MODEL FOR TRACKING MOVING TARGETS USING HETEROGENEOUS
CAMERA SYSTEMS

NAN NING AND TELE TAN

Abstract Sufficient camera coverage is critical to the maintenance of the track continuity of a moving target across a large scale surveillance area. Due to economic and bandwidth reasons, it is often not possible to ensure that cameras within a large distributed surveillance system always have overlapping coverage. Traditional tracking schemes that rely on fixed stationary cameras are prone to fail when the non-overlapping coverage between adjacent cameras becomes significant. In this paper, we present a novel approach on building a tracking model that uses heterogeneous camera systems to reduce the non-overlapping problem. We incorporate the surveillance cameras that are installed on public transport vehicles such as buses into the existing distributed camera system to dynamically increase the surveillance space, as when these vehicles move around the surveillance area the on-board cameras act as mobile video sensors. We adopt a grid-based filtering approach together with a road map to systematically integrate the multiple observations coming from the heterogeneous sensors. Simulated experimental results have shown that our approach has the potential to deal with large-scale observations that are prevalent in city-wide implementation.

Key Words, Heterogeneous camera system, grid-based filtering, large-scale geographical tracking.

1. Introduction

Surveillance cameras are ubiquitous today. The types of them are heterogeneous. It is not surprise that any person or vehicle moving in a city can be expected to be video-taped on many occasions on a daily basis by a system of distributed surveillance cameras. Automated tracking involves the task of automatically determining the track continuity of a target of interest as it moves within the surveillance space using the large image repositories generated by such a surveillance system. Due to economic and bandwidth reasons, it is sometimes not possible to ensure that cameras within the system have overlapping coverage. The challenge here is to address the issue of track continuity when the non-overlapping coverage between adjacent cameras becomes significant and the target’s movement is non-linear. Traditional tracking schemes that rely on a set of fixed stationary cameras can not cope well with this situation. Since the major problem is caused by the insufficient camera coverage, we focus on possible solutions that could increase the coverage.

We observe that the growing trend of fitting public transport vehicles like buses, trains and taxis with some form of surveillance system, mainly to counter vandalism has...
created the opportunity for these systems to be integrated into a larger network-based surveillance system that has the potential to span the entire city. As these vehicles move around the city area, their on-board cameras act as mobile video sensors. If these moving cameras are used together with the stationary cameras to track a target, this would introduce a very powerful technique for resolving the problem of insufficient surveillance coverage faced by the existing system.

As an example, Figure 1 shows a typical camera system configuration for wide outdoor surveillance. The existing system consists of 7 fixed cameras $C_1, C_2, \ldots, C_7$ installed around major road intersections. The camera field-of-view (FOV) is not overlapped. We assume the target is initially located in the FOV of $C_4$. From the road map information, we know it can only take one of the four possible paths to go next. However, due to the large non-overlapping areas between $C_4$ and the rest of the cameras, the temporal-spatial cue for prediction would quickly lose its effectiveness. As the target moves out of the current FOV and proceeds along the non-overlapping regions of the camera system, many possibilities of movement patterns could happen. The uncertainty of predicting and tracking the target before it moves into the FOV of another camera increases with the degree of non-overlapping areas. Therefore, after the target disappears from $C_4$’s FOV and before it reappears in another camera’s FOV (i.e. during the target’s disappearing time period), each of these paths have to be assigned an equal probability to contain the target as we do not know which path it is really in. The long disappearing time caused by the large non-overlapping areas in the system would quickly set each camera in the system to have the same probability to capture the target, and at this stage the tracking would fail since we are incapable of eliminating the false targets from the noisy observation data.

In this paper we will address this problem by the use of surveillance cameras from moving buses to reduce the gap between any adjacent fixed cameras. This is equivalent to adding additional sensors dynamically within surveillance space at each time stamp so that the total coverage of FOVs could increase significantly. The buses act as mobile surveillance agents that provide dynamic information during a target’s disappearing time.
period which consequently prolongs the effective predicting time and makes the tracking task more tractable. Figure 2 illustrates the integration of mobile bus cameras into the surveillance system depicted earlier.

In Figure 2, if we assume three buses $B_1$, $B_2$ and $B_3$ pass through three of the four possible paths that the target is most likely going along after it disappears from $C_4$’s FOV as depicted in Figure 1, and does not catch the target, then it is most likely that the target is going on the fourth path. Therefore, we can reset the fourth path to have the highest probability that contains the target and those three paths to have low probabilities. This enables the prior PDF to be updated dynamically to reflect this dynamic information although the target is still disappearing.

