Motion Pattern Analysis for Far-field Vehicle Surveillance

by

Chaowei Niu

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2006

Author .....
Department of Electrical Engineering and Computer Science
January 20, 2006

Certified by .....
W. Eric L. Grimson
Bernard Gordon Professor of Medical Engineering
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Motion Pattern Analysis for Far-field Vehicle Surveillance

by

Chaowei Niu

Submitted to the Department of Electrical Engineering and Computer Science
on January 20, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

The main goal of this thesis is to analyze the motion patterns in far-field vehicle tracking data collected by multiple, stationary non-overlapping cameras. The specific focus is to fully recover the camera's network topology, which means the graph structure relating cameras and typical transitions time between cameras, then based on the recovered topology, to learn the traffic patterns(i.e. source/sink, transition probability, etc.), and finally be able to detect unusual events. I will present a weighted statistical method to learn the environment's topology. First, an appearance model is constructed by the combination of normalized color and overall model size to measure the appearance similarity of moving objects across non-overlapping views. Then based on the similarity in appearance, weighted votes are used to learn the temporally correlating information. By exploiting the statistical spatio-temporal information weighted by the similarity in an object's appearance, this method can automatically learn the possible links between the disjoint views and recover the topology of the network. After the network topology has been recovered, we then gather statistics about motion patterns in this distributed camera setting. And finally, we explore the problem of how to detect unusual tracks using the information we have inferred.

Thesis Supervisor: W. Eric L. Grimson
Title: Bernard Gordon Professor of Medical Engineering

Acknowledgments

First of all, I would like to thank my advisor, Professor Eric Grimson, for his constant wise guidance through my graduate research, and for maintaining an open and interactive environment for me to thrive in. His ever-positive attitude toward work and life sets the standards that I always strive to achieve.

Thanks to Chris Stauffer for providing the tracker, Kinh Tieu for providing the map plotting code. I would also like to thank the fellow graduate students in the group: Gerald Dalley, Biswajit Bose, Tomas Izo, Joshu Migdle, Xiaoxu Ma, and Xiaogang Wang.

I want to thank my parents for their love and support, and especially for encouraging me all along to pursue my own independent development. Finally, thanks to my husband for his kind understanding and eternal support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Because of the development of technology, multi-camera visual surveillance applications are rapidly increasing in interest. Those applications include tracking moving objects throughout a set of views, classifying those moving objects into different categories (i.e. cars, people, animals), learning the network topology, getting statistics about the moving objects, and finally detecting and interpreting uncommon activities of the moving objects. In this thesis, we are focusing on the last three applications, assuming the first two tasks have been solved.

1.1 Wide-area Surveillance Problem

Consider the problem of wide-area surveillance, such as traffic monitoring and activity classification around critical assets (e.g. an embassy, a troop base, critical infrastructure facilities like oil depots, port facilities, airfield tarmacs). We want to monitor the flow of movement in such a setting from a large number of cameras, typically without overlapping fields of view (FOV). To coordinate observations in these distributed cameras, first we need to know the connectivity of movement between fields of view (i.e. when an object leaves one camera, it is likely to appear in a small number of other cameras with some probability). In some instances, one can carefully site and calibrate the cameras so that the observations are easily coordinated. In many cases, however, cameras must be rapidly deployed and may not last for long periods

of time. Hence we seek a passive way of determining the topology of the camera network. That is, we want to determine the graph structure relating cameras, and the typical transitions between cameras, based on noisy observations of moving objects in the cameras.

If we can in fact determine the "blind" links (i.e. links that connect the disjoint views which cannot be observed directly) between camera sites, we can gather statistics about patterns of usage in this distributed camera setting. We can then record site usage statistics, and detect unusual movements. To determine the network topology and to answer these questions, we must first solve the tracking problem, i.e. we must maintain a moving object's identity from frame to frame, through the same camera view, through overlapping camera views, and through non-overlapping camera views, as shown in Figure 1-1. The bounding box shows the moving object. In the field of view (FOV), vehicles tend to appear and disappear at certain locations. These locations may correspond to garage entrances, or the edge of a camera view, and been called sources and sinks, respectively [20]. Based on the visible tracking trajectories, one can easily learn the links between each source and sink[1].

Tracking through the same views and overlapping views has been widely studied[2] [3] [4] [5]. However, little attention has been paid to the non-overlapping tracking correspondence. Good understanding of the activity requires knowledge of the trajectories of moving objects. For the field out of view, however, the tracking correspondences are unavailable, even the tracking trajectories are unavailable, which makes this problem harder.

1.2 Observations

In this thesis, first we focus on how to learn the non-overlapping network topology, which means to detect the possible "blind" links between disjoint views, and determine the transition time (i.e., the time between disappearing at one location and reappearing at the other location). Our learning is based on the following observations:

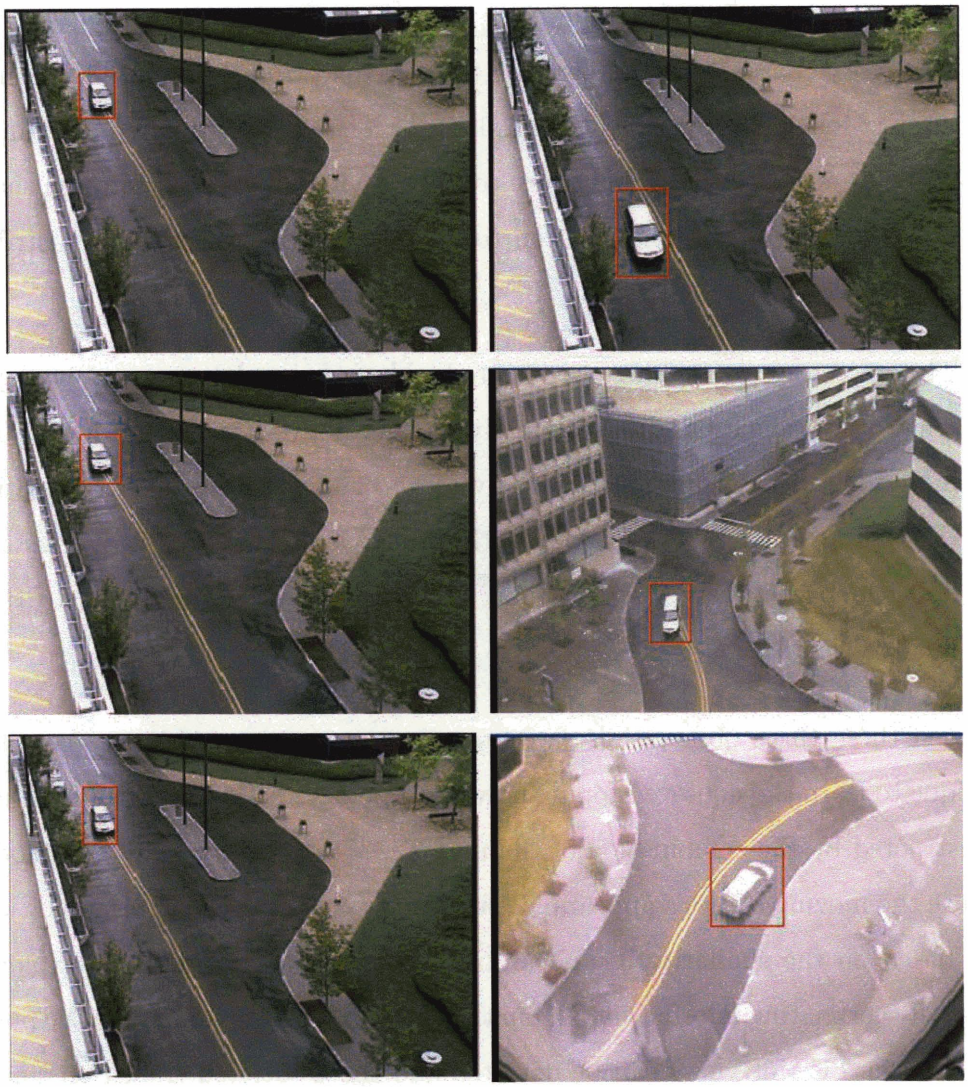

Figure 1-1: Tracking examples. The bounding box shows the moving object. The first row shows tracking through the same view, the middle row shows tracking through overlapping camera views, and the bottom row shows tracking through non-overlapping camera views.

1. Physical characteristics of moving objects do not change. For example, a red sedan in one view is still a red sedan in another disjoint view, it cannot become a white SUV.

2. Vehicles running on the same route roughly share the same speed and other trajectory characteristics. Based on the real road traffic, most vehicles on road are just following traffic. They will slow down and stop with a red light and will speed up when the green light turns on. This will make the assumption that the transition time from one location to another location is Gaussian distributed reasonable.

3. The trajectories of moving objects are highly correlated across non-overlapping views (i.e. vehicles are not randomly moving between different views). To be more illustrative, suppose a vehicle wants to go from location A to location C through location B. It will go directly from A to B and then to C, instead of doing loops between A and B (i.e. from A to B, then to A, then to B) and finally goes to C.

Given these three observations, we propose to use a weighted cross-correlation technique to learn the non-overlapping network topology. First, an appearance model is constructed by the combination of the normalized color and overall size model to measure the moving object's appearance similarity across the non-overlapping views. Then based on the similarity in appearance, the votes are weighted to exploit the temporally correlating information. From the learned correlation function the possible links between disjoint views can be detected and the associated transition time can be estimated. Given the possible (i.e. candidate) links, we can finally recover the network topology by the estimated mutual information. This method combines the appearance information and statistics information of the observed trajectories, which can overcome the disadvantages of the approaches which only use one of them. This method avoids the camera calibration, and avoids solving the tracking correspondence between disjoint views.

1.3 Related Work

One possible approach to learn the connectivity or spatial adjacency of the camera network is to use calibrated camera networks [14] [17]. Jain et al. [14] used calibrated cameras and an environmental model to obtain the 3D location of a person. Collins et al. [17] developed a system consisting of multiple calibrated cameras and a site model, and then used region correlation and location on the 3D site model for tracking. This kind of method usually requires detecting the same landmarks with known 3D coordinates from different cameras and using a complex site model.

Another possible approach is to solve the tracking correspondence problem directly. Ali et al.[9] uses MAP estimation over trajectories and camera pose parameters to calibrate and track with a network of non-overlapping cameras. Huang and Russell [7] present a Bayesian foundation for computing the probability of identity, which is expressed in terms of appearance probabilities. Their appearance model is treated as the product of several independent models, such as: lane, size, color and arrival time. They have used a simple Gaussian model to measure the transition probability between two disjoint views.

Javed et al. [8] adopted Huang and Russell's method[7] and used Parzen windows to estimate the inter-camera space-time (i.e., transition time between two views) probabilities and then solved the correspondence problem by maximizing the posterior probability of the space-time and appearance.

Kang et al. [15] used a combination of appearance model and motion model to track the moving objects continuously using both stationary and moving cameras, then learned the homography between the stationary, the moving cameras and the affine transform derived from the stabilization.

The above methods require that we establish the correspondence for individual tracks between non-overlapping views. The correspondence assignment problem can be found in time $O(n^3)$ by formulating the problem as a weighed bipartite matching problem, which is difficult and time consuming. However, appearance information between different views is still quite useful and should not be discarded.

Other approaches to estimate the spatio-temporal information uses statistical model[10] [11] [16] [13]. Petty et al.[11] proposed to estimate transition time from aggregate traffic parameters in a freeway scenario. Westerman et al. [16] used cumulative arrivals at successive detector sites to estimate vehicle arrivals. Ellis[13] proposed two stage algorithm to learn the topology. First detecting entry and exit zones in each view, then temporally correlating the disappearance and reappearance of tracked objects between those views to detect possible links. For those statistical methods, the performance is only based on information of appearing and leaving time of the detected moving objects at each soure/sink. It will not perform well under fair heavy traffic condition.

1.4 Thesis Organization

This thesis is organized as follows. In Chapter 2, we introduce the joint probability model (i.e. appearance model) for measuring the similarity in appearance between detected moving objects. In Chapter 3, the cross-correlation method is constructed to learn the spatio-temporal information, then, the proposed method, "weighted" cross-correlation method, to learn the possible link associated with transition time is discussed. Chapter 4 gives the experimental results and associated problems. Section 5 presents mutual information and how to fully recover the network topology based on the estimated mutual information followed by the results presented in Chapter 6. Given the recovered network topology, Chapter 7 discusses how to learn the transition probability and the source/sink information, finally be able to detect the unusual tracks.

Chapter 2

The Appearance Model

To coordinate observations in the distributed cameras, we need to know the connectivity of movement between fields of view (i.e. when an object leaves one camera, it is likely to appear in a small number of other cameras with some probability), which means we need to know the network topology. In the following three parts, we will focus on how to recover the camera network topology.

The far field vehicle tracking system we have been using is provided by Chris Stauffer[3]. The input to the tracking system is the video sequence, and the output of the tracking system is a set of tracking sequences, where each track is a sequence of observations of the same object (supposely) in the field of view. These tracks are provided as input to our topology learning system. Some sample observations are shown in Figure 2-1.

In different views, the same object can appear dramatically different, not only the size, but the color as well. In order to relate the appearance of an object from

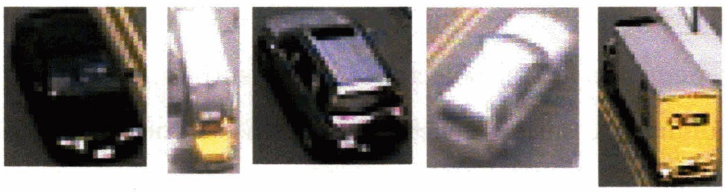

Figure 2-1: Examples of observations

view to view, the appearance model (i.e. color model, and size model) should be learned first. Learning the appearance model is carried out by assuming that there exists some known correspondences between disjoint views. One way to achieve the correspondence, is by driving the same car around the environment. An other possible way is to manually detect interesting vehicles (i.e. yellow cab, Fedex truck, blue bus) across the disjoint views. Since we only need to model color and overall size, unlike the traditional appearance-based correspondence method, which requires a significant amount of the known correspondence, only some small number of the best matches are needed in the training phase.

