Jin Yoon, "Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 1218–1225.

[16] Bing Wang, Gang Wang, Kap Luk Chan, and Li Wang, "Tracklet association with online target-specific metric learning," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 1234–1241.

[17] Xiaojing Chen, Zhen Qin, Le An, and Bir Bhanu, "An online learned elementary grouping model for multi-target tracking," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 1242–1249.

[18] Xinchu Shi, Haibin Ling, Weiming Hu, Chunfeng Yuan, and Junliang Xing, "Multi-target tracking with motion context in tensor power iteration," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on.* IEEE, 2014, pp. 3518–3525.

[19] Horst Possegger, Thomas Mauthner, Peter M Roth, and Horst Bischof, "Occlusion geodesics for online multi-object tracking," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 1306–1313.

[20] Anton Milan, Kaspar Schindler, and Stefan Roth, "Detection-and trajectory-level exclusion in multiple object tracking," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 3682–3689.

[21] Anton Milan, Stefan Roth, and Kaspar Schindler, "Continuous energy minimization for multitarget tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 1, pp. 58–72, 2014.

[22] Martin Hofmann, Michael Haag, and Gerhard Rigoll, "Unified hierarchical multi-object tracking using global data association," in *Performance Evaluation of Tracking and Surveillance (PETS), 2013 IEEE International Workshop on*. IEEE, 2013, pp. 22–28.

[23] Robert T Collins and Peter Carr, "Hybrid stochastic/deterministic optimization for tracking sports players and pedestrians," in *Computer Vision–ECCV 2014*, pp. 298–313. Springer, 2014.

[24] Bo Yang and Ramakant Nevatia, "Multi-target tracking by online learning a crf model of appearance and motion patterns," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 203–217, 2014.

[25] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg, "Multiple hypothesis tracking revisited," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4696–4704.

[26] Francois Fleuret, Jerome Berclaz, Richard Lengagne, and Pascal Fua, "Multicamera people tracking with a probabilistic occupancy map," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 2, pp. 267–282, 2008.

[27] Zheng Wu, Nickolay I Hristov, Tyson L Hedrick, Thomas H Kunz, and Margrit Betke, "Tracking a large number of objects from multiple views," in *Computer Vision, 2009 IEEE 12th International Conference on.* IEEE, 2009, pp. 1546–1553.

[28] Saad M Khan and Mubarak Shah, "Tracking multiple occluding people by localizing on multiple scene planes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 3, pp. 505–519, 2009.

[29] Jerome Berclaz, Francois Fleuret, and Pascal Fua, "Multiple object tracking using flow linear programming," in *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on.* IEEE, 2009, pp. 1–8.

[30] Jerome Berclaz, Francois Fleuret, Engin Türetken, and Pascal Fua, "Multiple object tracking using k-shortest paths optimization," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 9, pp. 1806–1819, 2011.

[31] Horesh Ben Shitrit, Jerome Berclaz, Francois Fleuret, and Pascal Fua, "Tracking multiple people under global appearance constraints," in *Computer Vision (ICCV), 2011 IEEE International Conference on.* IEEE, 2011, pp. 137–144.

[32] Laura Leal-Taixé, Gerard Pons-Moll, and Bodo Rosenhahn, "Branch-and-price global optimization for multi-view multi-target tracking," in *Com-*

puter Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 1987–1994.

[33] Horesh Ben Shitrit, Jérôme Berclaz, François Fleuret, and Pascal Fua, “Multi-commodity network flow for tracking multiple people,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 8, pp. 1614–1627, 2014.

[34] Martin Hofmann, Denis Wolf, and Gerhard Rigoll, “Hypergraphs for joint multi-view reconstruction and multi-object tracking,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on.* IEEE, 2013, pp. 3650–3657.

[35] Anurag Mittal and Larry S Davis, “M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene,” *International Journal of Computer Vision*, vol. 51, no. 3, pp. 189–203, 2003.

