KNIGHT^M: A REAL TIME SURVEILLANCE SYSTEM FOR MULTIPLE OVERLAPPING AND NON-OVERLAPPING CAMERAS

Omar Javed, Zeeshan Rasheed, Orkun Alatas and Mubarak Shah

Computer Vision Lab,
School of Electrical Engineering and Computer Science,
University of Central Florida

ABSTRACT

In this paper, we present a wide area surveillance system that detects, tracks and classifies moving objects across multiple cameras. At the single camera level, tracking is performed using a voting based approach that utilizes color and shape cues to establish correspondence. The system uses the single camera tracking results along with the relationship between camera field of view (FOV) boundaries to establish correspondence between views of the same object in multiple cameras. To this end, a novel approach is described to find the relationships between the FOV lines of cameras. The proposed approach combines tracking in cameras with overlapping and/or non-overlapping FOVs in a unified framework, without requiring explicit calibration. The proposed algorithm has been implemented in a real time system. The system uses a client-server architecture and runs at 10 Hz with three cameras.

1. INTRODUCTION

Urban surveillance of wide areas requires a network of cameras. One of the major tasks of a multiple-camera surveillance system is to maintain the identity of a person moving across cameras in the environment. Most of the automated surveillance approaches require overlapping field of views (FOVs) for tracking targets across multiple cameras. However, it is not always possible to have cameras with overlapping FOVs while covering large areas in urban environments. In addition, site models or calibrated cameras are not available in many situations. Furthermore, maintaining complete calibration of a large network of sensors is a significant maintenance task, since cameras can accidentally be moved.

Here, we propose a framework to reliably locate and track people and vehicles using *uncalibrated* cameras which can have overlapping and/or non-overlapping fields of view. A client-server architecture is used to implement the proposed algorithm. Workstations attached with each camera perform the single camera tracking and send the current trajectories to a central server. Initially, the relationship between the camera FOVs is learnt during a training phase, which assumes that the multi-camera correspondences are known. In the case of non-overlapping cameras the FOV's are virtually expanded so that they have an overlap in an extended image coordinate space. In the testing phase, initial detection and tracking is performed at the single camera level. As soon as an object enters the FOV of a camera it's associated client queries the server for the label. The server uses the inter-camera relationships to determine if the object is a new entry into a system or it is already being tracked by another camera. If the object is already being tracked, the server hands over the object label to the client that generated the particular query.

In the next section we discuss the related work. The single camera surveillance system is described in Section 3. In Section 4, the use of FOVs of different cameras to solve the multi-camera tracking problem is discussed. Results are given in Section 5.

2. RELATED WORK

In related work, Collins et. al. [8] have developed a system consisting of multiple calibrated cameras and a site model. The objects were tracked using correlation and 3D location on the site model. Lee et. al. [7] proposed an approach for tracking in cameras with overlapping FOV's that did not require calibration. The camera calibration information was recovered by matching motion trajectories obtained from different views and plane homographies were computed from the most frequent matches. Cai and Aggarwal [1], used calibrated cameras with overlapping FOV's. The correspondence between objects was established by matching geometric and appearance features.

Huang and Russell [2] used a combination of appearance matching and transition times (across cameras) of objects, in non-overlapping cameras with known topology, to establish correspondence. Kettnaker and Zabih [5], also used the transition times of objects across cameras for tracking. A Bayesian formulation of the problem was used to

