# Surveillance Event Detection

Mert Dikmen, Huazhong Ning, Dennis J. Lin, Liangliang Cao, Vuong Le,
Shen-Fu Tsai, Kai-Hsiang Lin, Zhen Li, Jianchao Yang, Thomas S. Huang
Department of Electrical and Computer Engineering University of Illinois Urbana
Coordinated Sciences Laboratory & Beckman Institute for Advanced Sciences
405 N Mathews Ave, Urbana, IL, 61801
{mdikmen, hning, djlin, vuongle, cao4, stsai8, khslin, zhenli3, jyang29, huang}@ifp.uiuc.edu

Fengjun Lv, Wei Xu, Ming Yang, Kai Yu, Zhao Zhao, Guangyu Zhu, Yihong Gong
NEC Laboratories America, Inc.
10080 N Wolfe Road, SW-350, Cupertino, CA 95014
{flv, xw, myang, kyu, zhaozhao, gzhu, ygong}@sv.nec-labs.com

May 14, 2009

## Abstract

We have developed and evaluated three generalized systems for event detection. The first system is a simple brute force search method, where each space-time location in the video is evaluated by a binary decision rule on whether it contains the event or not. The second system is build on top of a head tracker to avoid costly brute force searching. The decision stage is a combination of state of the art feature extractors and classifiers. Our third system has a probabilistic framework. From the observations, the pose of the people are estimated and used to determine the presence of event. Finally we introduce two ad-hoc methods that were designed to specifically detect OpposingFlow and TakePicture events. The results are promising as we are able to get good results on several event categories, while for all events we have gained valuable insights and experience.

## 1 Introduction

Event detection in uncontrolled environments is critical to video-based surveillance systems, which is one of the ultimate goals of vision technologies. The challenges are mainly on two-fold: the vast diversity of one event viewed from different view angles, at different scales, and with different degrees of partial occlusions, and the demand for efficient processing of huge amount of video data, not to mention the inherent semantic gap between motion patterns and events. Thus, we strive to extract efficient image features that are pertinent to events of interests and learn multiple one-against-all classifiers to grasp the essential of individual events. As different individuals may have dramatically different appearances, the most relevant image features of events are those capable of encoding the shape and motion patterns.

In the following section (2) we discuss our three generalized approaches for event detection. In contrast to the ad-hoc approaches we developed for TakePicture and OpposingFlow event (see section 3), these general methods can be trained and used for detection of any pre-defined action categories. We report the results in terms of Actual DCR and Minimum DCR scores in section 4 and finally conclude in section 5.

# 2 Main Approaches

**Data Pre Processing**

In addition to the labels provided by NIST, which only showed starting and ending points of the events, we recorded the spatial location of several instances of five selected events (CellToEar, Embrace, Pointing, ObjectPut, PersonRuns) by drawing a bounding box of fixed aspect ratio around the person performing the action. We used this new localized annotation set as our training data for training most of our algorithms.

## 2.1 Brute Force Action Search

The first system we consider treats the problem of action detection as a retrieval problem. The hypothesis space is kept as large as possible by considering every fixed sized space-time entity with significant foreground as a candidate for one of the action sought after. We exhaustively search over all possible space-time locations in the video over a range of scales. From every candidate space-time window, motion descriptor features are extracted, which have been shown to have good performance in similar tasks [4]. Then the distance to every single example in the database is measured. If there is an action in the database within the $R\text{-}neighborhood$ of the candidate window, we keep it and it is considered for detection in the next step, otherwise it is pruned out. The last step in the detection process is mean shift clustering. This is based on the assumption that, if there is a true instance of an action where a likely candidate has been found, there will be multiple detections with slight shifts in space and time around the candidate. Thus, through clustering of the candidate points we can obtain a more robust action detection process. See figure 1 for an overview of our system.

