2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Switched Video Feedback for Sensor Deployment and Target Tracking in a Surveillance Network*

Amit Goradia†, Zhiwei Cen‡, Clayton Haffner‡, Ning Xi† and Matt Mutka‡
†*Dept. of Electrical and Computer Engineering*
‡*Dept. of Computer Science and Engineering*
Michigan State University
East Lansing, MI 48824, USA

Abstract— **Network surveillance systems provide extended perception and distributed sensing capability in monitored environments through real time monitoring of the target area and target objects using multiple networked sensors. The development of wireless communication and sensing technology make it possible to deploy networked surveillance systems in various environments. We consider a surveillance network where the sensors are static. The task of tracking targets in a surveillance network is challenging because of the following reasons: (1) The location of the sensors need to be optimally deployed. (2) the view of the sensors need to be optimized so that at a given time the targets are shown with a discernable resolution for feature identification. (3) it is important to devise stable control algorithms for accomplishing the surveillance task. When the target moves, it is important to switch the sensing task between sensors to maintain the visibility of the target with adequate resolution. This paper presents a novel method to deploy static sensors given a target region and a dynamic programming method to optimally switch sensors when the target moves. Finally, simulation results demonstrate the efficacy of the proposed approach for tracking targets over an area.**

I. Introduction

Networked surveillance systems have received much attention from the research community due to their many pervasive applications [1]. Technological advances in wireless networking and distributed robotics have been leading to increasing research on distributed sensing applications using wireless sensor networks. Infrastructure-less surveillance and monitoring are important applications of such rapidly deplorable sensor networks.

Video feedback is an essential component of the surveillance system. A single human operator cannot effectively monitor a large area by looking at dozens of monitors showing raw video output. In [2] an approach is presented which provides an interactive, graphical user interface (GUI) showing a synthetic view of the environment, upon which the system displays dynamic agents representing people and vehicles. Another program called the the Modular Semi-Automated Forces (ModSAF) program provides a 2-D graphical interface similar to the VSAM GUI [3], with the ability to insert computer-generated human and vehicle avatars that provide simulated opponents for training [4].

Although automatic image analysis and video understanding tools [2] can be used to facilitate identification of targets and activation of alarms or logs for certain surveillance tasks, the operator needs the video feedback to make decisions about the tracking task which may not have been pre-programmed or to independently task the network based on the current feedback received from the sensors.

Since multiple cameras are deployed to track the identified targets, multiple, concurrent feedback video streams maybe required for monitoring the target. These sensors initiating these streams will be changing from time to time as the target moves out of range of the current sensors tracking it. However, providing multiple unnecessary (unrelated to the task) video feedback streams often causes loss of attention span of the operator and makes it hard to keep track of the various activities over the cameras. Hence only video streams from relevant sensors should be presented to the operator on a per activity basis. This is done through automatic or manual switching of the camera streams that are presented to the operator.

In this research we focus on the problem of optimally deploying multiple sensors to maximize the ability to monitor the target location. Also given a moving target, the proposed approach can predict the motion of the target and thus dynamically allocate an optimal switching strategy among the sensors. By automatically switch between cameras, the human operator can maintain an active view of the subject.

The remaining paper is organized as follows: Section II discusses optimal control of a discrete-time finite-state as a dynamic programming problem. Section III discusses switched video feedback and provides a dynamic systems model of a switched video feedback system. It further provides examples of switched video feedback algorithms and develops an assessment metric based on the camera locations. Section V proposes a dynamic programming based method to optimally minimize switching cost and maintain the target resolution. Simulation results of the proposed approach are provided in section VI. Finally, the conclusions and discussions are provided in section VII.

II. Optimal Control and Dynamic Programming

A controlled dynamical system [5] is a system $\Sigma = [\mathcal{X}, \Gamma, \mathcal{U}, \phi]$ where,

* This research work is partially supported under NSF Grants CNS-0551464, EIA-9911077, DMI-0115355, and OCI-0334035.

- $\mathcal{X}$ is an arbitrary topological space called the state space of Σ;
- Γ is the time set, which is a transition semigroup with identity;
- $\mathcal{U}$ is a nonempty set called the control-value space of Σ; and
- The map $\phi : \mathcal{X} \times \Gamma \times \mathcal{U} \mapsto \mathcal{X}$ is a continuous function satisfying the identity and semigroup properties.