Fig 2. Integration of bus mobile surveillance agents into existing camera system

The integration of bus mobile surveillance agents into existing surveillance systems has constructed a heterogeneous surveillance system. The rest of this paper will present the detailed development of such a system in order to verify its tracking performance. It is organized as follows: Section 2 reviews the background in the area of non-overlapping tracking. Section 3 introduces our tracking methodology and Section 4 presents the tracking results. The final section concludes the research work.

2 Background
2.1 Related work

Up to now, little research has explored the use of a heterogeneous camera system that consists of fixed cameras and mobile cameras for object tracking. However, various approaches have been reported which attempt to solve part of these problems. [12] proposed a Bayesian probabilistic approach for vehicle matching across two fixed cameras on a freeway. Although the distance between the cameras was large, the vehicles moved in a relatively linear fashion which restricted its applications. [13] extended [12]’s work by scaling up to multiple sensors. Their works also focused on freeway vehicle matching. [10] proposed an optimal Bayesian solution for general tracking tasks. However, they assumed the transition models were supplied as part of the input which is difficult to achieve for a real world tracking system. Inspired by the works of [12] and [10], [9] presented a general solution for tracking without the necessity to input the
transition models. These were obtained using a defined learning strategy. The solution was only evaluated in a small environment and the real world uncertainties have not been fully addressed. Other approaches include [15] who assumed a target moved in a smooth velocity and [2] who used enough calibrated cameras. Both approaches are not favorable for general motion tracking in large scale environments.

2.2 Bayesian filtering approach
Since we intend to develop a heterogeneous surveillance system to verify its feasibility in tracking moving objects over large non-overlapping areas, we need a suitable filtering approach that has good scalability as our underlying tracking framework. As described earlier, mobile surveillance agents could bridge the gaps by dynamically providing extra FOVs in the surveillance space at each time stamp, and hence could address the problem of incomplete and insufficient data existing in surveillance systems where the non-overlapping areas are significantly large. However, we still have to deal with the noisy measurements (i.e. identifying the real target from possible targets) each time when observation arrives. This can be achieved by an appropriate filtering approach.
We selected the Bayesian filtering approach as we assume the target’s initial location is known which gives us the prior information of the target’s movement. We can then use the observations received from our heterogeneous system to update the prior and obtain the posterior probability distribution of the target at each time stamp.

In our tracking task, the desired system state is the 2D position of a target at each time stamp. In order to optimally estimate this variable, it is necessary to construct the posterior probability distribution (i.e. posterior PDF) over the current state $X_k$ at the current time $t = k$, given all observations obtained so far. This means to compute $p(X_k \mid Z_{1:k})$, where $Z_{1:k}$ represents the observation measurements made from time $t = 1$ to $t = k$. The Bayesian formula for computing $p(X_k \mid Z_{1:k})$ is shown by equations (1)-(3) below (adopted from [16] and [1]).

$$p(X_k \mid Z_{1:k}) = \alpha p(Z_k \mid X_k)p(X_k \mid Z_{1:k-1})$$

In the above equation, $p(Z_k \mid X_k)$ is the observation measurement at the current time $k$ which can be directly obtained from observational data, and $p(X_k \mid Z_{1:k-1})$ is the prior probability distribution over $X_k$ that is computed at the previous time $t = k - 1$ by the following formula:

$$p(X_k \mid Z_{1:k-1}) = \int p(X_k \mid X_{k-1})p(X_{k-1} \mid Z_{1:k-1})dX_{k-1}$$

In equation (1), $\alpha$ is the normalizing constant that makes the total state probability sum to one. It is computed as below:

$$\alpha = 1 / p(Z_k \mid Z_{1:k-1}) = 1 / \{ p(Z_k \mid X_k)p(X_k \mid Z_{1:k-1})dX_k \}$$

Equations (1)-(3) are the essential components of the Bayesian filters for posterior PDF calculation. It actually consists of two stages: prediction and update. Equation (2) is the prediction equation and equation (1) is the update equation. The prediction stage computes the prior PDF over $X_k$ at time $t = k - 1$ while the update stage updates the prior PDF using observation evidence at time $k$ and constructs the posterior PDF over $X_k$. These two stages operate recursively which is a convenient solution for many
tracking tasks as it is not necessary to store the complete data set nor to reprocess existing data if a new measurement becomes available [1].