### 2.1.1 Candidate Region Selection

The evaluation video is exhaustively searched over all possible scales and locations. In this system the length of the action in time is fixed to be 30 frames long. This is consistent with the median length of most events provided in the annotations, and recently it has been argued [14] that a short snapshot of an action can be discriminative enough to distinguish it from everything else. We have also modelled the spatial scales of actions with respect to their vertical location in the frame. It can be deduced that there is a clear linear relationship between the spatial locations of the actions and their possible sizes. Therefore when exhaustively searching for the candidate regions, we only consider scales that were observed in the training set. We further prune out candidate regions that do not have enough foreground in them. This is done by estimating the scene foreground using Zivkovic et al.'s background subtraction algorithm using improved Gaussian Mixture Models [17].

### 2.1.2 Feature Extraction

We have tested several shape and flow descriptors [9, 3, 4] and have concluded that the generality and performance (both speed and accuracy) of Efros et al.'s [4] motion descriptor is the most suitable for this task. The feature extraction process can be described as follows. First the optical flow between consecutive frames is calculated in grayscale. In our system we have used Lukas-Kanade method for estimating the optical flow in horizontal and vertical directions. The horizontal and vertical channels are further divided into their respective positive and negative components, which eventually gives four images (i.e. 1. positive flow in horizontal direction, 2. negative flow in horizontal direction, 3. positive flow in horizontal direction, 4. negative flow in horizontal direction). These four images are resized to (7x7) images by linear interpolation. We further downsample in time domain, in which we extract these motion features only at every 6'th frame. Thus every candidate window eventually yields a $7 \times 7 \times (\frac{30}{6} + 1) \times 4 = 1176$ dimensional feature vector.

### 2.1.3 Matching

Each candidate window is compared with examples by measuring the Euclidean distance between the features of the current candidate window and the features of the events in the development set. Note that at this stage metric training can be considered for added accuracy [15]. However we intended this approach to be a baseline system, with as little *human in the loop* training as possible. Another consideration is while learning strategies are ultimately very rewarding in detection problems where the object has very distinct characteristics (e.g. faces),

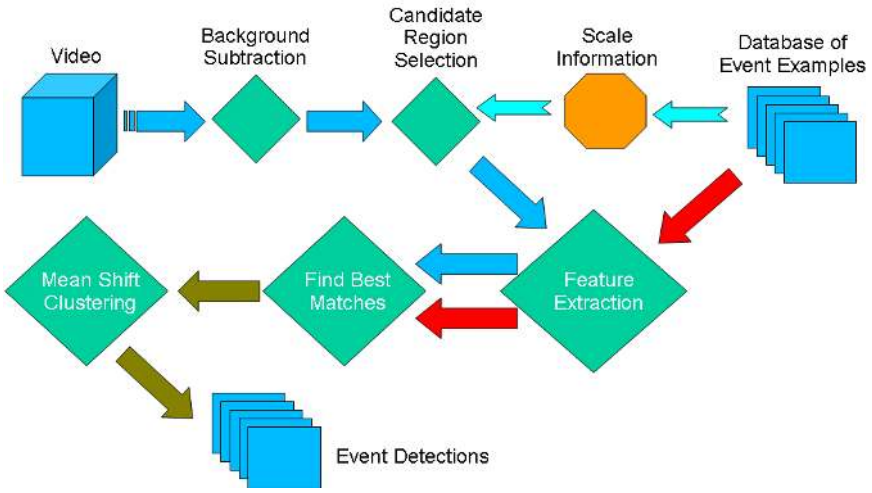


Figure 1: The system diagram for event detection by brute force search.

their efficiency in the context of event search is still an open question due to the large variability in the appearance individual events. After measuring finding the nearest neighbors, we only keep the candidate windows which have a neighbor in the development set, whose distance is less than $R$. The value of $R$ has been determined in the training stage such that only about 1% of the candidate windows would be retained after thresholding.

Finally the remaining candidate regions after thresholding, will be clustered in 3D $xyt$-space. Our system outputs each cluster center with the number of cluster members as the confidence measure for event detection. We have used popular mean shift [2] as the clustering method of choice. We employed a uniform 3D rectangular kernel (box kernel) for efficient implementation. Our submissions included three versions of this system each differing in kernel size. In the results section we report the results of the best performing kernel which was the smallest one (10, 10 and 1 in $x$, $y$ and $t$ directions respectively).