[36] Zheng Wu, Thomas H Kunz, and Margrit Betke, “Efficient track linking methods for track graphs using network-flow and set-cover techniques,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* IEEE, 2011, pp. 1185–1192.

[37] Simone Calderara, Andrea Prati, and Rita Cucchiara, “Hecol: Homography and epipolar-based consistent labeling for outdoor park surveillance,” *Computer Vision and Image Understanding*, vol. 111, no. 1, pp. 21–42, 2008.

[38] Ran Eshel and Yael Moses, “Homography based multiple camera detection and tracking of people in a dense crowd,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on.* IEEE, 2008, pp. 1–8.

[39] Ran Eshel and Yael Moses, "Tracking in a dense crowd using multiple cameras," *International journal of computer vision*, vol. 88, no. 1, pp. 129–143, 2010.

[40] Rafael Muñoz-Salinas, R Medina-Carnicer, Francisco José Madrid-Cuevas, and A Carmona-Poyato, "Particle filtering with multiple and heterogeneous cameras," *Pattern Recognition*, vol. 43, no. 7, pp. 2390–2405, 2010.

[41] Mustafa Ayazoglu, Binlong Li, Caglayan Dicle, Mario Sznaier, Octavia Camps, et al., "Dynamic subspace-based coordinated multicamera tracking," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2462–2469.

[42] Ye Liu, Hui Li, and Yan Qiu Chen, "Automatic tracking of a large number of moving targets in 3d," in *Computer Vision–ECCV 2012*, pp. 730–742. Springer, 2012.

[43] Horst Possegger, Sabine Sternig, Thomas Mauthner, Peter M Roth, and Horst Bischof, "Robust real-time tracking of multiple objects by volumetric mass densities," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2395–2402.

[44] Moonsub Byeon, Songhwai Oh, Kikyung Kim, Haan Ju Yoo, and Jin Young Choi, "Efficient spatio-temporal data association using multidimensional assignment for multi-camera multi-target tracking," in *Proc. BMVC*, 2015.

[45] Rudolph Emil Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[46] Michael Isard and Andrew Blake, “Condensation—conditional density propagation for visual tracking,” *International journal of computer vision*, vol. 29, no. 1, pp. 5–28, 1998.

[47] Tao Yang, Yanning Zhang, Xiaomin Tong, Xiaoqiang Zhang, and Rui Yu, “Continuously tracking and see-through occlusion based on a new hybrid synthetic aperture imaging model,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3409–3416.

[48] Nigel JB McFarlane and C Paddy Schofield, “Segmentation and tracking of piglets in images,” *Machine vision and applications*, vol. 8, no. 3, pp. 187–193, 1995.

[49] Thomas A Feo, Mauricio GC Resende, and Stuart H Smith, “A greedy randomized adaptive search procedure for maximum independent set,” *Operations Research*, vol. 42, no. 5, pp. 860–878, 1994.

[50] Donald B Reid, “An algorithm for tracking multiple targets,” *Automatic Control, IEEE Transactions on*, vol. 24, no. 6, pp. 843–854, 1979.

[51] Amir Roshan Zamir, Afshin Dehghan, and Mubarak Shah, “Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs,” in *Computer Vision–ECCV 2012*, pp. 343–356. Springer, 2012.

[52] Dimitri J Papageorgiou and Michael R Salpukas, “The maximum weight independent set problem for data association in multiple hypothesis tracking,” in *Optimization and Cooperative Control Strategies*, pp. 235–255. Springer, 2009.

[53] Una Benlic and Jin-Kao Hao, “Breakout local search for maximum clique problems,” *Computers & Operations Research*, vol. 40, no. 1, pp. 192–206, 2013.

[54] Yaakov Bar-Shalom, *Tracking and data association*, Academic Press Professional, Inc., 1987.

[55] Y Bar-Shalom, T Fortmann, and M Scheffe, “Joint probabilistic data association for multiple targets in clutter,” in *Proc. Conf. on Information Sciences and Systems*, 1980, pp. 404–409.

