Motion Detection and Segmentation Using Image Mosaics

Kiran S. Bhat[1] Mahesh Saptharishi[2] Pradeep K. Khosla[1,2]
1. Robotics Institute 2. Electrical And Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
{kiranb | mahesh | pkk}@cs.cmu.edu

Abstract

We propose a motion segmentation algorithm for extracting foreground objects with a pan-tilt camera. Segmentation is achieved by spatio-temporal filtering of the scene to model the background. Temporal filtering is done by a set of modified AR (Auto-Regressive) filters which model the background statistics for a particular view of the scene. Backgrounds from different views of the pan-tilt camera are stitched together into a planar mosaic using a real-time image mosaicing strategy. Our algorithms work in real-time, require no user intervention, and facilitate high-quality video transmission at low bandwidths.

1 Introduction

Consider a typical video conferencing application. Participants are either stationary or moving in a controlled environment such as a room. If we can segment the foreground (non-stationary objects) from the background (stationary objects), we can achieve low bit-rate videoconferencing by transmitting the background once and continuously transmitting the foreground. Additionally, if the participants move outside the field of view, the camera should pan or tilt such that they always stay within its field of view.

A system that implements the application described above requires robust algorithms for motion detection, segmentation and tracking with a pan-tilt camera. This paper describes a motion detection algorithm that learns the background statistics of a temporally consistent scene. We automatically build an image mosaic of the background by exploring the visibility range of the pan/tilt camera initially. We continuously adapt the background model contained in the viewable subset of the mosaic. We perform motion segmentation by comparing the current camera image with its corresponding background indexed from the mosaic. After segmentation, we track the users by appropriately controlling the camera rotations. Multiple users are temporally corresponded using a novel object correspondence scheme.

The novelty of the suggested approach lies in two main areas. First, we propose a simple motion detection and segmentation algorithm that relies on low-level processing, but has the capacity to adapt itself using feedback from higher-level processes such as the classification and correspondence algorithms. Second, we propose a method for automatically constructing and indexing image mosaics by locating and tracking salient features in the scene. We have implemented the proposed algorithms for real-time motion detection, segmentation and tracking. The system requires no help from the user in carrying out its task.

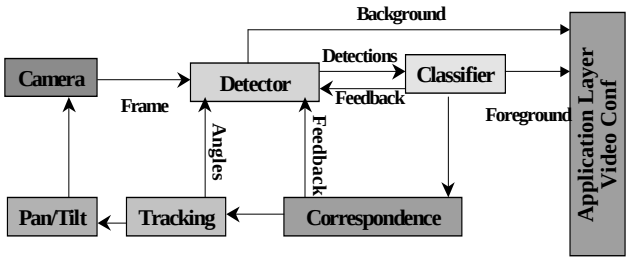


Figure 1. Modular organization of the system. Each block is part of the pipelined processing chain and executes concurrently. The arrows illustrate the information flow. The detector block is explained in sections 3 and 4.

2 System Architecture

We have developed an agent-based architecture where a series of modules can be cascaded to form a pipelined processing chain. Each module in the processing chain operates concurrently within its own thread. Figure 1 shows the connectivity of the architecture. The camera produces 320×240 8-bit grayscale images which are sent to the detector agent. The detector maintains the background mosaic and segments moving objects from the image. The classifier [1] labels the detected moving objects as “person” or “people” (group of persons). It rejects uninteresting detections such as shadows and false alarms, and feeds back this information to the detector. The detection process is described in sections 3 and 4. The correspondence agent [2] temporally associates the labelled objects. It feeds back the predicted future positions of the moving objects to the detector. The correspondence agent informs the tracker of the predicted future locations of non-stationary objects, which in turn controls the pan-tilt to track a selected object. The tracker informs the detector of the camera position for use with the mosaic indexing. The information feedback from the classifier

and correspondence agents is used to adapt the local sensitivity parameters of the detection filters. This simple feedback mechanism is extremely effective in improving the SNR, and therefore the reliability, of the detections.

3 Modeling the Background

This section addresses the problem of modeling and adapting the background contained within the current viewable subset of the image mosaic. Our basic algorithm is similar to those described in [3, 4, 5]. The algorithm described in [3] uses an IIR filter to model the background. In [4], Grimson et al. use a gaussian mixture to estimate the background of the scene. Both [3,4] use color imagery in their background modeling process. [5] describes a technique that estimates the maximum and minimum intensity differences for each pixel while there are no moving objects in the scene. This information is then used to detect moving objects. Like the system described in [5], we use grayscale imagery to estimate the background. A key difference in our work is the use of feedback from higher-level processes such as the classifier and the correspondence agents to adapt the detection process. This information feedback helps us simplify the detection algorithm while performing just as well as more complicated detection processes such as [4].