#### 2.1.4 Discussion

The results of our brute force searching method serves as a reliable baseline for more sophisticated approaches. It is interesting to observe that the simple nearest neighbor finding can produce competitive results in several cases. This can be attributed to the strength of the feature descriptor as well as the wide range of variations in event appearances, where nearest neighbor has a clear advantage due to the fact that there is no explicit or implicit modeling of event appearances.

### 2.2 Action Detection with a Tracker

Our second system mainly follows the framework of hypothesis generation, feature extraction, and classification. The candidate regions are generated based on human detection and tracking which can significantly reduce the solution space. Then, we aim to detect events of two categories: 1) events that require understanding of the articulated body motion of a single person, such as *CellToEar*, *ObjectPut*, and *Pointing*; 2) events that can be revealed by the moving trajectories of a single person, such as *OpposingFlow* and *ElevatorNoEntry*. For the first category, we combine three different classification methods based on bags of interest points and motion features, which will be elaborated in the following sections. For the second category, we apply rule-based classifiers on locations and trajectories. In the post-processing stage, the frame-based classification results are linked to event segments by heuristics. The system diagram is illustrated in Fig. 2.

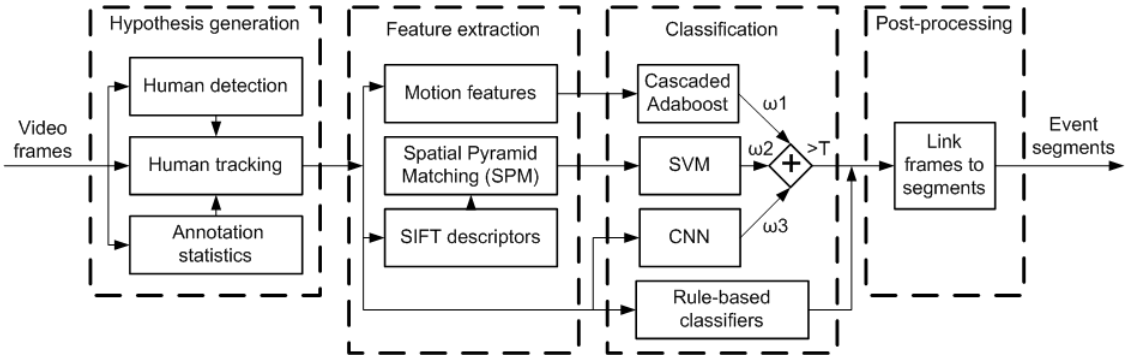


Figure 2: The system diagram for human event detection.

#### 2.2.1 Human detection and tracking

We apply a Convolutional Neural Network (CNN) [8] to detect human heads in an image and than track multiple human [5] by fusing color and contour information [1, 16]. In general, up to 30 candidates regions are evaluated for each frame. Some typical human detection and tracking results for different camera views are shown in Fig. 3.

#### 2.2.2 The combination of three learning methods

Given the human detection and tracking results, to detect the events that requires understanding of articulated body motion, *i.e.*, *CellToEar*, *ObjectPut*, and *Pointing*, we combine three machine learning algorithms: a cascaded Adaboost classifier based on motion features, an SVM classifier based on spatial pyramid matching [7] of a bag of interest point descriptors [10], and a CNN classifier based on raw images.

The motion feature extraction for the Adaboost classifier is explained as follows. For consecutive frames, we first calculate the frame difference images which only retain the motion information, and then we perform Canny edge detection to make the observations cleaner. The motion edges are accumulated to a single image with a forgetting factor. This is a tradeoff between retaining all relevant information and efficient processing. On one hand, this approach preserves some temporal shape information, on the other hand, to analyze one image is much computationally cheaper than analyzing spatio-temporal volumes. Afterwards, we extract Haar-like features from the accumulated motion edge image based on the detected or tracked human heads and train a cascaded Adaboost classifier. One example of the feature extraction process is illustrated in Fig. 4.