2.1 Normalized Color Model

Various methods have been proposed to model the color change of moving objects from one camera to another. For far-field vehicle surveillance, since a vehicle is the only moving object and usually contains one color, a single color model per vehicle would be sufficient. However, under different views, the same color may appear dramatically different due to the lighting geometry and illuminant color (figure 2-2). Based on this consideration, we adopt a normalized color model. First, we use the comprehensive color normalization(CCN) algorithm proposed by Finlayson et al. [18] to reprocess the input color images.

2.1.1 Comprehensive Color Normalization Algorithm

The light reflected from a surface depends on the spectral properties of the surface reflectance and of the illumination incident on the surface. In the case of Lambertian surfaces, the light is simply the product of the spectral power distribution of the light source with the percent spectral reflectance of the surface. Assuming a single point source light, illumination, surface reflection and sensor function, combining together forms a sensor response:

$$\bar{p}^{\widehat{x,E}} = \bar{e}^{x}.\bar{n}^{x} \int_{\omega} S^{x}(\lambda)E(\lambda)\bar{F}(\lambda)d\lambda \tag{2.1}$$

where λ is wavelength, $\bar{p}$ is a 3-vector of sensor responses (rgb pixel value), $\bar{F}$ is the 3-vector of response functions (red, green and blue sensitive), E is the illumination striking surface reflectance S^x at location x. Integration is over the visible spectrum ω. Bar denotes vector quantities. The light reflected at x, is proportional to $E(\lambda)S^x(\lambda)$ and is projected onto $\bar{x}$ on the sensor array. The precise power of the reflected light is governed by the dot-product term $\bar{e}^x.\bar{n}^x$. Here, $\bar{n}^x$ is the unit vector corresponding to the surface normal at x and $\bar{e}^x$ is in the direction of the light source. The length of $\bar{e}^x$ models the power of the incident light at x. Note that this implies that the function $E(\lambda)$ is actually constant across the scene. Substituting $\bar{q}^{x,E}$ for $\int_\omega S^x(\lambda)E(\lambda)\bar{F}(\lambda)$ allows us to simplify the above formula into:

$$\bar{p}^{\hat{x},E} = \bar{q}^{x,E}\bar{e}^x.\bar{n}^x \tag{2.2}$$

It is now understood that $\bar{q}^{x,E}$ is that part of a scene that does not vary with lighting geometry (but does change with illuminant color). Equation 2.2, which deals only with point-source lights is easily generalized to more complex lighting geometries. Suppose the light incident at x is a combination of m point source lights with lighting direction vectors equal to $\bar{e}^{x,i}$ $(i = 1, 2, \cdots, m)$. In this case, the camera response is equal to:

$$\bar{p}^{\hat{x},E} = \bar{q}^{x,E}\sum_{i=1}^{m}\bar{e}^x.\bar{n}^x \tag{2.3}$$

Of course, all the lighting vectors can be combined into a single effective direction vector:

$$\bar{e}^x = \sum_{i=1}^{m}\bar{e}^{x,i} \Rightarrow \bar{p}^{\hat{x},E} = \bar{q}^{x,E}\bar{e}^x.\bar{n}^x \tag{2.4}$$

This equation conveys the intuitive idea that the camera response to m light equals to sum of the responses to each individual light. Since we now understand the dependency between camera response and lighting geometry, it is a scalar relationship dependent on $\bar{e}^x.\bar{n}^x$, it is straightforward to normalize it:

$$\frac{\bar{p}^{\widehat{x,E}}}{\sum_{i=1}^{3} p_i^{\widehat{x,E}}} = \frac{\bar{q}^{x,E}\bar{e}^{x}.\bar{n}^{x}}{\sum_{i=1}^{3} \bar{q}_i^{x,E}\bar{e}^{x}.\bar{n}^{x}} = \frac{\bar{q}^{x,E}}{\sum_{i=1}^{3} \bar{q}_i^{x,E}} \tag{2.5}$$

when $\bar{p}^{\widehat{x,E}} = (r, g, b)$ then the normalization returns: $(\frac{r}{r+g+b}, \frac{g}{r+g+b}, \frac{b}{r+g+b})$.

Hence, we can define function $R()$:

$$R(I)_{i,j} = \frac{I_{i,j}}{\sum_{k=1}^{3} I_{i,k}} \tag{2.6}$$

where I is an $N \times 3$ image matrix with N image pixels, whose columns contain the intensity of 3 RGB color channels.

Let us now consider the effect of illuminant color. If we hold lighting geometry, the vectors $\bar{e}^{x}$, fixed and assume the camera sensors are delta functions: $F(\lambda) = \delta(\lambda - \lambda_i),\ i = (1, 2, 3)$. Under $E(\lambda)$ the camera response is equal to:

$$\bar{p}_i^{\widehat{x,E}} = \bar{e}^{x}.\bar{n}^{x} \int_{\omega} S^{x}(\lambda)E(\lambda)\delta(\lambda - \lambda_i)d\lambda = \bar{e}^{x}.\bar{n}^{x}S^{x}(\lambda_i)E(\lambda_i) \tag{2.7}$$

and under a different $E_1(\lambda)$:

$$\bar{p}_i^{\widehat{x,E_1}} = \bar{e}^{x}.\bar{n}^{x} \int_{\omega} S^{x}(\lambda)E_1(\lambda)\delta(\lambda - \lambda_i)d\lambda = \bar{e}^{x}.\bar{n}^{x}S^{x}(\lambda_i)E_1(\lambda_i) \tag{2.8}$$

Combining the above two equations together we can get:

$$\bar{p}_i^{\widehat{x,E_1}} = \frac{E_1(\lambda_i)}{E(\lambda_i)}\bar{p}_i^{\widehat{x,E}} \tag{2.9}$$

This equation informs us that, as the color of light changes, the values recorded in each color channel scale by a factor (one factor per channel). It is straightforward to remove the image dependence on illuminate color by function $C()$:

$$C(I)_{i,j} = \frac{N/3I_{i,j}}{\sum_{k=1}^{N} I_{k,j}} \tag{2.10}$$

where I is an $N \times 3$ image matrix with N image pixels, whose columns contain the intensity of 3 RGB color channels. The $N/3$ here is to ensure that the total sum of all pixels after the column normalization is N which is the same as that after the row

normalization. The comprehensive normalization procedure is defined as a loop:

1. $I_0 = I$

2. do $I_{i+1} = C(R(I_i))$ until $I_{i+1} = I_i$

Note that after the iteration, we get a lighting geometry and illuminant color independent image. An example is shown in figure 2-2 and 2-3. Because HSV color model is more similar in the way humans tend to perceive color, the example is shown in the HSV color model. Figure 2-2 is the color histograms of one car's two observations before color normalization. We can see that the two histograms for Hue and Saturation are pretty different. After the normalization, however, the histograms for Hue and Saturation match well (Figure 2-3).

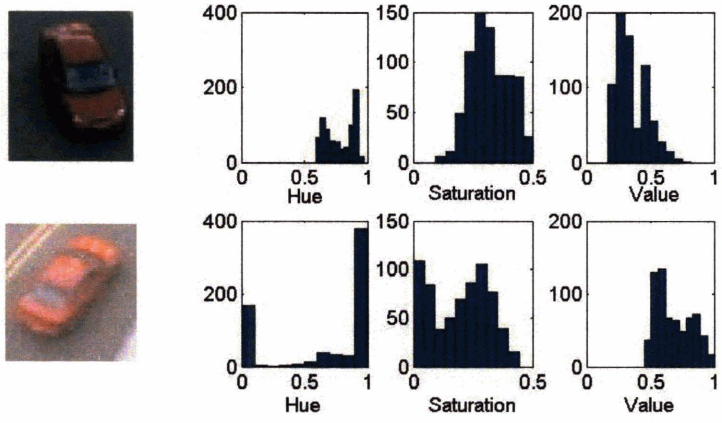

Figure 2-2: Color histograms of one car's two observations before color normalization

2.1.2 Color Model

After the color normalization procedure, we can use a color histogram (in color space HS) to fit a multivariate Gaussian distribution modeling the color change P_{color} throughout any two different scenes:

$$\begin{aligned} P_{color} &= P(h^{c_1}, s^{c_1}, h^{c_2}, s^{c_2} | O^{c_1} = O^{c_2}) \\ &= N_{\mu_{h,s}, \Sigma_{h,s}}(h^{c_1} - h^{c_2}, s^{c_1} - s^{c_2}) \end{aligned} \tag{2.11}$$

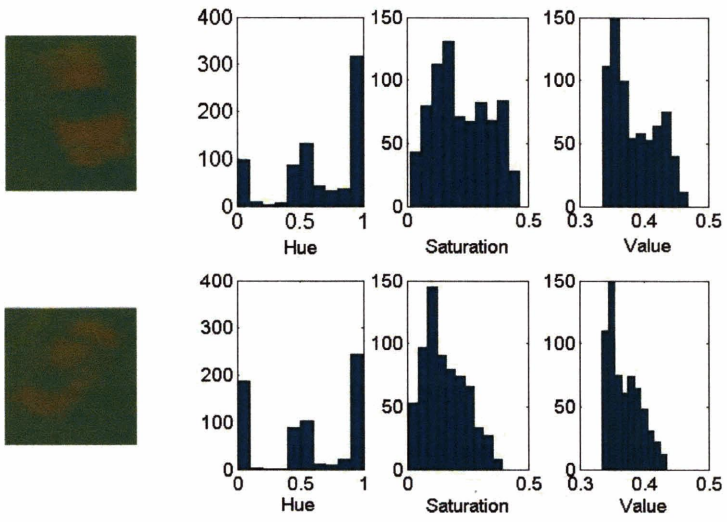

Figure 2-3: Color histograms of the same car's two observations after color normalization

where c_1, c_1 are the camera 1 and 2. O^{c_1}, O^{c_2} are the detected observation under camera 1 and camera 2 respectively. h, s are H and S information included in the observation. $\mu_{h,s}$ and $\Sigma_{h,s}$ are the mean and variance respectively. And $O^{c_1} = O^{c_2}$ means those two observations are actually generated by the same object. For each pair of different views, there is a multivariate Gaussian distribution associated with it. To learn the parameters of the Gaussian, for each pair of the camera views, we have used the quadratic distances of the normalized color histogram (i.e. H and S histograms) to compute the mean and variance.

2.2 Size Model

For far-field surveillance, even after successful detection, there are often very few image pixels per object, which makes it difficult to model the shape change throughout cameras. However, we know for sure that a sedan in one scene cannot be a truck in another scene, which means overall size information still plays an important role in correspondence. Here we use width and length of the bounding box to measure the overall size. This estimate of size is somewhat simplistic. However, given that objects

are fairly small in far field settings, it is unlikely that we will be able to recover the shape detail, so all we rely on is over size measures. Ideally, we should fit a best ellipse to the shape, to account for orientation relative to the camera, but in general given the small image size of objects, we find width and length to suffice.

We also adopt a multivariate Gaussian distribution to model the size change P_{size}.

$$\begin{aligned} P_{size} &= P(w^{c_1}, l^{c_1}, w^{c_2}, l^{c_2} | O^{c_1} = O^{c_2}) \\ &= N_{\mu_{w,l}, \Sigma_{w,l}}(w^{c_1} - w^{c_2}, l^{c_1} - l^{c_2}) \end{aligned} \tag{2.12}$$

where w^{c_1},l^{c_1} are the detected vehicle's width and length under camera 1. $\mu_{w,l}$ and $\Sigma_{w,l}$ are the mean and variance respectively. The imaging transformation of a perspective camera leads to distortion of a number of geometric scene properties. As a result, objects appear to grow larger as they approach the camera center and become smaller when they are far away from the camera[19]. So in the sense of simple normalization, the average size over the whole trajectory has been adopted, when we do the size model. The parameters of this Gaussian distribution can be estimated using the same procedure as described in Chapter 2.1.2.

2.3 Joint Probability Model

Given two observations o_a^i and o_b^j, where o_a^i is the observation a from camera i and o_b^j is the observation b from camera j, the similarity in appearance between those two observations can be calculated as the probability that these two observations are actually generated by the same object, which is called "appearance probability", denoted by $P(o_{a,i}, o_{b,j} | a = b)$. It is important to note that the appearance probability is *not* the probability $a = b$.

Assuming that color and size information of each observation is independent, the similarity in appearance between two observations can be described as the product of the color and size similarity:

$$
\begin{aligned}
& P_{similiarity}(o_{a,i}, o_{b,j}) \\
&= P(o_{a,i}, o_{b,j}|a = b) \\
&= P(color_{a,i}, color_{b,j}|a = b)P(size_{a,i}, size_{b,j}|a = b) \\
&= P_{color}P_{size}
\end{aligned}
\tag{2.13}
$$

Now we know how to model the appearance change of objects from view to view, and how to measure the similarity in appearance for two observations. This result will be used to help exploring the statistical spatio-temporal information (see Chapter 3).

Chapter 3

Weighted Cross Correlation Model

If we can determine the "blind" links (i.e. links that connect the disjoint views) between camera sites, we can then gather statistics about patterns of usage in this distributed camera setting. This would then allow us to detect unusual movements, to classify types of activities, to record site usage statistics. In this chapter, we will discuss how to incorporate the appearance similarity information into the cross correlation function, then use it to estimate the possible blind links between disjoint views.

3.1 Cross Correlation Function

In statistics, the term cross correlation is sometimes used to refer to the covariance $cov(X, Y)$ between two random vectors X and Y, in order to distinguish that concept from the "covariance" of a random vector X, which is understood to be the matrix of covariances between the scalar components of X.