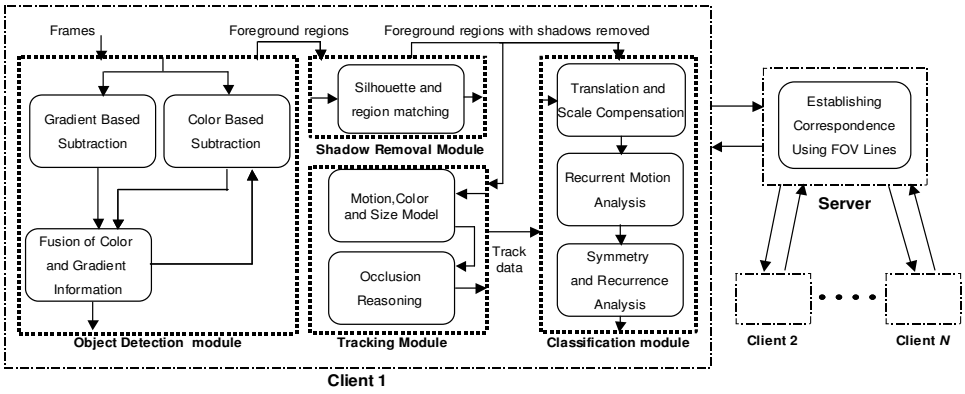

Fig. 1. Components of multiple camera tracking system.

reconstruct the paths of objects across multiple cameras. The topology of allowable paths of movement was manually given to the system.

In contrast to the above mentioned work, our proposed method combines tracking across overlapping and non overlapping cameras in a single framework. Also it does not require manual input of camera topology or calibration information.

3. SINGLE CAMERA SURVEILLANCE

The single camera system consists of the following components.

• **Advance Background Differencing:** Color based background subtraction methods are susceptible to sudden changes in illumination. Gradients of image are relatively less sensitive to changes in illumination and can be combined with color information effectively and efficiently to perform quasi illumination invariant background subtraction. The background differencing algorithm [3] performs subtraction at multiple levels. At the pixel level, statistical models of gradients and color are separately used to classify each pixel as belonging to background or foreground. In the second level, foreground pixels obtained from the color based subtraction are grouped into regions. Each region is tested for the presence of gradient based foreground pixels at its boundaries. If the region boundary does not have gradient based foreground then such regions are removed. The pixel based models are updated based on decisions made at the region level. This approach provides the solution to some of the common problems that are not addressed by most background subtraction algorithms such as quick illumination changes due to adverse weather conditions, repositioning of static background objects, and initialization of background model with moving objects present in the scene.

•**Tracking:** Each object is modeled by color and spatial *pdfs*. The spatial *pdf* is represented by a Gaussian distribution with variance equal to the sample variance of the object silhouette. The color *pdf* is approximated by a normalized histogram. Each pixel in the foreground region votes for the label of an object, for which the product of color and spatial probability is the highest. Each region in the current frame is assigned an object's label if the number of votes from the region's pixels for the object is a significant percentage, say T_p, of all the pixels belonging to that object in the last frame. If two or more objects receive votes, greater than T_p, from a region, it is assumed that multiple objects are undergoing occlusion. The position of a partially occluded object is computed by the mean and variance of pixels that voted for that particular object. In case of complete occlusion, a linear velocity predictor is used to update the position of the occluded object. This method takes care of both a single object splitting into multiple regions and multiple objects merging into a single region. The spatial and color models are updated for objects that are not undergoing occlusion. Fig. 2 shows the result of tracking under occlusion.

•**Object Classification:** People undergo a repeated change in shape while walking. Vehicles, on the other hand, are rigid bodies and do not exhibit repeating change in shape while moving. We have developed a specific feature vector called a 'Recurrent Motion Image' (RMI) [4] to calculate repeated motion of objects. We are able to distinguish between single persons, groups of persons and vehicles using this method.

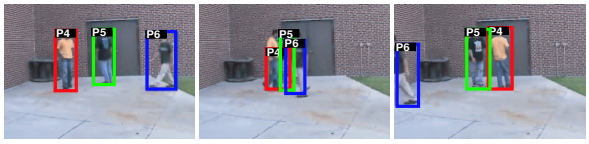

Fig. 2. Tracking results on single camera. The tracker is capable of handling multiple occluded people

4. MULTIPLE CAMERA SURVEILLANCE

We assume for multi-camera tracking that all cameras are viewing the same ground plane. In order to track people across cameras we first need to discover the relationship between the FOV's of the cameras. When the tracking is initiated there is no information about the FOV lines of the cameras. The system can, however find this information by observing motion in an environment. This 'training' phase is described in the next subsection.

