SFU at TRECVid 2009: Event Detection

Weilong Yang, Tian Lan, and Greg Mori
School of Computing Science
Simon Fraser University

Abstract

This paper describes the SFU entry for the TRECVid 2009 event detection challenge. We attempted to detect events involving either one or two individuals, and present algorithms for "running", "embrace", and "pointing" event detection. The system has three stages. First, preprocessing is performed to extract candidate space-time regions using background subtraction and photogrammetric context. These candidate regions are processed with specific event detection algorithms, utilizing motion for "running" and "embrace" and shape for "pointing". A post-processing step produces temporally localized responses for evaluation. The system is computationally efficient (0.2 seconds per frame) and achieved very good minimum DCR scores on the final evaluation.

1. Introduction

In the TRECVid 2009 event detection evaluation, the objective is to automatically detect the occurances of a pre-specified set of events in surveillance videos. In our submission, we focus on detecting events that comprise one or two individuals rather than large groups of people. Hence, we develop systems for the detection of "person running", "embrace", and "pointing" events.

The dataset consists of surveillance camera footage which was accquired at London Gatwick airport. Note that this task only requires locating times (i.e. frames) at which the events occur, rather than accurate spatial locations. The dataset is large in scale – an approximately 100 hour video dataset, which consists of videos from five fixed-view surveillance cameras in the airport, has been released for development purposes. Representative frames from each camera view are shown in Fig.1.

Compared with the benchmark human action datasets in common usage in the computer vision literature (e.g. KTH and Weizmann datasets), TRECVID is much more difficult and realistic. We list the main challenges of TRECVID below.

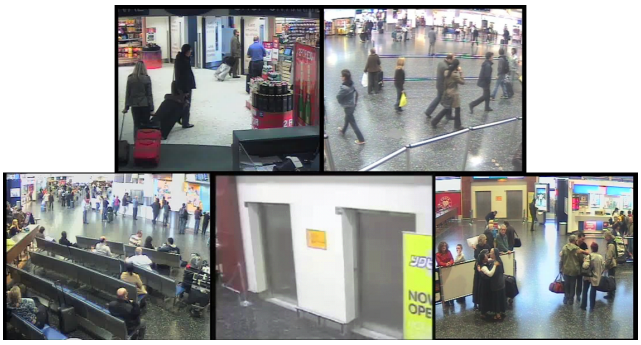


Figure 1. Representative Frames of TRECVID event detection dataset.

- **Rarity of events**: Detecting events in surveillance videos is a rare event detection problem. Finding instances of an event such as "person running" is akin to searching for a needle in a haystack. In contrast, the benchmark human action datasets formulate small scale forced choice problems with balanced categories. Each input video contains one of a small set of possible actions, with no large "background" or "other" category.
- **Cluttered background and occlusion**: The TRECVid videos all have cluttered backgrounds, which not only contain static background objects, but also dynamic ones, *e.g.* moving people. In addition, instances of events are often occluded by other moving people.
- **Action variation**: Different persons may perform the same action differently. Unlike many benchmark datasets, the actions in TRECVid are not choreographed, instead the captured actions are natural and realistic. This leads to a very large variation for the same pre-defined action. Further, actions are captured from a variety of different camera viewpoints.
- **Incomplete annotation information**: For the development videos, the annotations only provide temporal locations of the pre-defined events, and no spatial bounding box information is provided. This makes the training of a detector more difficult (bounding box

information is essential for many learning-based approaches).