In signal processing, the cross correlation (or sometimes "cross-covariance") is a standard method of estimating the degree to which two series are correlated, commonly used to find features in an unknown signal by comparing it to a known one[12]. Consider two discrete series $x(i)$ and $y(i)$ where $i = 0, 1, 2...N-1$. The cross correlation R at delay d is defined as:

$$R(d) = \sum_{i=0}^{i=N-1} x_i * y_{i+d} \tag{3.1}$$

If the above is computed for all delays d=0,1,2,...N-1 then it results in a cross correlation series of twice the length as the original series.

There is the issue of what to do when the index into the series is less than 0 or greater than or equal to the number of points ($i - d < 0$ or $i - d \geq N$). The most common approaches are to either ignore these points or assuming the series x and y are zero for $i < 0$ and $i \geq N$. In many signal processing applications the series is assumed to be circular in which case the out of range indexes are "wrapped" back within range, ie: $x(-1) = x(N-1), x(N+5) = x(5)$ etc.

The range of delays d and thus the length of the cross correlation series can be less than N, for example the aim may be to test correlation at short delays only. The denominator in the expression above serves to normalize the correlation coefficients such that $-1 \leq r(d) \leq 1$, the bounds indicating maximum correlation and 0 indicating no correlation. A high negative correlation indicates a high correlation but of the inverse of one of the series.

3.2 Cross Correlation Model

As mentioned in the chapter of Introduction, there are two observations: Transition time from one location to another location is Gaussian distributed. And the trajectories of moving objects are highly correlated across non-overlapping views. Under these two observations, we can see that the sequences of appearing vehicles under the connected cameras (i.e. there exist routes directly connecting those cameras) are highly correlated. Since cross correlation function can capture the degree of correlation between two signals, we present a simple cross-correlation model to estimate the existence of possible blind links and the associated transition time between different cameras.

For each traffic source/sink (i.e. locations where objects tend to appear in a scene and locations where objects tend to disappear from a scene), traffic can be represented

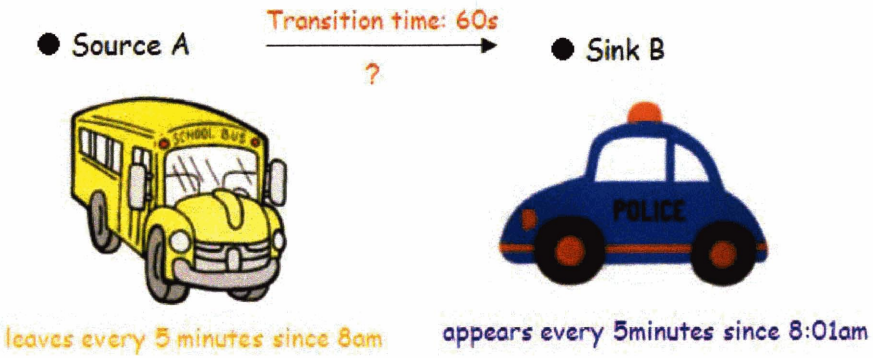

Figure 3-1: Example of the case which cross correlation doesn't work

as a discrete flow signal $V_i(t)$, which is defined as the list of observations (see Figure 2-1) appearing in a time interval around time t at source/sink i.

The cross-correlation function between signals $V_i(t)$ and $V_j(t)$ can indicate the possibility of a link, and be used to estimate the transition time if there exists such a link:

$$R_{i,j}(T) = \sum_{t=-\infty}^{t=\infty} ||V_i(t)|| * ||V_j(t+T)|| \tag{3.2}$$

If there is a possible link between source/sink i and j, there should exist a clear peak in $R_{i,j}(T)$ at time $T = t$, where t denotes the transition time from location i to location j. In this sense, a possible "blind" link from location i to location j has been learned.

However, there are some limitations to this method. For example, it would not perform well under heavy traffic conditions. To illustrate this problem, we present an extreme situation (See Figure 3-1). Suppose at source/sink A, an yellow school bus leaves every 5 minutes since 8am, at source/sink B, a blue police car appears every 5 minutes since 8:01am, and there is no possible link between A and B. However, if we use the cross correlation method directly, a possible link will be learned and the learned transition time would be 60 seconds.

Intuitively, at different source/sinks, only those observations which look similar in appearance can be counted to derive the spatio-temporal relation. In order to fix

this problem, we propose a weighted cross correlation technique

3.3 Weighted Cross Correlation Model

The weighted cross correlation technique is defined as :

$$R_{i,j}(T) = \sum_{t=-\infty}^{t=\infty} \sum_{O_{a,i} \subseteq V_i(t)} \sum_{O_{b,j} \subseteq V_j(t+T)} P_{similarity}(O_{a,i}, O_{b,j}) \tag{3.3}$$

Specifically, for a pair of disappearing vehicles at source/sink i at time t and appearing vehicles at source/sink j at time $t+T$, calculate the similarity in appearance between those two observations and update $R_{i,j}(T)$. Then peak values can be detected using the threshold estimated as:

$$threshold = mean(R_{i,j}(T)) + w * std(R_{i,j}(T)) \tag{3.4}$$

where w is a user-defined constant.

In this work, we assume there is only one popular transition time if there is a link between i and j. People in real life tend to choose the shortest path between the start location and the destination, which makes the single transition time reasonable with the assumption of constant velocity. Although we assume there is only one popular transition time between two disjoint views, this weighted cross correlation model can be applied to the cases with multiple transition times which will result in multiple peaks in $R(T)$. For our implementation, transition time is assigned with the time associated with the highest detected peak. Figure 5 gives an example when weighted cross correlation can detect a valid link, while general cross correlation fails. After applying the general and weighted cross correlation function on the data from two cameras located at an intersection, the results are shown in Figure 3-2 (a) and (b), respectively. (b) has a clear peak which suggests a possible link with transition time 11 seconds between those cameras, which (a) does not.

In this part, we learned how to use the weighted cross correlation model to estimate the possible blind links and the associated transition time between disjoint views. we

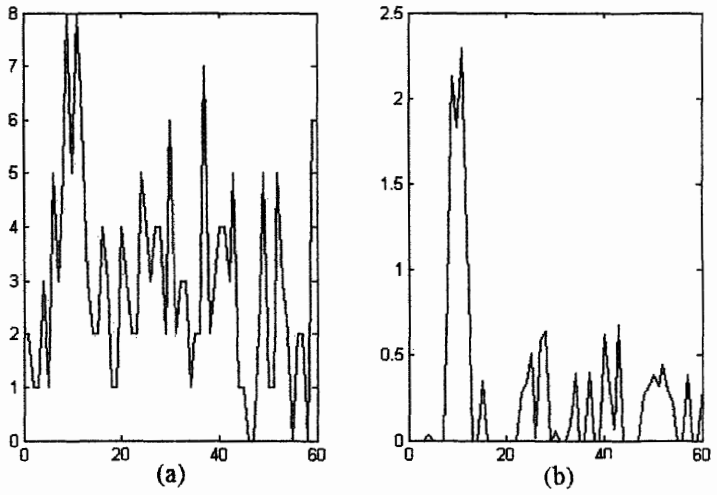

Figure 3-2: After applying the general and weighted cross correlation function on the data from two cameras located at an intersection, the results are shown in Figure 5 (a) and (b), respectively. (b) has a clear peak which suggests a possible link with transition time 11 seconds between those cameras, which (a) doesn't.

will present the experimental results in the next section using both real tracking data and synthetic tracking data.

Chapter 4

Experiments and Problems

In order to evaluate the proposed the weighted cross correlation method, we have tested it both on real data and synthetic data.

4.1 Real Data

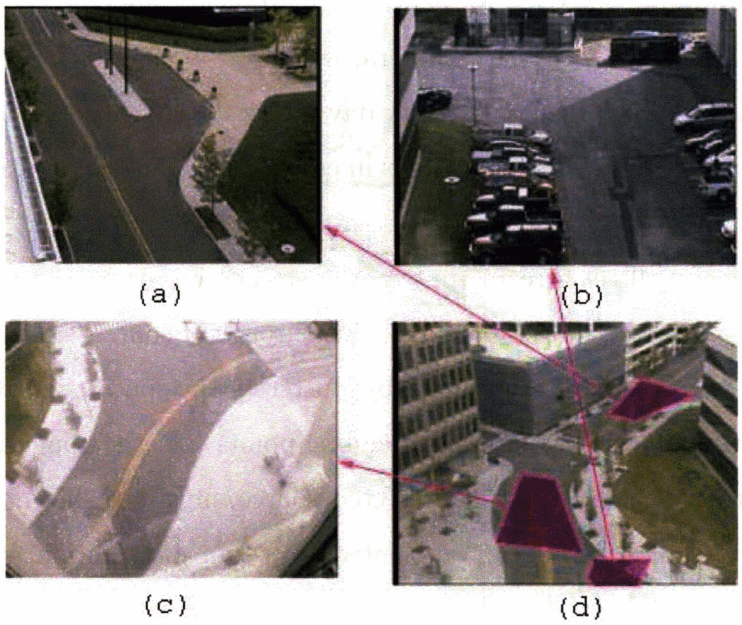

Figure 4-1: (a),(b),(c) are the three non-overlapping cameras we have used. The cameras' relative location is shown in (d) using the shaded quadrangle.

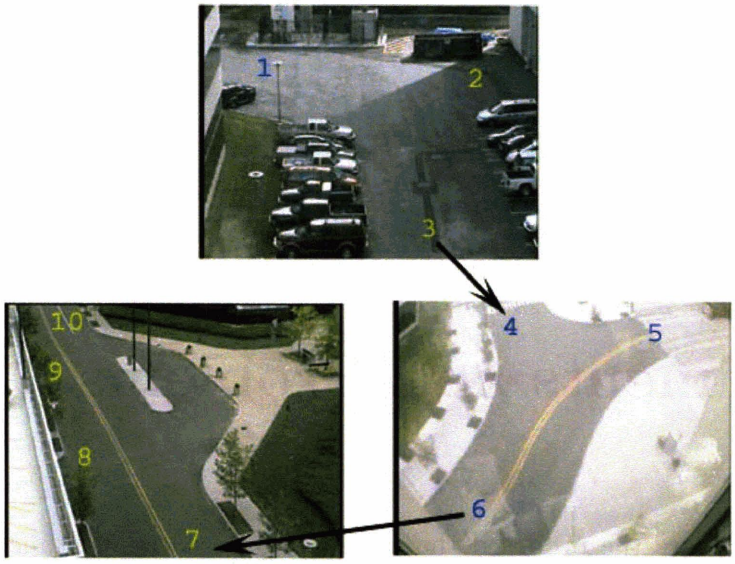

Figure 4-2: Detected sources/sinks. Black arrows indicate direct links between source/sink 3 and source/sink 4, source/sink 6 and source/sink 7

For the real data experiment, we used three non-overlapping cameras distributed around several buildings. The layout of the environment and the cameras' location are shown in Figure 4-1. For each camera, we have 1 hour of vehicle tracking data obtained from a tracker based on[3] every day for six days. There are total of 213 observations in camera(a),1056 observations in camera (b), 1554 observations in camera (c).

In our cameras, all the streets are two way streets, i.e. each source is also a sink. For simplicity, we merge sources and sinks into groups of source/sinks. The detected source/sinks in each camera are learned by clustering the spatial distribution of each observation's trajectory's beginning and ending points (i.e. the appearing coordinate and disappearing coordinate). The detected source/sinks are shown in Figure 4-2. For each source/sink, there is an associated Gaussian distribution with mean and variance. From the cameras' spatial relationship, we know that there exists directly links between source/sink 3 and source/sink 4, source/sink 6 and source/sink 7, and there is no other direct link among those sources/sinks. Visible links can be easily

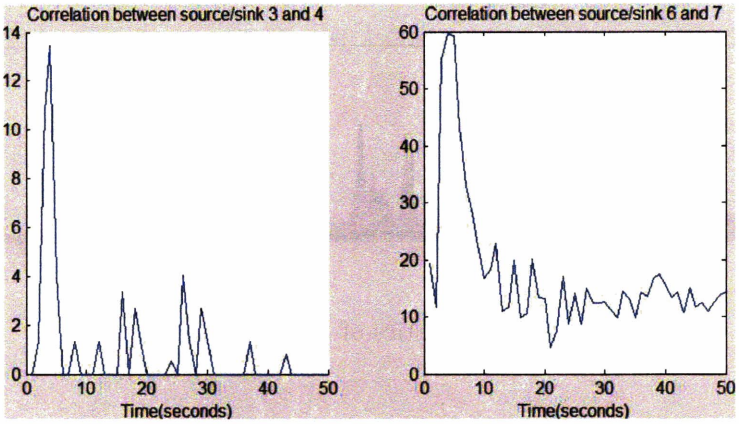

Figure 4-3: Cross correlation functions between different views. Left one gives the cross correlation between camera b,source/sink 3 and camera c, source/sink 4, with transition time 3 seconds; Right one shows correlation between camera c, source/sink 6 and camera a, source/sink 7, with transition time 4 seconds.

learned using trajectories information. Our goal is to learn such "blind" links.

Because we only focus on learning the "blind" link between disjoint views, we know that the transition time must be non-negative which is determined by the nature of traffic flow, i.e, the same vehicle must first disappear at one specific location, then can reappear at the other different location. However, if overlapping views have been considered, the transition time may be negative.

