國立交通大學

多媒體工程研究所

碩 士 論 文

非重疊攝影機間之
機率式交通流量建模方法

Probabilistic Modeling of Dynamic Traffic Flow between
Non-Overlapping FOVs

研 究 生：邱維辰

指導教授：莊仁輝　教授

指導教授：王聖智　教授

中 華 民 國 九 十 八 年 七 月

非重疊攝影機間之機率式交通流量建模方法

Probabilistic Modeling of Dynamic Traffic Flow between Non-Overlapping FOVs

研 究 生：邱維辰	Student：Wei-Chen Chiu
指導教授：莊仁輝	Advisor：Jen-Hui Chuang
指導教授：王聖智	Advisor：Sheng-Jyh Wang

國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年七月

非重疊攝影機間之機率式交通流量建模方法

學生：邱維辰　　　　　　　　　　　　　　　　　指導教授：莊仁輝、王聖智

國立交通大學多媒體工程研究所碩士班

摘　　　要

在延伸現有視覺監視系統至全面性交通監控的過程中，如何推斷多攝影機間之交通狀況是一項非常重要的課題。在本論文中，我們提出一個於非重疊攝影機間之機率式交通流量建模的有效方法。基於物體於攝影機間移動時其移動時間會符合某總體模型之假設，並藉由連續估計此模型之參數，我們可以推斷出在不可視區域中之動態交通狀況。原則上如果我們知道攝影機間之移動物體的對應關係，則移動時間的總體模型的參數即可被估計出來。然而尋找物體間之對應關係在電腦視覺領域中仍然是一個懸而未決的問題。在本文中，我們將非重疊攝影機間物體對應關係之建立與總體移動時間模型之參數估計視為一個統整性的最佳化問題，並利用期望值最大化演算法來反覆尋找最佳的物體對應關係以及總體模型之參數解。在實際場景中的實驗結果證明了我們的方法能有效地估計出總體移動時間模型之參數並準確推斷出動態的交通情況。

關鍵字：交通流量，非重疊攝影機

Probabilistic Modeling of Dynamic Traffic Flow between Non-Overlapping FOVs

Student：Wei-Chen Chiu

Advisors：Prof. Jen-Hui Chuang, Prof. Sheng-Jyh Wang

Institute of Multimedia Engineering
National Chiao Tung University

ABSTRACT

The ability to infer the traffic status across multiple cameras allows the extended use of existing vision-based surveillance systems to global traffic monitoring. In this paper, we propose an efficient algorithm to probabilistically model the dynamic traffic flow between non-overlapping FOVs. By assuming the transition time of object moving across cameras follows a global model and consecutively estimate the model parameters, we may infer the time-varying traffic status in the unseen region. In principle, the parameters of the transition time model can be estimated if the object correspondence between non-overlapping FOVs is known. However, object correspondence itself is still an unsolved problem in computer vision. In this paper, we model object correspondence and the parameters estimation as a unified problem in a proposed Expectation-Maximization (EM) based framework. By treating object correspondence as a latent random variable, the proposed framework can iteratively search for the optimal object correspondence and model parameters. Experimental results on real data show the accuracy of dynamic model estimation and the beneficial inference of the traffic status.

Keywords: Traffic flow, Non-overlapping FOVs

誌謝

首先感謝我的指導教授，莊仁輝老師與王聖智老師。從大四預研生時期進入實驗室，到碩士班結束，兩位老師不僅給予我許多研究方面的提點，也在處事態度及人生經驗上讓我獲益良多。能夠順利的拿到碩士學位，實在是因為有兩位老師的用心指導及包容。在此向辛苦的兩位老師致上最高的敬意與感謝。另外也要感謝我的口試委員，陳永昇老師和林嘉文老師，對這篇論文所提出的指導和建議。

接著要感謝實驗室的各位學長學弟們。尤其感謝敬群學長，從我大三於國外交換學生階段開始就給予我研究領域上的啟蒙，其中經歷過工研院實習及碩士班的研究期間，學長一直不吝於犧牲自己的時間與我討論，給我很多的建議以及參考資料、學習方向，更分享了許多人生的歷練，這都是我永遠無法忘記的，實是我生命中的貴人。很感謝慈澄學長、禎宇學長及國華學長平時的照顧，以及一起在實驗室中打拚畢業的瑞男、文中、庭瑋、世旻、聖中、永昌、尚一、宣良、邦展，大家的相互討論與分享，都是我研究的助力。周節、瑋國、育瑋等學弟妹平時的諸多幫忙、已經畢業的學長姐們的經驗傳承，更讓我在研究所期間助益良多。

最後感謝我的父母、家人對我的關懷、支持及鼓勵，感謝馨方的陪伴、包容與體諒，你們都是我心理上最大的支柱。沒有你們，我或許無法堅持到最後。感謝陪伴我走過學習歷程的眾多朋友與同學，我的一切都來自於你們的貢獻，由衷感謝。

Contents

List of Figures

1 Introduction

With the rapid development of economy, the increasingly serious problem of traffic system is becoming one of the crucial factors for the growth of national economy. As the urgent requirement of road using comes up, Intelligent Transport System (ITS) emerges to address the problems of road safety and congestion. ITS varies in the technologies applied, from basic management systems such as car navigation and traffic signal control system, to monitoring applications, such as security CCTV systems and to more advanced applications like prediction of transit vehicle arrival time. However, no matter what kind of application it is, the most commonly used quantity for the performance assessment of ITS is traffic flow. Hence, the modeling of traffic flow has become a key ingredient of ITS.

The traffic flow research has been widely discussed and studied for over 70 years. Among them, the development of traffic flow theory provides a tool to help transportation engineers understand and express the characteristics of traffic state. The basic idea of traffic flow theory is to model the microscopic and macroscopic relationships among traffic stream variables, including speed, flow and concentration (density). Many models have been proposed as analytical techniques such as traffic stream models including Greeshield's model and Edie's model, shock wave analysis, deterministic and stochastic queuing theories, and capacity analysis [1, 2, 3, 4]. Moreover, to make ITS more efficient, dynamic traffic flow model has gained popularity. The term "dynamic" here is to describe the traffic state changes caused by stochastic traffic flow uncertainties, such as traffic jams, accidents, weather condition, queue length in front of a traffic light, etc.

Consider an easier traffic flow modeling problem between two separate re-

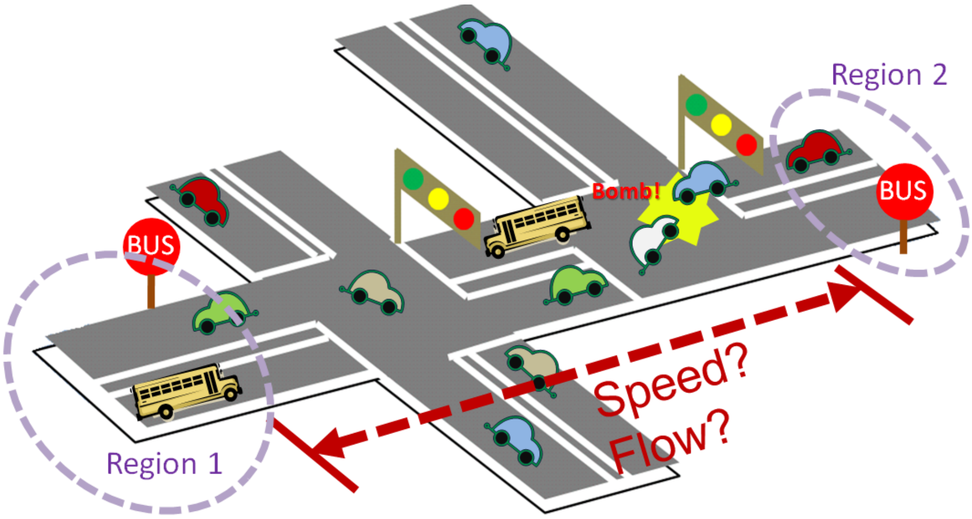


Figure 1.1: A simple case: monitor traffic between two regions.

gions, as shown in Figure 1.1. Given a vehicle moving from one region toward another, its traveling time may increase due to a heavy traffic state associated with an accident or traffic jam, or decrease in off–peak hours or due to other phenomena associated with smooth traffic flow. Hence, instead of using lots of variables to model the traffic flow, we can infer the traffic state by observing the "delay times" or "transition time" of objects moving across different regions.

Inspired by this intuitive idea, many research works and applications focus on dealing with the relevant problem of "delay times." For example, in [5] and [6] Francesco Viti et al. claimed that the transition time of vehicles in urban networks is determined half by the driving time and half by the delay at intersections that mainly depends on the arrival time, queue length and the signal frequency. Hence, they proposed a probabilistic model for delay times at controlled intersection. In addition, another major topic of ITS about the prediction of transit vehicle arrival time is also based on the concept of the delay modeling. In [7, 8, 9, 10], Dailey et al. presented an algorithm to predict transit vehicle arrival time by using an automatic vehicle location system (AVLS) to collect data that include time and

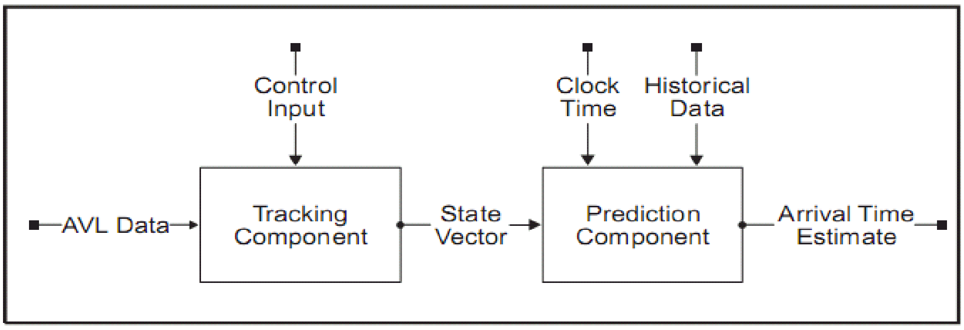


Figure 1.2: Overview of Dailey's System [8].

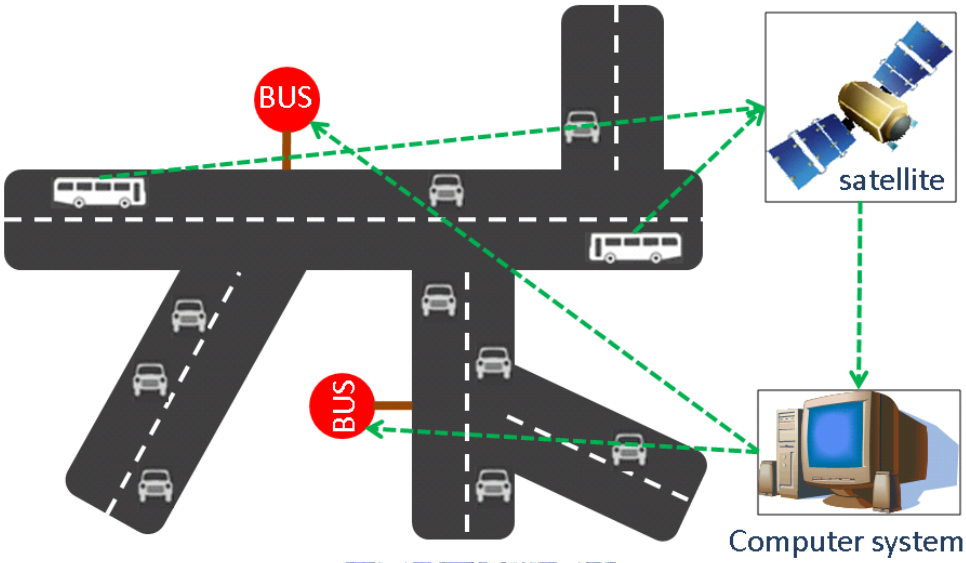


Figure 1.3: A GPS–based AVLS for bus arrival time prediction.

location pairs. The data are used with historical statistics in an optimal filtering framework to predict future arrivals. An overview of their system is shown in Figure 1.2.

There are still lots of methods trying to provide an accurate prediction of transit vehicle arrival times. For instance, in [11], Steven et al. used artificial neural networks (ANNS) trained with historical AVLS data to do dynamic prediction; in [12], the authors used support vector machines (SVM) instead. An example of GPS–based AVLS for bus arrival time prediction is illustrated in Figure 1.3.

However, all the methods mentioned above suffers from drawbacks of being expensive in installation and maintenance because they are highly dependent upon

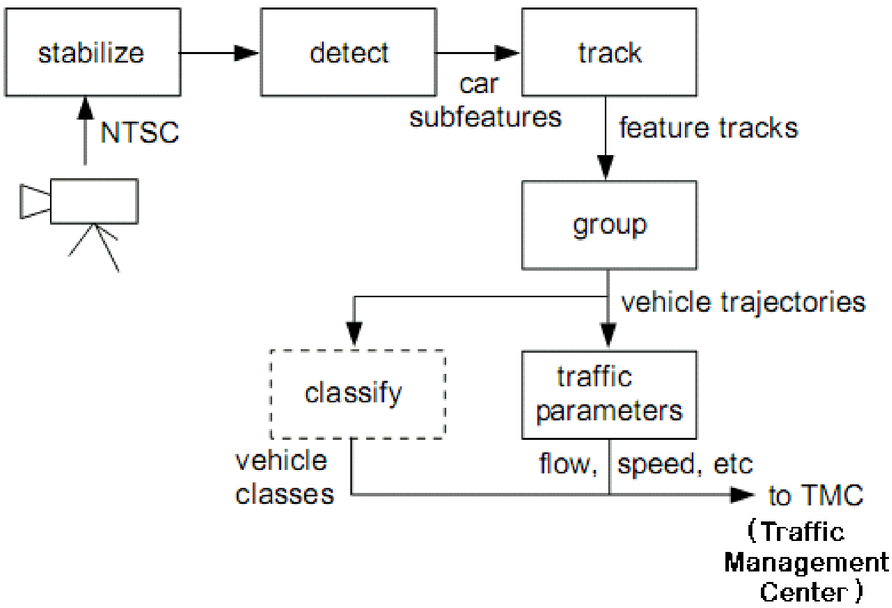


Figure 1.4: Overview of Beymer's system [15].

AVLS, which generally costs a lot. Besides, due to the privacy issue, promoting those AVLS–based applications is not as easy as it appears. Fortunately, in recent years, as a result of rising demand for security, many surveillance cameras have been installed in major intersections and some regions of interest, and the number of cameras is growing rapidly. This trend enables the development of another form of ITS technologies – "vision–based traffic surveillance system." This kind of system has the features of relatively low installation cost, easy operation and less traffic disruption during maintenance in contrast to AVLS [13, 14].

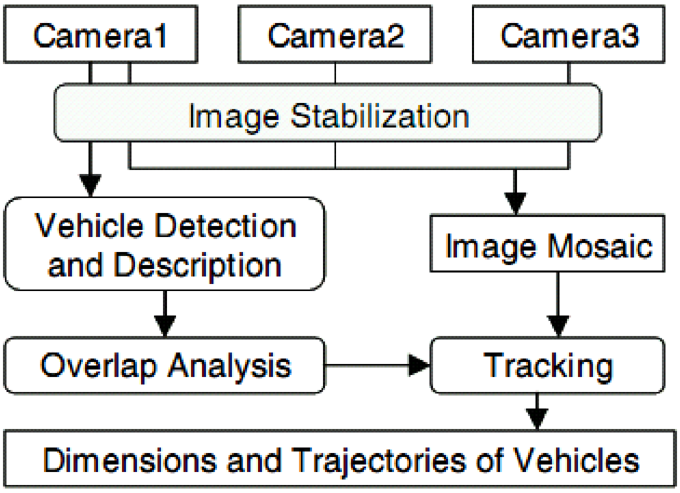
Actually, many image processing and computer vision techniques for the analysis of traffic flow video sequences have been widely used for road traffic monitoring and traffic data collection. Considerable amount of improvements have also been made [14]. For example, in [15, 16] Beymer et al. developed a feature based tracking approach for the task of tracking vehicle under congestion. By tracking individual vehicles, the system can measure conventional traffic parameters as well as new metrics suitable for improved automated surveillance.

(a) (b)

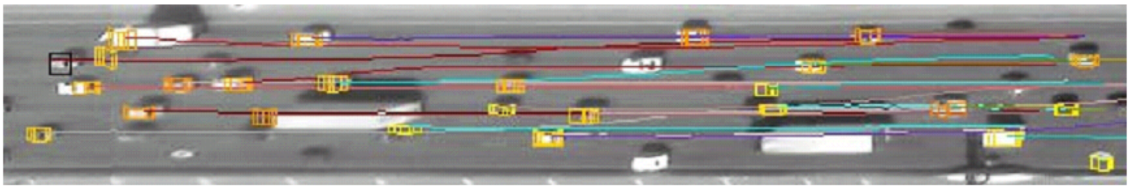
Figure 1.5: Results of Beymer's system [15]. (a) Sample feature tracks. (b) Sample feature groups.