The dynamical system can also be denoted by the system $\Sigma = [\mathcal{X}, \Gamma, \mathcal{U}, f]$ where the transition function f is the generator of the extended transition function ϕ [5].

A discrete-time dynamic system is a dynamic system Σ for which $\Gamma = \mathbb{Z}$ where, $\mathbb{Z}$ is the set of integers. The discrete-time dynamic system, Σ is finite dimensional if both $\mathcal{X}$ and $\mathcal{U}$ are finite dimensional and the dimension of the system Σ is the dimension of $\mathcal{X}$.

Consider a finite dimensional discrete-time dynamic system Σ and function $b : \mathcal{X} \times \Gamma \times \mathcal{U} \mapsto \mathbb{R}_+$ that takes on non-negative real values. We can now define a trajectory cost function as:

$$\mathcal{B}(\tau, \sigma, x, \omega) = \sum_{i=\sigma}^{\tau-1} b(i, \xi(i), \omega(i)) \tag{1}$$

where, ω is a sequence of inputs and $\xi = \varphi(x, \omega)$ is a sequence of states of the dynamical system given the initial state x and a sequence of control inputs ω.

The optimization problem can be stated as: Given a discrete-time finite dimensional dynamic system Σ, a trajectory cost function $\mathcal{B}$ and a pair of times $\sigma < \tau$, and an initial state $x(\sigma)$ find a sequence of control inputs ω admissible for state $x(\sigma)$ which minimizes $\mathcal{B}$.

More generalized problems can be introduced which require the total cost to be a function of the final state or by adding constraints to the final state etc.

In order to find a control input sequence ω that minimizes the trajectory cost function $\mathcal{B}$, we could list all possible control input sequences $\omega = u_\sigma, u_{\sigma+1}, ...u_\tau$ which are elements of $\mathcal{U}$ and compute the total cost of all the trajectories generated by all $\omega \in \mathcal{U}^{[\sigma,\tau)}$. This could indeed entail a prohibitively large computation cost.

Alternately we could use the Dynamic Programming method and inductively construct the Bellman function V and the optimal control input law K, backwards in time i.e, from τ towards σ as:

$$V : [\sigma, \tau] \times \mathcal{X} \mapsto \mathbb{R}_+ \tag{2}$$

$$K : [\sigma, \tau) \times \mathcal{X} \mapsto \mathcal{U} \tag{3}$$

The Bellman function $V(s, x)$ should satisfy for any $s \in [\sigma, \tau]$, and each $x \in \mathcal{X}$,

$$V(s, x) = \min_{\omega} \mathcal{B}(\tau, \sigma, x, \omega) \tag{4}$$

and the optimal control input satisfies the condition

$$\begin{aligned} \xi(j+1) = \phi(\xi(j), j, K(j, \xi(j))),\ j = s, s+1, ..., \tau \\ \xi(s) = x \end{aligned} \tag{5}$$

for each $s \in [\sigma, \tau)$ and each $x \in \mathcal{X}$.

The computation effort required to solve this problem will be significantly lower than tabulating all the control input sequences. However storage requirements for this procedure will be significantly large.

III. Switched Video Feedback as a Dynamic System

The video feedback switch is based on the capability of the tracking cameras to visually resolve the target and discern its features i.e., the resolution of the target sustained at the cameras.

The configuration space of N cameras $\mathcal{C}$ can be defined as the collection of all the parameters of the cameras involved. That is:

$$\mathcal{C} = \{C_1, C_2, \ldots, C_N\} \tag{6}$$

where, $C_i \in \mathbb{R}^p$ is a vector containing all the intrinsic and extrinsic parameters of the camera.

Consider a monitored region as a domain $E \subset \mathbb{R}^n$ under surveillance. A finite number (N) of cameras are distributed in this region and are involved in the surveillance task of maintaining visibility of a target as it moves within the monitored region. The monitored region E can be discretized on a finite dimensional grid $\mathcal{G} = \{V_i | V_i \in \mathbb{R}^n, \text{i=1,..,g}\}$ which consists of finite number, g, of vertices V_i. Note that grid $\mathcal{G}$ can be generated from the domain E using various approximate cell decomposition methods where the cells have a pre-defined shape and size in order to achieve a certain resolution.