For any pair of source/sinks, we have using the disappearing vehicles at one sink and the appearing vehicles at the other source to calculate the weighted cross correlation function. A possible link has been detected if there exists a significant peak in the cross-correlation function (See equation 3.4, in our experiments, w is set to 2). Only two possible links have been detected as shown in Figure 4-3. The left one gives the cross correlation between camera b,source/sink 3 and camera c, source/sink 4, with transition time 3 seconds; The right one shows correlation between camera c, source/sink 6 and camera a, source/sink 7, with transition time 4 seconds. Notice that the detected "blind" links don't include the links like the one between source/sink 10 to source/sink 6 through source/sink 7. The reason is that we have used the visible trajectory's information. If we want to check the possible "blind" link between source/sink 10 and source/sink 6, we would use the observations that

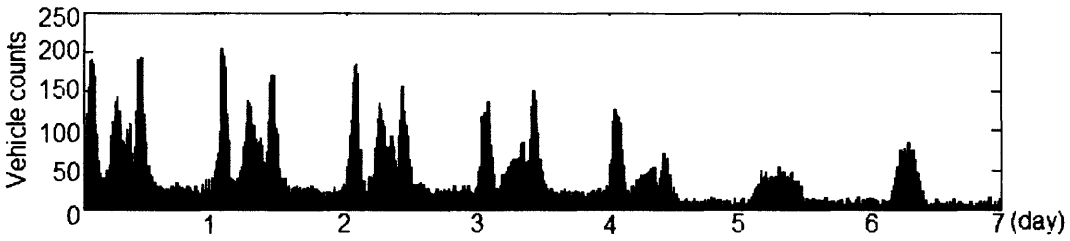

Figure 4-4: Statistics of the simulated data

leave the scene through source/sink 10 and the ones that enter the scene through source/sink 6, which wouldn't give the link through source/sink 7. So the cameras' topology can be fully recovered.

4.2 Simulated Data

We also tested our algorithm on a simulated network. This simulator synthetically generates the traffic flow in a set of city streets, allowing for stop signs, traffic lights, and differences in traffic volume (i.e. morning rush hours and afternoon rush hours have a higher volume, as well as lunch traffic). The network includes 101 cameras which are located at roads' intersections (including cross and T intersections). For each camera, there are two observers that look in the opposite directions of the traffic flow (i.e. Observer 1 and 2 belong to camera 1 , Observer 3 and 4 belong to camera 2, etc). Every observer can be treated as a source/sink. Tracking data has been simulated 24 hours every day for 7 week days, including 2597 vehicles (Fig. 4-4).

Transition time may change with the road condition. For example, it will be larger during rush hour than during non-rush hour. So in our experiment, we only pick one particular hour data (10am to 11am) each day for 5 days. For each camera, the only information we have is that vehicles appear then disappear from this location roughly the same time (i.e. the duration is very short), so we can treat it like a delta function.

For each pair of the observers, we first calculate the cross correlation function that has been learned for each pair of the observers. A possible link has been detected

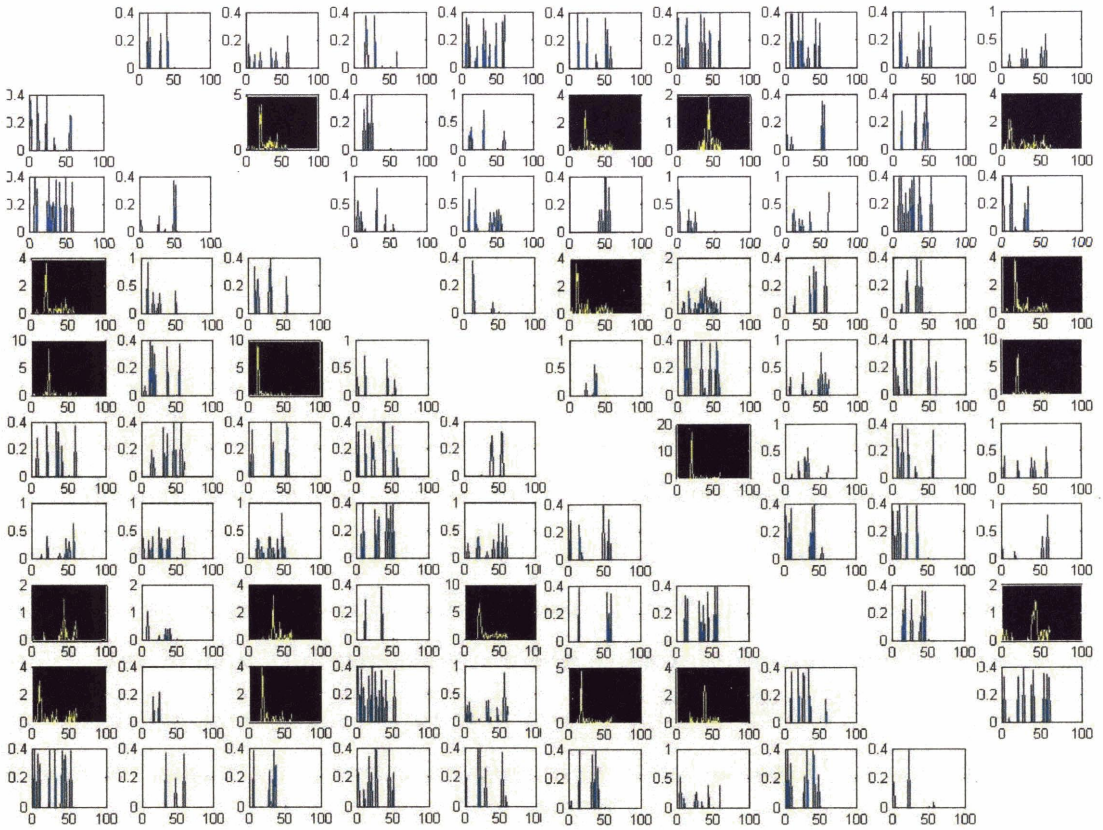

Figure 4-5: Cross correlation for each pair of the observers from 17,18,...,to 26. The column index from left to right is: observer 17, observer 18, ..., observer 26; The row index from up to bottom is: observer 17, observer 18, ..., observer 26.

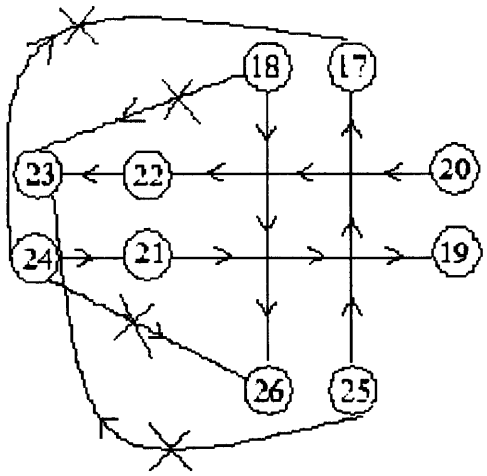

Figure 4-6: The recovered topology based on the weighted cross correlation,the red cross indicates the false link based on the group truth.

if there exists a significant peak in the cross-correlation function (See equation 3.4, in our experiments, w is set to 2). Figure 4-5 shows the cross correlation results for each pair of the observers from 17, 18,..., to 26, each row is the parent observer, each column is the child observer, detected possible links are highlighted in black background figures. From the detected links, however, the topology wasn't correctly recovered(see Figure 4-6). For example, there are detected links from observer 25 to 22, from observer 22 to 23 and from observer 25 to 23. We don't know if the link from observer 25 to 23 is actually through observer 22, or if there exists another link between them.

4.3 Problems

Unlike the real data, this camera view has only one source/sink and we have no information of any visible links, so we don't know where the vehicles are coming from and where they are going. Hence all the vehicles have been used to calculate the cross correlation function. In order to get rid of those "fake" links and recover the true topology, we think of mutual information.

Chapter 5

Mutual Information and Estimation

From the previous chapter, we know that weighted cross correlation function can only help us to detect "possible" blind links, which may include "fake" links. In this chapter, we will focus on how to solve this problem and discuss how to use mutual information to remove the fake links and fully recover the network topology.

5.1 Mutual Information

Mutual information is a measure of the amount of information that one random variable contains about another random variable. It is the reduction in the uncertainty of one random variable due to the knowledge of the other.

5.1.1 Entropy

First, we will introduce the concept of entropy, which is a measure of uncertainty of a random variable and a measure of the amount of information required on the average to describe the random variable. Let X be a random variable and $p(x)$ be the probability distribution function of X .

Definition[21]: The *entropy* $H(X)$ of a discrete random variable X is defined

by

$$H(X) = -\int p(x) log p(x) dx \tag{5.1}$$

The definition of entropy is related to the definition of entropy in thermodynamics; The higher the entropy of one random variable, the more uncertain of this random variable. Next, we will introduce the two relate concepts: relative entropy and mutual information.

5.1.2 Relative Entropy and Mutual Information

The relative entropy is a measure of the distance between two distributions. In statistics, it arises as an expected logarithm of the likelihood ratio. The relative entropy $D(p||q)$ is a measure of the inefficiency of assuming that the distribution is q when the true distribution is p. For example, if we knew the true distribution of the the random variable, then we could construct a code with average description length $H(p)$. If, instead, we used the code for a distribution q, we would need $H(p)+D(p||q)$ bits on the average to describe the random variable.

Definition[21]: The *relative entropy* or *Kullback-Leibler distance* between two probability distributions $p(x)$ and $q(y)$ is defined as:

$$D(p||q) = \int p(x) log \frac{p(x)}{q(y)} dxdy \tag{5.2}$$

It can be easily shown that relative entropy is always non-negative and is zero if and only if $p = q$. However, it is not a true distance between distributions since it is not symmetric and does not satisfy the triangle inequality. Nonetheless, it is often useful to think of relative entropy as a "distance" between distributions.

Now we are ready to introduce mutual information, which is a measure of the amount of information that one random variable contains about another random variable. It is the reduction in the uncertainty of one random variable due to the knowledge of the other.

Definition[21]: Consider two random variables X and Y with a joint probability

distribution $p(x,y)$ and marginal probability distribution functions $p(x)$ and $p(y)$. The mutual information $I(X;Y)$ is the relative entropy between the joint distribution and the product distribution $p(x)p(y)$:

$$I(X;Y) = \int p(x,y) log \frac{p(x,y)}{p(x)p(y)} dxdy = D(p(x,y)||p(x)p(y) \tag{5.3}$$

From the definition of mutual information, it can be easily shown that $I(X;Y) \geq 0$ with equality if and only if X and Y are independent (i.e. the joint distribution is the same as the product of the marginal distributions). In other words, the higher the mutual information between two variables, the more likely the two variables are dependent. Also, it can be shown[21] that for a Markov chain type topology between three random variables $X \rightarrow Y \rightarrow Z$, we have $I(X;Y) \geq I(X;Z)$.

Mutual information can also be interpreted by the concept of entropy.

$$I(X;Y) = H(X) - H(X|Y) \tag{5.4}$$

Mutual information between X and Y is the uncertainty of X minus the uncertainty of X given the information of Y. In other words, mutual information is the amount by which the uncertainty about X decreases when Y is given: the amount of information Y contains about X.

Mutual information has been used in many fields to recover the topology[23] [24], which means to find the dependency relationships between the variables involved. Graphical models provide a useful methodology for expressing the dependency structure of a set of random variables[27]. Random variables can be treated as nodes of a graph, while edges between nodes indicate the dependency, which can be estimated by the pairwise mutual information. It has been showed that the graph with the maximum edge weight will be the optimum tree dependency approximation.

5.1.3 Data Processing Inequality

As mentioned before, for a Markov chain type topology between three random variables $X \rightarrow Y \rightarrow Z$, we have $I(X;Y) \geq I(X;Z)$, this is called data processing inequal-

ity. Considering our camera network problem, the mutual information of neighboring cameras should be greater than non-neighboring cameras. We can use this property to refine the network topology.

5.2 Mutual Information Estimation

In order to calculate the mutual information, we need to estimate the joint and marginal distributions of X and Y, which is computationally hard. However, with the assumption that X and Y are jointly Gaussian distributed with correlation coefficient ρ_{xy}, the quantity of mutual information can be computed analytically as (ρ_{xy} can capture the linear dependence between X and Y regardless of their joint distribution)[26] [25]:

$$I(X;Y) = -\frac{1}{2}\log_2(1-\rho_{xy}^2) \tag{5.5}$$

From Chapter 3, we already know how to estimate the weighted cross correlation $R_{i,j}(T)$. So if there exists a clear peak in $R_{i,j}(T)$ at time $T = T_{peak}$, the correlation coefficient can be estimated as:

$$\rho_{i,j}^2 = \frac{R_{i,j}(T_{peak}) - median(R_{i,j}(T))}{\sigma_{V_i}\sigma_{V_j}} \tag{5.6}$$

Because the cross correlation function is under the assumption that the signals are transient, which is not accurate for our case, we have used median of $R_{i,j}(T)$ instead of mean of $R_{i,j}(T)$.

5.3 Overall Review of The Algorithm

To implement the proposed algorithm, four steps must proceed sequentially:

1. For each possible pair of source/sinks, learn the cross correlation function;

2. Detect the possible links using the peak detection algorithm;

3. For the detected links, estimate the cross correlation coefficients, otherwise, set the cross correlation coefficient to 0;

4. Recover the network topology based on the estimated mutual information.

Chapter 6

More Experiments

In this chapter, we will use the estimated mutual information to recover the simulated network topology based on the weighted cross correlation function.

6.1 Simulated Network cont'

As we discussed in Chapter 4, for the simulated network (there is only one source/sink per camera view), only using the weighted cross correlation function, the topology cannot be correctly recovered.

So after the cross correlation function has been learned, mutual information has been estimated as shown in Figure 6-1(a) with intensities corresponding to the magnitude of the mutual information. The brighter the figure, the higher the mutual information. From the data processing inequality, we know that mutual information for the neighboring cameras is higher than the mutual information for the non-neighboring cameras. So we can cluster the mutual information into two clusters based on the magnitude. The cluster with higher mutual information would be used to recover the network topology. Figure 6-1(b) is the recovered topology for observer 17 to observer 26. We can see that the link from observer 25 to 23 is actually through 22 which is consistent with the ground truth. Table 6-1 shows the learned associated transition time for each link. Finally, the fully recovered topology of the simulated network is shown in Figure 6-2. Number means the index of the observers.