2.3 Grid-based Bayesian filtering

However, Bayesian filters only provide a rigorous general framework for dynamic state estimation problems [8]. If the state variable is continuous, there is no general analytic (closed form) expression for the required PDF due to the use of integration in its Bayesian formulas. To tackle this problem, the simplest way is to divide the state space into discrete grid cells where each cell represents a discrete state. In this way, the continuous state variable is converted into a discrete state variable and therefore the integration can be replaced by a summation. Although this is a sub-optimal solution, it could approach the optimal state estimation if the cell size is small enough. This method is commonly known as a grid-based approach [7] and is a widely used Bayesian implementation method for location estimation [1][7]. It is also referred to as discrete piecewise constant approximation to density functions [11].

The grid partition strategy that represents a state space as a grid of points (2D or 3D coordinates) have been frequently adopted for effective surveillance tracking and target location estimation in distributed sensor networks[5]. They are also the core strategy used for wireless sensor network applications [6]. For indoor/outdoor location estimation, grid-based filters often partition the environment into small patches between 10cm to 1m in size [7]. Each grid patch contains the belief that the person or object is in that cell. The mobile-robot-localization community has shown that grid-based approximations provide accurate position estimates in combination with high robustness to sensor noise [7]. A grid-based method has been successfully applied for the position estimation by the museum tour-guide robots Rhino and Minerva [3][4].

An obvious advantage of the grid-based Bayesian filtering approach is its good scalability. It can be easily applied to large scale state space by simply dividing it into grid cells. Since we need a suitable approach that can potentially cope with large scale geographical area tracking, the grid-based filtering meets this need and hence was selected as our overall tracking framework. The next section will describe the detailed tracking methodology.

3 Tracking Methodology

3.1 The tracking environment

Figure 3 shows a simulated tracking environment used in this work (The map was obtained from [14]). It can be seen as a simple Geographical Information System (GIS) supported by a geo-rectified map. The GIS essentially stores the information of the fixed camera systems (i.e. location and footprint extend) within the map area. The locations of all buses moving within the environment are obtained by the on-board GPS receiver and displayed as moving bus icon in the environment. In this work, we assume that only forward looking cameras are used, although the environment can cope with other possible camera placement configurations. All videos are time-stamped using a common clock signal.

The People and Target objects in the environment are all created using a generic model
that assigns the location of the object and an appearance characteristic at every time interval. The system can generate any amount of evenly spaced People objects that is realistically possible. As this system is to be used for tracking a predetermined target, only one Target object is generated.

Fig 3. The simulated tracking environment used in this work.

3.2 Overview of the tracking system

Figure 4 shows the overview of our designed tracking system. The system adopts a Bayesian recursive estimation method to calculate each current state of the target by prediction and update. It comprises four major modules:

1. Image Processing Module: performs matching of an input entry with that of a reference “target” subject. The module returns a matching score between 0 and 1.0 to represent different degrees of match (0 being not a match and 1.0 represents a perfect match). The matching approach may use attributes of the target like color, texture and physical biometrics like facial recognition.

2. Prediction Module: predicts the state probability distribution for the next time stamp, given a target’s known initial state. This constructs the prior belief for the current state, or the prior PDF.

3. Update Module: uses the observed appearance probabilities to update the prior PDF and obtains the posterior PDF over the current state.

4. Target Reacquisition Module: selects the state that has the highest state probability from the posterior PDF and uses it to update the current state of the target or selects a number of states that have high state probabilities as candidate targets and manually identifies the real target and uses it to update the current state of the target. The latter is used for semi-automated tracking systems.
3.3 Grid-based filtering

We first define some notations used for this paper. Let \( k \) be a particular time that represents the current time, \( X_k \) be the random state variable at time \( k \) which is unobservable, and \( Z_k \) be the observable evidence variable at time \( k \). \( Z_k \) can be further divided into two sub-observations: \( Z_k^{\text{fix}} \) - observations from the fixed cameras and \( Z_k^{\text{bus}} \) - observations from the bus cameras. Therefore,

\[
Z_k = \{Z_k^{\text{fix}}, Z_k^{\text{bus}}\}
\]

a) Initialization

The first step is to divide the state space into grid cells. Each cell represents a state in the state space. Each cell has at least the following four parameters:

1. Cell position – represented by its center’s position.
2. Appearance probability of the target being in this cell at time \( k \).
3. Predicted state probability of the target being in this cell at time \( k \).
4. Current state probability of the target being in this cell at time \( k \).