Consider any continuous time target trajectory in two dimensions $k_c(\tau) : \Gamma_c \mapsto \mathbb{R}^2$. For every $\tau \in \Gamma_c$, $k_c(\tau)$ provides as an output the location of the target. This continuous time trajectory can be approximated as a discrete-time finite dimensional map on a grid as

$$k(t) : \Gamma \mapsto \mathcal{U}_\mathcal{G} \tag{7}$$

where, $\Gamma = \mathbb{Z}$ and $\mathcal{U}_\mathcal{G}$ represents the finite dimensional space of the locations of the vertices V_i of the grid $\mathcal{G}$.

The dynamic sensor switching problem can be modeled as a discrete-time, finite-dimensional dynamic system. The state space of this system is the finite set of camera states defined as: $\mathcal{Q} = \{q_i | q_i = (i, C_i), i \in (1, ..., N), C_i \in \mathcal{C}\}$.

Using the above definitions we can now describe the model of a dynamic sensor switched system as a finite-dimensional discrete-time dynamic system $\mathcal{V}$ as:

$$\begin{aligned} \mathcal{V} = [\mathcal{Q}, \Gamma, \mathcal{U}, \phi] \\ \phi : \mathcal{Q} \times \Gamma \times \mathcal{U}_\mathcal{G} \times \mathcal{U}_1 \mapsto \mathcal{Q} \end{aligned} \tag{8}$$

where $\mathcal{U} := \mathcal{U}_\mathcal{G} \times \mathcal{U}_1$. Here, $u \in \mathcal{U}_1$ is the control inputs while $k(t) \in \mathcal{U}_\mathcal{G},\quad t \in \Gamma$, the sequence of grid locations in $\mathcal{G}$ that the target visits, is the reference inputs to the system. The reference inputs evolve according to a predefined function of time t and depend only on the target motion which may not be known apriori. The control inputs are used to control the trajectory of the system to ensure the target is covered and can also be used for optimization of the various metrics being used.

A. Example Switched Video Feedback Algorithm - Best Resolution

Consider a set of N cameras $\mathcal{Q} = \{(i, C_i)|i = 1, ..., N \;\; C_i \in \mathcal{C}\}$. The sensor selection strategy is dependent on the resolution sustained by the target at each of the cameras and the one with the best resolution will be selected to provide feedback to the human operator.

This sensor selection strategy takes into account only on the best resolution sustained at all the cameras which implies that the control input u, is based solely on the camera configurations $\{C_i\}$, and the target locations $k(t)$. The strategy does not depend on the current camera tracking the target i.e., sensor switching costs are not taken into account.

In order to implement this strategy in the dynamic systems model, we define a function $R_{res} : \mathcal{U}_{\mathcal{G}} \times \Gamma \mapsto \mathcal{U}_1$ which maps the locations of the target at various time instances to the space of the control input variables u based on the resolution sustained by the target at each camera. Notice that R is not a function of the current state of the dynamic system i.e., the current camera. This implies that the system does not have memory.

The above algorithm can be implemented as follows. Let q_{best} represent the camera that has the least distance to the target, i.e., under the assumption of homogenous camera sensors, q$_{best}$ can view the target with the best resolution. For a given target location $k(t)$ and time t, q_{best} can be written as:

$$\begin{aligned} & q_{best}(t) = q_i | \min_{i=1\cdots N} \text{dist}(C_i(t), k(t)), \\ & \text{visible}(q(t), k(t)) == 1 \end{aligned} \tag{9}$$

Using the definition of q_{best} the next sensor to switch to i.e., $q(t+1)$ can be written as: $q(t+1) = q_{best}(k(t+1))$.

B. Example Switched Video Feedback Algorithm - Persistent Camera

In the best resolution based video feedback algorithm, the current camera tracking the target was not taken into account in the input to the dynamic system. Switching cameras in a surveillance network can lead to the disorientation of the human operator and is also generally associated with a switching time delay. Hence we should minimize the number of switches when tracking a target. However, the target should sustain a certain resolution at the tracking camera for recognition and classification purposes.