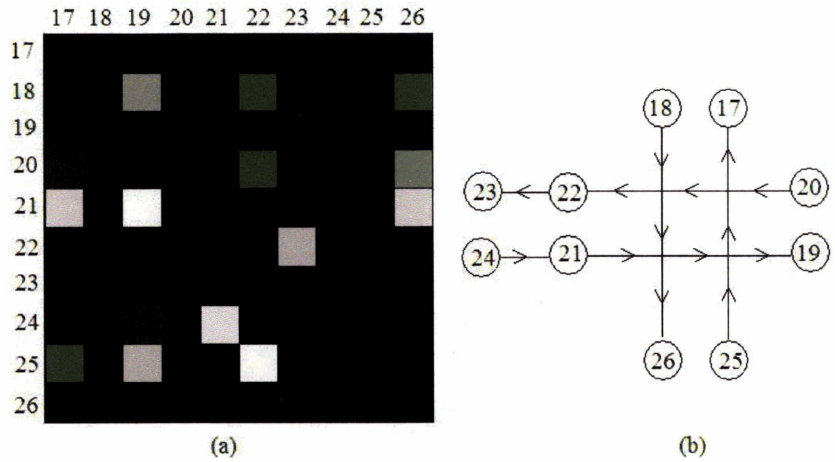

Figure 6-1: (a) The adjacency matrix of the mutual information. (b) The recovered corresponding topology.

For the real data, since there are multiple source/sinks per camera view, which means we can get information of visible trajectories, we can successfully recovery the topology without using mutual information. If there is only one source/sink per camera view (i.e. zooming in), or every camera view is treated as one large source/sink, however, mutual information will be needed to learn the network topology.

Parent observer	Child observer	Tran. time(Seconds)
18	19	21
18	22	23
18	26	11
20	17	20
20	22	10
20	26	17
21	17	24
21	19	14
21	26	20
22	23	21
24	21	20
25	17	10
25	19	19
25	22	18

Table 6.1: The learned associated transition time

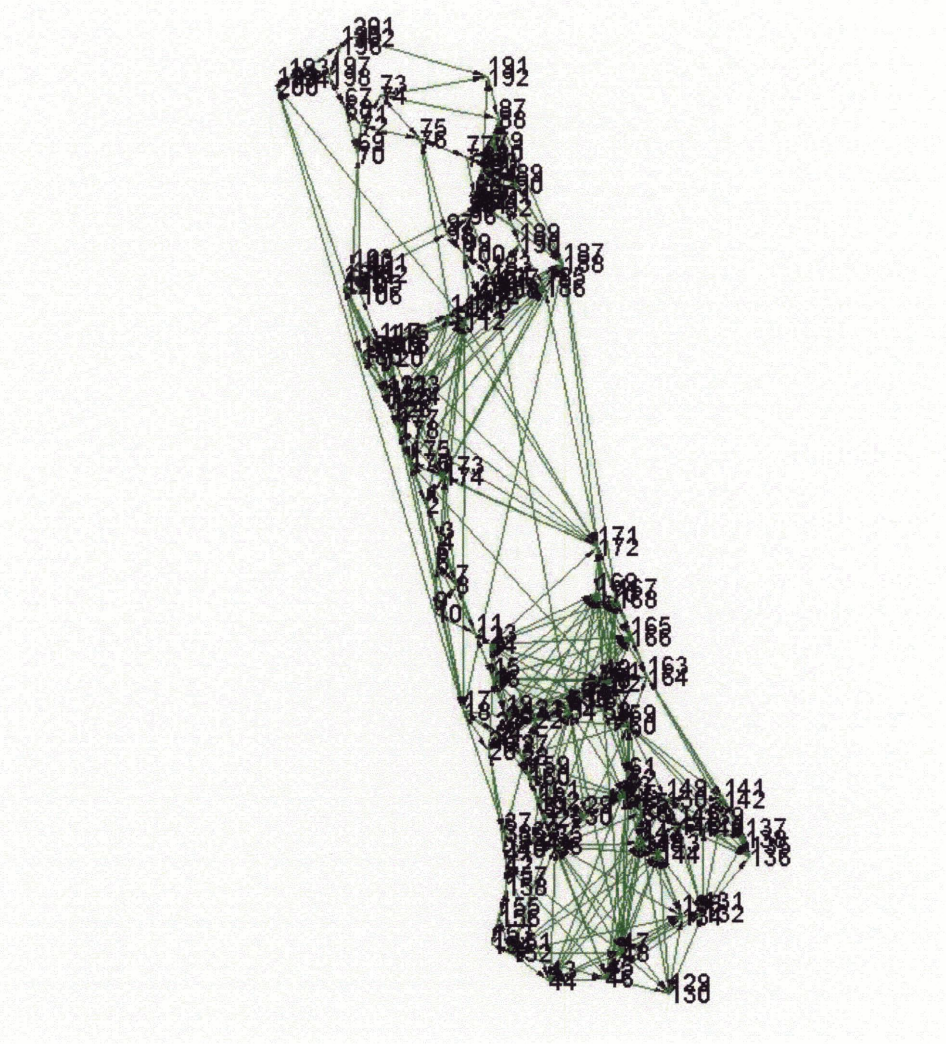

Figure 6-2: The fully recovered simulated network topology

Chapter 7

Information Inference and and Unusual Track Detection

In the previous chapter, we discussed how to recover the network topology using the far-field vehicle tracking data. Now given the network topology, we can learn the transition probability information, the source/sink information, and finally can detect unusual tracks.

7.1 Transition Probability Learning

7.1.1 Markov Process

A discrete-time stochastic process[29][28] is a collection X_n for $n \in 1 : N$ of random variables ordered by the discrete time index n. In general, the distribution for each of the variables X_n can be arbitrary and different for each n. There may also be arbitrary conditional independence relationships between different subsets of variables of the process-this corresponds to a graphical model with edges between most nodes.

Now we will consider discrete-time first-order Markov chains[30], in which the state changes at certain discrete time instants, indexed by an integer variable n. At each time step n, the Markov chain has a state, denoted by X_n, which belongs to a finite set S of possible states, called the state space. Without loss of generality,

and unless there is a statement to the contrary, we will assume that $S = 1, \cdots, m$, for some positive integer m. The Markov chain is described in terms of its transition probabilities p_{ij} : whenever the state happens to be i, there is probability p_{ij} that the next state is equal to j. Mathematically,

$$p_{ij} = P(X_{n+1} = j | X_n = i), i, j \in S \tag{7.1}$$

The key assumption underlying Markov processes is that the transition probabilities p_{ij} apply whenever state i is visited, no matter what happened in the past, and no matter how state i was reached. Mathematically, we assume the Markov property, which requires that

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_0 = i_0) = P(X_{n+1} = j | X_n = i) = p_{ij} \tag{7.2}$$

for all times n, all states $i, j \in S$, and all possible sequences $i_0, \cdots, i_{n-1}$ of earlier states. Thus, the probability law of the next state X_{n+1} depends on the past only through the value of the present state X_n. The transition probabilities p_{ij} must be of course nonnegative, and sum to one:

$$\sum_{j=1}^{m} p_{ij} = 1, for\ all\ i. \tag{7.3}$$

We will generally allow the probabilities p_{ii} to be positive, in which case it is possible for the next state to be the same as the current one. Even though the state does not change, we still view this as a state transition of a special type (a self-transition).

Specification of Markov Models

- A Markov chain model is specified by identifying

  1. The set of states $S = 1, \cdots, m$.
  2. The set of possible transitions, namely, those pairs (i, j) for which $p_{ij} > 0$.
  3. And, the numerical values of those p_{ij} that are positive.

- The Markov chain specified by this model is a sequence of random variables $X_0, X_1, X_2, \cdots$, that take values in S and which satisfy

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_0 = i_0) = P(X_{n+1} = j | X_n = i) = p_{ij} \tag{7.4}$$

for all times n. all states $i, j \in S$, and all possible sequences $i_0, \cdots, i_{n-1}$ of earlier states.

All of the elements of a Markov chain model can be encoded in a transition probability matrix, which is simply a two-dimensional array whose element at the ith row and jth column is p_{ij}:

$$\begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{pmatrix}$$

It is also helpful to lay out the model in the so-called transition probability graph, whose nodes are the states and whose arcs are the possible transitions. By recording the numerical values of p_{ij} near the corresponding arcs, one can visualize the entire model in a way that can make some of its major properties readily apparent.

7.1.2 Transition Probability Learning

After we know the connectivity of the network, we can fit a first order Markov model to this network, hence to learn the transition probability from nodes to nodes.

In the real world, traffic patterns do not remain the same all the time. We wouldn't expect the traffic of morning rush hour to have the same pattern as that of evening non-rush hour. In other words, the transition probability of the network will change with time. Therefore, we would also like to learn the transition probability in the function of time.

We will continue to use the simulated network to demonstrate. As we said before, this simulator synthetically generates the traffic flow in a set of city streets, allowing for stop signs, traffic lights, and differences in traffic volume (i.e. morning rush hours

and afternoon rush hours have a higher volume, as well as the lunch traffic). The network includes 101 cameras which are located at roads' intersections (including cross and T intersections). For each camera, there are two observers that look in the opposite directions of the traffic flow (i.e. Observer 1 and 2 belong to camera 1 , Observer 3 and 4 belong to camera 2, etc). So there are total of 202 observers. Every observer can be treated as a source/sink. Tracking data has been simulated 24 hours every day for 7 week days, including 2597 vehicles. The learned transition probability is shown in the Figure 7-1. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

We have studied the transition probability for every two hours from morning to evening. Figure 7-2 shows the Transition probability of the network between 8am to 9am. Figure 7-3 shows the transition probability of the network between 12pm to 1pm. And Figure 7-4 shows the transition probability of the network between 6pm to 7pm. From those figures, we can see that transition probability does change with the time.

The traffic patterns do not remain same for all types of vehicles either. We wouldn't expect the traffic of buses to have the same pattern as that of gas trucks. In other words, the transition probability of the network will change with different kinds of vehicles. So we try to learn the transition probability in the function of vehicle type. In the simulator, there are total 7 kinds of vehicles: sedan, bus, SUV, minivan, pickup, gas truck, and panel truck. And the results are shown from Figure 7-5 to 7-8. Figure 7-5 shows the transition probability for sedan. Figure 7-6 shows the transition probability for bus. Figure 7-7 shows the transition probability for gas truck. And Figure 7-8 shows the transition probability for panel truck. From those figures, we can see different type of vehicles has totally different transition probability. Especially for gas truck, it only contains activity at limited regions.

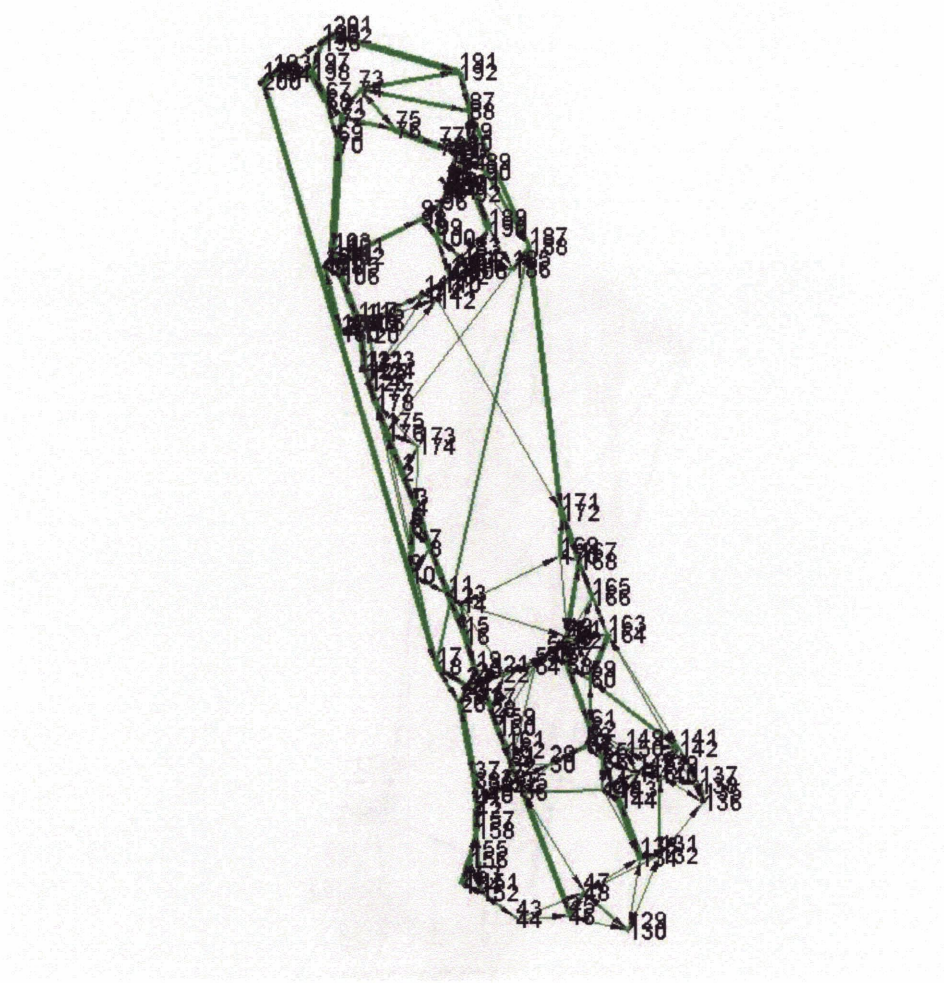

Figure 7-1: Transition probability of the network. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