Figure 1.4 shows an overview of their system. Some results are shown in Figure 1.5. In [17], the authors used probabilistic line feature grouping algorithm to perform model–based 3–D vehicle detection and description. With the tracking algorithm based on the zero–mean correlation matching, their approach provides a method for vehicles monitoring. The flow diagram and a result of their system are shown in Figure 1.6.

Nevertheless, almost all existing vision–based traffic surveillance systems are constrained by cameras' field of views (FOVs). Moreover, these systems only concern about the visible information in the video sequences and only indicate the "local" traffic status at the camera location. However, for a camera surveillance system that covers a wide area, it is not always possible to have overlapping views; the observed regions can be widely separated in time and space. As shown in Figure 1.7, cameras only have "local "information from their FOVs. Without the traffic status in the unseen region between non–overlapping cameras, we cannot infer a "global" traffic flow. Hence, under the condition that most existing cameras' FOVs are non–overlapping, if we want to build a practical vision–based wide–area surveillance system for traffic flow monitoring, we have to estimate the



(a)



(b)

Figure 1.6: (a) Flow diagram of Kim's system. (b) An example tracking result [17].

traffic state in the unseen regions between FOVs.

In this thesis, our main objective is to extend existing vision–based surveillance systems from local–range monitoring toward global–range monitoring of traffic flow by inferring the traffic state between non–overlapping FOVs. The key barrier we will meet is having no visual information of the road, which links the non–overlapping FOVs, to observe the traffic state directly. In general, if we can identify the same vehicle across cameras (e.g., using license plate recognition), it will be easy to achieve our objective. However, there is no guarantee that we can always achieve the correspondences of objects across cameras. Hence, instead of solving the problem by identifying the same vehicle across cameras, here we propose a different method.

We first assume that the transition time of most vehicles across two non–overlapping FOVs satisfies a global distribution model. Accordingly, if we have

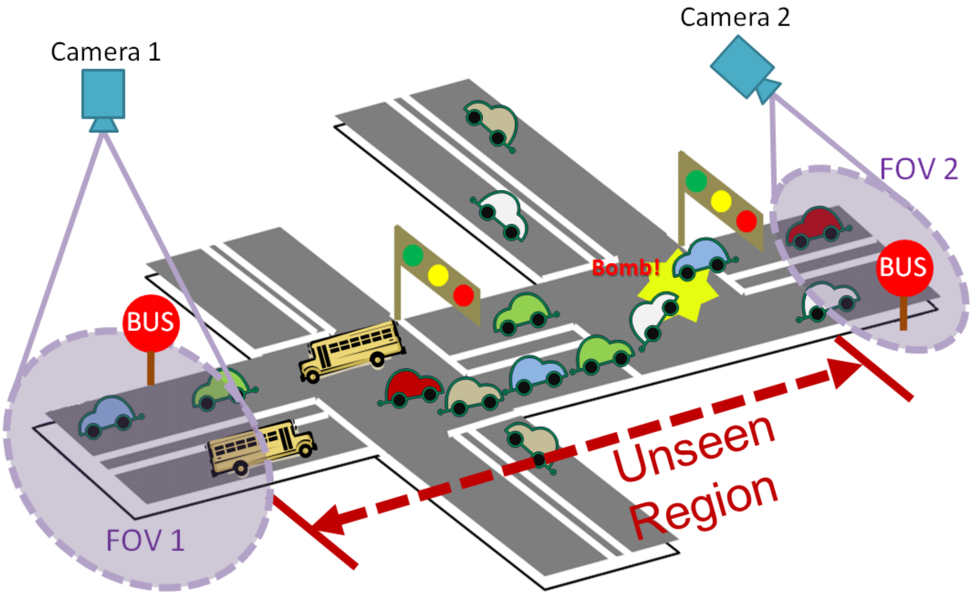


Figure 1.7: An example of traffic surveillance system with non–overlapping cameras.

the prior knowledge of the global transition–time distribution, we may infer the rough correspondence among our targets. Unfortunately, that prior knowledge is not available in this problem. From another point of view, every correspondence of vehicles can be treated as an observation from the global transition–time model. By collecting these observations of correspondence, we can derive the global model. Again, we do not have any information about correspondences. Hence, in our system, we exploit the physical dependence between the correspondences of buses and the global transition–time distribution, and formulate the object correspondence and the parameters estimation for the transition–time distribution as a unified problem in a proposed Expectation–Maximization (EM) based framework. By treating object correspondence as a latent random variable, the optimized solution can be found through iterations.

As for, the transition–time model, since it may might vary with time due to dynamic changes of traffic state, a sequential EM are used to adapt to such changes over time. A simple interface of our system is shown in Figure 1.8. Such

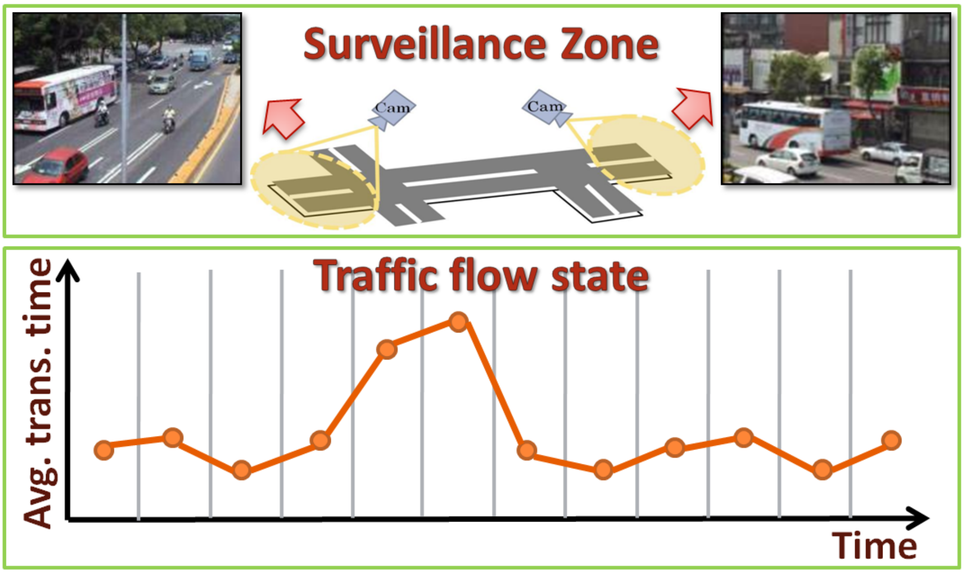


Figure 1.8: A simple interface of proposed system. It provides the information of the time–variant traffic state between non–overlapping FOVs.

a system can also be combined with the map services, such as Google Map or the CCTV Live–cam service on some important traffic spots, so that users can fetch the current traffic state from the Internet.

This thesis is organized as follows. In Chapter 2, we will first introduce some related works and mathematical techniques. In Chapter 3, we will present the proposed algorithm for modeling the dynamic changes of traffic state between non–overlapping FOVs. Some experimental results are shown in Chapter 4. Finally, we give our conclusion in Chapter 5.

2 Related Work

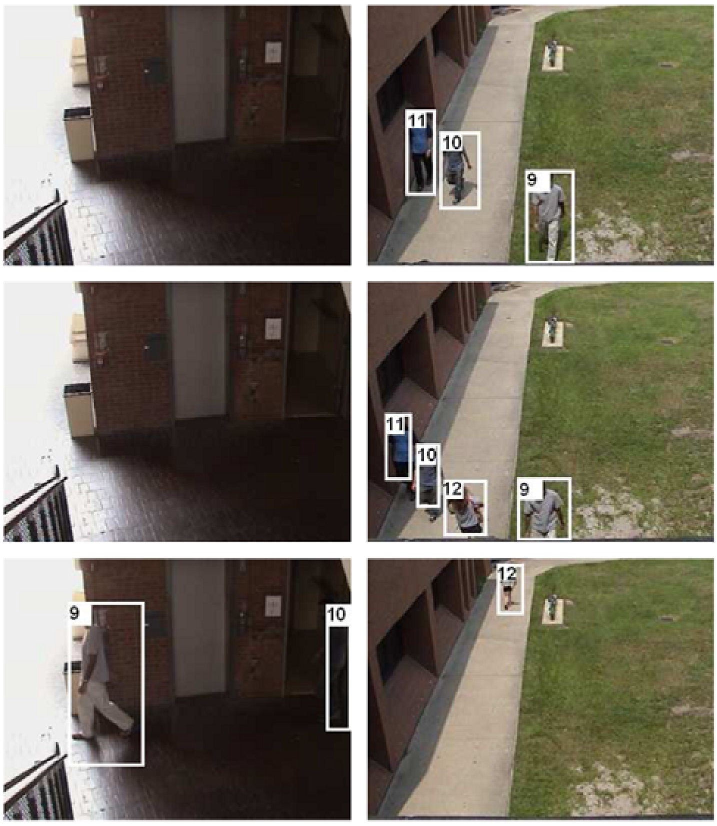
In this chapter, we will first introduce a few related researches on surveillance systems with non–overlapping cameras. Since in the proposed method we use some mathematical techniques such as Monte Carlo Markov Chain and Expectation–Maximization, we will also give some brief introductions of these mathematical techniques.

2.1 Surveillance systems with non–overlapping cameras

In general, the key problem of surveillance systems with non–overlapping cameras is to build the relationships between objects moving through different FOVs, i.e., the association of objects across cameras. Once the association is established for all objects, problems such as co–operative object tracking, object counting, and monitoring of objects activities become easier to resolve.

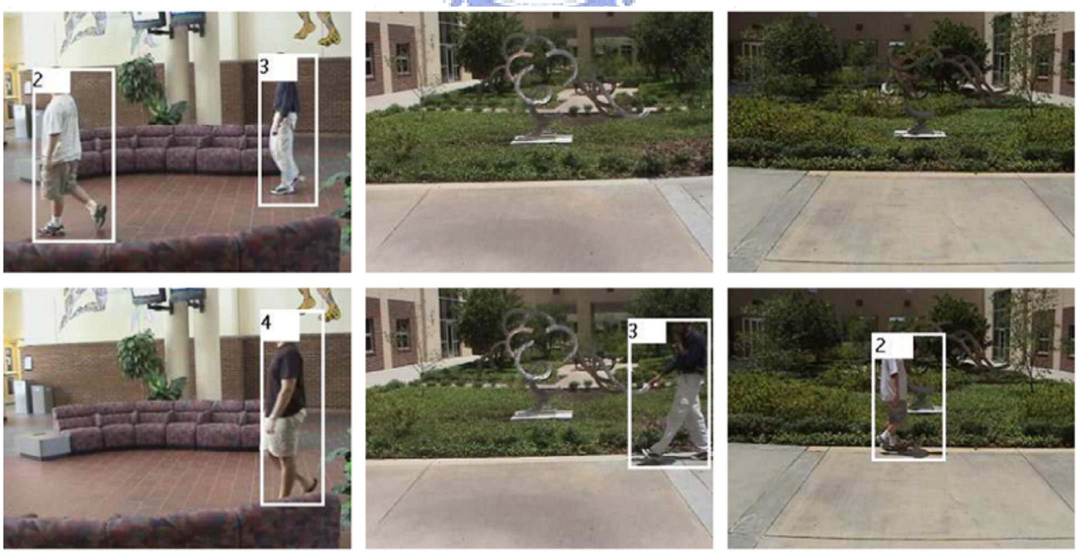
In [18, 19], Javed et al. proposed a method to establish the correspondence between observations across cameras. During a training phase, they manually established the correspondence to discover the relationships between cameras. They took the locations of exits and entrances between cameras, directions of movement, and the average transition time across cameras as space–time clues. Also they learned the inter-camera illumination for modeling the changes of appearance of objects across cameras as appearance clues. With these clues, tracks of targets moving through FOVs are corresponded by maximizing the posterior probability. Some results of their approach are shown in Figure 2.1.

The method proposed in [20] by Song et al. used feature matching for object tracking in a camera network. They treated similarities between features (ap-



Cam 1 Cam 2

(a)



Cam 3 Cam 2 Cam 1

(b)

Figure 2.1: Two results from [18].

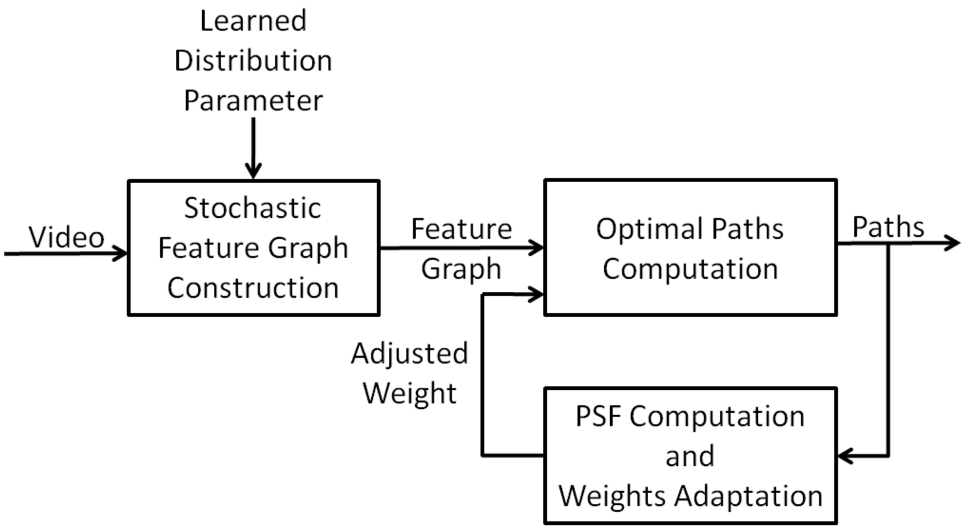
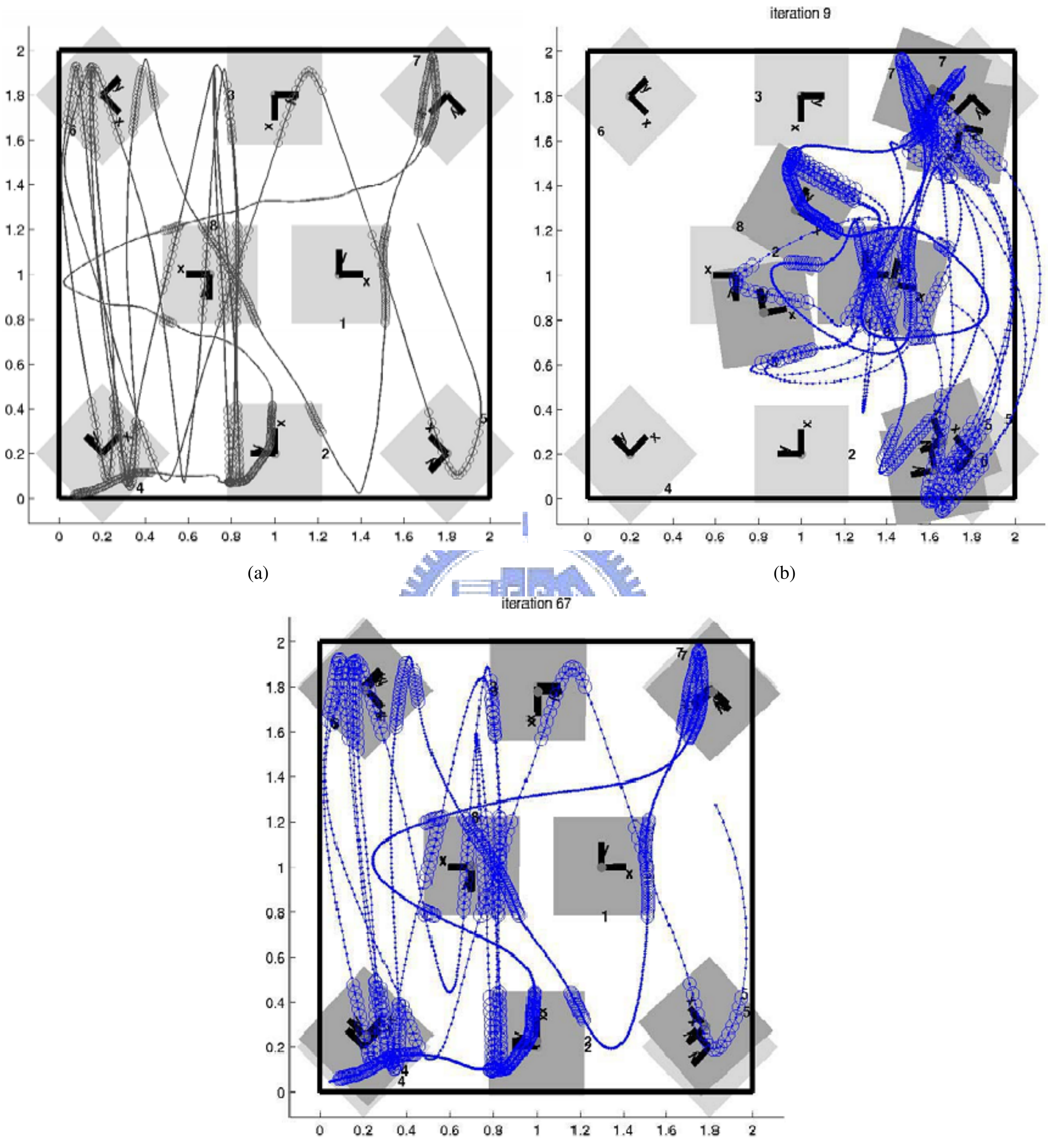


Figure 2.2: The framework of Song's system, redrawed from [20].

pearance and biometric) as random variables, whose probability distribution were created via supervised learning. By building the feature graph containing the feature vector and similarity score observed over space and time, tracks of people can be found as the optimal paths in this graph. Besides, to deal with the interdependence between features, they add a path smoothness function to correct wrong correspondences and adjust the tracking result. The framework of their system is shown in Figure 2.2.