3.1 The AR Filter

We seek to model the background by using a set of AR filters to represent each pixel. Let $I_{n,x,y}$ represent a pixel at time n and at position (x, y) in a 320×240 8-bit grayscale image. Similarly, let $B_{(n-1),x,y}$ represent the predicted background value for that pixel. A significant difference $D_{n,x,y} = I_{n,x,y} - B_{(n-1),x,y}$ between the image and the background values suggests the presence of a moving object. If $|D_{n,x,y}| - b_{n,x,y} > 0$, the pixel is classified as foreground. If $|D_{n,x,y}| - b_{n,x,y} \le 0$, the pixel is classified as background, where $b_{n,x,y}$ represents the threshold for a particular pixel. At each step, we wish to gradually minimize the difference between the background model $B_{(n-1),x,y}$ and the image $I_{n,x,y}$ by using the update rule $B_{n,x,y} = B_{(n-1),x,y} + \eta D_{n,x,y}$, where η represents the learning rate constant. The update rule can also be posed as an AR filter, as shown in equation (1). Rather than making the threshold $b_{n,x,y}$ a constant, we adapt it just as we adapt the background, using an AR filter.

$$B_{n,x,y} = (1-\eta) B_{(n-1),x,y} + \eta I_{n,x,y} \qquad (1)$$

$$0 \le \eta \le 1$$

This simple AR filter-based background model is quite effective, but suffers when objects in the scene are moving too slowly. Often when the moving objects are slow, the background incorrectly acquires part of the object, resulting in false alarms. To alleviate this problem, we introduce a conditionally lagged background model. The conditionally lagged background model, $B^{cond}_{n,x,y}$ is set to the continuously updated model, $B_{n,x,y}$, if the pixel is classified as a foreground pixel. If the pixel is classified as a background pixel, then we don't update the conditionally lagged background. If after some T time steps, the magnitude of the difference between the conditionally lagged background and the image, $D^{cond}_{n,x,y}$, is less than the magnitude of $D_{n,x,y}$, the value of $B_{n,x,y}$ is reset with the value of $B^{cond}_{n,x,y}$. On the other hand, if the magnitude of $D_{n,x,y}$ is less than the magnitude of $D^{cond}_{n,x,y}$, the value of $B^{cond}_{n,x,y}$ is set to $B_{n,x,y}$. The classification of a pixel as foreground or background now depends on $|D^{cond}_{n,x,y}| - b_{n,x,y}$. When T is chosen appropriately, this technique prevents the false alarms caused by the movement of slow objects.

Given a binary map of foreground detections, connected components analysis is used to segment the moving object from the background. Prior to this step, morphological operations such as closing and erosion are performed to remove stray detections. Each positive binary value (i.e., a detection) is replaced with the actual grayscale value from the original image. Thus, the segmented image contains just the grayscale values of the moving object without the background.

3.2 The Feedback Mechanism

The feedback mechanism allows for adjusting the sensitivity parameters of the detections. This contains information about the labels (as "people" or "person") and correspondences of each detected pixel. The information feedback is used to adapt a *Perceptron* to better classify a pixel as foreground or background. The classification of a pixel as foreground depends on the inequality $|D_{n,x,y}| - b_{n,x,y} > 0$. We can redefine the classification rule as $\omega_0 - \omega_1 |D_{n,x,y}| - \omega_2 b_{n,x,y} > 0$, where each of the weights, ω_i, is adaptable. The weights are updated using the standard perceptron learning rule. If a pixel was classified as foreground and was part of a successfully labelled and corresponded object, the classification (as foreground) is considered correct. On the other hand, if the pixel was part of a rejected object, then the classification is considered incorrect. In practice we have found that adapting ω_0 alone provides a considerable increase in detection performance.

For videoconferencing applications, it is important to track foreground objects that become stationary after a while. The information feedback from the correspondence agent is used to determine which pixels represent a stationary object. A new AR filter is initialized to keep track

of the changing intensity of the pixel. This new AR filter can be visualized as an additional background layer. As long as the difference between the value of the new AR filter and the original AR filter is significant (as determined by the perceptron), the layer is maintained. This ensures that the now stationary foreground object is not regressed into the original background of the scene.

4 Mosaic Generation and Indexing

This section describes the algorithm used to construct an image mosaic from a sequence of background images. The background represented by the viewable subset of the image mosaic is continuously updated by the algorithm described in the previous section. Several techniques have been proposed to create an image mosaic from sequences of images [6, 7, 8]. They obtain the registration between images by minimizing the sum squared error of image intensities at each pixel. Although these techniques produce very accurate registration results, they tend to be slow, and typically require user interaction to initialize the registration. We create an image mosaic in near-real time (5 fps) by locating and tracking feature points in the image sequence. This technique is much faster than previous techniques, and does not require any user intervention. We also propose a method to accurately index the viewable portion of the image mosaic corresponding to a particular camera rotation.