The spatial pyramid matching (SPM) [7] of a bag of interest point descriptors demonstrates superb performance in object and scene categorization due to its power to delineate the local shape patterns. However, the original spatial pyramid matching feature is extracted from a figure in a single frame. The occurrence of an event is a temporal process, so it is unable to capture the comprehensive event character without considering the temporal information. Therefore, we improve SPM features by incorporating temporal information. As shown in Fig. 5, after extracting the dense SIFT features [10], we construct the original SPM features from a single human figure in one spatial-temporal cube at each frame. The spatial-temporal cube is defined as the aggregation of the regions in the successive frames which are along the temporal axis with the same image coordinates *w.r.t* the base human figure. Then, for each cube two statistical features, *i.e.* the mean and the difference-power of the SPM features, are calculated and fed to the SVM learner.

The CNN classifier is trained based on raw images in a single frame given the human detection and tracking results.

For each candidate region, the classification confidences of aforementioned three classifiers are linearly weighted combined. If the combined confidence is larger

Figure 3: Sample human detection and tracking results for camera 1,2,3,5.

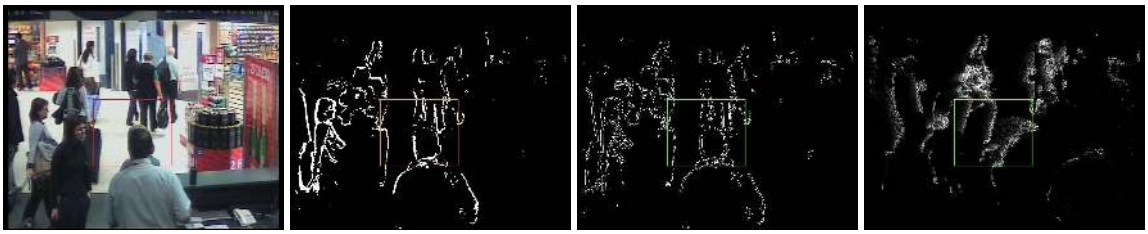

Figure 4: Illustration of the motion feature extraction. From left to right: (a) the original frame, (b) the frame difference image, (c) Canny edge detection, (d) accumulated motion edge image

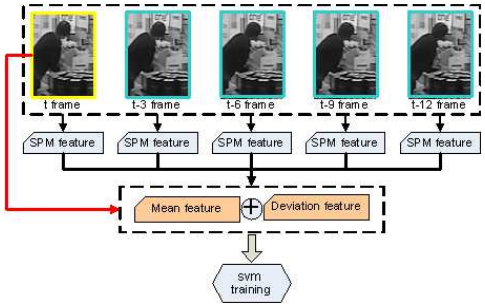


Figure 5: Illustration of the improved SPM feature.

than a threshold $T$, this frame is regarded as positive. The weights $\{\omega_1, \omega_2, \omega_3\}$ and the threshold $T$ are determined by cross-validation on the development set. The frame based results are linked to generate the event segments by heuristics considering the spatial and temporal smoothness and consistency.

#### 2.2.3 The Rule-based method

For the events *OpposingFlow* and *ElevatorNoEntry*, the location and trajectory are sufficient to reveal their occurrences. Given the human detection and tracking results, we train rule-based classifiers utilizing the location, velocity, orientation, and trajectory. The parameters and the rules are determined by the cross-validation on the development set.

#### 2.2.4 Discussion

From the 5-fold cross-validation results on the development set, we observe that the false positives rates are still fairly high. A considerable portion of the false positives appear similar in terms of the motion patterns, *e.g.* touching hair is occasionally misclassified to *CellToEar* and it is very hard to distinguish between *ObjectPut* and *ObjectGet*. The majority of the false positive are induced by cluttered background, the occlusions in a crowd, and the complicated interactions among people.

The combination of three classification methods outperforms individual ones. However, the combination

weights vary dramatically *w.r.t* different events in different cameras, which indicates that the performance and the generalization ability are not stable. Moreover, the heuristics in the post-processing stage are not trivial and also play an important role in determining the final performance.