In addition, a cell is set to be 1 if it is observable, otherwise to be 0. A cell is observable if it is in a FOV of a fixed camera or it is on a bus’s route. The unobservable cells will not participate in the grid-based filtering calculations. This could significantly reduce the total computational cost. Figure 5 shows an example of the decomposition of a state space into equal size grid cells (The map was obtained from [14]). Note that the cell size in this figure is just for illustration purpose. In real implementation, the cell size would be much smaller.
b) Appearance probability calculation

In order to evaluate the performance of the target prediction and tracking strategy as objectively as possible, we will not consider the influences of the image processing module at this stage. To achieve this, we simulate the results of the similarity cost matrix (performed in Module 1 of the system described in Section 3.2) which mimic the color matching scores of a subject and that of the reference target. We generate a random match score for an arbitrary observed person. This is an unbiased simulation since the process of assigning the match score is the same for all subjects, including the target.

c) Prior PDF calculation

The formula for prior PDF calculation is derived from the Bayesian formula for it as in equation (2). Since we have decomposed the state space into discrete cells, the integration in equation (2) can be substituted by a summation expression. Besides, the observation variable comprises two parts in our case as mentioned earlier (equation (4)). Hence, equation (2) can be rewritten as:

\[
\sum_{X} p(X_k | X_{k-1}) p(X_{k-1} | Z_{1:k-1}^{fix}, Z_{1:k-1}^{bus})
\]

As can be seen from the above equation, the prior PDF calculation comprises two parts: \(p(X_{k-1} | Z_{1:k-1}^{fix}, Z_{1:k-1}^{bus})\) which is the posterior PDF at time \(k-1\) and is known in our case, since we assume the target’s initial state is known; and \(p(X_k | X_{k-1})\) which is the state transition probability distribution from time \(k-1\) to time \(k\), which is uncertain in our case, since we do not assume the target’s state transition model is known a priori. The approach we used to compute it is introduced next.

State Transition Function

Let \(d_{ij}\) be the distance between two cells \(i\) and \(j\). We use an exponential state
transition function to model the state transition probability between two cells as below:

\[
p(X_i^t | X_{i-1}^t) = 1 / e^{-\lambda d_{ij}}
\]  

Equation (6) means the state transition probability from cell \( j \) at time \( k - 1 \) to cell \( i \) at time \( k \) decreases exponentially with the distance between these two cells. This makes sense as a person is more likely to move to its nearby cells than move to remote cells in one step. Moreover, equation (6) sets remote cells’ state probabilities to be very low but not zero which preserves their possibilities to contain the target.

**Using Bus Mobile Cameras to Overcome the Degeneracy Problem of the Prior PDF**

For non-overlapping systems, the time period that a target is not within the FOV of any camera is highly uncertain. If we only rely on a pre-defined state transition function (e.g. equation (6)) the accuracy of the prior PDF degenerates with time. If a target’s disappearing time is long, eventually all cells will have the same probabilities to have the target. At this time, the prior PDF is useless for prediction. To overcome this problem, we use the information derived from the video images of the moving bus cameras to update the prior PDF dynamically to produce better prediction results. The theory behind this is that if a bus passes through a road and does not see the target, then it is very likely the target is not traveling along that road. The system can then automatically assign a low probability for that road containing the target. Therefore, at each time stamp, if a bus camera does not capture the target, then all cells within its camera’s FOV will be set low state probabilities. After that, a cell still uses the state transition function (equation (6)) to increase its state probability exponentially with time. In this way, these moving bus cameras essentially play a role in reducing the disappearing time of a target and increasing its track continuity. The track continuity is difficult to maintain if we only rely on fixed location cameras.

d) Posterior PDF calculation and target reacquisition

When observation arrives, the posterior PDF at the current time can be calculated using the equation below which is derived from its Bayesian formula (equation (1)) and equation (4):

\[
p(X_k | Z_{1:k}) = \alpha \times p(Z^f_k, Z^{bus}_k | X_k) \times p(X_k | Z^f_{1:k-1}, Z^{bus}_{1:k-1})
\]

Same as equation (1), the above equation consists of three parts:

- \( p(X_k | Z^f_{1:k-1}, Z^{bus}_{1:k-1}) \) : The prior PDF over the current state of the target which has been computed from part c).
- \( p(Z^f_k, Z^{bus}_k | X_k) \) : The observation measurement over the current state which can be obtained directly from the camera (fixed and mobile) measurements.
- \( \alpha \) : Normalizing constant that makes the total probabilities sum to be 1.