4.1 Mosaic building

We have implemented a fully automatic algorithm to stitch together images captured by a pan-tilt camera. The algorithm uses a feature tracker to robustly identify and track feature points through the background image sequence [9]. Consider a pixel $\vec{x} = (x, y)$ in an image I that translates to $\vec{x}' = (x + t_x, y + t_y)$ in the subsequent image J. We estimate the translation $\vec{t} = (t_x, t_y)$ by minimizing the intensity error between I and J where W is a

$$E(\vec{t}) = \sum_W [J(\vec{x} + \vec{t}) - I(\vec{x})]^2 \qquad (2)$$

7x7 feature window containing $\vec{x}$. After a first-order Taylor series expansion of equation (2), we obtain equation (3), where $\vec{e} = J(\vec{x}) - I(\vec{x})$ and $\vec{g}^T = \nabla_{\vec{x}} J(\vec{x})$. This optimi-

$$E(\vec{t}) = \sum_W (\vec{e} + \vec{g}^T \vec{t})^2 \qquad (3)$$

zation problem has the least-square solution (4).

$$\left(\sum_W \vec{g}\vec{g}^T\right) t = -\left(\sum_W \vec{e}\vec{g}\right) \qquad (4)$$

The eigenvalues of the 2x2 matrix Z (defined in (5)),

$$Z = \sum_W \vec{g}\vec{g}^T \qquad (5)$$

give a good measure of the texture of the image window. In particular, we choose a feature point if the minimum eigenvalue is greater than a set threshold. We select and track N feature points (N = 80 to 100) through the image sequence. We use the coordinates of the tracked feature points to fit an affine model between the two images J and I in the sequence. The affine model is expressed in equation (6),. where $\vec{x}'$ is the coordinate of one feature

$$\vec{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax + by + f \\ cx + dy + h \end{bmatrix} \qquad (6)$$

point in image J and $\vec{x}$ is its corresponding feature point coordinate in image I. We can obtain the affine parameters by solving the overconstrained system (7) in the form of equation (8). The least square solution is given by the

$$\vec{b} = A\vec{w} \qquad (7)$$

$$\begin{bmatrix} x'1 \\ x'2 \\ \cdots \\ x'N \end{bmatrix} = \begin{bmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ \cdots & \cdots & \cdots \\ xN & yN & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ f \end{bmatrix} \quad \begin{bmatrix} y'1 \\ y'2 \\ \cdots \\ y'N \end{bmatrix} = \begin{bmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ \cdots & \cdots & \cdots \\ xN & yN & 1 \end{bmatrix} \begin{bmatrix} c \\ d \\ h \end{bmatrix} \qquad (8)$$

pseudoinverse, $\vec{w} = (A^T A)^{-1} A^T \vec{b}$.

Once the affine parameters $\vec{w} = [a, b, f, c, d, h]^T$ have been calculated, we can warp all the images with respect to a common coordinate system. For our experiments, we have arbitrarily chosen the first image as the reference, and warped all other images into the first image's coordinate system. We use a triangular weighting function (with maximum weight at the image center and zero weight at the edges) to blend overlapping regions between different warped images.

The affine transformation allows for representation of the motion of a planar surface under orthographic projection. A more accurate motion model is the perspective transformation, which is characterized by 8 parameters. However, for our application, we have found that an affine model is sufficiently accurate. Moreover, it is extremely simple to implement and is very fast.

4.2 Mosaic Indexing

Our system initially builds a background mosaic of the entire viewable environment. During videoconferencing, as the camera rotates, we index and update the corresponding viewable subset of the background mosaic to perform motion detection and segmentation. During the mosaic construction, we store the pan and tilt angles for all the