## 2.3 Action Detection using Latent Pose Conditional Random Fields

In our third system, we consider only three events: *CellToEar*, *ObjectPut*, and *Pointing*. And three kinds of event classifiers are trained on the development set: CRF (*conditional random field*) [6], LDCRF (*latent dynamic CRF*) [11], and LPCRF (*latent pose CRF*). CRF and LDCRF are two existing models and LPCRF is our own work (the details are given below). More specifically, we first use the NEC tracker (section 2.2.1) to obtain the motion trajectories for the human in the scenario. Then for each considered event, we trained three classifiers (CRF, LDCRF, and LPCRF), using the manually labeled events in the development set as positive samples. The negative samples are randomly selected from tracking trajectories. Then for each camera and each event, we choose the best classifier. The best classifiers take the motion trajectories in the evaluation set as input and decide the occurrences of the corresponding events.

Fig. 6 gives the graphical structures of three models: CRF, LDCRF, and LPCRF. Much like a Markov random field, a CRF is an undirected graphical model in which each vertex represents a random variable whose distribution is to be inferred, and each edge represents a dependency between two random variables. In a CRF, the distribution of each discrete random variable in the graph is conditioned on an input sequence. The LDCRF model [11] incorporates hidden state variables into the traditional CRF to model the sub-structure of human actions, and combines the strengths of CRFs and HCRFs [13] to capture both extrinsic dynamics and intrinsic structure. Interested readers are referred to [6, 11].

### 2.3.1 The Model

Our latent pose conditional random fields (LPCRF) model is a generalization of CRF and LDCRF. Fig. 6 illustrates its graphical structure. The latent pose estimator learns to convert an observation vector $\mathbf{x}$ into a more compact and informative representation $\mathbf{y}$, and the model recognizes human actions based on the pose sequence $Y = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n\}$. Later we denote the latent pose estimator as $P(\mathbf{y}|\mathbf{x}, \Theta)$ in probabilistic form or $\mathbf{y} = \Psi(\mathbf{x}, \Theta)$ in deterministic form, where $\Theta$ is the set of parameters of the latent pose estimator and is jointly optimized with the random fields using a gradient ascent algorithm.

### 2.3.2 Formulation of Our LPCRF Model

Our model is defined as

$$P(\mathbf{z}|X, \Omega) = P(\mathbf{z}|Y, \Phi) = \sum_{\mathbf{h} \in \mathcal{H}_{\mathbf{z}}} P(\mathbf{h}|Y, \Phi) \qquad (1)$$

where $Y = \Psi(X, \Theta)$ is the optimal estimation of the latent pose estimator given observations $X$ and parameters $\Theta$, and $\Omega = \{\Phi, \Theta\}$ represents all the model parameters. The joint distribution over the hidden state sequence $\mathbf{h}$ given $Y$ still has an exponential form

$$P(\mathbf{h}|Y, \Phi) = \frac{\exp\left(\sum_j V_\Phi(j, h_j, Y) + \sum_j E_\Phi(j, h_{j-1}, h_j, Y)\right)}{K_\Phi(Y, \mathcal{H})}, \qquad (2)$$

where $K_\Phi(Y, \mathcal{H})$ is the observation dependent normalization.

If the parameters $\Theta$ for the latent pose estimator are fixed, our LPCRF model collapses into an LDCRF model. If each class label $z \in \mathcal{Z}$ is constrained to have only one hidden sub-action, *i.e.*, $|\mathcal{H}_z| = 1$, the LDCRF model further collapses into a CRF model. Hence, our LPCRF model is a more general framework of CRF and LDCRF. However, our LPCRF model is essentially different from both CRF and LDCRF in some aspects. In our model, input features used by the random fields are trainable and are jointly optimized with the random fields, while in CRF and LDCRF, the input features are fixed and cannot be tuned for the given recognition task. The latent pose estimator encodes the knowledge of multimodal image-to-pose relationship and provides optimal feature representation for action recognition. This knowledge can be acquired from existing well-trained models (if available) and adapted for action recognition in the learning process.