[56] Samuel S Blackman, “Multiple-target tracking with radar applications,” *Dedham, MA, Artech House, Inc., 1986, 463 p.*, vol. 1, 1986.

[57] Robert W Sittler, “An optimal data association problem in surveillance theory,” *Military Electronics, IEEE Transactions on*, vol. 8, no. 2, pp. 125–139, 1964.

[58] Harold W Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[59] Katta G Murty, “Letter to the editor—an algorithm for ranking all the assignments in order of increasing cost,” *Operations Research*, vol. 16, no. 3, pp. 682–687, 1968.

[60] Thomas Kurien, “Issues in the design of practical multitarget tracking algorithms,” *Multitarget-Multisensor Tracking: Advanced Applications*, vol. 1, pp. 43–83, 1990.

[61] Lngemar J Cox and Sunita L Hingorani, “An efficient implementation of reid’s multiple hypothesis tracking algorithm and its evaluation for the

purpose of visual tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 2, pp. 138–150, 1996.

[62] Xiaoyi Ren, Zhipei Huang, Shuyan Sun, Dongyan Liu, and Jiankang Wu, "An efficient mht implementation using grasp," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 50, no. 1, pp. 86–101, 2014.

[63] Thomas A Feo and Mauricio GC Resende, "Greedy randomized adaptive search procedures," *Journal of global optimization*, vol. 6, no. 2, pp. 109–133, 1995.

[64] Elder Magalhães Macambira and Cid Carvalho De Souza, "The edge-weighted clique problem: valid inequalities, facets and polyhedral computations," *European Journal of Operational Research*, vol. 123, no. 2, pp. 346–371, 2000.

[65] Patric RJ Östergård, "A fast algorithm for the maximum clique problem," *Discrete Applied Mathematics*, vol. 120, no. 1, pp. 197–207, 2002.

[66] Pablo San Segundo, Diego Rodríguez-Losada, and Agustín Jiménez, "An exact bit-parallel algorithm for the maximum clique problem," *Computers & Operations Research*, vol. 38, no. 2, pp. 571–581, 2011.

[67] Wayne Pullan and Holger H Hoos, "Dynamic local search for the maximum clique problem," *Journal of Artificial Intelligence Research*, pp. 159–185, 2006.

[68] Wayne Pullan, "Phased local search for the maximum clique problem," *Journal of Combinatorial Optimization*, vol. 12, no. 3, pp. 303–323, 2006.

[69] Andrea Grosso, Marco Locatelli, and Wayne Pullan, "Simple ingredients leading to very efficient heuristics for the maximum clique problem," *Journal of Heuristics*, vol. 14, no. 6, pp. 587–612, 2008.

[70] Qinghua Wu and Jin-Kao Hao, "An adaptive multistart tabu search approach to solve the maximum clique problem," *Journal of Combinatorial Optimization*, vol. 26, no. 1, pp. 86–108, 2013.

[71] William Brendel, Mohamed Amer, and Sinisa Todorovic, "Multiobject tracking as maximum weight independent set," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* IEEE, 2011, pp. 1273–1280.

[72] Afshin Dehghan, Shayan Modiri Assari, and Mubarak Shah, "Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4091–4099.

[73] Fred Glover and Manuel Laguna, *Tabu Search*, Springer, 2013.

[74] Weina Ge and Robert T Collins, "Multi-target data association by tracklets with unsupervised parameter estimation.," in *BMVC*. Citeseer, 2008, vol. 2, p. 5.

[75] Ben Benfold and Ian Reid, "Stable multi-target tracking in real-time surveillance video," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* IEEE, 2011, pp. 3457–3464.

[76] Michael Roth, Martin Bauml, Ramakant Nevatia, and Rainer Stiefelhagen, "Robust multi-pose face tracking by multi-stage tracklet association,"

in *Pattern Recognition (ICPR), 2012 21st International Conference on.* IEEE, 2012, pp. 1012–1016.