In order to implement this strategy, define a function $R_{per} : \mathcal{Q} \times \mathcal{U}_{\mathcal{G}} \times \Gamma \mapsto \mathcal{U}_1$ which maps the current tracking camera and the locations of the target at various time instances to the space of the control input variables u. This implies that the state of the system i.e. $q(t)$ is used along with the resolution in order calculate the control input to the dynamic system.

The above algorithm can be implemented using the definition of q_{best} from previous section. The sensor selected at the next time step $q(t+1)$ can be written as:

$$q(t+1) = \begin{cases} \min(\text{dist}(C_{best}(t+1), k(t+1) + \epsilon, \text{dist}(q(t), k(t+1))), \\ \quad \text{if visible}(q(t), k(t+1)) = 1 \\ q_{best}(t+1), \text{if visible}(q(t), k(t+1)) = 0 \end{cases} \tag{10}$$

IV. Assessment Metric for Camera Deployment using Switched Video Feedback

In this section we propose a metric as a performance measure of the video feedback switching algorithm based on the configuration of the cameras.

The switching metric $M \in \mathcal{R}_+$ maps the configuration space of the video algorithm [6] (which includes the previous feedback camera) and a scalar potential field over the configuration space of the video algorithm to a positive scalar. Given the configuration of the cameras $\mathcal{C}$, the video algorithm can be thought of as a mapping from the combined space of current feedback camera and target location to the feedback camera space. The scalar potential $\varphi(p) \in \mathcal{R}_+, p \in E$ provides a relative importance to the current target location and can be chosen to bias the importance of the target locations.

$$M = \int_F \nabla \mathcal{V}(f, \mathcal{C}) \; \varphi(p) \; df \tag{11}$$

where, $f \in F \subset \mathcal{Q} \times E$ and $p \in E$. $\nabla \mathcal{V} \in [0, 1]$ is the spatial derivative of the output of the video feedback algorithm and represents at which points in the video feedback configuration space F the output of the video feedback algorithm changes. Integration of these points over the configuration space represents the number of switching surfaces present in the space E. A lower number of switching surfaces will imply that the feedback camera location does not switch a lot with the free motion of the target. The term $\varphi(p)$ is just a scalar potential which reflects the importance of the particular point $p \in E$ and can be used to bias the surveillance space E.

The switching metric M in conjunction with other metrics such as target resolution at various locations in E can be computed for a large number of randomly generated configurations and a sub-optimal solution can be derived for the placement of the cameras.

V. Optimized Target Tracking using Switched Video Feedback

Given the scenario having a large number of static cameras distributed in an environment with significant overlap of their viewing regions, the problem is to identify the minimum number of cameras required to track a given target trajectory.

We propose to use dynamic programming as an optimal control strategy in order to minimize the total number of cameras required to view the target with adequate resolution.

In order to minimize the switching cost and maintain the resolution of the target, we construct a graph based on the camera location and the visibility of the targets to the cameras, given the predicted motion of the target on the grid

$\mathcal{G}$. Figure 1 shows a part of the graph constructed for each grid point in $\mathcal{G}$ that the target traverses. All the cameras that can observe a particular grid point are listed as the possible nodes to switch to. An extra node column is added to accommodate the resolution metric as shown in the Figure 1. The switching cost between the different nodes is tabulated as '1' if the tracking sensor is switched at this grid point and as '0' if the same camera is retained.

The entire graph is constructed for all the points sequentially visited by the target on the grid $\mathcal{G}$. Note that this procedure does not mandate any contiguity requirements on the path traversed by the target in order to construct the graph. Once the entire graph is constructed, dynamic programming can be used in order to find the optimal switching sequence.

In order to construct the entire graph, it was assumed that the complete motion of the target on the grid was known. However, it is a very strict requirement to know the entire path of the target. Given the current and past observations of the target, the future trajectory of the target can be predicted. Based on this finite time prediction, the graph can be constructed for the predicted trajectory and the camera switching sequence can be calculated for the predicted trajectory. As the target location evolves, the future trajectory of the target can be predicted and a camera switching sequence can be computed based on the prediction.