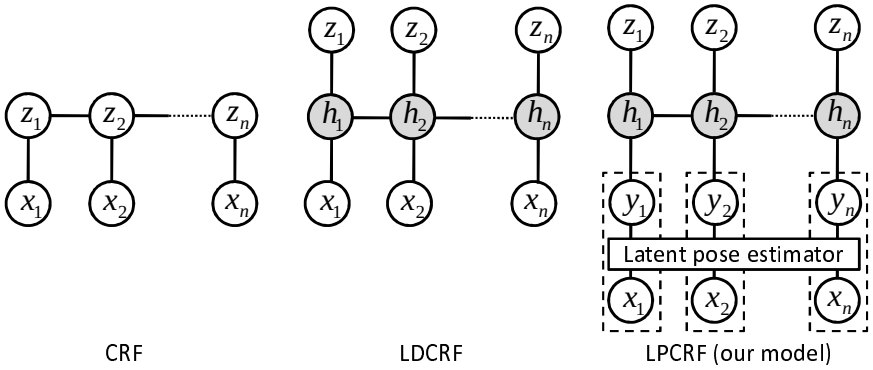


Figure 6: Graphical structures of our LPCRF model and two existing models: CRF [6] and LDCRF [11]. In these models, $\mathbf{x}$ is a visual observation, $z$ is the class label (e.g., walking or hand waving) assigned to $\mathbf{x}$ , and $h$ represents a hidden state of human actions (e.g., left-to-right/right-to-left walking). The subscripts index the frame number of the video sequence. In our LPCRF model, the observation layer of the random fields is replaced with a *latent pose estimator* that learns to compress the high dimensional visual features $\mathbf{x}$ into a compact representation (like human pose) $vy$. Our model also enables transfer learning to utilize the existing knowledge and data on image-to-pose relationship. The dashed rectangles means that $\mathbf{y}$'s are technically deterministic functions of $\mathbf{x}$ when the parameters of the latent pose estimator are fixed.

In all, the latent pose estimator is seamlessly integrated and globally optimized with the random fields.

The model parameters $\Omega = \{\Phi, \Theta\}$ are learned from training data consisting of labeled action sequences $(X^{(t)}, \mathbf{z}^{(t)})$. The labeled image-to-pose data $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$, if available, can also be utilized as auxiliary data. The optimal parameters $\Omega^*$ is obtained by maximizing the objective function:

$$L(\Omega) = \underbrace{\sum_t \log P(\mathbf{z}^{(t)}|X^{(t)}, \Omega)}_{L_1(\Omega)} \underbrace{-\frac{1}{2\sigma^2}\|\Phi\|^2}_{L_2(\Omega)} + \underbrace{\eta \sum_t \log P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \Theta)}_{L_3(\Omega)} \quad (3)$$

where the first term, denoted as $L_1(\Omega)$, is the conditional log-likelihood of the action training data. The second term $L_2(\Omega)$ is the log of Gaussian prior $P(\Phi) \sim \exp\left(-\frac{1}{2\sigma^2}\|\Phi\|^2\right)$ with variance $\sigma^2$ and it prevents $\Phi$ from drifting too much. And the third term $L_3(\Omega)$ is the conditional log-likelihood of the image-to-pose training data. $\eta$ is a constant learning rate. Note that our model enables the image-to-pose data to be naturally added to the learning process.

### 2.3.3 Discussion

Our LPCRF model can bridge the gap between the high dimensional observations and the random fields. This model replaces the observation layer of random fields with a latent pose estimator that learns to convert the high dimensional observations into more compact and informative representations under the supervision of labeled action data. The structure of our model also enables transfer learning to utilize the existing knowledge and data on image-to-pose relationship.

Our model works better than CRF and LDCRF for some cameras, while the latter works better for other cameras. So in the testing stage, we dynamically choose the model according to video scenarios.