[77] Weizhi Nie, Anan Liu, and Yuting Su, “Multiple person tracking by spatiotemporal tracklet association,” in *Advanced Video and Signal-Based Surveillance (AVSS), 2012 IEEE Ninth International Conference on.* IEEE, 2012, pp. 481–486.

[78] Keinosuke Fukunaga and Larry D Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition,” *Information Theory, IEEE Transactions on*, vol. 21, no. 1, pp. 32–40, 1975.

[79] Tao Zhao and Ram Nevatia, “Tracking multiple humans in crowded environment,” in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on.* IEEE, 2004, vol. 2, pp. II–406.

[80] Jianbo Shi and Carlo Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR’94., 1994 IEEE Computer Society Conference on.* IEEE, 1994, pp. 593–600.

[81] Cheng-Hao Kuo, Chang Huang, and Ramakant Nevatia, “Multi-target tracking by on-line learned discriminative appearance models,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.* IEEE, 2010, pp. 685–692.

[82] Roger Y Tsai, “An efficient and accurate camera calibration technique for 3d machine vision,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 1986*, 1986.

[83] Abraham Savitzky and Marcel JE Golay, "Smoothing and differentiation of data by simplified least squares procedures.," *Analytical chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.

[84] Larry C Andrews, *Special functions for engineers and applied mathematicians*, Macmillan, 1985.

[85] John M Robson, "Finding a maximum independent set in time o (2n/4)," Tech. Rep., Technical Report 1251-01, LaBRI, Université Bordeaux I, 2001.

[86] Yaakov Bar-Shalom, "Multitarget-multisensor tracking: advanced applications," *Norwood, MA, Artech House, 1990, 391 p.*, vol. 1, 1990.

[87] Anna Ellis, Ali Shahrokni, and James Michael Ferryman, "Pets2009 and winter-pets 2009 results: A combined evaluation," in *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on.* IEEE, 2009, pp. 1–8.

[88] Anton Andriyenko, Stefan Roth, and Konrad Schindler, "An analytical formulation of global occlusion reasoning for multi-target tracking," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1839–1846.

[89] Woonhyun Nam, Piotr Dollár, and Joon Hee Han, "Local decorrelation for improved pedestrian detection," in *Advances in Neural Information Processing Systems*, 2014, pp. 424–432.

[90] Rangachar Kasturi, Dmitry Goldgof, Padmanabhan Soundararajan, Vasant Manohar, John Garofolo, Rachel Bowers, Matthew Boonstra, Valentina Korzhova, and Jing Zhang, "Framework for performance evaluation of face,

text, and vehicle detection and tracking in video: Data, metrics, and protocol," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 319–336, 2009.

[91] Yuan Li, Chang Huang, and Ramakant Nevatia, "Learning to associate: Hybridboosted multi-target tracker for crowded scene," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE, 2009, pp. 2953–2960.

[92] Frank Hutter, Holger H Hoos, Kevin Leyton-Brown, and Thomas Stützle, "Paramils: an automatic algorithm configuration framework," *Journal of Artificial Intelligence Research*, vol. 36, no. 1, pp. 267–306, 2009.

[93] Helena R Lourenço, Olivier C Martin, and Thomas Stützle, *Iterated local search*, Springer, 2003.

[94] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[95] Edward Rosten and Tom Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006*, pp. 430–443. Springer, 2006.

[96] Anthony Dick Konrad Schindler Ian Reid Anton Milan, Seyed Hamid Rezatofighi, "Online multi-target tracking using recurrent neural networks," *arXiv preprint arXiv:1604.07866*, 2016.

[97] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler, "Learning by tracking: Siamese cnn for robust target association," *arXiv preprint arXiv:1604.07866*, 2016.

[98] Hironori Hattori, Vishnu Naresh Boddeti, Kris M Kitani, and Takeo Kanade, "Learning scene-specific pedestrian detectors without real data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3819–3827.