This procedure enables us to extend the dynamic programming algorithm for computing the switched camera sequence to a case where the target trajectory is not know in advance but can be predicted (for a finite look ahead time) using the current and past observations of the target location. Using this trajectory prediction based algorithm a sub-optimal solution to the camera selection problem can be computed.

The procedure for constructing the graph is shown in Algorithm 1. After the graph is constructed, we use the dynamic programming method, a modified version of Dijkstra algorithm to find the optimal switching strategy (Algorithm 2). In the graph generating stage, we enumerate all the cameras that can see the current grid point with an acceptable resolution. The resolution metric is marked over each camera as a weight on the vertex. However, in order to run the dynamic programming method, it is desirable to have all the weight on the edges. Thus each camera is represented by two vertexes in the graph, with an edge connecting them that has a weight equaling to the resolution metric. We also enumerate all the cameras that can see the grid points specified by the prediction vector. Between two prediction grid points we connect the cameras based on their switching metric. If the two cameras are the same, the switching metric is set to 0, otherwise a non zero switching metric is set on the edge. In Figure 1, all nonzero metrics are set to 1 for the sake of simplicity.

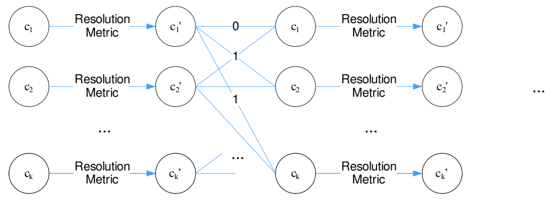

Fig. 1. Generating Graph for the Camera Switching Strategy

Algorithm 1 $[G, W] = \mathrm{GraphGen}(p_v)$

```
1: for all $cam$ in $CameraSet$ do
2:     if $p_v(1)$ is visible by $cam$ then
3:         Add two nodes representing $cam$ to $G$
4:         Connect the two nodes with the resolution metric of $cam$ in $W$
5:     end if
6: end for
7: $p_{prev} = p_v(1)$
8: Delete $p_v(1)$ from $p_v$
9: for all $p$ in $p_v$ do
10:     for all $cam$ in $CameraSet$ do
11:         if $p$ is visible by $cam$ then
12:             Add two nodes representing $cam$ to $G$
13:             Connect the two nodes with the resolution metric of $cam$
14:             Connecting the nodes generated by $p$ and $p_{prev}$ and set the switching cost to $W$ if they are for different cameras.
15:         end if
16:     end for
17: end for
```

VI. Simulation Studies

A. Sensor Deployment

Consider a switched surveillance scenario with two cameras as shown in Figure 2. Camera 1 is kept static at location $(0, 0)$ while the location of camera 2 is changed from 0-100 on both the X and Y axes. Both the cameras are directed at 45 degrees and have a viewing cone of 90 degrees. Figures 3, 4 and 5 depict the change in the three metrics namely simple distance based camera switching metric, persistent distance metric and the resolution metric, when the location of camera 2 is moved all over the grid.

Figure 3 shows the result of simple distance based metric simulation. Simple Distance switching is based on the following. If the target is being monitored by the current camera, and the distance between the target and that camera is greater than the distance between the target and some other camera, then the system will switch which camera is tracking the target, thus creating a cost. The total of the costs at each point within the grid is the total cost for a given camera configuration.

Persistence distance (Figure 4) is nearly exactly the same as simple distance, except that there is threshold related to when you switch, i.e. if the distance between the current