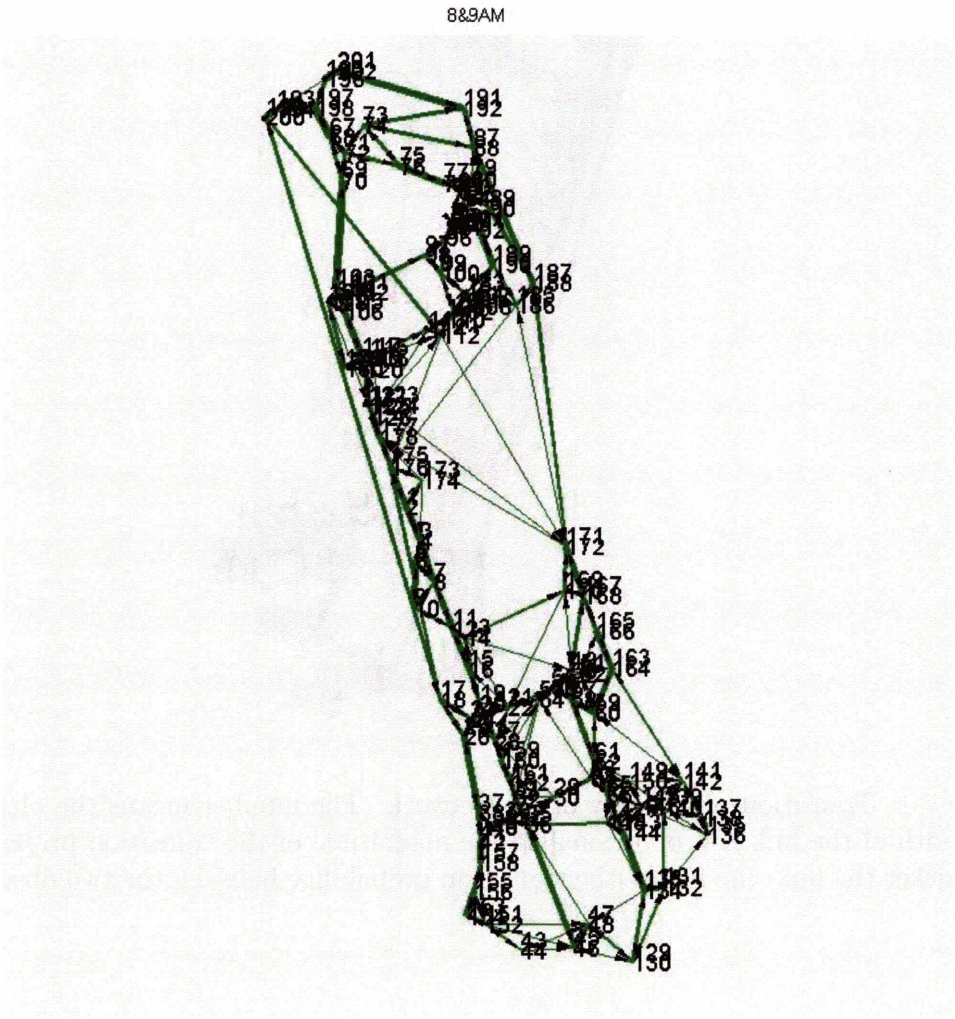

Figure 7-2: Transition probability of the network between 8am to 9am. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers

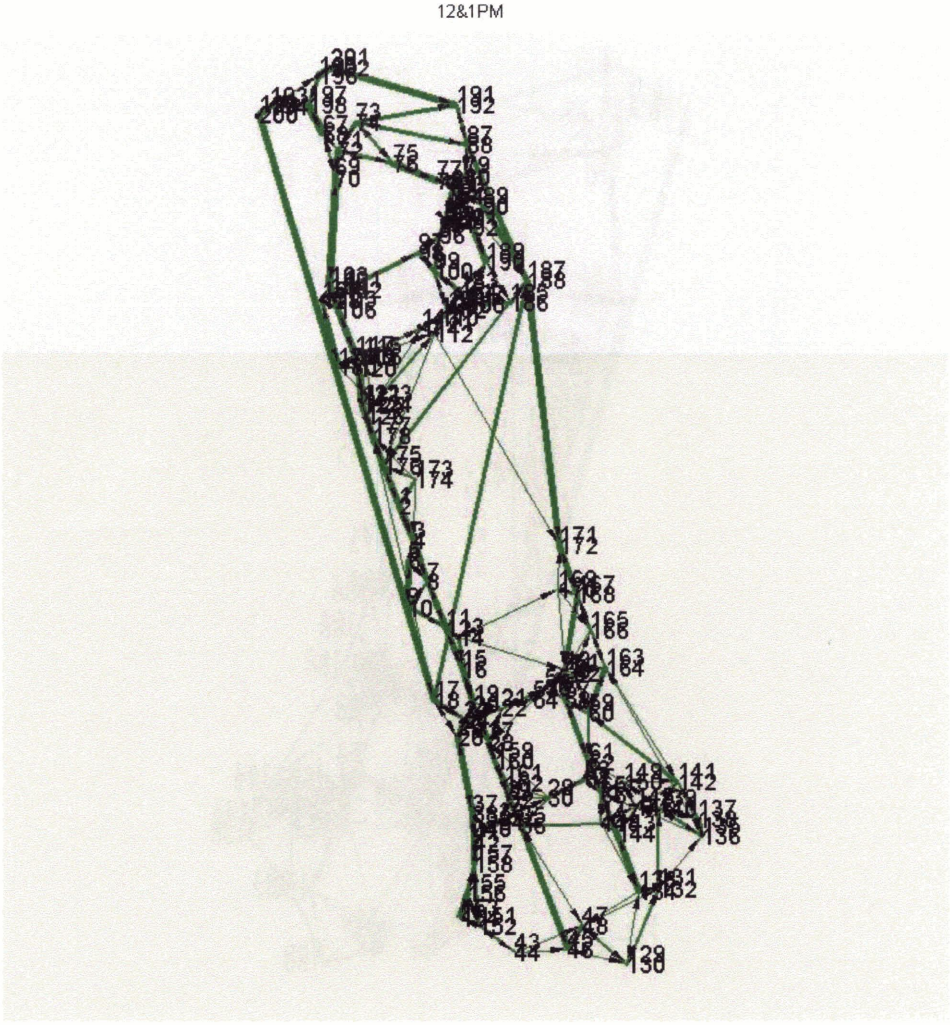

Figure 7-3: Transition probability of the network between 12pm to 1pm. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

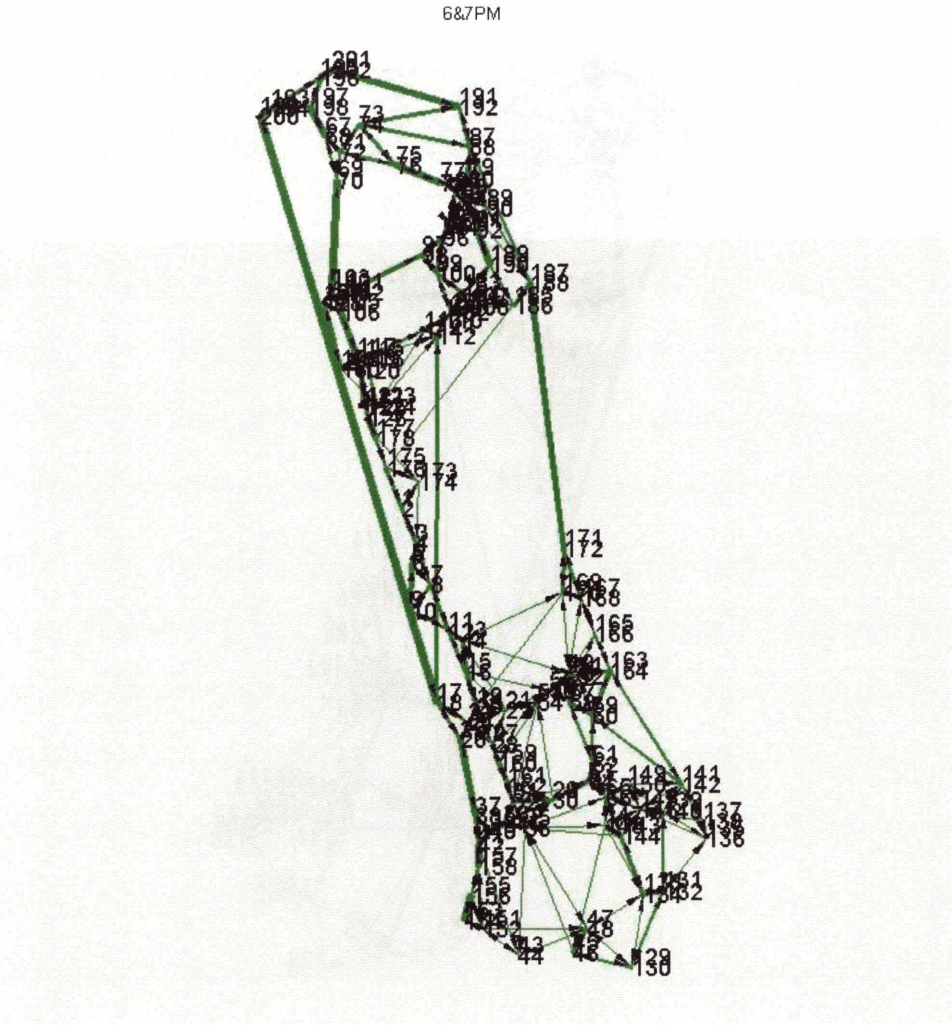

Figure 7-4: Transition probability of the network between 6pm to 7pm. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

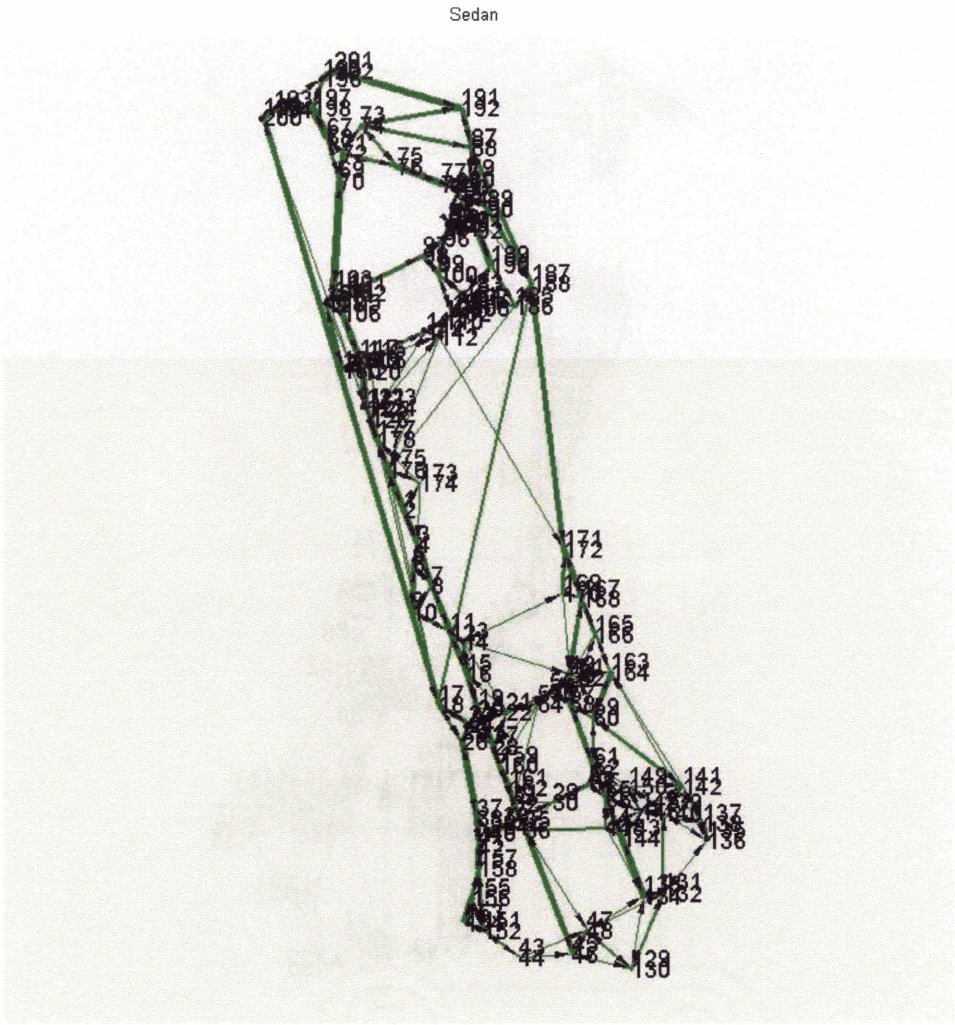

Figure 7-5: Transition probability of the network for sedan. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