2. Event Detection System

Given a pre-defined event, the objective of event detection is to temporally localize all similar events in test videos. We develop an event detection system for the TRECVid evaluation that first detects events in specific spatial and temporal locations in test videos, and then post-processes them to produce temporally localized output for evaluation. The system is based on space-time window-scanning, computing a score for the presence of each event at space-time locations. Exhaustively searching all possible space-time windows in a video is time-consuming. In order to reduce the searching space, we apply pre-processing steps using background subtraction and *photogrammetric context* to discard the majority of regions in a video. Foreground regions are detected using the standard Gaussian Mixture Model (GMM), and those which do not cover enough foreground are discarded. In our system, we use the background subtraction code provided by Zivkovic *et al*. [5]. Example results are shown in Fig. 3(b). To further reduce the searching space, we employ photogrammetric context information to roughly estimate the human height on images and determine where events are likely to occur (Sec. 3). We then apply specific event detection algorithms for each event on these candidate regions to produce spatially and temporally localized events (Sec. 4). Finally, these are post-processed to produce temporally localized output (Sec. 5).

The details of these steps are provided below. An overall flowchart view of our system is provided in Fig. 2. Note that the overall system is computationally fast. The C++ and OpenCV implementation processes videos in roughly 0.2s per frame.

3. Contextual Cues

Divvala *et al*. [1] use the term *photogrammetric context* to describe a variety of camera parameters, such as camera height, focal length, orientation, etc. Hoiem *et al*. [3] model camera parameters to reason about the relative scales of objects, which can greatly improve object detection performance. Our work is similar to [3]. In the TRECVID event detection videos, the camera view is fixed. In the absence of full camera calibration data, we learn the relationship between person height (in the image coordinates) and his distance to the camera from the development video dataset. After leaning, given the distance of a person to the camera, we can roughly estimate his height in the frame. Note that we assume all the people in videos have roughly the same height in world coordinates, and in all the five camera views, a person's height in image coordinates has a linear relationship with his distance to the bottom of the

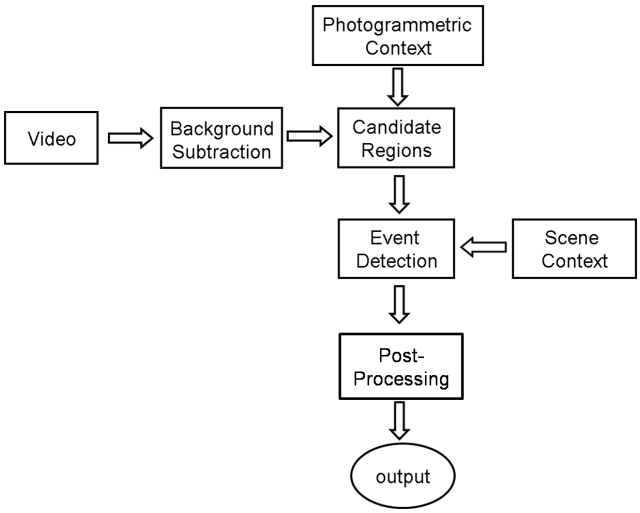
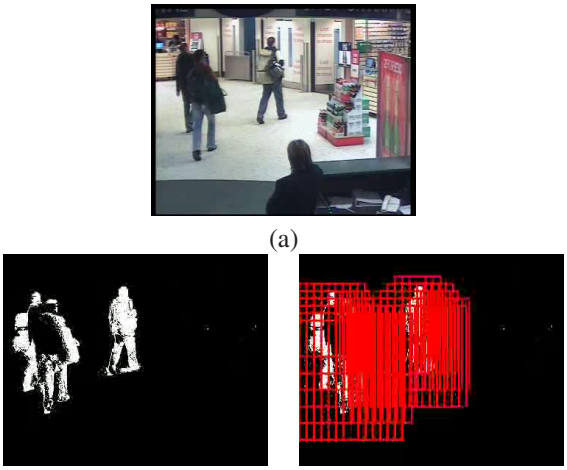


Figure 2. The general flowchart of our event detection system. Given an input video, we first extract the candidate regions by background subtraction and photogrammetric cue. Then we run the event detection on those candidate regions, whose results are further processed by the post-processing stage.



(a)

(b) (c)

Figure 3. (a) Example input video frame (b) Background substraction result (c) Candidate regions obtained using photogrammetric context and pruning based on foreground percentage.