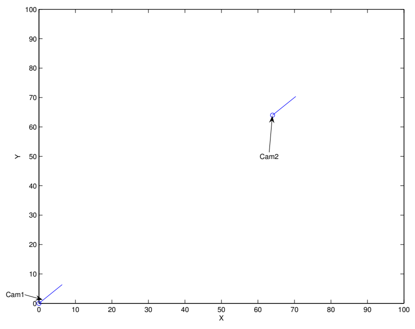

Fig. 2. Two camera setup

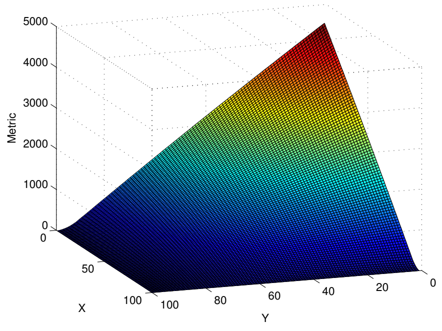

Fig. 3. The Metric Plot When Simple Distance Is Used

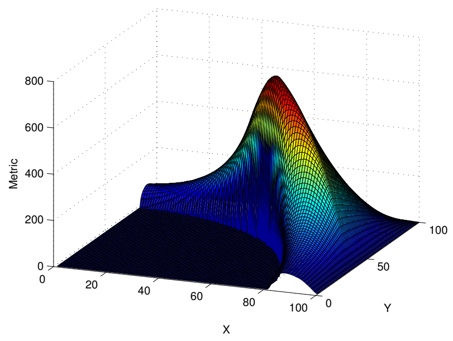

Fig. 4. The Metric Plot When Persistence Is Used

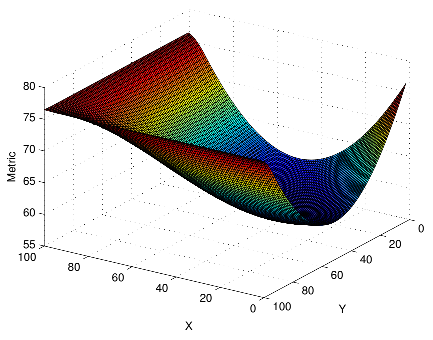

Fig. 5. The Resolution Metric Plot

Algorithm 2 $p = \text{Dijkstra}(G, W, s, t)$

```
1:  for all $v$ in $V(G)$ do
2:      $d(v) \leftarrow +\infty$ {Initialize the distance vector}
3:      $p(v) \leftarrow$ undefined {Initialize the previous node vector}
4:  end for
5:  $d(s) \leftarrow 0$ {The source has 0 distance to itself}
6:  $C \leftarrow \emptyset$ {Initialize the checked node set}
7:  $Q \leftarrow V(G)$ {Copy the vertex set into a working set}
8:  while $Q \neq \emptyset$ do
9:      $u \leftarrow ExtractMin(Q)$ {Extract the vertex with minimum value in $d$}
10:     $C \leftarrow C \cup \{u\}$
11:     if $u = t$ then
12:         return
13:     end if
14:     for all edge $(u, v)$ do
15:         if $d(u) + W(u, v) < d(v)$ then
16:             $d(v) \leftarrow d(u) + W(u, v)$
17:             $p(v) \leftarrow u$
18:         end if
19:     end for
20: end while
21: if $!t \in C$ then
22:     $p \leftarrow \emptyset$
23: end if
```

camera and the next best camera is above some threshold, then you switch and assign a cost. The main purpose of this persistence method, is to avoid a constant switching, which could lead to user disorientation. This will require that the switch will provide the user with not only a better view, but a significantly better view based on you persistence threshold.

The resolution metric (Figure 5), is based on the concept that the closer the sensor is to the target the higher the resolution is for that sensor. So for the resolution metric, the cost is the distance between the grid point and the best sensor to view that grid point. Then the total cost of a given sensor configuration is the sum of the cost at each grid point within the viewing area.

We have developed a series of metrics and used a Monte Carlo simulation method to find the optimal location of the cameras based on this metric. The simulation results of a 10 camera placement problem on a 100x100 grid is presented in this section.

The metric used to calculate the cost of the camera placement is a combination of the resolution based switching metric and a best resolution metric. The configuration of the cameras was varied and over 100,000 sets of random camera locations were generated for the Monte-carlo simulation. The Monte-carlo simulations were conducted in 10 iterations with each iteration simulating 10,000 sets of random locations. The convergence of the Monte-carlo scheme was verified by noting that the minimum cost of all the 10 batches was nearly identical.