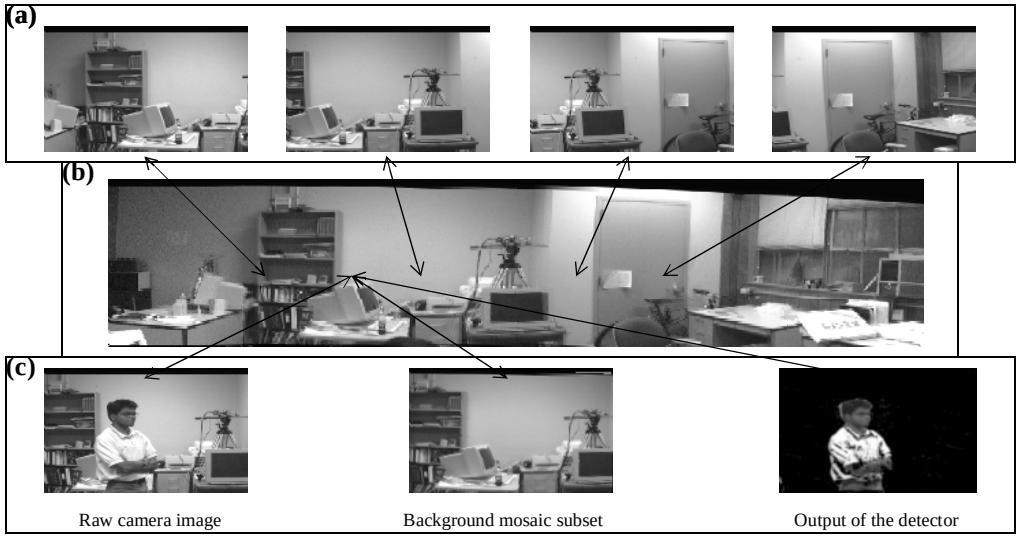


Figure 2. Results of mosaicing and detection. (b) shows the mosaic constructed using a spatial sequence of 70 images. 4 images from this sequence are shown in (a). (c) shows a moving person at a particular camera rotation, the corresponding subset indexed from the background mosaic, and the extracted foreground.

images in the sequence, in addition to their affine warp parameters. For given rotation angles (obtained from the tracking agent) we can coarsely index the mosaic using the stored affine parameters. However, due to hysteresis and other errors in the pan-tilt unit, the indexed image is offset relative to the actual camera image by around 2-3 pixels. We solve this problem by performing registration between the indexed mosaic image and the actual camera image. We approximately mask the large moving regions in both images by subtracting the images and ignoring regions with large errors. Then we solve for the translation between the two masked images using the same techniques described in equations (2-4), with the exception that W now indicates the entire masked image. Once we find the translation parameter $\hat{t}$, we can index the correct portion of the mosaic and can perform accurate motion segmentation.

5 Results and Conclusions

Figure 2(b) shows the mosaic of a room created from a sequence of 70 images, four of which are shown in Figure 2(a). Figure 2(c) shows (l-r) an image of the user, the corresponding indexed background from the mosaic and the detected foreground object. Space limitations prevent us from showing a sequence of segmented images of the user as he moves around in the room. The mosaicing algorithm initially registers images at 5Hz, and the subsequent indexing and detection algorithms execute at 10 Hz on a Pentium 266MHz laptop. As seen from the figure, the detector successfully builds a background mosaic and segments the foreground objects. The proposed algorithm for segmentation can be improved by incorporating other relevant visual cues. Our algorithms provide a powerful infrastructure for transmitting video in real time at low bandwidths.

References

[1] C. Diehl, M. Saptharishi, J. Hampshire, and P. Khosla. Collaborative surveillance using both fixed and mobile unattended ground sensor platforms. In SPIE Proceedings on Unattended Ground Sensor Technologies and Applications, vol. 3713, pp. 178-185, 1999.

[2] M. Saptharishi, J. Hampshire, and P. Khosla.Agent-Based moving object correspondence using differential discriminative diagnosis. Submitted to CVPR, 2000.

[3] A.J. Lipton, H. Fujiyoshi and R.S. Patil. Moving target classification and tracking from real time video. In IEEE Workshop on Applications of Computer Vision, pp. 8-14, 1998.

[4] W.E.L. Grimson, L. Lee, R. Romano, and C. Stauffer. Using adaptive tracking to classify and monitor activities in a site. In CVPR98, pp. 22-31, 1998.

[5] I. Haritaoglu, D. Harwood and L. Davis. W^4: Who? When? Where? What? A real time system for detecting and tracking people. In IEEE International Conference on Automatic Face and Gesture Recognition, pp. 222-227, 1998.

[6] R.Szeliski and H.Shum. Creating full view pararomic image mosaics and environment maps. Computer Graphics Proceedings, Annual Conference Series, pp. 251-258, 1997

[7] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu, Mosaic representations of video sequences and their applications. Signal Processing: Image Communication, special issue on Image and Video Semantics: Processing, Analysis, and Application, Vol. 8, No. 4, pp.327-351, May 1996.

[8] F. Dufaux and F Moscheni. Background Mosaicking for low bit rate video coding.IEEE ICIP'96, pp. 673-676, September 1996

[9] J.Shi and C.Tomasi. Good Features to Track. In CVPR94, pp. 593-600, June 1994.