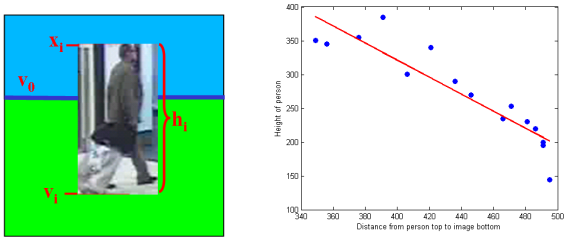
image.

In [3], object image height is estimated with the following equation:

$$h_i = -\frac{y_i}{y_c}(v_0 - v_i). \quad (1)$$

where y_i is the object height in world coordinates, y_c is the camera height, v_0 is horizontal position and v_i is object bottom position. In our experiments, we use object top position x_i to represent the distance from people to the image bottom, and Eqn. 1 can be adapted as follows:

$$h_i = -\frac{y_i}{y_c}(x_i - h_i) + \frac{y_i}{y_c}v_0. \quad (2)$$



(a) (b)

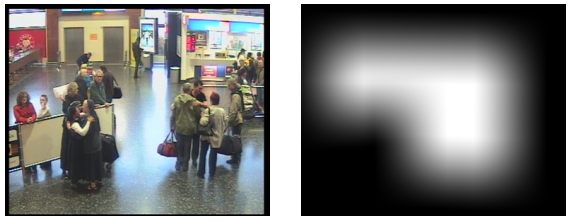
Figure 4. (a) Illustration of object image height h_i, horizontal position v_0, object bottom position v_i, object top position x_i ;(b) The linear relationship between the person image height and his distance to the camera.

From the above equation, we can see that the object's height in image coordinates has a linear relationship with its distance to the image bottom (illustrated in Fig. 4(a)), which is in accordance with our assumption. To simplify the notation, we only use two parameters to represent this linear relationship:

$$h_i = ax_i + b. \tag{3}$$

The parameters a and b are learned for each of the five camera views. In the training data, we extract some representative frames and manually label people's height in different locations of the image. Each pair of person height and location (h_i, x_i) can be expressed as a point in coordinates where x is the distance from a person to the image bottom and h is the person's height, as shown in Fig. 4(b). Then, the parameters a and b can be obtained by linear regression. Fig. 4(b) shows an example of the relationship between h and x for camera view 1, where the coordinate of each blue point corresponds to a pair of manually labeled person height and location (h_i, x_i). The same approach is used for the other camera views.

In the TRECVID2009 airport videos, each camera view represents a scene. For each scene, based on the ground truth annotations for the development dataset, we can build the confidence map for each pre-defined action. The example confidence map of camera view 3 for the embrace action is shown in Fig. 5. Because the ground truth annotations only contain the frame location of the action, without the spatial information, we manually label the spatial location of the action and then produce the confidence map. This idea of using confidence map to represent the scene contextual information was proposed in [4], here we use the same idea but build the confidence map by ourselves. For the embrace action, by our observation, it often happens in the public greeting region which corresponds to camera view 3, but rarely happens in other scenes. We simply set the confidence map all zeros for all the camera views except view 3. The experimental results show that incorporating the scene contextual information, the detection performance can be improved.



(a) (b)

Figure 5. (a) The representative frame of camera view 3 (public greeting area) (b) The confidence map of camera view 3 for embrace action.

4. Event Detection

The pre-processing steps will result in a set of candidate space-time regions which we will analyze to determine whether they contain events of interest. We develop specific event detection algorithms for each of the three event categories we consider: "person runs", "embrace", and "pointing". The first two categories contain significant motion, and so we develop algorithms based on analyzing patterns of optical flow. We used features similar to that in Efros *et al.* [2], which we briefly summarize below, before providing the details of the three detection algorithms.