# 3 Event Specific Approaches

## 3.1 Take Picture

The training data provided included instances of people taking picture where the camera flash was activated. This is an indoor environment and the shots being taken mostly were compositions of several people. Under these imaging conditions it is reasonable to expect that the hand held cameras will produce a burst of flash. To look for picture taking events we utilize a flash detector. Our flash detection algorithm looks for an substantial increase in the number of pixels in the top portion of the red channel histogram. Formally:

$$FlashScore_t = \sum_{k=200}^{255} (RedHistogram_t(k) - RedHistogram_t(k)) \quad (4)$$

where the range of $k$ covers the brightest pixel values.

## 3.2 Opposing Flow

Under the setting of provided data, there are three doors in the view of Camera 1 through which people can walk through in the wrong direction. We have performed our experiments on the rightmost and center doors on Camera 1, which had all the instances of opposing flow in the training data. In our basic approach we selected a rectangular region, with its center about the shoulder height of an average person and performed continuous 3D filtering with our space-time Gabor filter [12], which was specifically tuned to detect right-to-left motion patterns.

$$\begin{aligned} G(x,y,t) = & \exp\left[-\left(\frac{X^2}{2\sigma_x^2}+\frac{Y^2}{2\sigma_y^2}+\frac{T^2}{2\sigma_t^2}\right)\right] \\ \times & \cos\left(\frac{2\pi}{\lambda_x}X\right)\cos\left(\frac{2\pi}{\lambda_y}Y\right) \end{aligned} \quad (5)$$

$$\begin{aligned} \begin{pmatrix} X \\ Y \\ T \end{pmatrix} = & \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix} \\ \times & \begin{pmatrix} \cos\omega & 0 & -\sin(\omega) \\ 0 & 1 & 0 \\ \sin(\omega) & 0 & \cos(\omega) \end{pmatrix} \begin{pmatrix} x \\ y \\ t \end{pmatrix} \end{aligned} \quad (6)$$

here $\omega$ and $\theta$ determine the 3D orientation of the filter in space-time and $\lambda$ determines the effective support of the filter. We empirically evaluated several combinations of parameters and chose the best performing set for final method.

The response of the space-time Gabor filter is averaged over the rectangular regions corresponding to both doors and thresholded for detection only when the door is in an open state. We learn the models for door open/close states by clustering the average value of a $5 \times 5$ region on the top left corner of each door using the Expectation Maximization algorithm.

# 4 Results

We have submitted results for individual runs of the systems, as well as 3 combined results. When combining the outputs of the event detectors, we used a simple weighted combination scheme. Each system is assigned a weight according to their relative strength and for each frame we multiply this weight by the confidence output of the detector for that particular frame. If the weighted combination of detector confidences for a frame is above a given threshold we deem that the frame has the event. The results of our systems can be seen in tables 1 and 2

It can be seen that the first system has a high Actual DCR score as there was no tuning of the confidence measure. In terms of minimum DCR scores, all systems become competitive. This may be attributed to the fact that all systems perform their best with either one false positive or one true detection output. However it is encouraging to note that some systems were able to get a DCR score below 1.0 with proper tuning of the confidence parameter. The OpposingFlow detection was quite reliable with false positives only being produced when a person walks right-to-left in the shopping area and is tall enogh to occlude one of the doors. The detector for TakePicture

| Actual DCR | Brute Force Search (1) | Tracking & Detection (2) | LPCRF (3) | (1) & (2) Combined | (1) & (3) Combined | (1),(2) & (3) Combined |
|---|---|---|---|---|---|---|
| CellToEar | 1.3689 | **0.9985** | 1.0258 | 1.0166 | 1.0080 | 1.0092 |
| ObjectPut | 1.2537 | **1.0044** | 1.0437 | 1.0208 | 1.0094 | 1.0099 |
| Pointing | 1.2205 | **1.0029** | 1.0902 | 1.0440 | 1.0429 | 1.0293 |
| OpposingFlow | **0.4296** | 0.7632 | | 1.0000 | 1.0000 | 0.4296 |
| TakePicture | **0.9577** | | | 1.0000 | 1.0000 | 0.9577 |
| PersonRuns | **1.0019** | | | 1.0089 | 1.0000 | 1.0089 |
| Embrace | **1.4042** | | | 4.0653 | 1.0000 | 4.0653 |
| Elevator No Entry | | NA | | | | |