Some works have been focused on the recovery of object tracks as well as the estimation of poses between cameras with non-overlapping FOVs. Rahimi et al., in [21], presented an approach that reconstructed the trajectory of a target across non–overlapping cameras and simultaneously computed external calibration parameters. Each camera is assumed to measure the location and velocity of a moving target within its field of view with respect to the camera's ground-plane coordinate system. With the assumption that the target's dynamic state of location and velocity evolves according to linear Gaussian Markov dynamics, they recover the target's trajectory and the calibration parameters of the cameras using



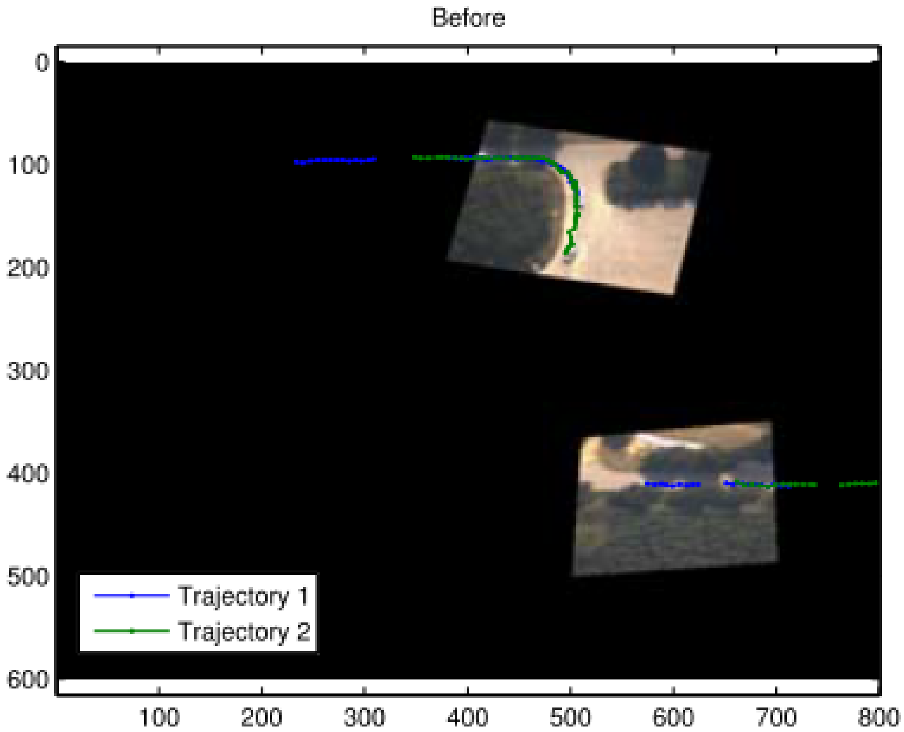
(a) (b) (c)

Figure 2.3: An example from [21].(a)Ground truth of sensor locations and targets' trajectories. (b) After 9 iterations. Blue denotes the recovered trajectories. Darker gray squares are estimated sensor locations. (c) After 67 iterations.

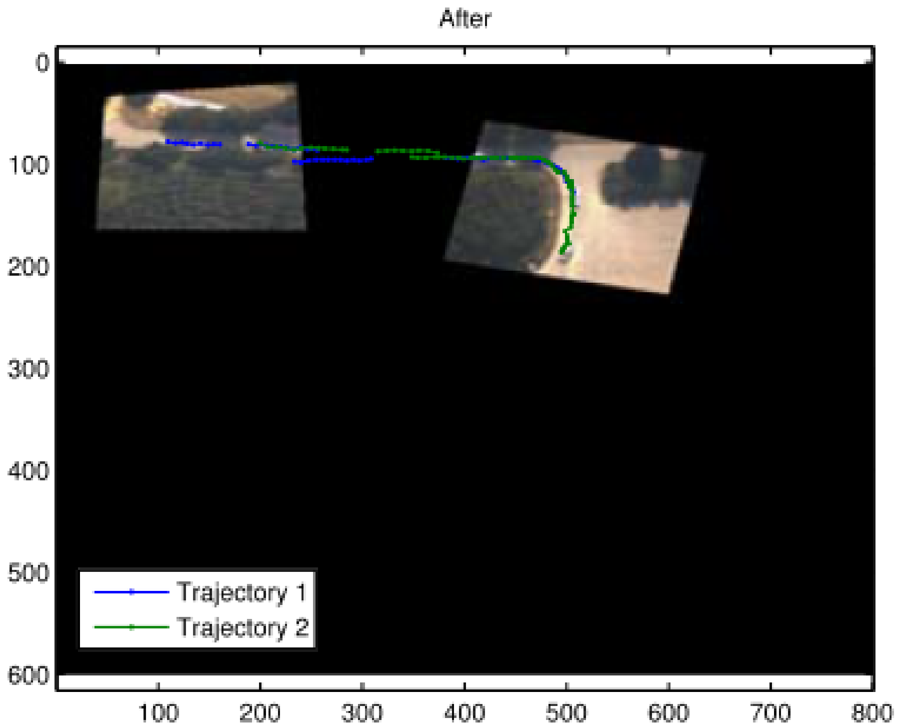
maximum a posterior (MAP) estimation. Figure 2.3 shows an example of their algorithm.

In [22], Sheikh et al. presented a unified framework for the association of multiple objects across multiple cameras in planar scenes. They modeled the scene as a plane in 3–D space and used homographies to describe the relationships between cameras. For object motion, they made an assumption that the object trajectory can be described by a polynomial kinetic model. With such an assumption, their system is able to recover object associations, inter–camera transformations and canonical trajectories across cameras, irrespective of whether the cameras are stationary or dynamic, or whether the fields of view are overlapping or not as long as the kinetic model is valid. Then they recovered the assignment of associations, while simultaneously computing the maximum likelihood estimates of the inter–camera homographies and the trajectory parameters by using the Expectation Maximization algorithm. An example of object association across multiple non–overlapping cameras is shown in Figure 2.4.

Instead of using the appearance feature or object motion model to directly infer the correspondence of objects across non–overlapping cameras, some research works try to recover the topology of a number of cameras based on co–occurrence of entries and exits. During the process of topology recovering, some indirect information about the correspondence of objects can also be extracted. In [23], Makris et al. assumed a single mode transition time distribution between FOVs and exhaustively search for the location of the mode by applying cross–correlation to the arrival and departure time of objects across cameras. Their method assumes all departures and arrivals within a time window are implicitly having one–to–one correspondences. The transition–time distribution obtained from the correspon-



(a)



(b)

Figure 2.4: Object association across multiple non–overlapping cameras [22]. (a) Initialization. (b) Converged solution with trajectories from two cameras merged into one.

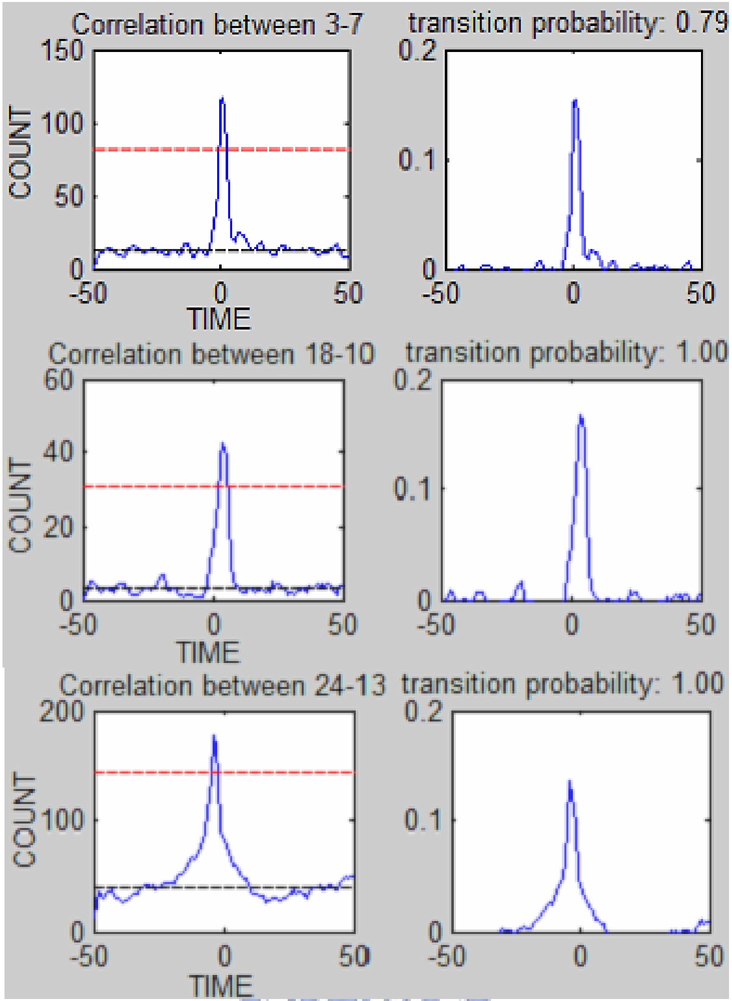


Figure 2.5: Cross–correlation functions and transition probability functions for selected pairs of entry/exit zones. Time is measured in seconds. Red line indicates the peak detection threshold, while the black line indicates the level of cross–correlation if the zones were uncorrelated [23].

dences is examined for a peak by thresholding based on the mean and standard deviation. Figure 2.5 shows examples of cross–correlation function and transition probability function for selected pairs of entry/exit zones.

Markis' approach has been extended by Stauffer [24] and Tieu et al. [25] by providing a more rigorous definition of transition based on statistical significance. Stauffer tested the hypothesis that the correlation between exit and entry events that may or may not contain valid object transitions is similar to the expected correlation when there are no valid transitions. Tieu et al. also presented that the method of [23] suffers from the assumption of unimodal transition distribution,

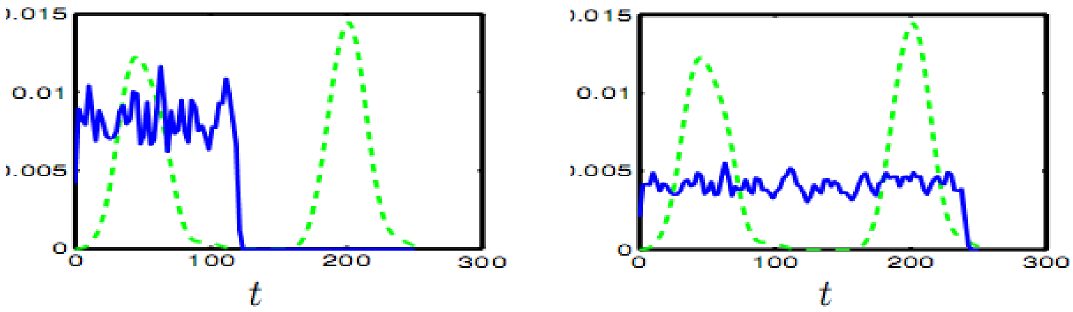


Figure 2.6: Transition distributions obtained using correlation with different time windows all fail to match the simulated multi–modal distribution (dashed plot).In addition, there is no clear maximum peak indicating statistical dependence [25].

as shown in Figure 2.6. Unlike using cross–correlation in [23], their approaches used information–theoretic mutual information or entropy to compute statistical dependence between observations across cameras.

However, most of the features or models used in the above methods are insufficient in real life scenario for traffic flow surveillance. For example, in the wide–area surveillance, the appearance model may become invalid because the observation may occupy only a few pixels or looks very different from different viewing angles of cameras ,or under sudden lighting changes. Likewise, for vehicles in the real life traffic which might contain complicated behavior patterns such as car–following and queuing, the aforementioned object motion won't be able to provide accurate descriptions. Moreover, due to the dynamic changes of real life traffic, the transition–time distribution may not be correctly found by cross–correlation or by maximizing the dependence between observations in two cameras.

2.2 Markov Chain Monte Carlo

Markov chain Monte Carlo (MCMC) method is a general computing technique that is often applied to solving integration and optimization problems in large dimensional spaces. As a result, it has been widely used in machine learning, physics, chemistry, biology statistics, and computer science. MCMC can be regarded as a strategy for sampling from some complex distributions of interest. It is named after the fact that the next sample is randomly generate from the previous sample by following a Markov chain mechanism, under which the transition probabilities between sample values are only a function of the most recent sample values. This mechanism can facilitate the chain to spend more time in more important regions [26]. In the literature, different algorithms of MCMC have already been well developed. We will introduce two commonly used ones in this section.

2.2.1 The Metropolis–Hastings algorithm

One original motivation for applying MCMC comes from the attempt to handle the situations when we cannot directly draw samples from some complex probability distributions. Metropolis–Hastings (MH) algorithm provides a general approach for obtaining a sequence of random samples from those distributions. The sequence can then be used to approximate the distribution (i.e., to generate a histogram). This algorithm is the most popular MCMC method [27, 28]. Many practical MCMC algorithms can be interpreted as special cases or extensions of this algorithm [29].

Suppose our goal is to draw samples from a probability distribution $p(x)$, MH algorithm creates a Markov chain for sampling a candidate value x^* given the current value x according to a proposal distribution $q(x^*|x)$. This new candidate is accepted with acceptance probability $A(x, x^*) = min\{1, [p(x)q(x^*|x)]^{-1}p(x^*)q(x|x^*)\}$; otherwise, the current value of x is retained. The pseudo–code of MH algorithm is listed in Figure 2.7.

Algorithm 1 Metropolis–Hastings algorithm

Initialize $x^{(0)}$
for $i = 0$ to $N-1$ **do**
  Sample u from $U(0,1)$
  Sample x^* from $q(x^*|x^{(i)})$
  if $u < A(x^{(i)}, x^*) = min\{1, \frac{p(x^*)q(x^{(i)}|x^*)}{p(x^{(i)})q(x^*|x^{(i)})}\}$ **then**
    $x^{(i+1)} = x^*$
  else
    $x^{(i+1)} = x^{(i)}$
  end if
end for

Figure 2.7: Metropolis–Hastings algorithm.

The main idea of MH algorithm is to simulate sampling from the target distribution $p(x)$ by sampling from another easy–to–sample proposal distribution $q(x^*|x)$. Hence, the more $q(x^*|x)$ matches the shape of $p(x)$, the better MH algorithm works. Besides, the term $[p(x)q(x^*|x)]^{-1}p(x^*)q(x|x^*)$ in the acceptance probability give us a measure about how much the Markov chain would like to move toward the candidate value x^*.

As shown in Figure 2.8, the authors in [29] give an example of running the MH algorithm based on a Gaussian proposal distribution $q(x^*|x^{(i)}) = N(x^{(i)}, 100)$ and a bimodal target distribution $p(x) \propto 0.3exp(-0.2x^2)+0.7exp(-0.2(x-10)^2)$ for 5000 iterations. As expected, the histogram of the samples approximates the target distribution.