In our real implementation, we did not do the normalization since our goal is finding the cells that have high state probabilities. In order to decide whether a cell possibly contains a target, a threshold is used (i.e. we use threshold to obtain a candidate target list). If a cell’s state probability is greater than the threshold, then this cell is likely to contain the target and will be inserted into a candidate target list. If at the end, the candidate target list is not empty, it means the target is present at the current time and the
The system will select a cell that has the highest state probability from the candidate list and use it to update the target’s current state position. If the candidate list is empty, then it means the target is not present at the current time, and the tracking process will continue predicting. For semi-automated tracking systems, a number of candidates from the list could be returned and the real target will be identified by an operator. The setting of an appropriate threshold is important as a low threshold would possibly allow many false positives to enter the list while a high threshold would possibly exclude the actual target. A range of threshold values from 0.0 to 1.0 have been tested in order to select the most suitable one. The test results are shown in section 4.

Strategies to Improve Tracking Accuracy

Note there are two situations where the cell that has the highest state probability may not contain the real target: the real target is not in the candidate list and the real target is in the list, but its state probability is not the highest. These problems are caused by the noisy observations. In either situation, the system would return a false match and hence affect the tracking accuracy. In order to overcome these problems, we applied two correspondent strategies:

1. Delay making a decision

   It is possible that during the target’s disappearing time, another object that has the similar appearance features as the target enters a camera’s FOV. In this case, the candidate target list is not empty but it does not contain the real target and the system would falsely claim the non-target person as a real target. In order to reduce this problem, we simply postpone making the decisions of who is the real target for a certain time period in order to wait for the real target to turn up and to be included into the candidate list. A range of different delay times have been tested to verify the improvement of the tracking accuracy. The results are shown in section 4.

2. Return multiple candidate targets

   Even though the real target is in the candidate list, its state probability may not be the highest due to the noisy measurements (e.g. occlusion, illumination changes etc). In this case, we resort to human interactions. The strategy is to return multiple candidate targets from the candidate list, and let the human operator decide which one is the target. We have tested a range of returned values of the number of candidates and the results will be shown in the next section.

4 Simulation results

The tracking system is tested using simulated models. A tracking interface is implemented to visualize the tracking process. The interface contains a resized road map [14], and a number of fixed cameras on major road intersections. In addition, a number of buses and people are simulated to move on it. Figure 3 is a snapshot of the tracking interface. We set 28 fixed cameras in the map. The bus transit frequency is set to be between 3 and 5 time units. This means that every 3 to 5 time units at least one bus passes through each major route. The total number of buses used for each run is 50, and the total number of persons moving in the map for each run is 100.

4.1 Data collection

On any given run, a target travels along a pre-defined path from origin to destination. During each run, the following data are computed: the total number of footprints, the total number of hits, the total number of misses and the total number of false alarms. The
meanings of these terms are explained below:

**Footprint**: the instance when the target is caught by a camera (fixed or mobile).

**Hit**: whenever the system claims a target is found, and it is the real target, we call it a hit or a true positive.

**Miss**: whenever a target’s is present, but the system fails to claim it. We call it a miss, or a false negative.

**False alarm**: the system claims the target is found, but the claimed target is not the real target. We call it a false alarm, or a false positive.

Total footprints of the target = Total hits of the target + Total misses of the target

### 4.2 Evaluation metrics

We use the **coverage** and **accuracy** measure as determined by [12] as our cost metrics. **Coverage** refers to the percent of footprints of the target observed by both fixed and bus cameras for which targets are proposed to be found, and **accuracy** refers to the percent of the proposed claims of targets that is in fact correct. They are expressed below:

\[
\text{coverage} = \frac{\text{Total number of hits} + \text{Total number of false alarms}}{\text{Total number of footprints}} \times 100\%
\]

\[
\text{accuracy} = \frac{\text{Total number of hits}}{\text{Total number of hits} + \text{Total number of false alarms}} \times 100\%
\]

### 4.3 Tracking accuracy

We first tested the tracking system using different state probability thresholds as discussed in section 3.3 part d). Table 1 shows the results.

