TT+GT at TRECVID 2010 Workshop

Nakamasa Inoue Toshiya Wada
Yusuke Kamishima Koichi Shinoda
Department of Computer Science,
Tokyo Institute of Technology
{inoue, wada, kamishi}@ks.cs.titech.ac.jp
{shinoda}@cs.titech.ac.jp

Ilseo Kim Byungki Byun
Chin-Hui Lee
School of Electrical and Computer Engineering,
Georgia Institute of Technology
{ilseo, yorke3, chl}@ece.gatech.edu

1 Overview

In this paper, we present our systems for semantic indexing and surveillance event detection in TRECVID 2010.

Semantic Indexing

We propose a statistical framework for semantic indexing using Gaussian mixture model (GMM) supervector kernels with visual and audio features. For classifier we used a Maximal Figure-of-Merit (MFoM) classifier and support vector machines (SVMs). Our four submitted runs are summarized in Table 1. Our best method was ranked 10-th in all full submission runs and 5-th among 31 teams.

Table 1: Description of TT+GT models

Model Name	Features and kernels	Learning Scheme	Mean Inf. AP
F_A_TT+GT+run1_1	GMM supervector kernels with MFCC and SIFT features	SVM	7.36%
F_A_TT+GT+run2_2	LSI features with color histogram and Gabor coefficients	MFoM-AP	3.72%
F_A_TT+GT+run3_3	GMM supervector kernels with SIFT extracted from Harris and Hessian affine regions	SVM	6.37%
F_A_TT+GT+run4_4	GMM supervector kernel with SIFT extracted from Harris affine regions	MFoM-F_1	3.56%

Surveillance Event Detection

We combine local features and global features and use them for detecting events. Local features are based on a person detector and they represent movements of individuals. Global features are based on optical flow features in video frames and they represent the flow of people. We tried to detect PersonRuns, PeopleMeet and PeopleSplitUp events. The results of three events are as follows; ANDCR of PersonRuns was 1.002, ANDCR of PeopleMeet was 1.003, and ANDCR of PeopleSplitUp was 1.008.

This paper is organized as follows. Section 2 describes the GMM supervector kernel and the MFoM classifier for semantic indexing. Section 3 describes our system for surveillance event detection using local and global features.

2 Semantic Indexing

2.1 Tokyo Tech's System

This section describes the GMM supervector kernels with MFCC and SIFT features.

2.1.1 Feature extraction

We extract three types of visual and audio features as follows:

1. **SIFT features with Harris affine detector**

   The SIFT feature proposed by Lowe [1] is invariant to image scaling and changing illumination so that it is widely used for object detection and categorization. Moreover, the Harris affine region detector [2], which is an extension of the Harris corner detector, provides affine-invariant regions. We use 32 dimensional SIFT features whose dimension is reduced by applying principal component analysis (PCA). The SIFT features are extracted not only from keyframes but also from a half of all the image frames in a shot.

2. **SIFT features with Hessian affine detector**

   We also extract SIFT features with the Hessian affine detector [2], which is complementary to the Harris affine detector. The combination of several different detectors can improve the robustness against noises.

3. **MFCC features**

   For audio features, we extract mel-frequency cepstrum coefficients (MFCCs), which are widely used in speech recognition. MFCCs describe the short-time spectral shape of audio frames. We also use Δ MFCC, $\Delta\Delta$ MFCC, Δ log power, and $\Delta\Delta$ log power. The total dimension of the audio features is 38.

2.1.2 GMM supervector kernel

The GMM supervector technique is proposed by Campbell [4] as a speaker recognition method and applied to visual event detection by Zhou [5]. We also showed the effectiveness of the GMM modeling and the combination of visual and audio models in the TRECVID high-level feature extraction task ([6]). In this work, we provide a unified processing of visual and audio features by using the GMM supervector kernel.

Let $X_s = \{x_i\}_{i=1}^{n_s}$ be a set of MFCC or SIFT features extracted from the s-th shot. Here we estimate a GMM parameter for each shot. The probability density function (pdf) of a GMM is given by

$$p(x|\theta) = \sum_{k=1}^{K} w_k \mathcal{N}(x|\mu_k, \Sigma_k), \tag{1}$$

where $\theta = \{w_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ is a set of parameters, K is the number of mixture components (vocabulary size), w_k is a mixing coefficient, and $\mathcal{N}(x|\mu_k, \Sigma_k)$ is a pdf of Gaussian distribution with mean vector μ_k and variance matrix Σ_k.

The maximum a posteriori (MAP) adaptation technique is used to estimate GMM parameters. For prior distribution for MAP adaptation, we use a universal background model (UBM), which is estimated by using all the features in video data. In this work, we adapt only mean vectors from the UBM as follows:

$$\hat{\mu}_k^{(s)} = \frac{\tau \mu_k^{(U)} + \sum_{i=1}^{n_s} c_{ik} x_i}{\tau + C_k}, \tag{2}$$

where $\mu_k^{(U)}$ is the k-th mean vector of the UBM, τ is a hyper parameter, and c_{ik}, C_k are given by

$$c_{ik} = \frac{w_k \mathcal{N}(x_i|\mu_k^{(U)}, \Sigma_k^{(U)})}{\sum_{k=1}^{K} w_k \mathcal{N}(x_i|\mu_k^{(U)}, \Sigma_k^{(U)})}, \quad C_k = \sum_{i=1}^{n_s} c_{ik}. \tag{3}$$

The GMM supervector is created by combining (normalized) mean vectors as follows:

$$\phi(X_s) = \begin{pmatrix} \tilde{\mu}_1^{(s)} \\ \tilde{\mu}_2^{(s)} \\ \vdots \\ \tilde{\mu}_K^{(s)} \end{pmatrix}, \quad \tilde{\mu}_k^{(s)} = \sqrt{w_k^{(U)}} (\Sigma_k^{(U)})^{-\frac{1}{2}} \hat{\mu}_k^{(s)}. \tag{4}$$

Finally, we obtain an RBF-kernel in the GMM supervector space given by

$$k(X_s, X_t) = \exp\left(-\gamma \|\phi(X_s) - \phi(X_t)\|_2^2\right). \tag{5}$$

For classifier, we use SVM and MFoM (description of the MFoM classifier is in Section 2.2). When we use SVM, we combine SVM scores for audio and visual features by calculating weighted sum of the scores. The weight coefficients (that add up to one) are trained by using 2-fold cross validation.