2.2.2 The Gibbs sampler

Gibbs sampler [30], is a special case of the Metropolis–Hastings algorithm [26]. It is an algorithm to generate a sequence of samples from the joint probability distribution of two or more random variables. The key to the Gibbs sampler is that

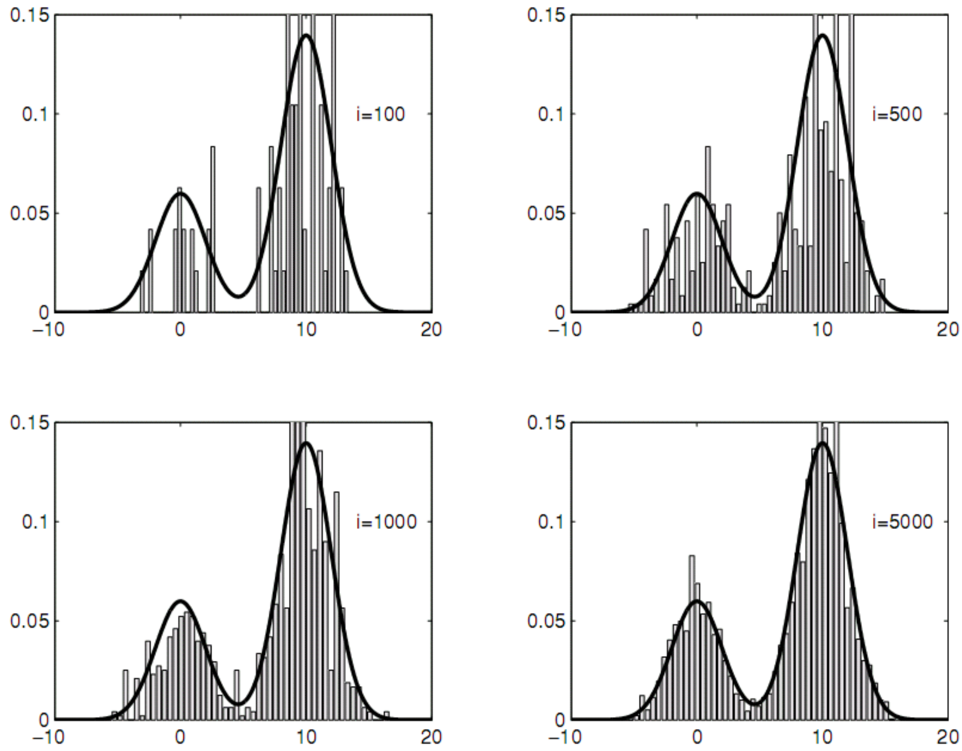


Figure 2.8: Target distribution and histogram of the MCMC samples at different iteration points [29].

one only considers univariate conditional distributions — the distribution when all of the random variables but one are kept fixed. Hence, this method is applicable when the joint distribution is not known explicitly but the univariate conditional distribution of each random variable is known.

Suppose we have an n–dimensional vector x and the expression for the full univariate conditional distribution $p(x_j|x_{-j}) = p(x_j|x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$. By taking the following proposal distribution for $j = 1, \ldots, n$

$$q(x^*|x^{(i)}) = \begin{cases} p(x_j^*|x_{-j}^{(i)}) & \text{If } x_{-j}^* = x_{-j}^{(i)} \\ 0 & \text{Otherwise.} \end{cases} \tag{2.1}$$

Algorithm 2 Gibbs sampler.

Initialize $x_{(0,1:n)}$
for $i = 0$ to $N-1$ **do**
  Sample $x_1^{(i+1)} \sim p(x_1|x_2^{(i)}, x_3^{(i)}, \ldots, x_n^{(i)})$.
  Sample $x_2^{(i+1)} \sim p(x_2|x_1^{(i+1)}, x_3^{(i)}, \ldots, x_n^{(i)})$.
  $\vdots$
  Sample $x_j^{(i+1)} \sim p(x_j|x_1^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_n^{(i)})$.
  $\vdots$
  Sample $x_n^{(i+1)} \sim p(x_n|x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_{n-1}^{(i+1)})$.
end for

Figure 2.9: Gibbs sampler.

The corresponding acceptance probability is:

$$\begin{aligned} A(x^{(i)}, x^*) &= \min\{1, \frac{p(x^*)q(x^{(i)}|x^*)}{p(x^{(i)})q(x^*|x^{(i)})}\} \\ &= \min\{1, \frac{p(x^*)p(x_j^{(i)}|x_{-j}^{(i)})}{p(x^{(i)})p(x_j^*|x_{-j}^*)}\} \\ &= \min\{1, \frac{p(x_{-j}^*)}{p(x_{-j}^{(i)})}\} \\ &= 1 \end{aligned} \tag{2.2}$$

Hence, the acceptance probability for each new candidate random value x^* is one. In other words, each new proposal sample is always accepted. The pseudo–code of Gibbs sampling is presented in Figure 2.9.

2.3 Expectation–Maximization

The Expectation Maximization algorithm (Hartley, 1958 [31]; Dempster et al., 1977 [32]; McLachlan and Krishnan, 1997 [33]) performs maximum likelihood estimation for parameters in probabilistic models, where the model depends on some unobserved variables. EM is an iterative optimization method consisting of an expectation (E) step, which finds the distribution of the unobserved variables, given the known values for the observed variables and the current estimate of the parameters; and a maximization (M) step, which computes the parameters which maximize the expected likelihood found in the E step, under the assumption that the distribution found in E step is correct. These parameters are again used to determine the distribution of the unobserved variables in the next E step [34]. To understand the EM algorithm, we give a simple description of EM in the following.

Suppose that we have observed the value of some random variable, Z, but not the value of another latent variable, Y. The joint probability for these two variables is parameterized using θ, as $P(y, z|\theta)$. We want to find the maximum likelihood estimate for the parameter θ of a model for Y and Z, but this problem cannot be easily solved directly. Alternatively, the corresponding problem in which Y is also known would be more tractable.

The basic idea of this problem is to maximize the log likelihood of the parameter θ, given the data z, by marginalizing over the latent variable Y:

$$\begin{aligned}\theta^* &= \arg\max_{\theta} \log \sum_{Y} P(y, z|\theta) \\ &= \arg\max_{\theta} \log P(z|\theta) \\ &= \arg\max_{\theta} L(\theta).\end{aligned} \tag{2.3}$$

This gives the intuition behind the EM algorithm: alternate between estimating the unknowns θ and estimating the hidden variable Y. The EM algorithm starts with some initial guess at the maximum likelihood parameters, $\theta^{(0)}$, and then proceeds to iteratively generate successive estimates, $\theta^{(1)}, \theta^{(2)}, \ldots$ by repeatedly applying the following two steps, for $t = 1, 2, \ldots$

– E Step: Compute a distribution $\tilde{P}^{(t)}$ over the range of Y such that

$\tilde{P}^{(t)}(y) = P(y|z, \theta^{(t-1)})$.

– M step: Set $\theta^{(t)}$ to the θ that maximizes $E_{\tilde{P}^{(t)}}[\log P(y, z|\theta)]$.

Here $E_{\tilde{P}}[\cdot\,]$ denotes the expectation with respect to the distribution $\tilde{P}$ over the range of Y [35]. As shown by Dempster, et al. [32], EM algorithm has the property of increasing the likelihood $L(\theta)$, or leave it unchanged, at each step. In fact, although an EM iteration does not decrease the observed data likelihood function, it is not guaranteed to converge to a maximum likelihood estimate. For most models, the algorithm may converge to a local maximum of $L(\theta)$. Actually, we may also view EM algorithm as two alternating maximization steps for the same function $F(\tilde{P}, \theta)$ which is defined as follows:

$$F(\tilde{P}, \theta) = E_{\tilde{P}}[\log P(y, z|\theta)] + H(\tilde{P}), \tag{2.4}$$

where $H(\tilde{P}) = -E_{\tilde{P}}[\log \tilde{P}(y)]$ is the entropy of the distribution $\tilde{P}$. Hence, an iteration of EM algorithm can be expressed in terms of the function F as follows:

– E Step: Set $\tilde{P}^{(t)}$ to the $\tilde{P}$ that maximizes $F(\tilde{P}, \theta^{(t-1)})$.

– M step: Set $\theta^{(t)}$ to the θ that maximizes $F(\tilde{P}^{(t)}, \theta)$.

Once the EM iterations have been expressed in this form, it is clear to tell that the algorithm will eventually converge to the value $\tilde{P}^{*}$ and θ^{*} that locally maximize $F(\tilde{P}, \theta)$ [35].

3 Proposed Method

In this chapter, we present an algorithm to infer the traffic state between non–overlapping FOVs by modeling the traffic flow with a transition–time distribution which may change dynamically. We first introduce the targets selected for the analysis of traffic status in our system. By utilizing the specific motion pattern of the targets of interest, we can reduce the complexity of traffic flow analysis and improve the system performance. Then we will describe the traffic flow model of our system in Section 3.2. In Section 3.3, the problem formulation and our algorithm are presented. In Figure 3.1, we show the flowchart of the proposed system.

We have conducted some experiments based on a real life traffic environment. Two cameras are installed to monitor two separate regions, which are linked by a road with some intersections in–between. More detailed description of the setting of our experimental environment and setting in the next chapter.

3.1 Selected Target

In order to monitor the traffic state between two non–overlapping FOVs linked by a road, we need to observe and analyze objects(cars) moving between the two FOVs. In general, however, since there may be intersections along the road, vehicles may leave or enter the road in–between. This circumstance will complicate the whole problem since we may have to identify the objects that leave halfway and ignore them in the traffic analysis. Accordingly, to alleviate the potential difficulty of analysis incurred by this case, we prefer to infer the traffic state between non–overlapping FOVs by choosing the vehicles are more likely to have

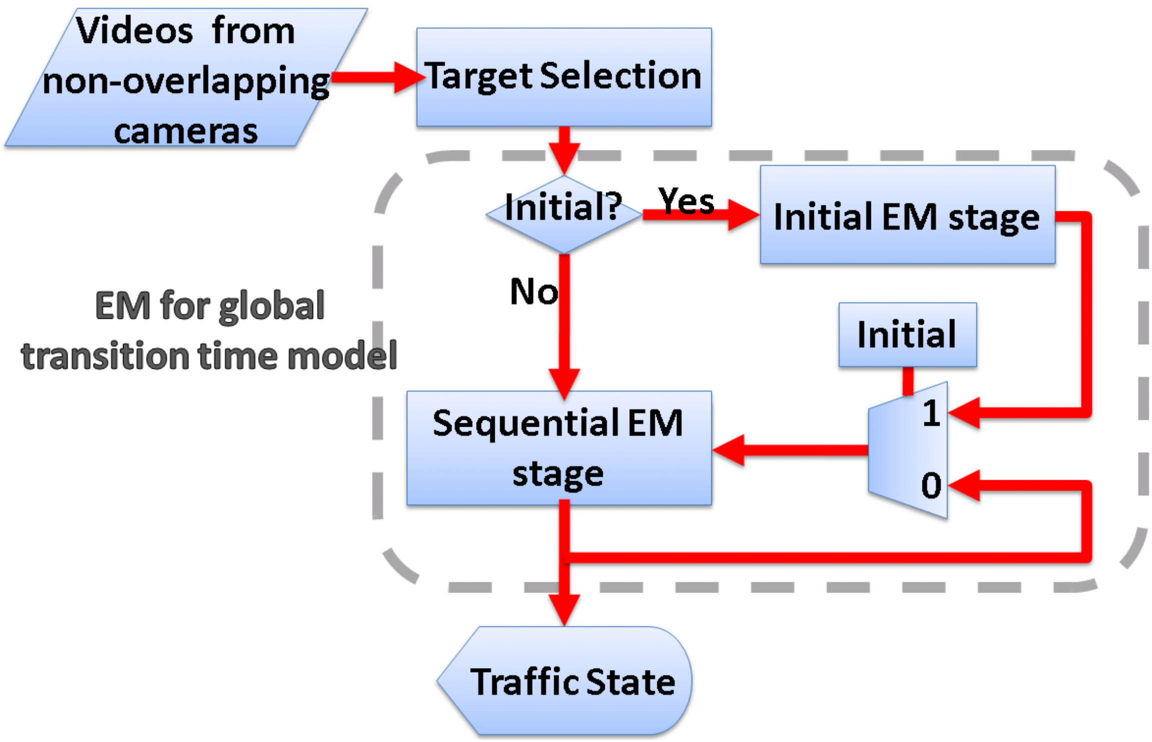


Figure 3.1: The block diagram of the proposed system.

fixed route (e.g., bus) across FOVs as our main targets. Although we still cannot guarantee that all the buses seen in one FOV will definitely show up in another, this target selection can indeed improve the accuracy of our analysis. By selecting buses in the traffic videos obtained from two FOVs, we record the entry/exit times of all the buses in the regions of interest. A result of the target selection is shown in Figure 3.2.

3.2 Traffic Flow Model

From a microscopic point of view, every vehicle has its own behavior pattern, which is affected by individual driver. However, if observed from a macroscopic point of view, one may find that most vehicles typically follow a global trend that is affected by the interactions among vehicles such as car–following and queuing. Hence, the main assumption of our proposed method is that the transition time of most vehicles across two non–overlapping FOVs will satisfy a global distribu-

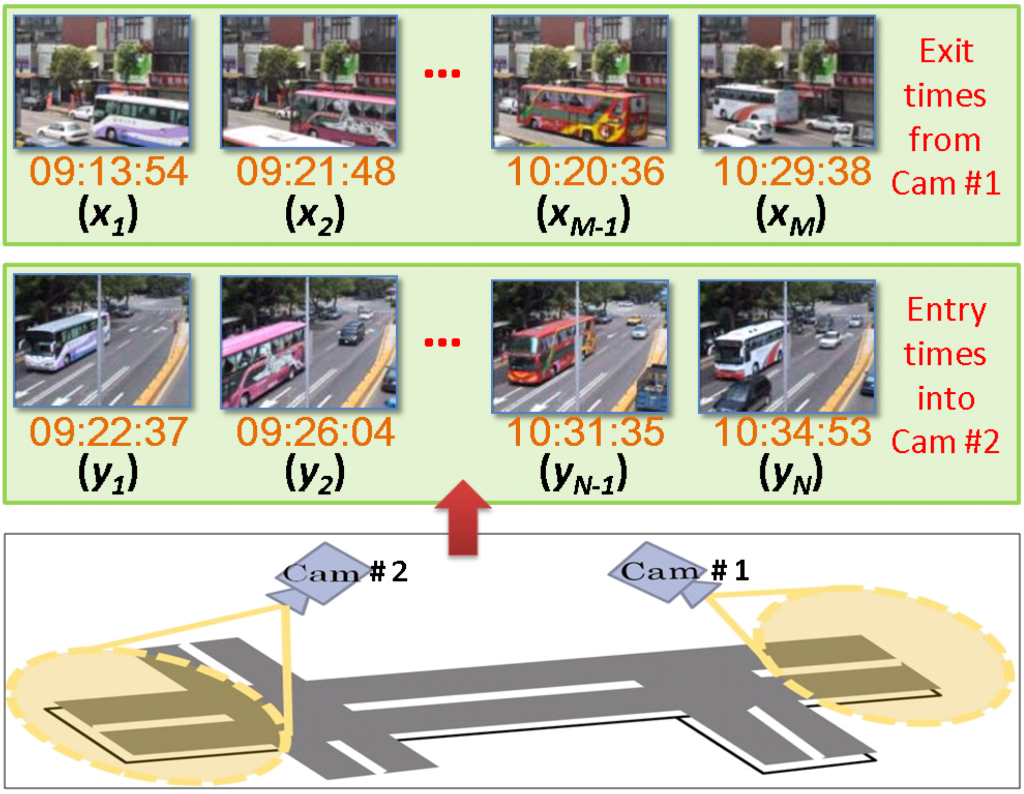


Figure 3.2: A result of target selection. We choose buses as our targets and record the exit time ($\{x_m\}_{m=1\sim M}$) and entry time ($\{y_n\}_{n=1\sim N}$) of them.

tion model. Here we use the Gaussian mixture model (GMM) to approximate this distribution because this model is general and can well approximate a lot of continuous functions under normal conditions. Some examples of distributions in the real life traffic are shown in Figure 3.3, in which we can see that the distributions can be well modeled by GMM. With the assumption of GMM model, our objective is to derive the parameters of this global transition time model to describe the traffic state. Moreover, since traffic flow state may dynamically change over time, we divide the timeline into many overlapped time–windows to analyze the transition time distributions at different times of the day, as shown in Figure 3.4.