4.1. Optical Flow

In this work, we use the motion descriptors proposed by Efros *et al.* [2], which has been widely used in action detection and recognition. We first compute the optical flow at each frame. The optical flow vector field F is then split into two scalar fields, F_x and F_y corresponding to the horizontal and vertical components of the flow vector. F_x and F_y are further half-wave rectified into four non-negative channels F_x^+, F_x^-, F_y^+, F_y^-, so that $F_x = F_x^+ - F_x^-$ and $F_y = F_y^+ - F_y^-$. Then, those four channels are blurred using a Gaussian kernel to obtain the final four channels Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-.

4.2. Event *running*

One of important features of the running event is that its motion magnitude is higher than other confusing actions (*e.g.* walking or jogging). Besides, the directions of the running actions in the TRECVID videos are often parallel to the camera. Based on this observation, the detection of running action is achieved by only checking the magnitude of the horizontal motion components. The output score of the running detection is computed as follows:

$$\begin{aligned} Score_{x+}^i &= \frac{\sum_{j \in C_i} fb_{x+}^j}{|C_i|} \\ Score_{x-}^i &= \frac{\sum_{j \in C_i} fb_{x-}^j}{|C_i|} \\ Score_i &= p_i \cdot \max(Score_{x+}^i, Score_{x-}^i) \end{aligned} \tag{4}$$

where, C_i denotes the candidate region at location i, fb^j_{x+} denotes the j-th element of the motion component Fb^+_x, $|C_i|$ refers to the total element number in the candidate region i. Because the optical flow computed in Section 4.1 is in image coordinates, if a running person is far from the camera, his optical flow will be very small. The ideal way to compute the optical flow for running detection is to first convert the image coordinate to the world coordinate. However, there is no camera calibration information provided. Therefore, we only multiply a parameter p_i to the output score in order to alleviate this effect of the motion inconsistency problem. p_i has a linear relationship with the distance between candidate region and camera, as in the photogrammetric context (Sec. 3). If the candidate region is close to the camera, p_i will be a small number. Otherwise, it will be large.

4.3. Event *embrace*

For the detection of the embrace action, we choose a template matching approach similar to that in [2]. We first crop some template action clips from the development videos. Given a template clip T, we slide it over all candidate regions in the test videos. The similarity between template T and the candidate clip S can be computed using normalized correlation. Suppose the four channels of the descriptor for clip S are $\mathbf{s}_1$, $\mathbf{s}_2$, $\mathbf{s}_3$, $\mathbf{s}_4$, and each channel has been concatenated to a vector. Similarly, the four channels for clip T are $\mathbf{t}_1$, $\mathbf{t}_2$, $\mathbf{t}_3$, $\mathbf{t}_4$. We denote $\hat{\mathbf{s}_k} = [s^1_k - \bar{s_k}, s^2_k - \bar{s_k}, ..., s^n_k - \bar{s_k}]$, and $\hat{\mathbf{t}_k} = [t^1_k - \bar{t_k}, t^2_k - \bar{t_k}, ..., t^n_k - \bar{t_k}]$, where $\bar{s_k}$ and $\bar{t_k}$ are the mean values of channel $\mathbf{s}_k$ and $\mathbf{t}_k$ respectively, s^i_k denotes the i-th element in channel vector $\mathbf{s}_k$. The output score for candidate clip S is computed as follows:

$$d(S,T) = C - \sum_{k=1}^{4} \frac{\hat{\mathbf{s}_k}^T \hat{\mathbf{t}_k} + \varepsilon}{\sqrt{(\hat{\mathbf{s}_k}^T \hat{\mathbf{s}_k} + \varepsilon)(\hat{\mathbf{t}_k}^T \hat{\mathbf{t}_k} + \varepsilon)}} \quad (5)$$

$$D(S,T) = w_S \cdot d(S,T) \quad (6)$$

where w_S denotes the confidence value at the location of candidate clip S.

4.4. Event *pointing*

In TRECVid event detection videos, the variation of the event pointing is very large. For example, people point while walking, some people stand still and point; the direction in which a person points is also highly variable. Therefore, the motion feature is less reliable for the pointing event. But we can observe that most of the pointing events share a common feature, that is the "pointing-out" arms are in the horizontal direction. So, instead of using the motion-based matching scheme, we choose to run a horizontal arm detector to detect the event of pointing.