Table 1: Actual DCR Scores by method and event

| Minimum DCR | Brute Force Search (1) | Tracking & Detection (2) | LPCRF (3) | (1) & (2) Combined | (1) & (3) Combined | (1),(2) & (3) Combined |
|---|---|---|---|---|---|---|
| CellToEar | 1.0284 | **0.9971** | 0.9986 | 0.9978 | 1.0012 | 0.9987 |
| ObjectPut | 1.0019 | **0.9993** | 1.0020 | 1.0037 | 1.0036 | 1.0013 |
| Pointing | **1.0007** | **1.0007** | 1.0055 | 1.0000 | 1.0014 | **1.0005** |
| OpposingFlow | **0.4268** | 0.7632 | | 1.0000 | 1.0000 | 0.4237 |
| TakePicture | **0.9577** | | | 1.0000 | 1.0000 | 0.9577 |
| PersonRuns | **1.0019** | | | 1.0089 | 1.0000 | 1.0089 |
| Embrace | **1.0046** | | | 1.0124 | 1.0000 | 1.0124 |
| Elevator No Entry | | NA | | | | |

Table 2: Minimum DCR Scores by method and event

event performed much below expectations because of the fact that in the evaluation set there were many instances of people taking picture with cameras that didn't fire a flash.

# 5 Conclusions

Event detection in video is an emerging application area. Literature on this subject is advancing fast and existing test datasets are quickly being rendered too easy. Trecvid surveillance event detection task is an interesting challenge to test the applicability of such algorithms in a real world setting. Based on our key observation that there is a wide variety in the appearance of the event types, we have implemented and tested various algorithms and features to detect eight of the ten required event categories. The results reflect the magnitude of the difficulty of the problem at hand, while we believe we have gained much insight to the practical problems, and future evaluations have the potential to produce much better results.

# References


[1] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'98)*, pages 232–237, Santa Barbara, CA, Jun 23-25 1998. 4

[2] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603 – 619, May 2002. 3

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1:886–893 vol. 1, June 2005. 2

[4] 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, Nice, France, 2003. 2

[5] M. Han, W. Xu, H. Tao, and Y. Gong. An algorithm for multiple object trajectory trackingt. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'04)*, volume 1, pages 864–871, Washington, DC, Jun.27-Jul.2 2004. 4

[6] J. D. Lafferty, A. Mccallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML*, pages 282–289, 2001. 6, 7

[7] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178, New York, NY, Jun 17-22 2006. 4

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based leraning applied to document recognition. 86(11):2278–2324, Nov. 1998. 4

[9] X. Li. Hmm based action recognition using oriented histograms of optical flow field. *Electronics Letters*, 43(10):560–561, 10 2007. 2

[10] D. G. Lowe. Distinctive image features from scale invariant keypoints. *Int'l Journal of Computer Vision*, 60:91–110, 2004. 4

[11] L.-P. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. *CVPR*, 2007. 6, 7

[12] H. Ning, T. X. Han, D. B. Walther, M. Liu, and T. S. Huang. Hierarchical space-time model enabling efficient search for human actions. *IEEE Trans. on Circuits and Systems for Video Technology*, to appear. 8

[13] A. Quattoni, M. Collins, and T. Darrell. Conditional random fields for object recognition. *Neural Information Processing Systems*, 2004. 6

[14] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require? *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008. 2

[15] D. Tran and A. Sorokin. Human activity recognition with metric learning (pdf). *European Conference on Computer Vision*, 2008. 2

[16] M. Yang, Y. Wu, and S. Lao. Intelligent collaborative tracking by mining auxiliary objects. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 697–704, Jun 17-22 2006. 4

[17] Z. Zivkovic and F. Van der Heijden. Efficient adaptive density estimapion per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, 2006. 2