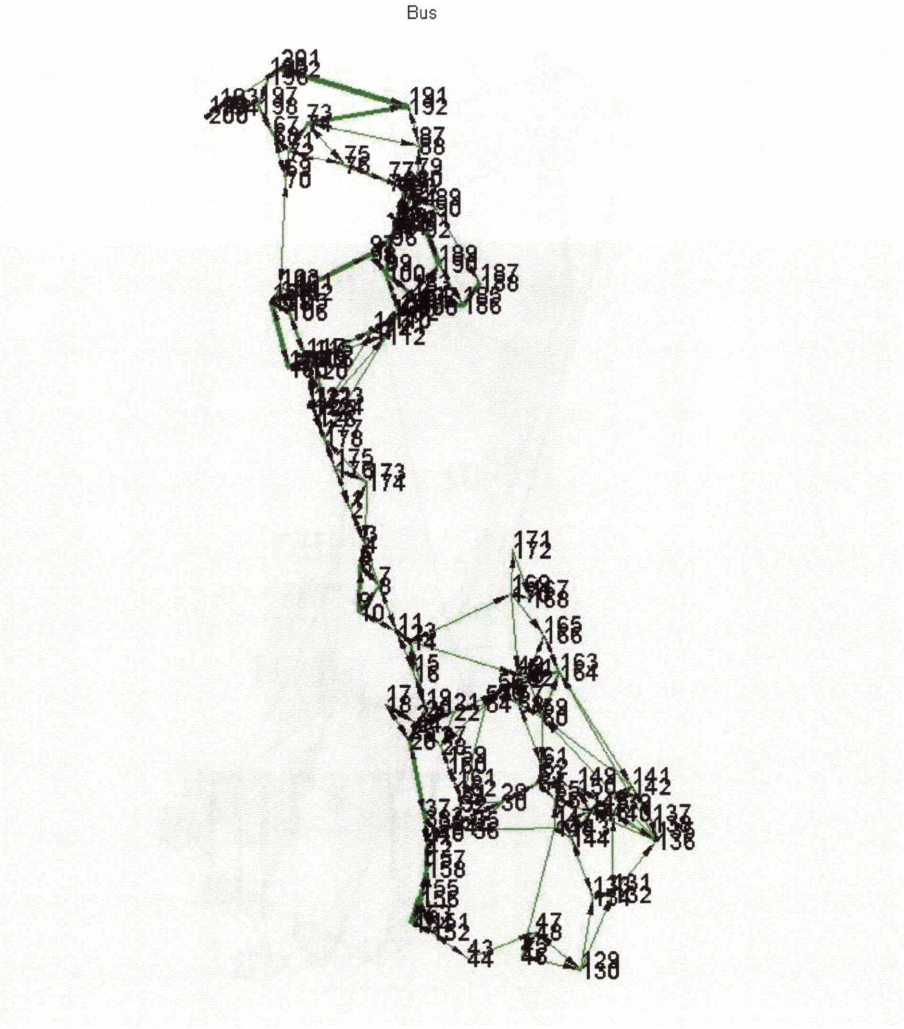

Figure 7-6: Transition probability of the network for bus. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

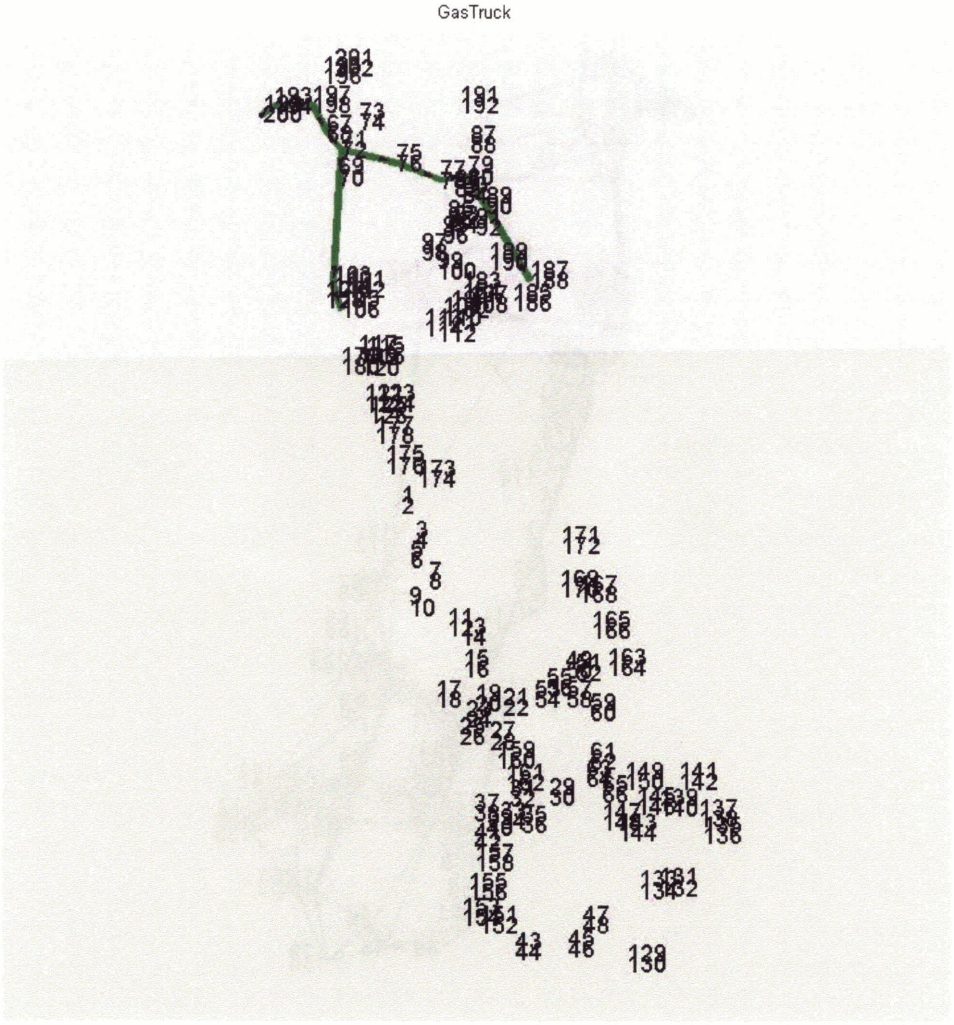

Figure 7-7: Transition probability of the network for gas truck. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

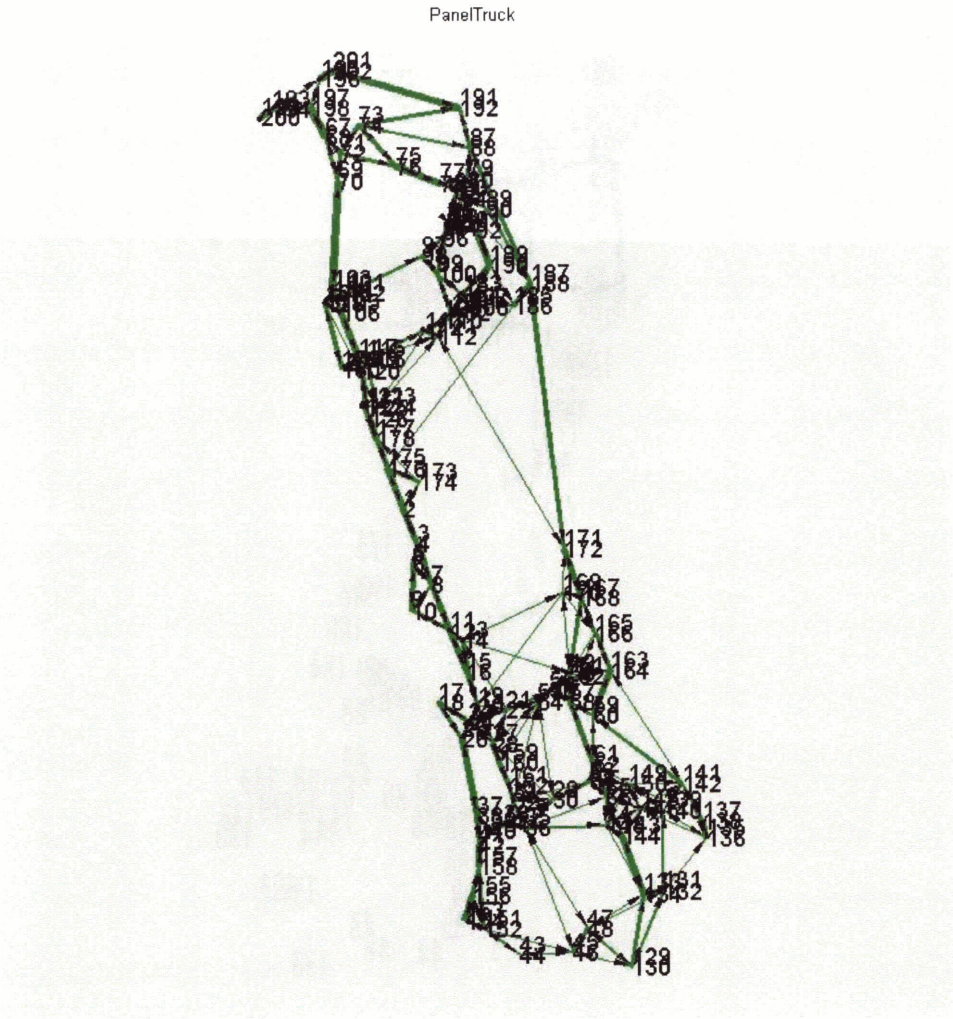

Figure 7-8: Transition probability of the network for panel truck. The number means the observer. The width of the link is proportional to the magnitude of the transition probability. The thicker the link, the higher the transition probability between the two observers.

7.2 Source and Sink Learning

In the field of view, vehicles tend to appear and disappear at certain locations. These locations may correspond to garages entrances, or the edge of a camera view, which have been called sources and sinks, respectively [20]. Source/sink information is also important for motion pattern analysis. It will help us to get an overall sense of what type of vehicles tend to appear and disappear at specific locations and get to know existence of certain infrastructure. For example, if we observe that the gas truck always disappear at one location, we may conclude that there might be a gas station around. It will also help us to detect some unusual events. So in this part, we focus on to learn the source and sink distribution for all the data and also learn the source and sink distribution for different type of vehicles.

Figure 7-9 , 7-10 and 7-11 show the source and sink distribution for all the tracking data, for the tracking data of gas truck and for the tracking data of panel truck, respectively. In the figures, the size and color of the number is corresponding to the probability of that observer to be a source or a sink. The larger and brighter the number, the higher the probability of that observer to be a source or a sink. From those figures, we can see that different type of vehicles yields different source/sink distribution. For example, for gas truck, it only tends to appear at observer 200 and observer 75, and disappear at observer 188 and 71. For panel truck, however, it tends to appear at observer 17, observer 172, observer 187, etc, and disappear at observer 18, observer 171, observer 188, etc.

7.3 Unusual Track Detection

The ultimate goal of most surveillance system is the automatic detection of unusual activities thereby triggering alarms. How to define unusual? In Merriam-Webster dictionary, "Unusual" is defined as "uncommon, rare". For the motion pattern analysis, unusual tracks means the tracks which are different from the normal tracks. Given the large number of observations, after we get the statistics(i.e. normal pattern) of

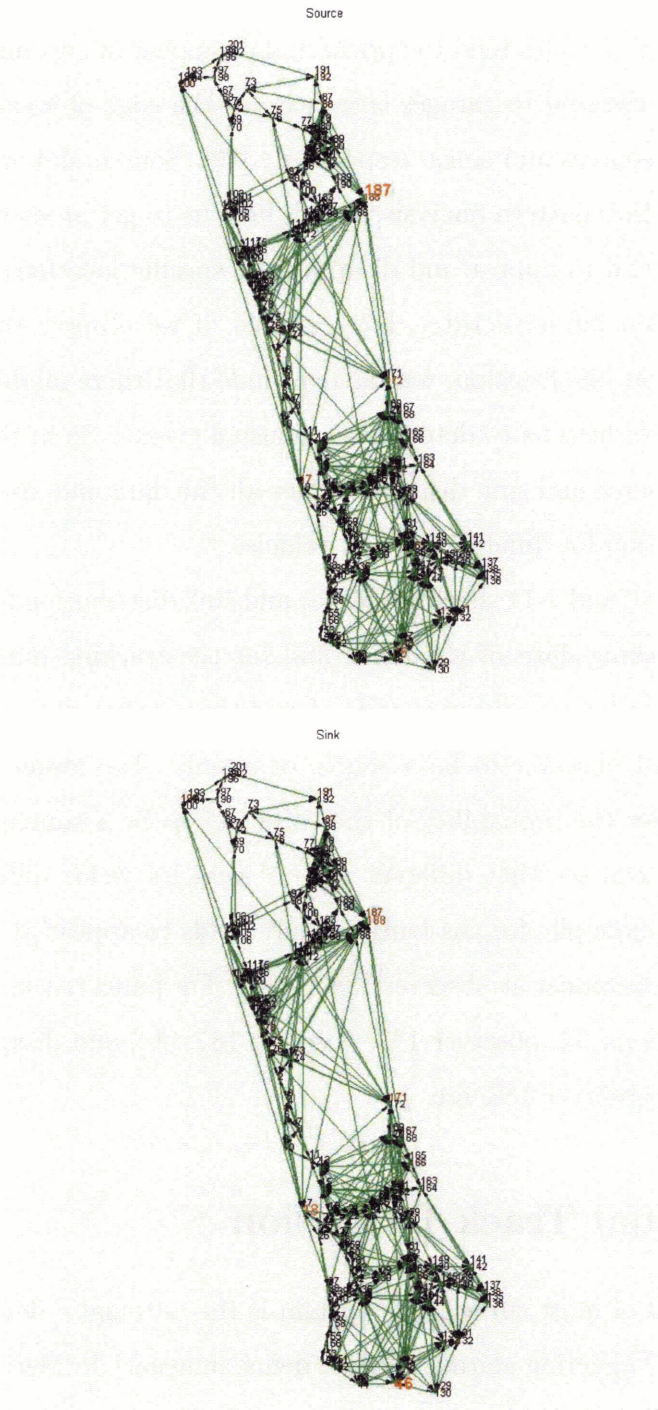

Figure 7-9: Source and sink distribution. The size and color of the number is corresponding to the probability of that observer to be a source or a sink