In an attempt to prune out false positives, such as straight lines on clothing and the background, we apply a head detector to candidate regions first. Rather running the head detector at multiple scales, we employ the photogrammetric context again to estimate the head size. Then, for each candidate location, the detector is only run in one single scale, which can significantly improve the efficiency.

After head detection, we use an edge-based limb detector to detect the pointing event based on the locations of detected heads. Since in pointing, the horizontal arm always appears around the head, so we do not need to scan the arm detector over the edge map of the whole frame, but only the specified area around the heads. Note that the edge map is obtained from the results of background subtraction, which can help to avoid false positives of static limb shaped objects, *e.g.* bars.

5. Post-Processing

For embrace detection, we use motion based matching scheme. The system will generate one score for each test clip which has the same frame length as the template clip. This score will be directly used to generate the final score files without any post-processing. For the event of running and pointing, the system will output a score for each frame. However, running and pointing events normally lasts more than 20 frames. Thus, we merge *n* consecutive frames into one event, where *n* is the pre-defined parameter (*e.g.* 30). The maximum score among those *n* frames will be chosen as the score of this event.

6. Experiments on Development Data

To demonstrate the advantages of contextual information, we did two experiments to compare our system with the baseline system which does not use contextual information. We first give a brief overview of the dataset we will use in the experiment, then a short introduction about the baseline system, followed by the experimental results.

6.1. Dataset for Experiments

The development dataset of the TRECVID evaluation consists of 50 videos of approximately 100 hours of surveillance data, each video lasts for approximately two hours. We choose four videos from the development dataset as our test videos. Each video has a different camera view. Since the camera view with elevator does not contain the actions of running, embrace and pointing, we simply remove this camera view. A representative frame of each camera view is shown in Fig. 1. To learn the parameters a and b in Section 3, we choose another four videos to form our training

(a) (b)

Figure 6. Example results of our detector on running detection.

set. Note again that the videos in our training set and test set are different. For embrace detection, we crop out ten consecutive frames of one typical embrace action from the training set as our template.

6.2. Baseline Systems

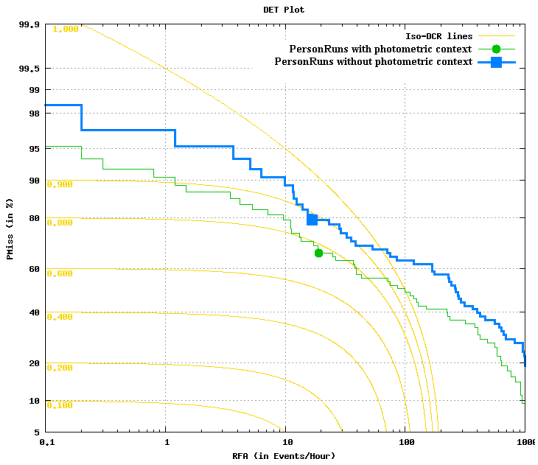
For comparison, we develop baseline systems for running and embrace detection. For the running baseline system, we remove the photogrammetric context. This context helps us to estimate the scale of the person in the frame. Therefore, after removing the photogrammetric context, we choose a standard sliding window approach with multiple scales for the running detection. Note that in the running baseline system, we do not use background subtraction. We simply scan the candidate regions with eight different scales over all locations in test videos.

For the embrace baseline system, we remove the scene context. In our system, scene context is represented by the confidence map. In the baseline system, we discard the confidence map and compute the distance between template T and the candidate clip S using Eqn. 5, without multiplying the confidence value.