The parameters of the cameras that were varied were the

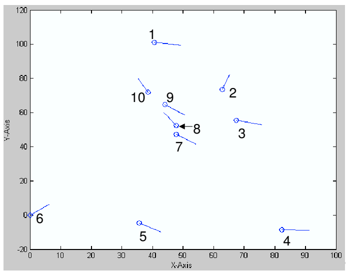

Fig. 6. Optimal camera placement for 10 cameras using Monte-carlo based optimization

X and Y position and the pointing angle θ. The viewing cone angle of the cameras was kept constant. Figure 6 shows the optimal placement of the cameras.

Based on the defined switches, we run the Monte Carlo simulation a number of iterations to obtain the best camera placement.

B. Optimized Target Tracking using Switched Video Feedback

In the simulation scenario, we deployed a set of cameras in a 100x100 grid area. A trajectory of sine wave is simulated and the switching of the cameras are captured along with the switching cost and resolution cost. In the simulation 11 cameras are placed as indicated by circles in Figure 7. The line under the camera circle indicates the facing of the camera. In the first simulation scenario, it is assumed that the whole trajectory is know at the starting point, thus a better switching strategy is given with only six switches. The number along the sine trajectory indicates that camera is switched to be the active camera at the point (Figure 7). In another more realistic scenario, we assume that the trajectory is only know for the next two grid points, and the third grid point is predicated through linear extrapolation. It is shown that more switches are needed in this case (Figure 8). However we expect less switches are needed when a more sophisticated prediction method is used. The cost associated with Figure 7 and Figure 8 are 1772.58 and 2094.47 respectively. We see a 18% improvement of the costs if we can see the whole path.

VII. Conclusion

In this paper we examined the problem of optimally placing a number of cameras in a given field to facilitate a surveillance task. We develop a series of metrics and use Monte Carlo method to obtain optimal camera placement strategies. Another problem we studied is related to camera switching strategy given a full or part of a trajectory path of a moving target. We develop a dynamic programming based approach to generate the switching strategy and optimize both the switching and resolution metrics. The simulation

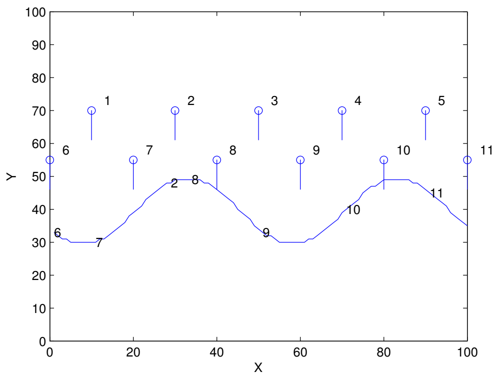

Fig. 7. Camera Switching for the Whole Path

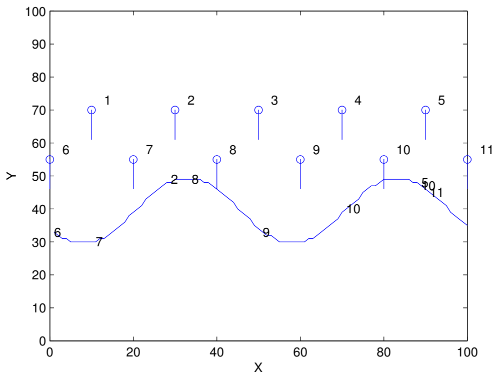

Fig. 8. Camera Switching with Trajectory Prediction

results show our method works well for surveillance tasks and scale well for large deployments.

References

[1] C. Regazzoni, V. Ramesh, and G. E. Foresti, "Special issue on third generation surveillance systems," *Proceedings of the IEEE*, vol. 89, Oct. 2001.

[2] R.T.Collins, A.J.Lipton, H.Fujiyoshi, and T.Kanade, "Algorithms for cooperative multisensor surveillance," *Proceedings of the IEEE*, vol. 89, pp. 1456–1477, 2001.

[3] R. T. Collins, A. J. Lipton, and T. Kanade, "Introduction to the special section on video surveillance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, p. 745746, August 2000.

[4] A. J. Courtemanche and A. Ceranowicz, "Modsaf development status," in *Proceeding of the 5th Conference on Computer Generated Forces and Behavioral Representation*, Orlando, FL, May 1995, p. 313.

[5] E. Sontag, *Mathematical Control Theory*. Springer-Verlag, 1990.

[6] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, December 1998.