4.1. Establishing Relationships between FOV's of Cameras

In order to perform consistent labelling across multiple cameras, we need to determine the FOV lines [6] of each camera as viewed in other cameras. These lines are determined during a training phase in which a single person walks in the environment.

Suppose, without loss of generality that L_l^{ij} and L_r^{ij} are the projections of the left and right FOV lines of camera C^i on camera C^j such that $i, j \in \{1 \dots n\} \wedge i \neq j$, where n is the total number of cameras. Suppose, the object being tracked in camera C^j enters or exits camera C^i from its left side. The point in C^j, at which the bottom of object touches the ground plane, actually lies on the projection of FOV of C^i on camera C^j. A least squares method is used to obtain L_l^{ij} from multiple such observations. In case of non-overlapping cameras, suppose an object exits from C^j. We keep on predicting the position of the object for a certain time interval T. The prediction is done by using a linear velocity model. If the object enters C^i from the left side, within interval T, the predicted position in C^j provides a constraint to determine L_l^{ij}. Basically, we are extending the coordinate space of C^j to obtain a virtual overlap between FOVs of C^i and C^j. Note that all correspondences are known during the training phase since there is a single object in the environment. Thus, using the above mentioned method, we can find the relationships between the FOVs of all pairs of cameras in which transition of objects is possible within time T.

4.2. Establishing Correspondence Across Multiple Cameras

The correspondence problem occurs when an object enters the FOV of a camera. We need to determine if the object is already being tracked by another camera or it is a new object in the environment. Suppose an object O enters camera C^i from the left side. Let S be the set of the cameras which contains the projection of left FOV line of C^i. Let L_l^{ij} denotes the projected line in camera $C^j \in S$. Let $\mathbf{P}$ be the set consisting of objects, that are currently visible in C^j or have exited the camera within T. For each object $P_k \in \mathbf{P}$, where k being the object's label, a Euclidean distance, $D(P_k^j, L_l^{ij})$, is computed from line L_l^{ij}. Note that the positions of exited objects are continuously updated by linear velocity prediction. If the object O is present in any C^j, its distance from L_l^{ij} should be small. Therefore the object O is assigned a label based on following criteria:

$$Label(O) = \arg\min_k \quad D\left(P_k^j, L_l^{ij}\right). \tag{1}$$

Finally the object O is given a label as described above. Note that the distance from the line only puts a spatial and temporal constraint for label assignment. If an object exits the environment while in the non-overlap area and a new person enters in the same time frame then it will be assigned the wrong label. To cater for this situation an appearance based distance measure can be combined with the distance function. An example of label assignment is shown in Fig 3.

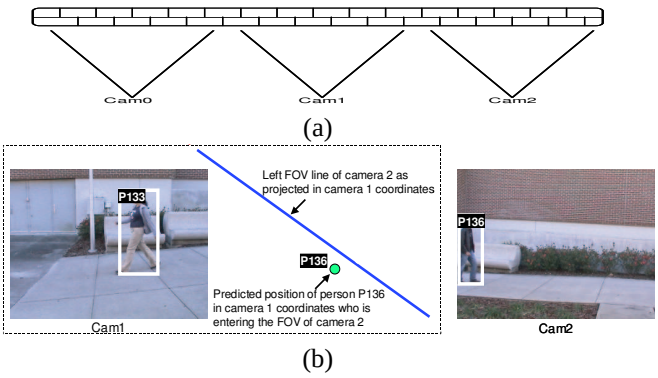

Fig. 3. (a) Arrangement of cameras. (b) Correspondence between two cameras. The left FOV line of camera 2 is shown in coordinates of camera 1. A person is entering into the view of camera 2. The predicted position of the exited person is shown with a circle in camera 1. Note that this point is very close to the FOV line of camera 2 in camera 1. By using our approach, correct label is assigned in camera 2.