초록

본 학위 논문에서는 시야각이 겹치는 여러 대의 카메라를 이용하여 실시간으로 온라인하게 다중 물체를 추적하는 알고리즘을 제안한다. 영상을 이용한 다중 물체 추적은 그 실용적 효용성 때문에 지난 수십 년 동안 매우 집중적으로 연구되어 왔으며 특히, 한 대의 카메라로는 해결하기 어려운 물체 가려짐이나 미탐지 문제의 해결 방안으로 시야각이 겹치는 다중 카메라를 사용하는 방식이 제안되었다. 허나 다중 카메라 다중 물체 추적 방식은 단일 카메라를 사용하는 방식에 비하여 문제의 복잡도가 높기 때문에 최근에 발표된 대부분의 연구들은 온라인 방식이 아닌 일괄 처리방식에 기반하고 있다. 일괄 처리 방식은 입력 영상의 처음부터 마지막까지 모두 활용하여 물체를 추적하기 때문에, 그 성능이 온라인 방식보다 훨씬 뛰어나지만 즉각적인 결과를 요구하는 다양한 실용 분야에서는 사용하기 어렵다는 단점이 있다. 그러므로 본 학위 논문의 목적은 적은 계산량에도 불구하고 일괄 처리 방식을 기반으로 하는 최신 다중 카메라 다중 물체 추적 알고리즘에 버금가는 성능을 유지하는 온라인 다중 카메라 다중 물체 추적 알고리즘을 제안하는 것이다.

제안하는 알고리즘은 다중 카메라로부터 탐지된 입력 디텍션(detection)들 사이의 가능한 모든 조합에 대해 트랙 가정(track hypothesis) (혹은 줄여서 '트랙(track)')을 생성하고 이 중에서 현재의 추적 상황을 가장 잘 표현하는 트랙들을 선택하는 방식을 취한다. 이 때, 어떠한 트랙이 좋은 트랙인지를 판별하기 위해 모든 트랙들에 대해 우리가 제안하는 트랙 스코어(score)를 계산하는데, 이럴 경우 추적 결과를 찾는 것은 결국 합산 스코어가 가장 큰 트랙의 집합을 구하는 것이다. 우리는 생성된 트랙 중에서 합산 스코어가 가장 큰 트랙 집합을 구하는 문제를 최대 가중 클릭 문제(The maximum weighted clique problem)으로 변환하여 풀었다. 최대 가중 클릭 문제는 변수들 간 서로 공존/불공존 관계가 있는 조합문제를 풀 때 많이 차용되는 문제 형식이다. 최대 가중 클릭 문제는 잘 알려진 NP-Complete

문제이기 때문에 우리의 추적 문제를 최대 가중 클릭 문제로 변환하였을 경우, 이를 풀기 위해 소요되는 최대 계산 시간은 트랙의 총 수에 지수적으로 비례한다. 추적이 진행됨에 따라 생성되는 트랙의 개수 역시 지수적으로 증가하기 때문에 이러한 상황에서 최대 가중 클릭 문제를 푸는 것은 매우 어렵다.