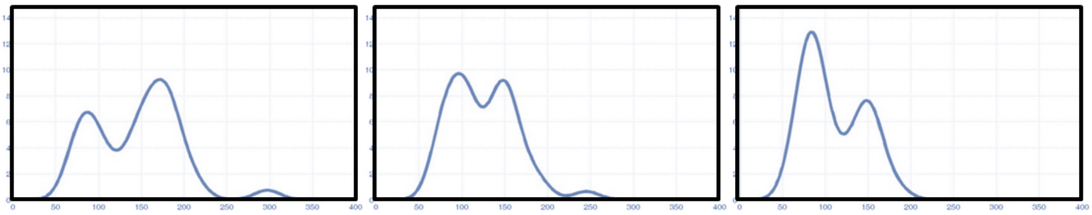


Figure 3.3: Some examples of distributions in the real life traffic.

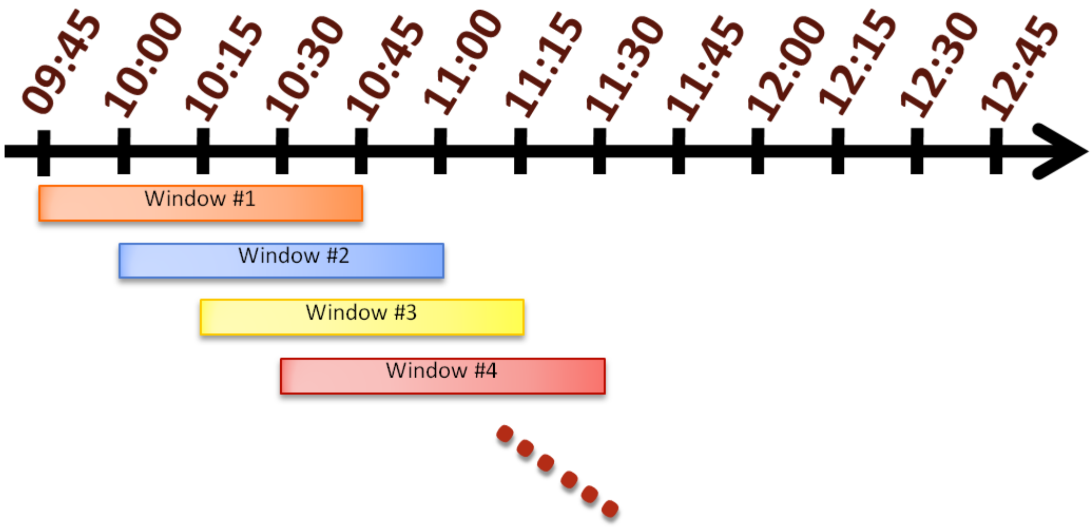


Figure 3.4: Division the timeline of the day into overlapped time–windows.

3.3 Problem Formulation

Before discussing how to determine the GMM parameters, we first consider two problem from two opposite viewing angles in the following paragraphs. First, suppose that we have the prior knowledge of the global transition–time distribution, we may infer the rough correspondence among buses. That is, we can approximately determine how the buses in two FOVs match one another. On the other hand, if we know how the buses in two FOVs match one another and thus the transition time of each correspondence, the global transition time distribution can be derived. Hence, in our system, we exploit this physical dependence between the correspondences of buses and the global transition–time distribution, and proposed a new solution to monitor the traffic between FOVs. Note that it is reasonable to assume that the traffic statuses on different sides of the road are independent of each other. Without loss of generality, we only discuss the traffic problems in a single side.

To determine the model parameters and to further infer the traffic state within a time window, we formulate the problem as an optimization process expressed as follows:

$$\Theta^* = \arg\max_{\Theta} p(\Theta|Z), \tag{3.1}$$

where $Z = (\{x_m\}_M, \{y_n\}_N)$ represents the combination of observations, including the exit time set $(x_1, x_2, \ldots, x_m, \ldots, x_M)$ and the entry time set $(y_1, y_2, \ldots, y_n, \ldots, y_N)$ of all the buses in two non–overlapping FOVs, as shown in Figure 3.2. However, it is difficult to build the probability model $P(\Theta|Z)$ in Equation 3.1, owing to the lack of physical connection between Θ and Z. To compensate the physical gap between parameters Θ and our observations Z, we introduce the correspondences between the exit time $\{x_m\}_M$ and entry time $\{y_n\}_N$ as an unknown random variable $C = \{c_m(x_m) = y_n\}_M$, where $c_m(.)$ indicates the entry time y_n that an exit time x_m will correspond to. Therefore, if a bus leaves one FOV at x_m, travels through the road, and then enters another FOV at y_n, we can express this correspondence by $\{x_m, c_m(x_m) = y_n\}$. The transition time of this correspondence is $t_m = c_m(x_m) - x_m$. Figure 3.5 shows the schematic diagram of x_m, y_n, $c_m(.)$, and t_m. By collecting all the transition times of each correspondence into a data set $T = \{t_m\}_M$, we may estimate the model parameters Θ.

The problem of the above formulation is that we do not have the measurement of C and we cannot derive the Θ directly. Hence, we treat the correspondence C as a latent variable and reformulate Equation 3.1 as Equatiion 3.2 by

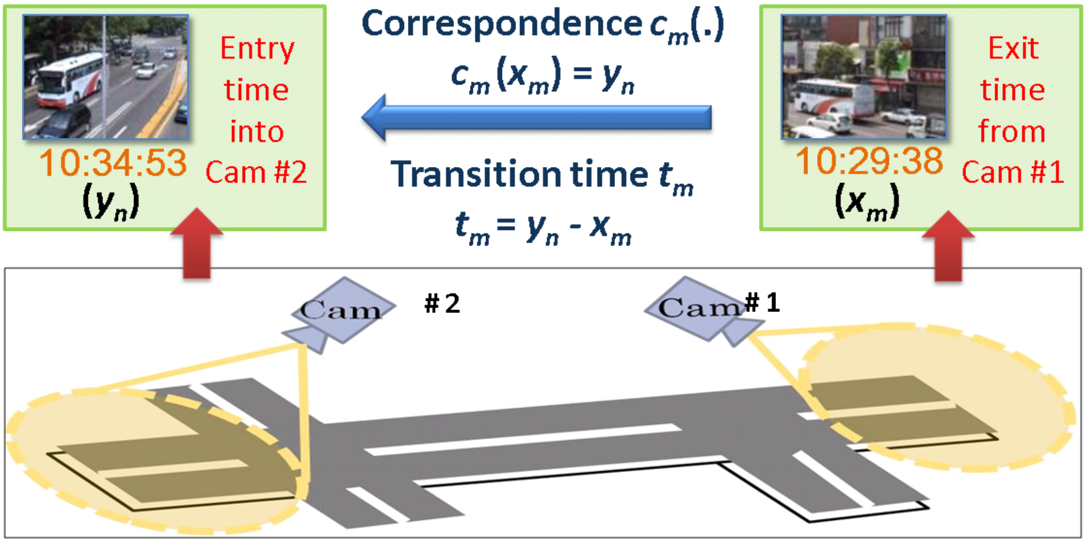


Figure 3.5: The schematic diagram of x_m, y_n, $c_m(.)$, and t_m.

marginalizing out every possible correspondence C.

$$\Theta^* = \arg\max_{\Theta} \sum_{C} p(\Theta, C|Z). \tag{3.2}$$

Although a direct approach to solving this optimization problem is generally intractable, the Expectation Maximization algorithm provides a mean to do the maximization by iteratively calculating the following two steps:

1. E Step: Calculate the expected log likelihood function $Q(\Theta)$:

$$Q(\Theta) = \sum_{C} log(p(Z, C|\Theta))p(C|Z, \Theta^{(old)}). \tag{3.3}$$

where the expectation is taken with respect to the posterior distribution $p(C|Z, \Theta^{(old)})$ over all possible correspondences C, given the data Z and the $\Theta^{(old)}$ at the previous time step.

2. M step: Find the Maximum Likelihood estimate $\Theta^{(new)}$ by maximizing $Q(\Theta)$:

$$\Theta^{(new)} = \arg\max_{\Theta} Q(\Theta). \tag{3.4}$$

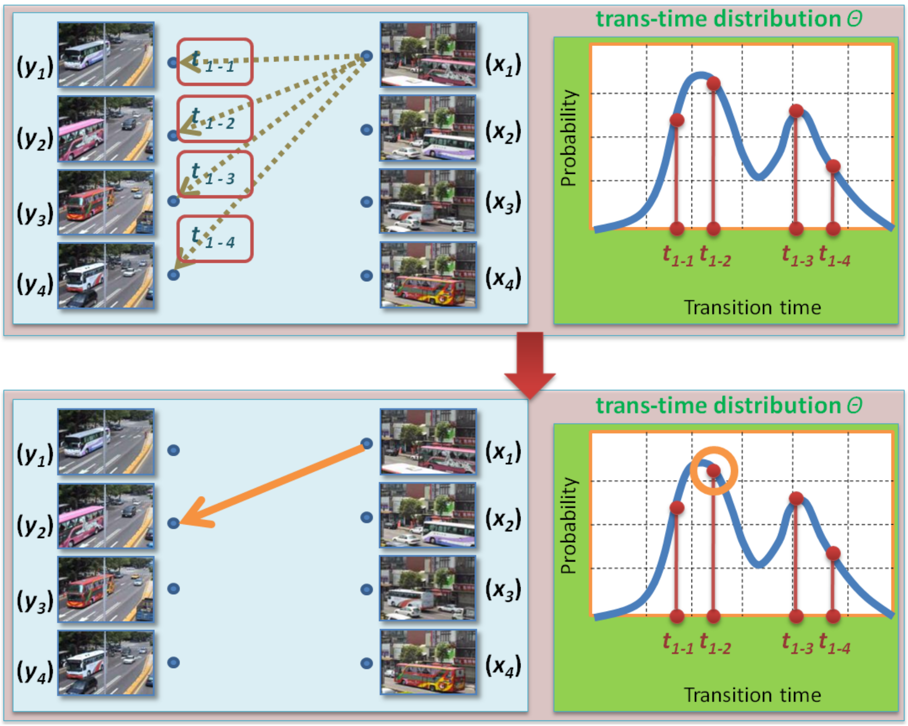
However in the E step, the expectation is difficult to compute since the number of possible correspondences explode combinatorially. A solution is to introduce

Algorithm 3 Gibbs sampler in proposed method.

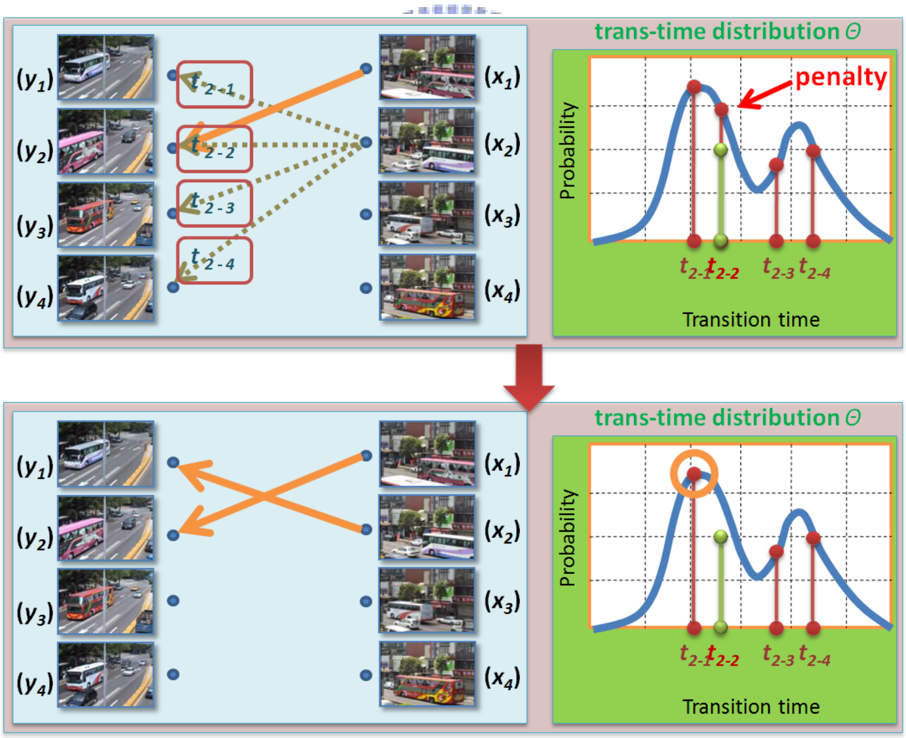
Initialize $c_{(1:m)}^{(0)}$
for $r = 0$ to $R-1$ **do**
  Sample $c_1^{(r+1)} \sim p(c_1|c_2^{(r)}, c_3^{(r)}, \ldots, c_m^{(r)}, \Theta)$.
  Sample $c_2^{(r+1)} \sim p(c_2|c_1^{(r+1)}, c_3^{(r)}, \ldots, c_m^{(r)}, \Theta)$.
  $\vdots$
  Sample $c_j^{(r+1)} \sim p(c_j|c_1^{(r+1)}, \ldots, c_{j-1}^{(r+1)}, c_{j+1}^{(r)}, \ldots, c_m^{(r)}, \Theta)$.
  $\vdots$
  Sample $c_m^{(r+1)} \sim p(c_n|c_1^{(r+1)}, \ldots, c_{m-1}^{(r+1)}, \Theta)$.
end for

Figure 3.6: Gibbs sampler in the proposed method.

Markov chain Monte Carlo (MCMC) to sample from the posterior probability distribution $p(C|Z, \Theta^{(old)})$ and replace the expectation in the E step over all possible correspondences with MCMC samples. More formally, this can be justified in the context of a Monte Carlo EM or MCEM. The sampling method we use here is the Gibbs sampling. In our setting, it is prefered that the entry/exit times are having one–to–one correspondences, but still allow the possibility that one entry time could be matched with more than one exit times. Therefore, $p(c_j|c_1^{(r+1)}, \ldots, c_{j-1}^{(r+1)}, c_{j+1}^{(r)}, \ldots, c_M^{(r)}, \Theta)$, the conditional probability of c_j given the other correspondences $\{c_m\}_{M \neq j}$ and global transition time distribution Θ, will be proportional to the likelihood of transition time $t_j = c_j(x_j) - x_j$ determined by Θ, while the conditional probability may decrease if the exit time $y_n = c_j(x_j)$ has been matched with others (see the algorithm in Figure 3.6). A simple example of the process in Gibbs sampling is shown in Figure 3.7. By performing sampling R times, it will generate a sequence of C. Since we expect that the correct C will have a much higher likelihood than others, we will choose the best sampling result to enter the M step in the EM algorithm.



(a)



(b)

Figure 3.7: A simple example of the Gibbs sampling. (a) In the beginning, for sampling c_1, there are four candidates: $c_1(x_1) = y_1$, $c_1(x_1) = y_2$, $c_1(x_1) = y_3$, $c_1(x_1) = y_4$ and therefore four possible transition times: t_{1-1}, t_{1-2}, t_{1-3}, t_{1-4}. For every transition time we can measure its likelihood to the distribution Θ. The Gibbs sampling will sample c_1 according to the probability proportional to these likelihoods. Here we assume the Gibbs sampling chooses $c_1(x_1) = y_2$. (b) Given the distribution Θ and the existing correspondence, we want to determine c_2. The same method is used to measure the conditional probability of every candidate of c_2. However, because $c_2(x_2) = y_2$ has been mapped by $c_1(x_1) = y2$, we will give $c_2(x_2) = y2$ a penalty to lower down its probability for the preference of 1–to–1 correspondence. Accordingly the Gibbs sampling chooses $c_2(x_2) = y_1$.

With the correspondences $C = \{c_m\}_M$ from the Gibbs sampling, we can calculate all the transition times $\{t_m = c_m(x_m) - x_m\}_M$ of buses traveling across two non–overlapping FOVs. In the M step, from $\{t_m\}_M$, we will derive the parameters Θ of the global transition time distribution. Assume we use a GMM with K Gaussians to approximate the global transition time distribution, then the distribution Θ will be parameterized by the weight w_j, the mean μ_j, and the variance σ_j^2 of the jth Gaussian, with $j = 1, \ldots, K$. To estimate these parameters given the transition times $\{t_m\}_M$, we can use a standard EM algorithm designed for GMM that is based on the following iteration formulae:

$$w_j^{(r+1)} = \frac{1}{n} \sum_{m=1}^{M} P(j|t_m) \tag{3.5}$$

$$\mu_j^{(r+1)} = \frac{\sum_{m=1}^{M} P(j|t_m) t_m}{\sum_{m=1}^{M} P(j|t_m)} \tag{3.6}$$

$$\sigma_j^{2(r+1)} = \frac{\sum_{m=1}^{M} P(j|t_m)(t_m - \mu_j^{(t+1)})^2}{\sum_{m=1}^{M} P(j|t_m)} \tag{3.7}$$

where

$$P(j|t_m) = \frac{w_j^{(r)} p(t_m|j; \mu_j^{(r)}, \sigma_j^{(r)})}{\sum_{k=1}^{K} w_k^{(r)} p(t_m|k; \mu_k^{(r)}, \sigma_k^{(r)})}. \tag{3.8}$$

A demonstration of the M step is shown in Figure 3.8.