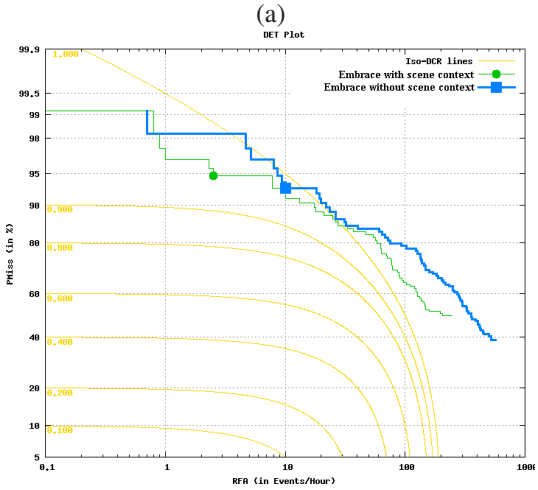
6.3. Results

After learning the parameters and cropping the embrace template from the training set, we test our system and the baseline system on the test set. The experimental results are evaluated using detection error tradeoff (DET) curves, which is a tradeoff between two error types: missed detection and false alarm. Basically, a good detection system would achieve few missed detections, as well as few false alarms. The DET curves of our system and baseline system on running detection and embrace detection are shown in Fig. 7. We also show the example results of our system on running detection in Fig. 6. We can see the red bounding boxes accurately locate running people, which not only demonstrates the effectiveness of our detection algorithm, but also the robustness of our person scale estimation.

For the comparison with baseline systems, we can see from Fig. 7 our system outperforms the baseline systems. By incorporating the photogrammetric context, there is almost a 10% miss rate decrease when the RFA $= 10$. RFA



(a)



(b)

Figure 7. DET curves of our system and baseline systems on (a) Running; (b) Embrace. The blue curve represents the performance of our system, while the green one represents the performance of the baseline system.

denotes the average number of false alarms per hour of video. It demonstrates the effectiveness of the photogrammetric context. For the embrace detection, the scene context can also improve the performance of our system. There is also a 9% miss rate decrease when RFA $= 100$. The details of performance improvements by the contextual information are summarized in Table 1 and Table 2.

	RFA = 0.1	RFA = 1	RFA = 10	RFA = 100
Context	95.2	90.5	79.4	49.2
Baseline	98.4	96.8	88.9	63.5

Table 1. Miss rate of our system and the baseline system under different RFA for running detection. RFA denotes the average number of false alarms per hour video.

	RFA = 0.1	RFA = 1	RFA = 10	RFA = 100
Context	99.1	97.4	91.2	70.2
Baseline	N/A	98.1	93.0	78.9

Table 2. Miss rate of our system and the baseline system under different RFA for embrace detection. RFA denotes the average number of false alarms per hour video.

7. Evaluation Results on the Formal Run of TRECVid 2009

The results of our system on the TRECVid 2009 evaluation formal run are shown in Table 3. We have achieved very competitive minimum DCR results on the events of running and embrace. We did not extensively tune parameters with the aim of producing low actual DCR score; our actual DCR looks relatively higher (the lower the score, the better the performance). But our system achieved very good minimum DCR scores.

Event	Actual DCR	Minimum DCR
Running	1.804	0.986
Embrace	1.053	0.998
Pointing	1.793	1.005

Table 3. Actual DCR and Minimum DCR by events on the formal run

8. Conclusion

We presented a system for the TRECVid 2009 event detection challenge. We focussed on events involving small numbers of people, and present algorithms for "running", "embrace", and "pointing" event detection. The system utilizes pre-processing to extract candidate space-time regions based on background subtraction and contextual cues. These candidate regions are processed with specific event detection algorithms, utilizing motion for "running" and "embrace" and shape for "pointing". A post-processing step produces temporally localized responses for evaluation. The system is computationally efficient (0.2 seconds per frame) and achieved very good minimum DCR scores on the final evaluation.

References

[1] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2009.

[2] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *IEEE International Conference on Computer Vision*, pages 726–733, 2003.

[3] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2137 – 2144, June 2006.

[4] P. Wilkins, P. Kelly, and C. Conaire. Dublin city university at trecvid 2008. In *TRECVID 2008 workshop*, 2008.

[5] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, May 2006.