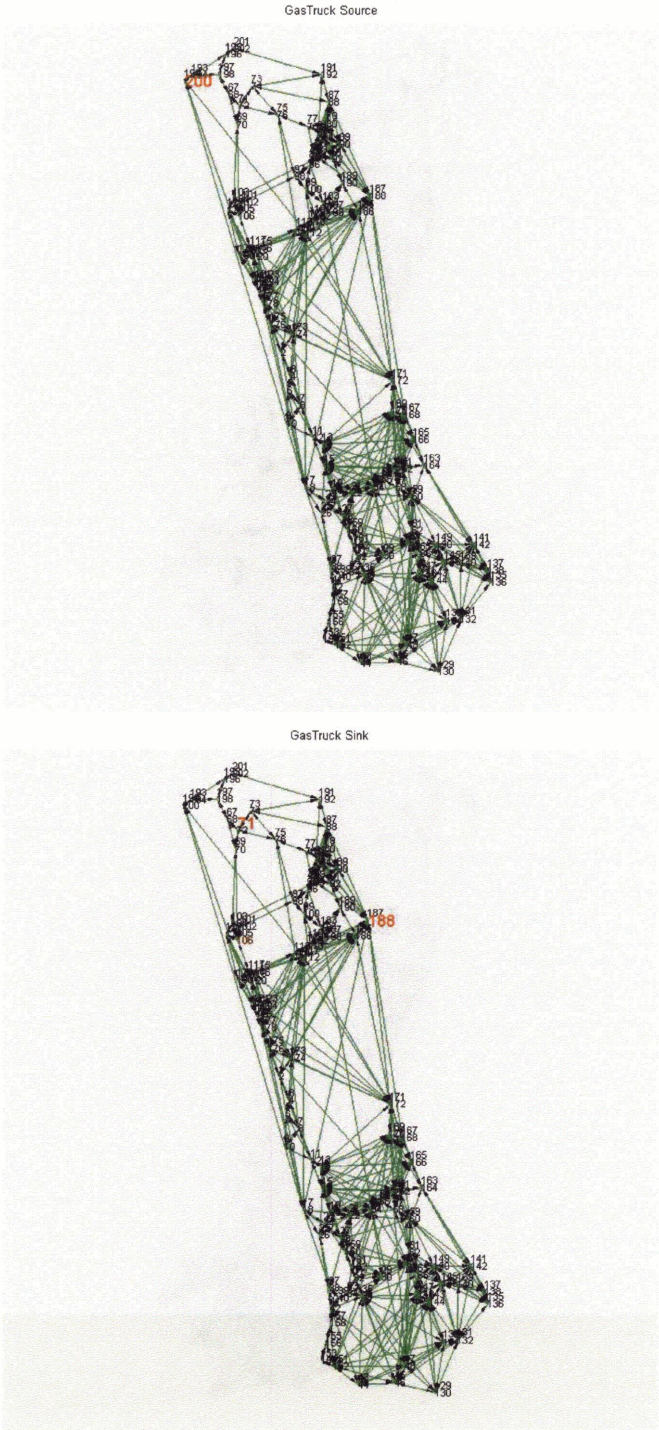

Figure 7-10: Gas truck source and sink distribution. The size and color of the number is corresponding to the probability of that observer to be a source or a sink.

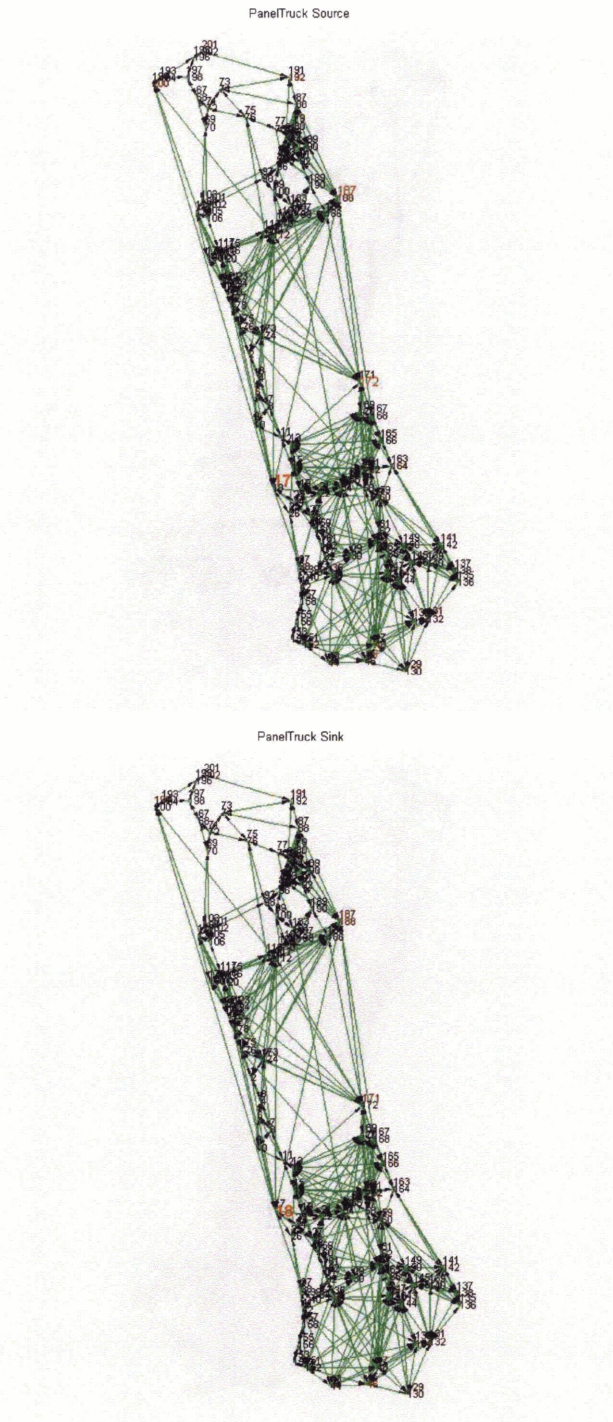

Figure 7-11: Panel truck source and sink distribution. The size and color of the number is corresponding to the probability of that observer to be a source or a sink

the tracking data, we could explore the unusual tracks in the following aspects:

- Does the track include any unusual path given the time and vehicle type ? In another word, is this track associated with very low transition probability?
- Does the track coming from an unusual source or disappearing at an unusual sink given the vehicle type?
- Does the track has repeated pattern?

If any track has yes to above questions, it will be flagged. In the following part, we will discuss in detail how to detect an unusual track using two examples.

Case 1: For the surveillance problem, a gas truck is one special kind of vehicle. In most of cases, it follows a specific route, say appearing from a certain location to the gas station, then, reappearing from the gas station to a specific sink. When it has a different route from its routine route, there may exist some kind of situation we should pay attention to. For example, if we detect a gas truck has a track of following sequence: observer 72, observer 70, observer 104, observer 106. From the source and sink distribution, we know that for a gas truck, the probability that it will disappear at observer 106 is 0.02, which is too small. We may flag this as an unusual track.

Case 2: For the surveillance problem, we should also pay attention to the repeated motion pattern. Consider this scenario: A terrorist plans on bombing a certain critical asset, say a power supply facility. He transports supplies using a pickup truck from his hideout to the parking lot of the facility in several separate, consecutive trips to load and unload supplies. The track looks like: observer 50, 58, 62, 149, 150, 61, 57, 49, 50, 58, 62, 149,150, 61, 57, 49, 50, 58, 62, 149, 150, 61, 57, 49, 50, 58, 62, 149, 150. This track has repeated under camera 25, 29, 31, 75 for 7 times. We need to find and flag this track. In the one week data, for pickup truck, there are total 1 track which has a pattern repeated for 7 times, 8 tracks which have a pattern repeated for 6 times, 46 tracks which have a pattern repeated for 5 times. The track we are interested ranks number 1 in all the repeated tracks, and will stand out from all the tracks. Therefore, we can flag this track as unusual track.

Chapter 8

Summary and Discussion

In this thesis, we have studied how to learn the motion pattern of the vehicles using far-field vehicle tracking data. The first and most important step is to recover the network's topology. In order to solve this problem, we proposed a weighted cross-correlation technique. First, an appearance model is constructed by the combination of the normalized color and overall size model to measure the moving object's appearance similarity across the non-overlapping views. Then based on the similarity in appearance, the votes are weighted to exploit the temporally correlating information. From the learned correlation function the possible links between disjoint views can be detected and the associated transition time can be estimated. Based on the learn cross correlation, the network topology can be recovered based on the estimated mutual information.

This method combines the appearance information and statistics information of the observed trajectories, which can overcome the disadvantages of the approaches which only use one of them. This method avoid doing the camera calibration, avoid solving the tracking correspondence between disjoint views.

However, our algorithm is based on three assumptions: (a) The appearance of the moving objects doesn't change. (b)The objects are moving at a roughly constant velocity. (c)The trajectories of the moving objects are highly correlated across non-overlapping views. If any of these three assumption fails, the proposed algorithm would present uncorrect results.

Another limitation of this method is that it can only solving the topology with one popular transition time between disjoint views; If the transition time is multi-model, one possible way to solve it is to estimate the mutual information directly, which means to estimate the joint distribution and marginal distribution of variables.

After we discover the topology of the network, we then gather statistics about motion patterns in this distributed camera setting. This would then allow us to record site usage statistics, to classify types of activities, and to detect unusual movements. First , we fit a first order of markov model to this network, hence to learn the transition probability from nodes to nodes, to learn the transition probability in the function of time, as well as the transition probability in the function of vehicle type. Then we infer the information of the source/sink distribution, and the source/sink distribution in the function of vehicle type. Finally, we explore the problem how to detect unusual tracks using the information we have inferred.

In future, we would like to explore the problem of recovering the network topology associated with multi transition time. Estimating the mutual information directly from estimating the joint and marginal distribution of the variables is probably good given the condition of multi transition time. The next work is to find a more "general" way to define unusual tracks using the statistics information of the tracking data.

Bibliography

[1] Chris Stauffer, Eric Grimson, "Learning Patterns of Activity Using Real-Time Tracking", IEEE Transactions on Pattern Recognition and Machine Intelligence (TPAMI), 22(8):747-757, 2000.

[2] Reid D B. "An algorithm for tracking multiple targets", AC, 24(6):843-854, December 1974.

[3] Chris Stauffer, "Adaptive Background Mixuture Models for Real-Time Tracking", Proc. IEEE Conf. on Computer Vision and Pattern Recognition, July, 1999.

[4] Isard M, MacCormick J P, "Bramble: A Baysian Multiple-Blob Tracker", IEEE International Conference on Computer Vision, October, 2001.

[5] Williams O, Blake A and Cipolla R. "A Sparse Probabilistic learning algorithm for real-time tracking". IEEE International Conference on Computer Vision, October, 2003.

[6] Chris Stauffer, Kinh Tieu, "Automated multi-camera planare tracking correspondence modeling", Proc. IEEE Conf. on Computer Vision and Pattern Recognition, July, 2003.

[7] Huang T, Russell S, "Object identification in a Bayesian context",in Proc. of IJCAI, 1997.

[8] Omar Javed, Zeeshan Rasheed, Khurram Shafique, Mubarak Shah, "Tracking Across Multiple Cameras With Disjoint Views", IEEE International Conference on Computer Vision, October, 2003.

[9] Ali Rahimi, Brian Dunagan, Trevor Darrell,"Simultaneous Calibration and Tracking with a Network of Non-overlapping Sensors", IEEE International Conference on Computer Vision and Pattern Recognition, June, 2004.

[10] Dailey D, "Travel Time Estimation Using Cross Correlation Techniques", Transportation Research Part B, vol 207B, No 2, pp97-107, 1993

[11] Petty, K, et al, "Accurate Estimation of Travel Times From Single Loop Detectors", paper presented at the 76th annual Transprotation Research Part meeting, 1997.

[12] Http://en.wikipedia.org/wiki/Cross-correlation

[13] Makis D, Ellis T, et al, "Learning a Multicamera Topology", Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, October, Nice, France, 2003.

[14] Jain R, Wakimoto K, "Multiple perspective interactive video", IEEE International Conference on Multimedia Computing and System, 1995.

[15] Kang J, Cohen I, Medioni G,"Continuous Tracking Within and Across Camera Streams",Proc. Computer Vision and Pattern Recognition, July, 2003.

[16] Westerman M, Immers L, "A Method for Determining Real-Time Travel Times on Motorways", Road Transport Informatics/Intelligent Vehicle Highways Systems, ISATA, pp 221-228, 1992.

[17] Collins R, Lipton A, Fujiyoshi H, and Kanade T, "A system for video survillance and monitoring", In Proc. American Nuclear Society(ANS) Eighth International Topical Meeting on Robotic and Remote Systems, April 1999.

[18] Finlayson G D, Schiele B, and Crowley J L, "Comprehensive Colour Image Normalization". Proc. the European Conf. on Computer Vision, 1998.

[19] Biswajit Bose, Eric Grimson, "Ground Plane Rectification by Tracking Moving Objects". Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS), Nice, France, October 2003.

[20] Chris Stauffer. "Estimating Tracking Sources and Sinks," Conference on Computer Vision and Pattern Recognition Workshop, 2003.

[21] Cover T M, Thomas J A, "Elements of Information Theory". Wiley, 1991.

[22] Cormen T H, Leiserson C E, and Rivern R L, "Introduction to Algorithms", MIT Press, 1990.

[23] Singhal M, "A Dynamic Information-Structure Mutual Exclusion Algorithm for Distributed Systems", IEEE Transactions on Parallel and Distributed Systems, 1992.

[24] Brillinger D R,"Second-order moments and mutual information in the analysis of time series", Recent Advances in Statistical Methods. Imperial College Press, London, 2002.

[25] Kullback S, "Information Theory And Statistics". Dover, 1968.

[26] Wentian Li, "Mutual Information Functions Versus Correlation Functions". Journal of Statistical Physics, 60(5-6):823-837 1990.

[27] Finn Jensen. "An Introduction to Bayesian Networks". Spring, 1996.

[28] Queesland University of Technology, "Introduction to Markov Chains courtesy of QUT".

[29] Jeff Bilmes, "What HMMs Can Do". UWEE Technical Report Number UWEETR-2002-2003, 2002.

[30] Dimitri P. Bertsekas, John N. Tsitsiklis, "Introduction to Probability". Athena Scientific, 2002.