The global transition–time distribution parameterized by $\{w_j, \mu_j, \sigma_j^2\}_K$ discovered in the M step will be propagated into the E step of the next EM iteration. After a few iterations, the EM algorithm will converge to a maxima of $p(\Theta|Z)$. The proposed EM algorithm for traffic state monitoring within a time–window is shown in Figure 3.9.

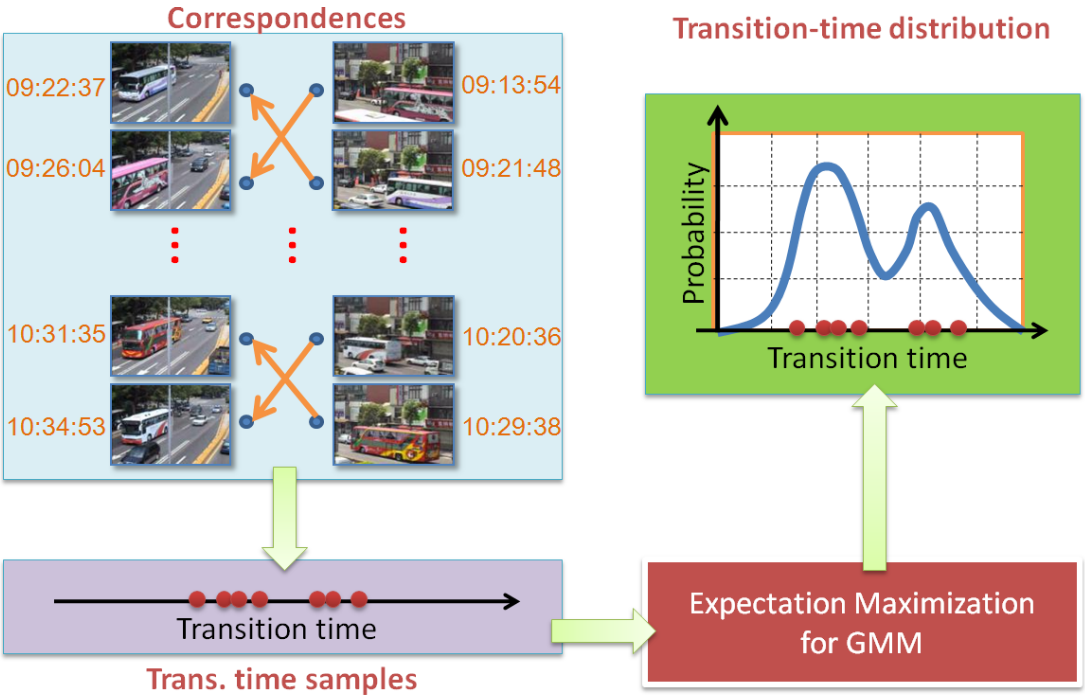


Figure 3.8: A demonstration of the M step. With the transition times calculated from the correspondence from the Gibbs sampling, we use the EM algorithm to estimate the parameters of the GMM in order to model the transition time distribution.

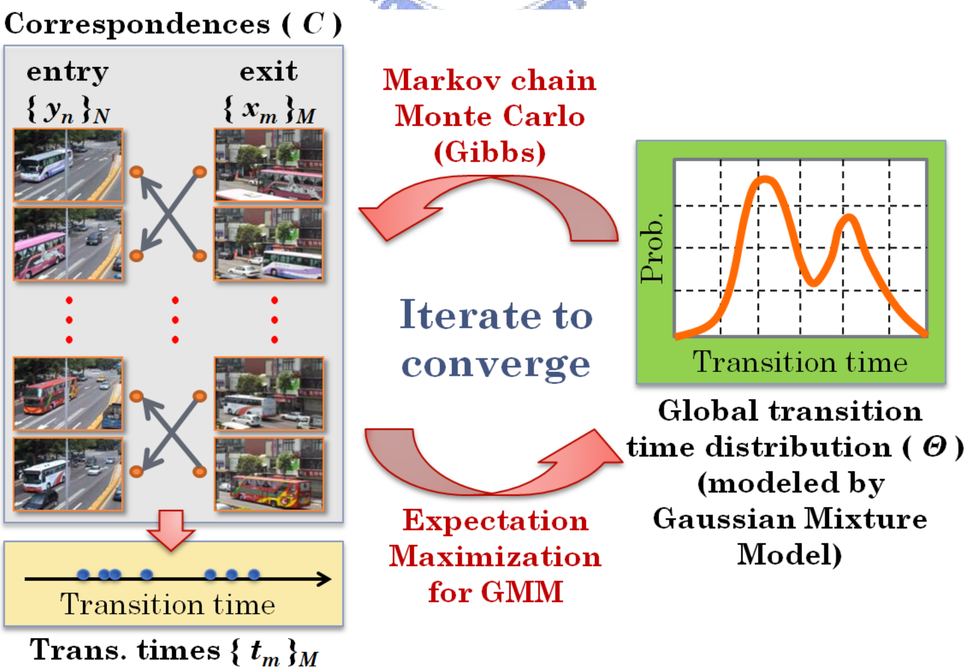


Figure 3.9: Proposed Expectation–Maximization algorithm for simultaneously estimating object correspondence and the parameters of the transition time distribution.

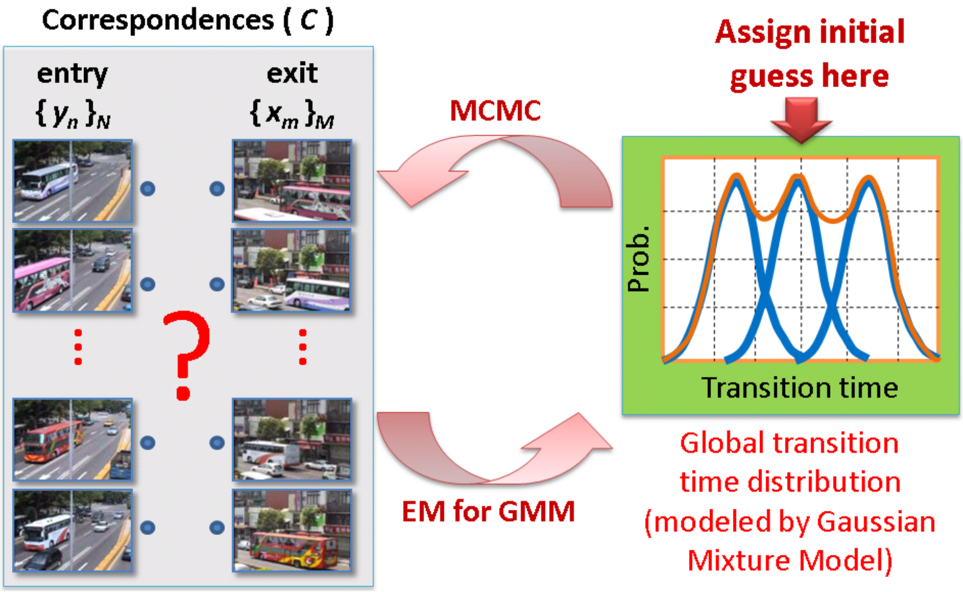


Figure 3.10: Initialization of the EM algorithm for the first time–window. We start the EM iteration by initializing the global transition–time distribution Θ with the Gaussian mixture model of K equal weighting, broadly separated, and wide–bandwidth Gaussians. We assume this initialization of GMM can cover most regions of the transition time.

3.4 Initialization

In this paper, we divide the timeline into several overlapped time–windows to handle the dynamic changes of the traffic flow, as shown in Figure 3.4, and apply the proposed EM algorithm to every time–window. An important step for performing the proposed EM algorithm in each time–window is the initial setting. For the first time–window, we start the EM iteration by initializing the global transition–time distribution Θ with the Gaussian mixture model of K equal weighting, broadly separated, and wide–bandwidth Gaussians, as shown in Figure 3.10. We assume this initialization of GMM can cover most regions of the transition time.

On the other hand, for the initial setting of the subsequent time–windows, since they are partially overlapped with the previous ones, we would like to propagate some information from the previous window into the current window as the initial condition. Hence, we have two choices for the initialization – propagating either the parameters of the Gaussian mixture model or the correspondences from

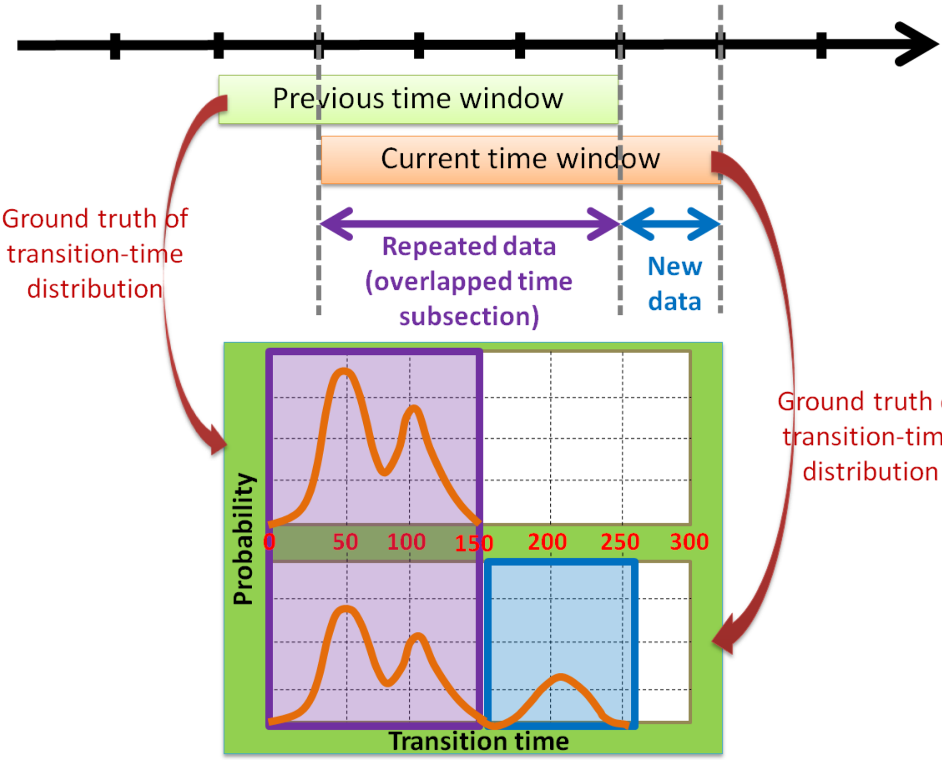


Figure 3.11: A possible situation for the current time–window and the previous one — the transition–time distribution of the current time–window is more widespread then it of the previous window. The increased range of the transition–time is caused from the data in the non–overlapped subsection which is new to the last time window (noted with the blue bounding–box), and the GMM of the previous time–window only has the information of the overlapped subsection in current time–window (noted with the purple bounding–box).

the previous window. Consider a possible situation wherein the transition–time distribution of the current time–window is more widespread than that of the previous one, as shown in Figure 3.11. The increased range of the transition time due to the new data (marked with the blue bounding–box) is not covered by the GMM of the previous time–window (marked with the purple bounding–box). As a result, if we start the EM iterations by initializing the parameters of GMM which are propagated from the previous time window, the algorithm may only cover the partial range of the transition time and lead to incorrect result.

Therefore, for the EM initialization of the subsequent time–windows, in addition to propagate the correspondences of the overlapped subsections from the previous window into the current window and use them as the initial conditions, for the data in the non–overlapped subsection which is new to the last time win-

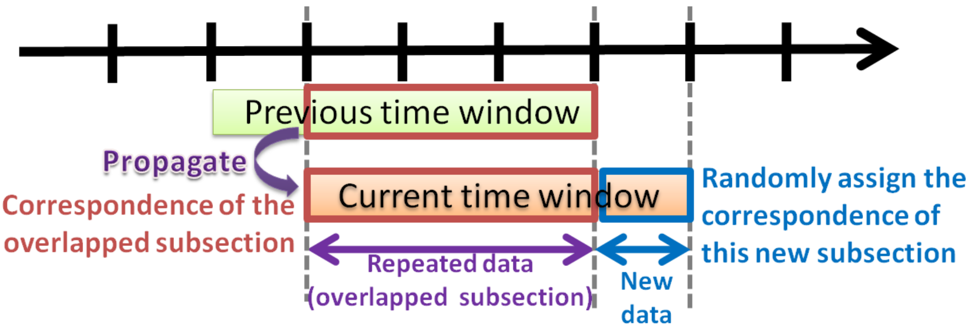


Figure 3.12: Initialization of the EM for the following time–windows. We propagate the correspondences of the overlapped subsection from the previous window into the current window, and randomly assign the correspondences of the new data in the non–overlapped subsection.

dow, we randomly assign the correspondences within this period. That is, we start the EM iterations by initializing the correspondences C which consist of both the propagated and the randomly assigned correspondences, as shown in Figure 3.12. This method for initialization helps the algorithm to maintain some information from the previous time–window, while still have the ability to extract useful information from the new data.

4 Experiment and Results

Out experimental environment is shown in Figure 4.1. We mounted two cameras at the Mackay Memorial Hospital and the overpass in front of National Tsing Hua University. In Figure 4.1, we can see that FOVs of these two cameras are non-overlapped and these two scenes are linked by Kuang Fu Road (indicated by the red line). Besides, there are three intersections and some bus stops along the road between these two FOVs. This makes the traffic situations more complicated. The traffic videos were taken from about 9:45 in the morning till 19:00 in the afternoon. We divide the time–line into 34 overlapped time–windows. The length of each time–window is 60 minutes, with 45 minutes overlapped with the previous window, as shown in Figure 4.2

We first check the traffic videos and manually select frames that contain buses. All the entry/exit times of buses are recorded. Then we apply the proposed EM algorithm for each time–window. Here we use a Gaussian mixture model with 3 Gaussians to model the transition–time distribution. For the first time–window, the parameters of the GMM are initialized with $\mu = \{50, 150, 250\}$, $\sigma^2 = \{200, 200, 200\}$ and $w = \{0.33, 0.33, 0.33\}$. Then the initialization for EM in all the other time–windows follows the method we described in Section 3.4. Figure 4.3, Figure 4.4, and Figure 4.5 show the computed transition–time distributions and the ground–truths in different time–windows. We can see that the distribution calculated by our algorithm is reasonably similar to the ground truth.

Here we also give some results by applying the method in [25] to our experimental data, as shown in Figure 4.6. We can see that the results are not as accurate as ours. The reason is that in our method we use a stronger assumption about the

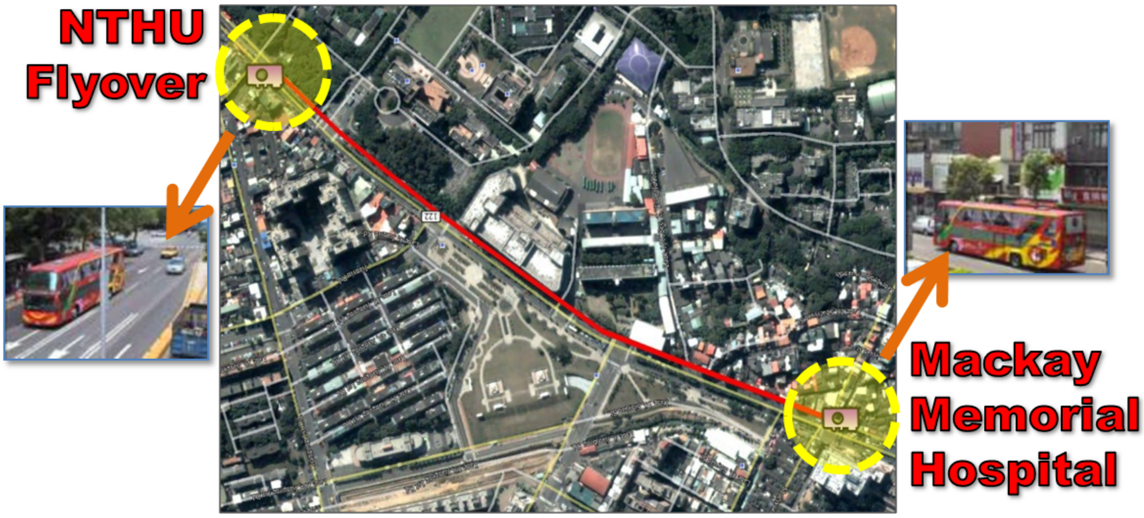


Figure 4.1: The experimental environment: monitor the traffic between two non–overlapping FOVs at Mackay Memorial Hospital and the NTHU overpass.