5. EXPERIMENTS

To evaluate the performance of proposed algorithm, seven sequences were tested with three different camera setups. The amount of non-overlap between cameras for which our algorithm can work is bounded by our capability of predicting the position of object once they exit from the frame. After experimenting with people walking outdoors, we found out that in most situations we can reasonably predict positions for six to seven seconds. For the detection of FOV lines in each camera setup, we had a person walked around for short periods. The system was able to compute FOV lines for both overlapping and non-overlapping cases. The

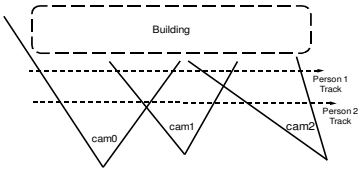

Fig. 4. The topology of cameras is shown. Note that the overlap between camera 1 & 2 is much less than the overlap between camera 0 & 1.

plan of one camera settings is shown in Fig. 4. The figure also shows tracks of two different persons. As seen from the track, person 1 was always visible as it passed though the system. Person 2 was not visible when it passed from camera 1 to camera 2. However the system was able to predict its position and correct correspondence was established. In the results shown in Fig. 6 the cameras did not have overlapping field of views. It took 3 to 5 seconds for persons to exit from camera and enter another. Fig. 5(a) shows the tracks of all people walking in the environment. There were a total of 9 tracks in individual cameras. The multi-camera tracker established correspondence and correctly recognized that there were only 3 persons in the environment. Fig. 5(b) shows the correspondence among the individual tracks for the cameras setup shown in Fig. 3. The system has been implemented in VC++ using a

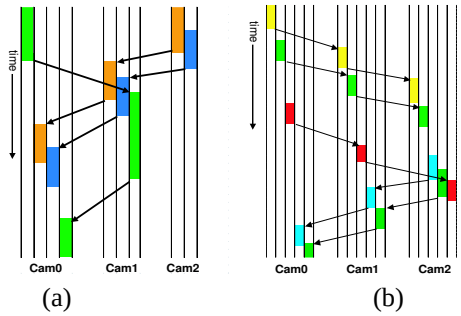

Fig. 5. Tracks of people in two different camera setups with 3 cameras. (a) A total of 9 tracks were seen. (b) A total of 15 tracks were seen. In both cases, correct labels were assigned to the tracks.

client server architecture and runs at approximately 10 Hz with 3 cameras. Each client sends track data to the server, which performs the multi-camera correspondence. The calculation intensive tasks (background subtraction, classification) are done on the client side, thus the system is scalable. For more information, visit: http://www.cs.ucf.edu/∼vision/projects/Knight/KnightM.html

Fig. 6. Tracking in multiple non-overlapping cameras. Frame 3062: Three people are seen in the cameras. Frame 3093: No person is seen in any camera as they walk through the non-overlapping area. Frame 3125: Both P101 and P103 are correctly re-assigned to people as they become visible in Cam 1. Frame 3210: Tracking continues with correct correspondences.

6. REFERENCES

[1] Q. Cai and J. K. Aggarwal. "Tracking human motion in structured environments using a distributed-camera system". *IEEE Tans. on PAMI*, 2(11), Nov 1999.

[2] T. Huang and S. Russell. "Object identification in a bayesian context.". In *Proceedings of IJCAI,*, 1997.

[3] O. Javed, K. Shafique, and M. Shah. "A hierarchical approach to robust background subtraction using color and gradient information". In *IEEE Workshop on Motion and Video Computing*, 2002.

[4] O. Javed and M. Shah. "Tracking and object classification for automated surveillance". In *Proceedings of ECCV*, 2002.

[5] V. Kettnaker and R. Zabih. " Bayesian multi-camera surveillance". In *Proceedings of CVPR*, 1999.

[6] Sohaib Khan, Omar Javed, Zeeshan Rasheed, and Mubarak Shah. "Human tracking in multiple cameras". In *Proceedings of ICCV*, 2001.

[7] R. Romano L. Lee and G. Stein. "Monitoring activities from multiple video streams: Establishing a common coordinate frame". *IEEE Trans. on PAMI*, 22(8):758–768, Aug 2000.

[8] H. Fujiyoshi R. Collins, A. Lipton and T. Kanade. "Algorithms for cooperative multisensor surveillance". *Proceedings of the IEEE*, 89(10), October 2001.