<table>
<thead>
<tr>
<th>threshold</th>
<th>coverage</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>0.70</td>
<td>0.36</td>
</tr>
<tr>
<td>0.4</td>
<td>0.67</td>
<td>0.43</td>
</tr>
<tr>
<td>0.5</td>
<td>0.63</td>
<td>0.61</td>
</tr>
<tr>
<td>0.6</td>
<td>0.53</td>
<td>0.62</td>
</tr>
<tr>
<td>0.7</td>
<td>0.33</td>
<td>0.64</td>
</tr>
</tbody>
</table>

Table 1. Test results for tracking accuracy using different state probability thresholds

From the above table, we can see that the coverage values decrease with the increase of the threshold while the accuracy values increase with the increase of the threshold. When the threshold is 0.3, the coverage reaches 70% of the target’s total footprints; however, only 36% of them are in fact correct. On the other hand, when the threshold is 0.7, the coverage drops to 33%, but the accuracy increases to 64%. This is because the use of a low threshold would introduce more false positive targets into the candidate list, and hence the coverage would appear to be high. Since many of them are false positives, the accuracy would be low. On the other hand, a high threshold would prevent many false positives to enter the list, and hence the coverage would appear to be smaller. As
there are fewer false positives, the accuracy would be high. It seems the most appropriate threshold is 0.5 with 63% coverage and among them 61% are correct claims.

We next tested the system using the strategy of delaying the decision made for claiming the target as discussed in section 3.3 part d). The testing was done using different delay values. The state probability threshold was set to 0.5. The results are shown in Table 2.

<table>
<thead>
<tr>
<th>delay</th>
<th>coverage</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.63</td>
<td>0.61</td>
</tr>
<tr>
<td>1</td>
<td>0.62</td>
<td>0.72</td>
</tr>
<tr>
<td>2</td>
<td>0.22</td>
<td>0.66</td>
</tr>
<tr>
<td>3</td>
<td>0.14</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 2. Test results for tracking accuracy using the strategy of delaying the decision made.

The above table shows that the coverage values decreases significantly with the increase of the delay times. This is because the system postpones reports of the proposed targets during certain time stamps. More delays means there are fewer reports for the target’s footprints and the total number of recovered footprints becomes smaller. We also see that the accuracy values do not increase consistently with the increase of delay times. This may be caused by the uncertainties encountered by increasing the delay intervals. Although the real target gets more chances to enter the candidate list with the increase of time, more false positive targets may also have chances to be included in as well. The testing results show that the delay value should not be high. We can see that one time delay is preferable, as it maintains the same coverage as that when there is no delay, but the accuracy improves to 72%.

We finally tested the program by returning multiple candidate targets. The state probability threshold and delay time used were set to 0.5 and 1 respectively. The results are shown in Table 3.

<table>
<thead>
<tr>
<th>Number of candidates returned</th>
<th>coverage</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.63</td>
<td>0.72</td>
</tr>
<tr>
<td>3</td>
<td>0.75</td>
<td>0.76</td>
</tr>
<tr>
<td>5</td>
<td>0.83</td>
<td>0.90</td>
</tr>
<tr>
<td>10</td>
<td>0.91</td>
<td>0.96</td>
</tr>
</tbody>
</table>

Table 3. Test results for tracking accuracy using the strategy of returning multiple candidate targets.

From Table 3, we can see the coverage and accuracy values increase significantly with the increase of the number of candidate targets returned. When the candidate number returned reaches 10, the coverage could reached 91% of the target’s total footprints, and 96% of it are correct. These figures strongly indicate that returning multiple targets is a suitable choice for semi-automated tracking systems.
5 Conclusion

In this paper, we proposed a novel approach that integrates surveillance cameras fitted on moving transport vehicles such as buses into existing camera-based surveillance systems in order to overcome the problem of insufficient surveillance coverage experienced by the non-overlapping nature of current systems. The footage obtained from these mobile cameras produce time-varying footprints that are useful to maintain track continuities of moving targets across a wide surveillance area like in a big city. The major advantage of this approach is to enable a tracking system to better model the targets’ dynamic state transitions over time and space thus generating more accurate tracking results. The simulated test results have shown that the developed tracking system can give a high accuracy tracking result for semi-automated systems.

REFERENCES