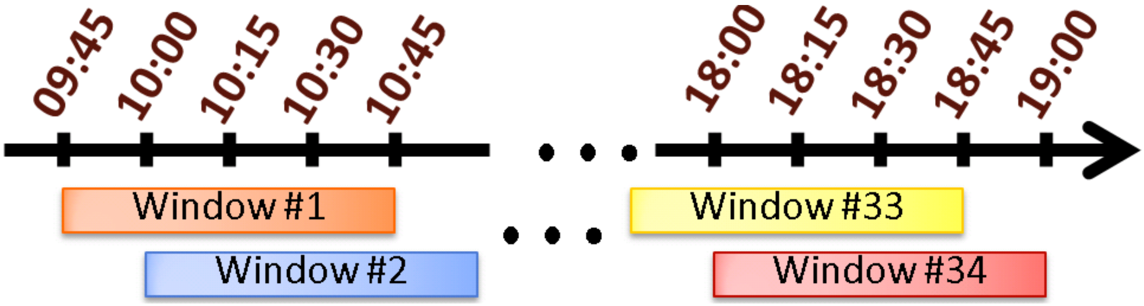


Figure 4.2: We divide the timeline into 34 overlapped time–windows. The length of each time–window is 60 minutes, with 45 minutes overlapped with the previous window.

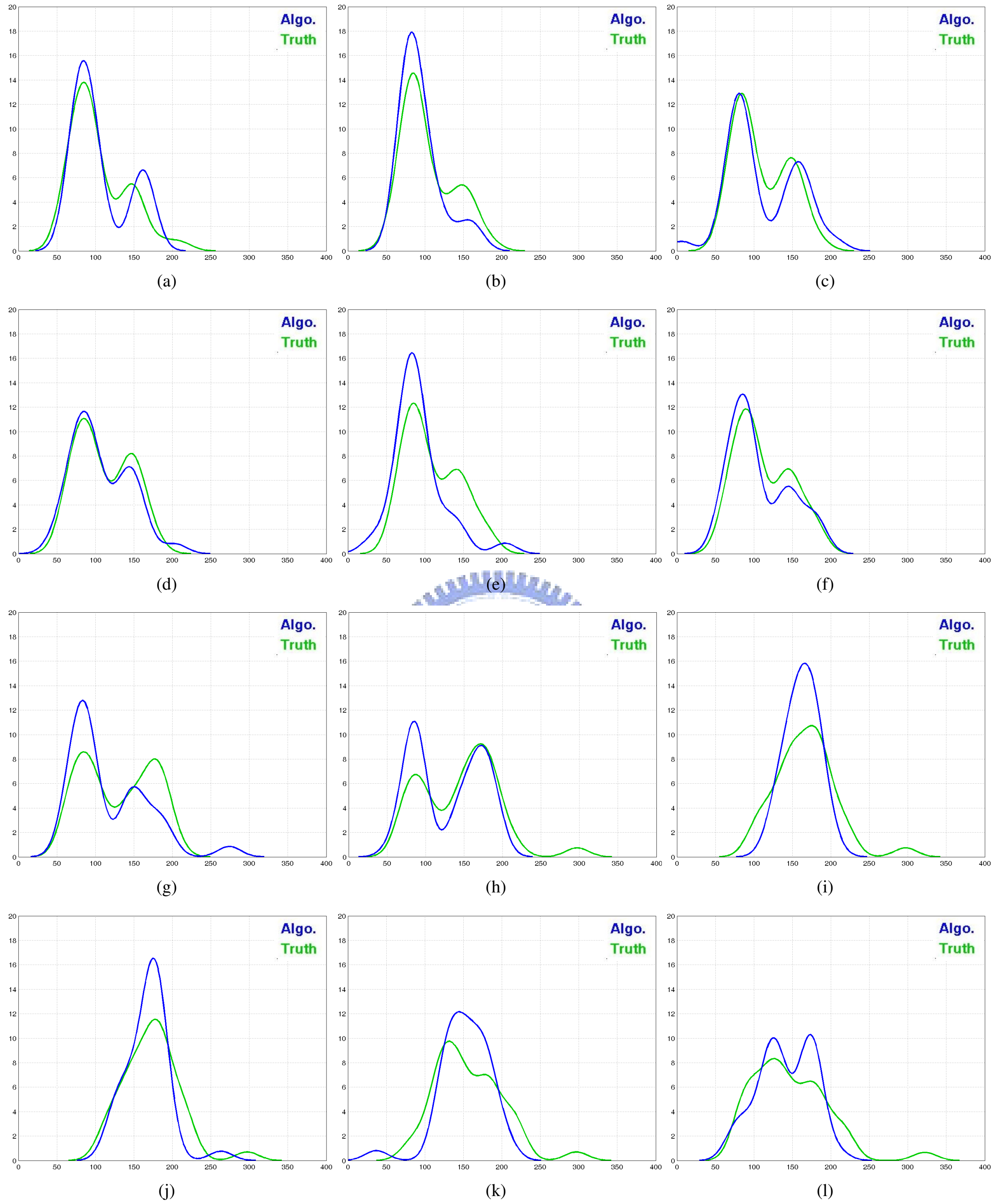


Figure 4.3: The transition–time distribution from time–window # 01 to time–window # 12.
(a) Time–window $09:45 \sim 10:45$. (b) Time–window $10:00 \sim 11:00$. (c) Time–window $10:15 \sim 11:15$. (d) Time–window $10:30 \sim 11:30$. (e) Time–window $10:45 \sim 11:45$. (f) Time–window $11:00 \sim 12:00$. (g) Time–window $11:15 \sim 12:15$. (h) Time–window $11:30 \sim 12:30$. (i) Time–window $11:45 \sim 12:45$. (j) Time–window $12:00 \sim 13:00$. (k) Time–window $12:15 \sim 13:15$. (l) Time–window $12:30 \sim 13:30$. (x–axis: transition time; y–axis: milli–probability).

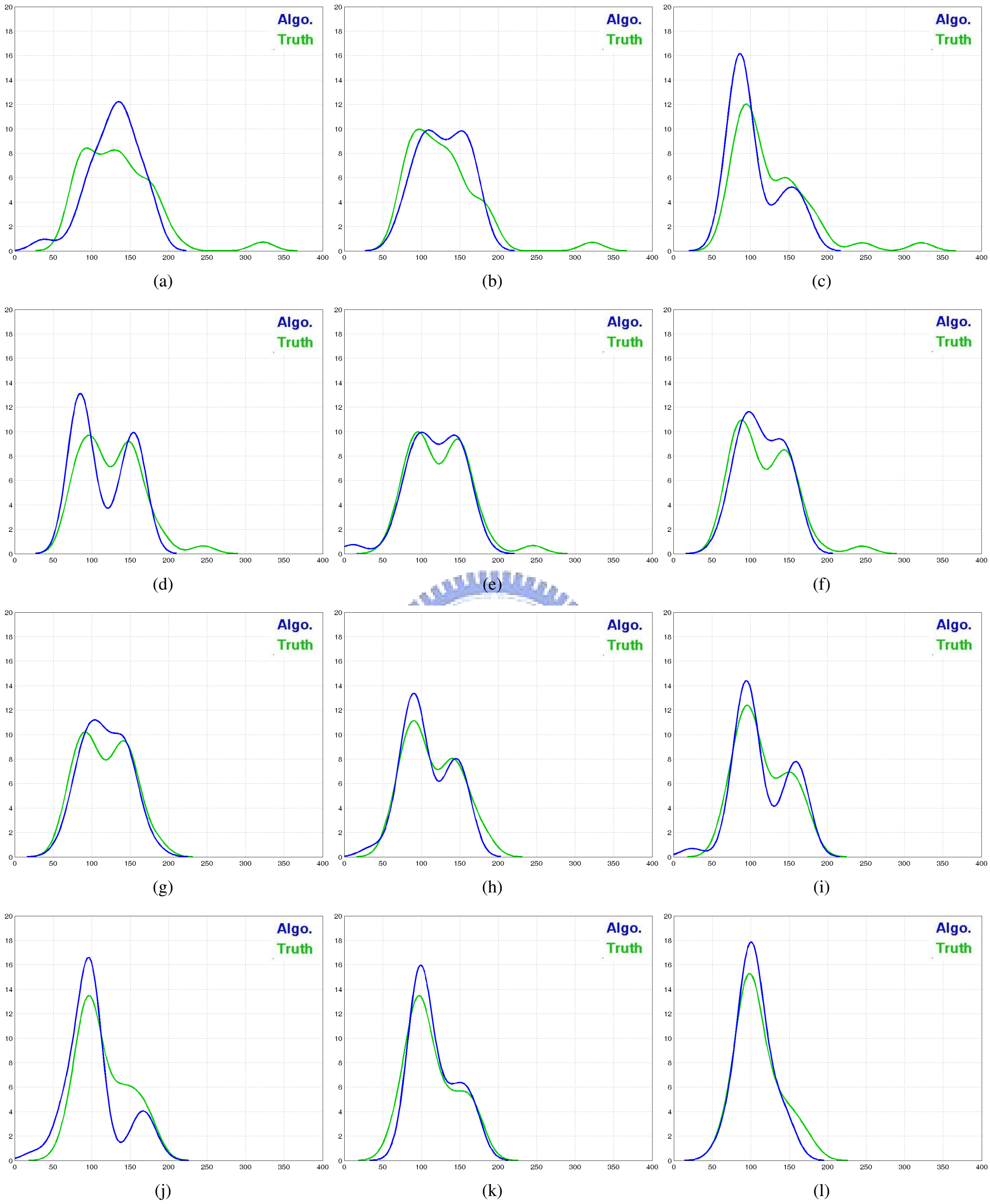


Figure 4.4: The transition–time distribution from time–window # 13 to time–window # 24.
(a) Time–window $12:45 \sim 13:45$. (b) Time–window $13:00 \sim 14:00$. (c) Time–window $13:15 \sim 14:15$. (d) Time–window $13:30 \sim 14:30$. (e) Time–window $13:45 \sim 14:45$. (f) Time–window $14:00 \sim 15:00$. (g) Time–window $14:15 \sim 15:15$. (h) Time–window $14:30 \sim 15:30$. (i) Time–window $14:45 \sim 15:45$. (j) Time–window $15:00 \sim 16:00$. (k) Time–window $15:15 \sim 16:15$. (l) Time–window $15:30 \sim 16:30$. (x–axis: transition time; y–axis: milli–probability).

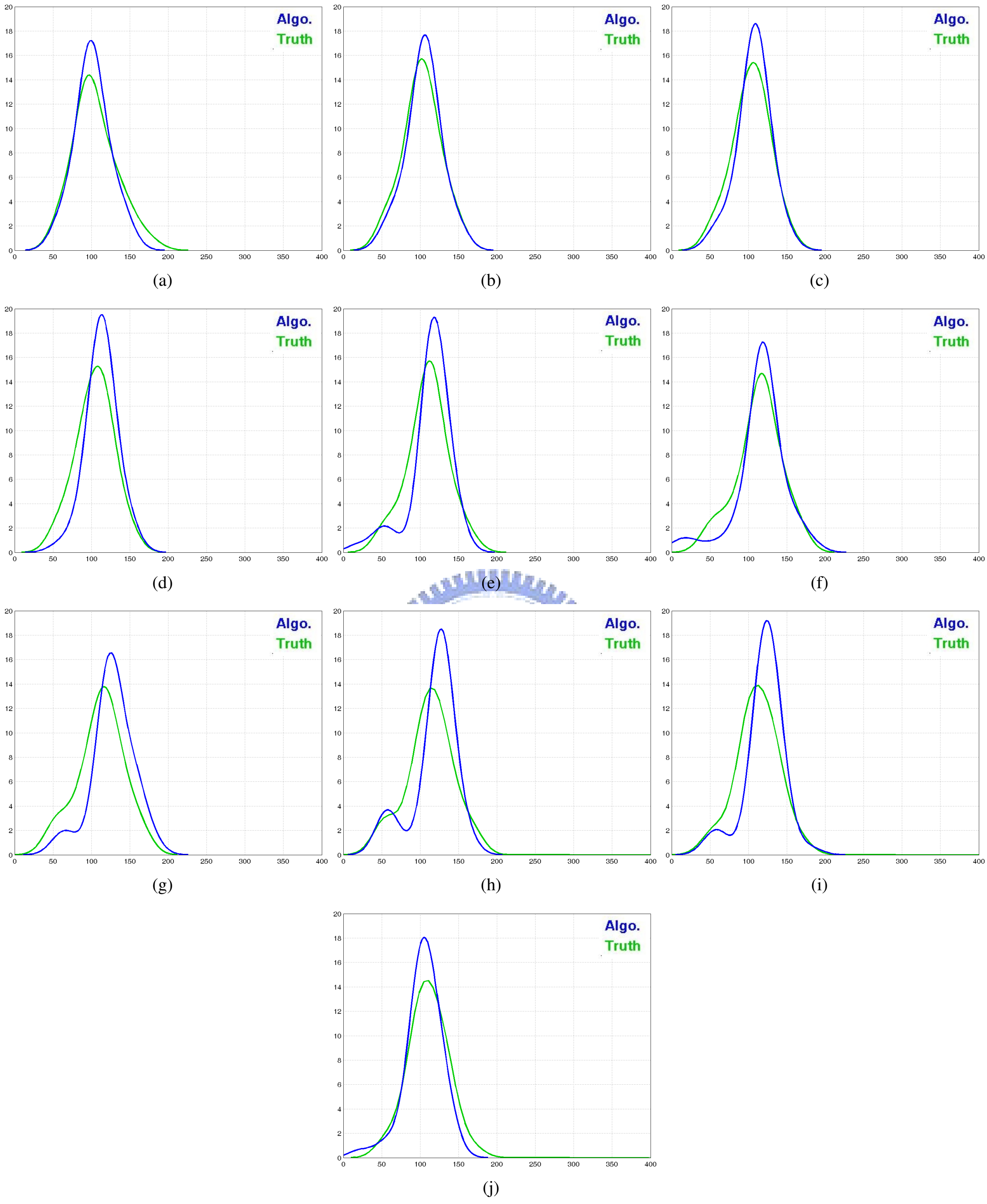
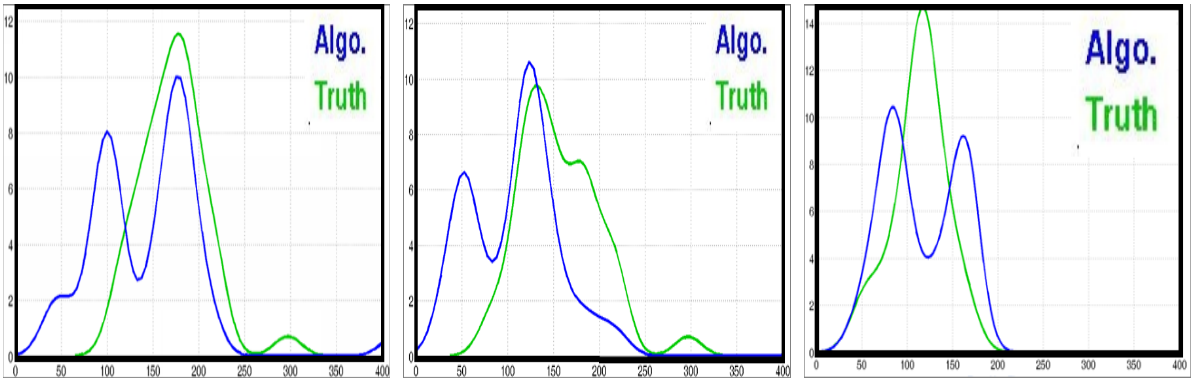


Figure 4.5: The transition–time distribution from time–window # 25 to time–window # 34.
(a) Time–window $15:45 \sim 16:45$. (b) Time–window $16:00 \sim 17:00$. (c) Time–window $16:15 \sim 17:15$. (d) Time–window $16:30 \sim 17:30$. (e) Time–window $16:45 \sim 17:45$. (f) Time–window $17:00 \sim 18:00$. (g) Time–window $17:15 \sim 18:15$. (h) Time–window $17:30 \sim 18:30$. (i) Time–window $17:45 \sim 18:45$. (j) Time–window $18:00 \sim 19:00$.
(x–axis: transition time; y–axis: milli–probability).



(a) (b) (c)

Figure 4.6: Some results from applying the method in [25] to our experimental data. (a) Time–window $12:00 \sim 13:00$. (b) Time–window $12:15 \sim 13:15$. (c) Time–window $17:00 \sim 18:00$.

characteristics of the real life traffic while the method in [25] only depends on the weaker minimum entropy assumption about the transition time distribution. For example, in Figure 4.6(c), the distribution of the ground truth has a wider range and thus has a larger entropy value than that of the distribution found by the algorithm in [25].