우리는 이렇게 폭발하는 계산량을 완화하기 위해 매 프레임 적은 수의 트랙들로 이루어진 최대 가중 클릭 문제들의 동적으로 생성하는 방법을 제시한다. 우리는 연속되는 두 프레임 사이 추적 대상들의 움직임이 급격하지 않기 때문에 추적 결과들 역시 유사할 것이라고 가정하였다. 이러한 가정에 기반하여, 만약 이전 프레임에서 어떠한 추적 결과가 추적 대상들을 제대로 잘 표현했다면 그 다음 프레임에서는 해당 추적 결과에 포함된 트랙과, 그로부터 파생된 트랙들 외 몇몇 유관 트랙들만이 제대로되 추적 결과에 포함될 수 있다는 결론을 얻을 수 있다. 이러한 정보를 이용하면, 매 프레임 해답 탐색의 영역을 대폭 줄일 수 있다. 허나, 이렇게 하나의 정답만을 계속하여 전달할 경우, 중간에 잘못된 추적 결과를 채택한 이 후, 이로부터 복원할 수 있는 방법이 요원하다. 우리는 이러한 문제를 해결하기 위해 매 프레임 찾은 여러 개의 추적 결과 중, 가장 좋은 최대 K개의 추적 결과를 다음 프레임으로 전달하였다. 전달받은 프레임에서는 K개의 추적 결과 각각을 이용하여 최대 가중 클릭 문제들을 생성하고, 각 문제에서 또다시 복수 개의 추적 결과를 탐색한다. 우리가 제안하는 알고리즘은 위에서 설명한 과정을 추적이 종료되는 시점까지 반복하여 수행한다. 이러한 방식을 사용하면 여러 개의 최대 가중 클릭 문제를 동시에 풀어야 함에도 하나의 큰 최대 가중 클릭 문제를 푸는 것에 비해 소요되는 연산 시간이 적어지는데, 이는 앞 서 언급한 바와 같이 최대 가중 클릭 문제의 전체 연산량이 한 번에 고려되는 트랙의 개수에 지수적으로 비례하기 때문이다. 뿐만 아니라, 우리가 제안하는 알고리즘은 제한된 시간에 더 많은 탐색 영역을 고려할 수 있기 때문에 성능 면에서도 우월하다.

비록 최대 가중 클릭 문제를 동적으로 생성하는 방법을 사용한다 하더라도, 각 문제를 푸는 것은 여전히 많은 계산량을 요구한다. 우리는 이러한 계산량 문제를 보다 완화하기 위해 세 가지 추가적인 전략을 사용하였다. 첫째로, 트랙을 생성할

시, 디텍션들을 사용하는 것 대신에 짧은 구간 신뢰성 높은 경로 조각인 트랙렛 (tracklet)을 사용하였다. 이를 통해 실존 가능성이 많은 트랙들만을 생성하여, 결과적으로 후보 트랙의 개수를 줄일 수 있다. 두 번째로, 각각의 최대 가중 클릭 문제를 정확하게 풀지 않고 발견적 탐색 방법 중 하나인 Breakout Local Search (BLS) 알고리즘을 기용하였다. BLS를 이용하면 짧은 시간 안에 다수의 좋은 해답들을 효과적으로 구할 수 있다. 마지막으로, K개의 최선 추적 결과들을 이용하여 각 트랙이 다른 트랙들과의 관계에서 얼마나 좋은 트랙인지를 확률로써 측정하고, 이를 반영하여 트랙들을 가지치기하였다. 이러한 확률은 보다 적합한 가지치기를 보장해준다.

수준 기표가 되는 실험 데이터(benchmark dataset)에서 진행한 실험에서는 우리가 제안하는 알고리즘이 최신의 추적 성능을 달성하였다. 특히, 제안하는 알고리즘은 온라인하게 추적을 진행함에도 불구하고 일괄 처리에 기반을 둔 최신 다중 카메라 다중 물체 추적 알고리즘에 비견되는 성능을 보여주었다. 더욱이, 제안하는 알고리즘은 매우 빠른 시간 내에 만족할 만한 추적 결과를 도출하였기에 실시간 성능을 가짐을 확인하였다. 우리는 제안하는 최대 가중 클릭 문제의 동적 생성법이 실제로 추적 성능 및 계산 속도에 어떠한 영향을 미치는지도 자체 비교를 통해 확인하였고, 특정 숫자 이상의 K 값을 사용하였을 때, 모든 트랙을 고려하는 하나의 최대 가중 클릭 문제를 푸는 것에 비하여 더 좋은 성능을 훨씬 빠른 속도로 도출하는 것을 확인하였다.

주요어: 영상 추적, 다중 카메라 다중 물체 추적, 다중 가정 추적, 최대 가중 클릭 문제, 온라인 추적, 실시간 추적

학번: 2009-20848