To infer the traffic flow state, we can first use the average transition time of the last 15–minute observation within each time–window as a statistic to represent the state. As shown in Figure 4.7, the chart of the traffic–flow state may give us the direct message about how the traffic dynamically changes over time. For instance, we may probably guess that the increase of the average transition time at about 12:00 is due to the lunch–time traffic, which is always a rush hour. Moreover, we could also express the traffic flow state by classifying the traffic changes as "stable", "increasing", and "decreasing," as shown in Figure 4.8. This figure indicates how the traffic changes with respect to the previous stage. This kind of description is much more user–friendly.

In addition to the aforementioned setting of the time–line division and the initialization method, here we provide some examples of different settings to support that our setting is reasonable. In the first example of different settings, we divide the time–line into 60–minute time–windows with 30 minutes overlapped with the previous window. For the EM initialization of the following time–windows, we propagate the parameters of GMM from the previous window. Figure 4.9 shows the transition–time distributions of some time–windows. We can see that the computed transition–time distributions cannot adapt quickly enough to the changes of the ground truth. One reason for this phenomenon is due to the problem of initialization we presented in Figure 3.11. Moreover, since the information propagated from the previous window only dominates 50% of the data in the processing window, the degree of change for the transition–time distribution might be too big to be tracked by the algorithm. Then, in the second example of different setting, we use the same time–windows as in the first example; and for the EM initialization of the following time–windows, we follows the method described in Section 3.4. Figure 4.10 shows the transition–time distributions of the time–windows which are the same as those in Figure 4.9. We can see that even there are some improvement in Figures 4.9(b), 4.9(d), and 4.9(e), the problem of miss–tracking still exists, as shown in Figure 4.9(c). Hence, in our experiment, we choose to divide the timeline into time–windows with much more overlaps with the previous time-window.

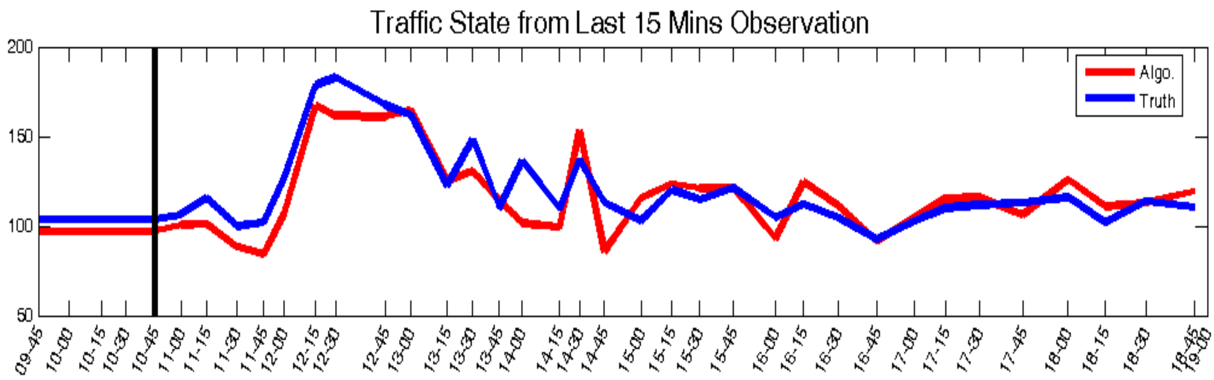


Figure 4.7: Traffic flow state expressed by the average transition time of the last 15–minute observation within each time–window. The red line is computed by our proposed method; The blue line is from the ground truth.

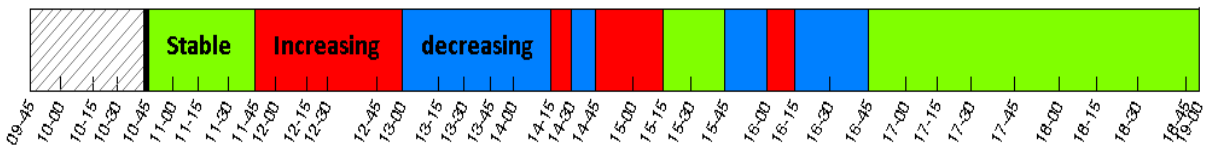


Figure 4.8: Express the traffic flow state by classifying the traffic changes as "stable", "increasing", and "decreasing." (Green color: stable traffic; Red color: increasing traffic flow; Blue color: decreasing traffic flow).

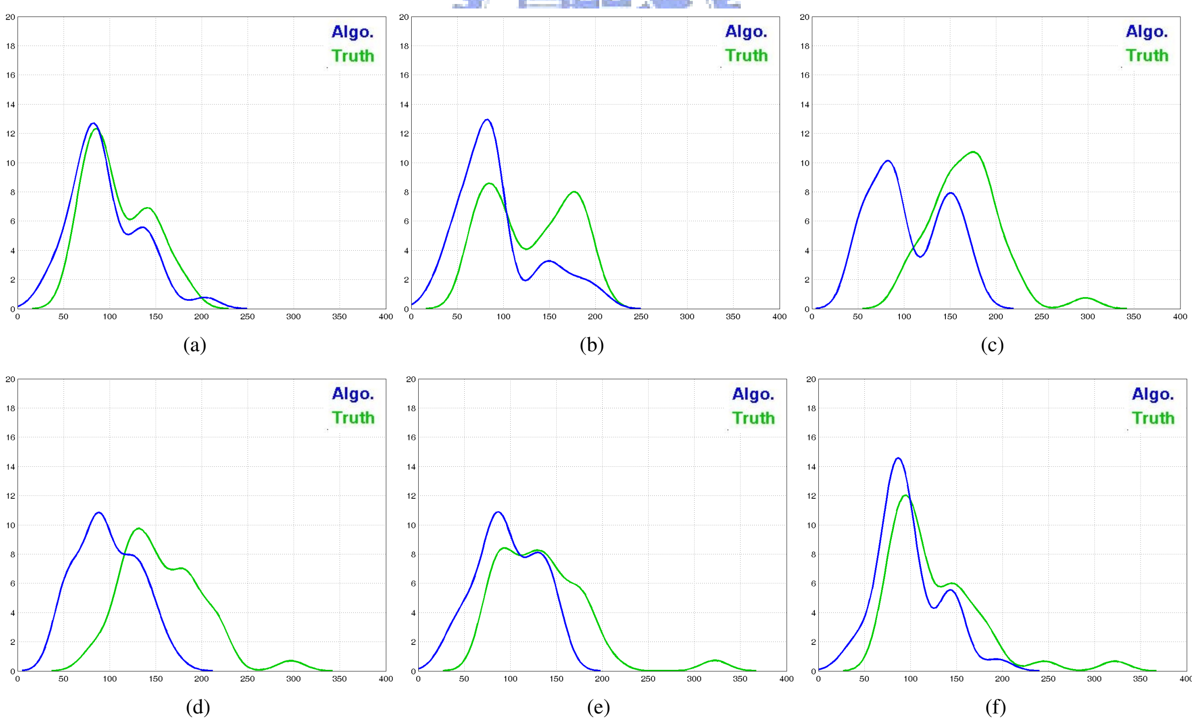


Figure 4.9: Some transition–time distributions from the first example of different setting. Each time–windows is 60 minutes long including 30 minutes overlapped with the previous window. The EM initialization of the following time–windows is with the parameters of GMM propagated from the previous window.
(a) Time–window $10:45 \sim 11:45$. (b) Time–window $11:15 \sim 12:15$. (c) Time–window $11:45 \sim 12:45$. (d) Time–window $12:15 \sim 13:15$. (e) Time–window $12:45 \sim 13:45$. (f) Time–window $13:15 \sim 14:15$. (x–axis: transition time; y–axis: milli–probability).

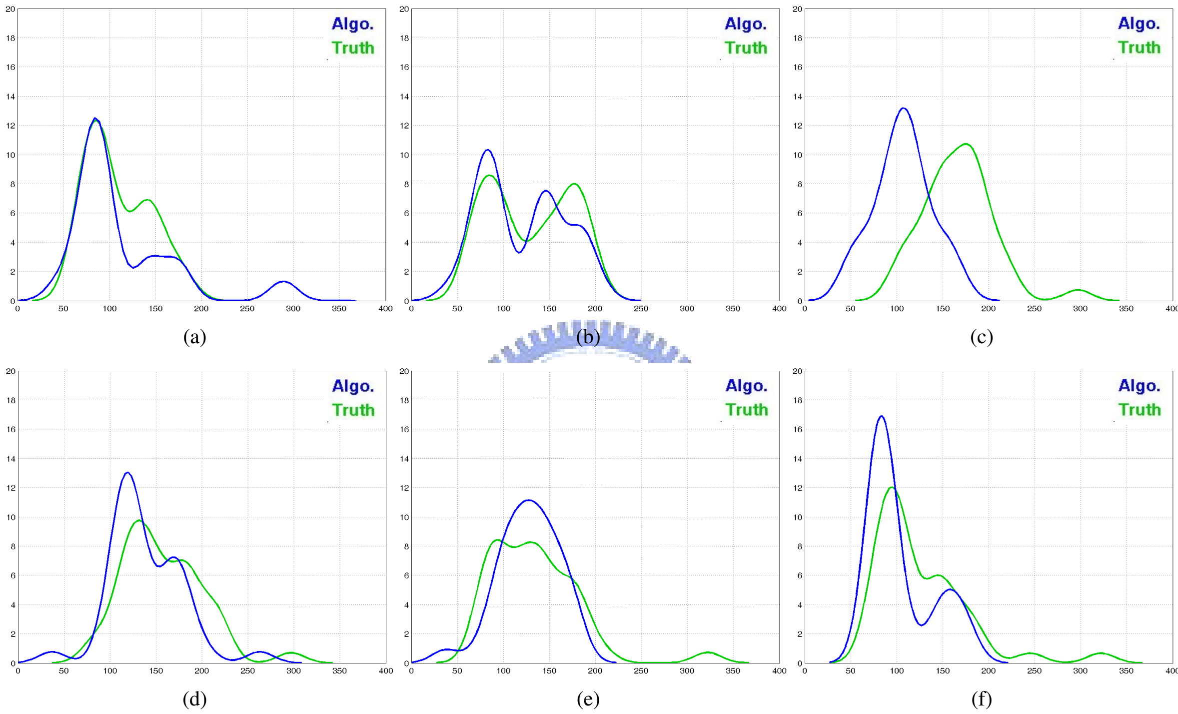


Figure 4.10: Some transition–time distributions from the second example of different setting. Each time–windows is 60 minutes long including 30 minutes overlapped with the previous window. The EM initialization of the following time–windows follows the method described in Section 3.4.
(a) Time–window $10:45 \sim 11:45$. (b) Time–window $11:15 \sim 12:15$. (c) Time–window $11:45 \sim 12:45$.
(d) Time–window $12:15 \sim 13:15$. (e) Time–window $12:45 \sim 13:45$. (f) Time–window $13:15 \sim 14:15$.
(x–axis: transition time; y–axis: milli–probability).

5 Conclusion

We propose an efficient method to probabilistically model the dynamic traffic flow between non–overlapping FOVs. Unlike previous works, our approach does not attempt to directly build the object correspondence across non–overlapping cameras. Instead, we model object correspondence and the parameters estimation of the transition time model as a unified problem. By building the physical connection between the transition time model and the object correspondence, the proposed EM–based framework can iteratively determine the optimal object correspondence and the model parameters. In addition, by dividing the time–line into many overlapped time–windows, our method can sequentially infer the time–varying traffic flow and recognize the dynamic changes of the traffic status over time. Moreover, our system is efficient and may provide a new thinking to well utilize the existing surveillance cameras for wide–area traffic monitoring. The experiments have shown that our approach performs well in a complicated traffic environment in real life.

Bibliography

[1] F.A. Haight. Mathematical theories of traffic flow. *Academic Press*, 1963.

[2] D.L. Gerlough and M.J. Huber. Traffic flow theory. *Transportation Research Board*, pages 199–201, 1975.

[3] A.D. May. Traffic flow fundamentals. *Prentice Hall*, 1990.

[4] N.H. Gartner, C.J. Messer, and A. Rathi. Special Report 165: Revised Monograph on Traffic Flow Theory. *Transportation Research Board, Washington, DC*, 1997.

[5] F. Viti. The dynamics and the uncertainty of delays at signals. *PhD thesis, Delft University of Technology*, 2006.

[6] H.J. van Zuylen and F. Viti. Delay at Controlled Intersections: The Old Theory Revised. In *IEEE Intelligent Transportation Systems Conference*, pages 68–73, 2006.

[7] D.J. Dailey, S.D. Maclean, F.W. Cathey, and Z.R. Wall. Transit vehicle arrival prediction: Algorithm and large–scale implementation. *Transportation Research Record: Journal of the Transportation Research Board*, 1771(-1):46–51, 2001.

[8] D.J. Dailey, Z.R. Wall, S.D. Maclean, and F.W. Cathey. An algorithm and implementation to predict the arrival of transit vehicles. *IEEE Intelligent Transportation Systems*, pages 161–166, 2000.

[9] F.W. Cathey and D.J. Dailey. A prescription for transit arrival/departure prediction using automatic vehicle location data. *Transportation Research Part C*, 11(3-4):241–264, 2003.

[10] Z.R. Wall and D.J. Dailey. An algorithm for predicting the arrival time of mass transit vehicles using automatic vehicle location data. *University of Washington*, 1998.

[11] I. Steven, Y. Ding, and C. Wei. Dynamic bus arrival time prediction with artificial neural networks. *Journal of Transportation Engineering*, 128:429, 2002.

[12] Y. Bin, Y. Zhongzhen, and Y. Baozhen. Bus arrival time prediction using support vector machines. *Journal of Intelligent Transportation Systems*, 10(4):151–158, 2006.

[13] B. Fishbain, I. Ideses, D. Mahalel, and L. Yaroslavsky. Real–time vision–based traffic flow measurements and incident detection. In *Proceedings of SPIE*, volume 7244, page 72440I, 2009.

[14] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359–381, 2003.

[15] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real–time computer vision system for measuring traffic parameters. In *IEEE Computer Society*

Conference on Computer Vision and Pattern Recognition, pages 495–501, 1997.

[16] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real–time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C*, 6(4):271–288, 1998.

[17] Z. Kim and J. Malik. Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *Proc. 9th IEEE International Conference on Computer Vision, Nice, France*, volume 1, pages 524–531, 2003.

[18] O. Javed, K. Shafique, Z. Rasheed, and M. Shah. Modeling inter–camera space–time and appearance relationships for tracking across non–overlapping views. *Computer Vision and Image Understanding*, 109(2):146–162, 2008.

[19] O. Javed, Z. Rasheed, K. Shafique, and M. Shah. Tracking across multiple cameras with disjoint views. In *Ninth IEEE International Conference on Computer Vision*, pages 952–957, 2003.

[20] B. Song and A.K. Roy-Chowdhury. Stochastic adaptive tracking in a camera network. In *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[21] A. Rahimi, B. Dunagan, and T. Darrell. Simultaneous calibration and tracking with a network of non–overlapping sensors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–187–I–194, 2007.

[22] Y. Sheikh, X. Li, and M. Shah. Trajectory association across non–overlapping moving cameras in planar scenes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2007.

[23] D. Makris, T. Ellis, and J. Black. Bridging the gaps between cameras. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, 2004.

[24] C. Stauffer. Learning to track objects through unobserved regions. In *IEEE Computer Society Workshop on Motion and Video Computing*, volume 2, 2005.

[25] K. Tieu, G. Dalley, and W.E.L. Grimson. Inference of non–overlapping camera network topology by measuring statistical dependence. In *Tenth IEEE International Conference on Computer Vision*, volume 2, 2005.

[26] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. Markov chain Monte Carlo in practice. *Chapman & Hall/CRC*, 1996.

[27] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[28] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computational machine. *Journal of Chemical Physics*, 21(6):1087–1091, 1953.

[29] C. Andrieu, N. De Freitas, A. Doucet, and M.I. Jordan. An introduction to MCMC for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

[30] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1984.

[31] H.O. Hartley. Maximum likelihood estimation from incomplete data. *Biometrics*, pages 174–194, 1958.

[32] A.P. Dempster, N.M. Laird, D.B. Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[33] G.J. McLachlan, T. Krishnan, and W. InterScience. The EM algorithm and extensions. *Wiley New York*, 1997.

[34] F. Dellaert. The expectation maximization algorithm. *Georgia Institute of Technology, Technical Report Number GIT-GVU-02-20*, 2002.

[35] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368, 1998.