2.2 Georgia Tech's System

Our learning framework is based on Linear Discriminative Function (LDF) classifiers trained with a Maximal Figure-of-Merit (MFoM) algorithm proposed in [14, 15]. This year, we tried two new schemes, namely: (1) maximizing the average precision (AP) metric directly by approximating it with a staircase function and smoothing it with a set of sigmoid functions; and (2) kernelizing the LDFs with features generated at Tokyo Tech as a collaboration system with Tokyo Tech. They are discussed next.

2.2.1 Maximal Figure of Merit Learning Scheme

The ranking score for ordering instances according to their relevance or similarity is based on a separation function that measures the distance between the target class and other competing classes. $T = \{(X, Y)|X \in R^D, Y \subset C\}$ be a given training set, where X is a D-dimensional feature for an instance and Y is the corresponding label for X. For binary classification, $C = \{C^+, C^-\}$, and C^- can be also divided into several subclasses as $C^- = \{C_1^-, \cdots, C_M^-\}$. Then, we learn a discriminant function, $g_j(X; \Lambda_j)$, with the parameter set, $\Lambda = \{\Lambda_j, 0 \leq j \leq M\}$. The class separation function is defined as follows:

$$d(X; \Lambda) = -g(X; \Lambda_0) + g^-(X; \Lambda) \tag{6}$$

where $g^-(X; \Lambda)$ is called an anti-discriminant function for the positive class, defined as:

$$g^-(X; \Lambda) = log\left[\frac{1}{M}\sum_{i=1}^{M} \exp\left(g_i(X; \Lambda_i)\right)^\eta\right]^{1/\eta} \tag{7}$$

which represents a geometric average of the competing scores among all negative subclasses. Here, if $M = 1$, which means that only one class is used for the negative class, $g^-(X; \Lambda)$ is just a confidence score for the negative class. For the $g(\cdot)$ function, we used a linear discrimination function (LDF), $g_j(X; \Lambda_j) = \omega_j \cdot X$, where ω_j is a D-dimensional parameter vector.

Then, a loss function for a given objective performance metric, such as precision, recall, or F_1, can be defined by approximating the four items in a contingency table with 0-1 sigmoid functions. With a generalized probabilistic descent(GPD) algorithm, a classifier is learned to minimize the desired loss function.

2.2.2 AP maximization with MFoM

We propose an extended MFoM called MFoM-AP [16] to directly maximize the ranking performance metric, AP, which is used for the evaluation of the TRECVID 2010 SIN task. In [9], it is shown that a learning algorithm to minimize classification errors cannot guarantee to maximize any desired ranking quantity. Even a classifier maximizing another ranking performance metric, Area Under the ROC curve (AUC), cannot guarantee to optimize AP [10]. Especially, for uneven distributions of positive and negative instances which is often the case in our evaluation, a learning scheme attempting to optimize the AP value is expected to lead to better AP values than the one that minimizes the error rate or maximize other ranking metrics. Our preliminary experiments using two-fold cross-validation sets showed that MFoM-AP outperforms MFoM-F_1 by a relative improvement of 10.7% in AP.

One of the difficulties in optimizing AP is that AP, like other ranking performance metrics, is a discrete measure and all related errors need to be approximated by 0-1 error functions, such as a sigmoid. There have been approaches to approximate pair-wise rankings [12, 11]. However, pair-wise training is almost prohibitive in the TRECVID case, where the size of training data is extremely large.

In contrast MFoM-AP uses an efficient list-wise method to approximate AP. First, AP is considered as a function of all the positive and negative sample scores as follows:

$$AP = f(s_1^+, s_2^+, \ldots, s_{M_p}^+, s_1^-, s_2^-, \ldots, s_{M_n}^-) \tag{8}$$

where M_p and M_n indicate the numbers of positive and negative samples. And, s_i^+ and s_j^- are the i^{th} and j^{th} highest positive and negative scores, respectively. Assuming we can change the value of a sample score, s_i^+ or s_j^-, remaining the other scores not changed, S_{AP} behaves like a staircase function with regard to the individual sample score. The stair way function is approximated using a combination of sigmoid functions to $\widehat{AP}_i^+$ or $\widehat{AP}_j^-$ as a function of s_i^+ or s_j^-. Then, with chain rule, we can compute the gradient of AP with respect to model parameters as follows:

$$\frac{\partial \widehat{AP}}{\partial \omega} = \sum_{i=1}^{M_p} \frac{\partial \widehat{AP}_i^+}{\partial s_i^+} \frac{\partial s_i^+}{\partial \omega} + \sum_{j=1}^{M_n} \frac{\partial \widehat{AP}_j^-}{\partial s_j^-} \frac{\partial s_j^-}{\partial \omega} \tag{9}$$

where ω indicates the classifier parameters. With this approximation, we only need to calculate $(M_p + M_n)$ gradients, where the computational complexity is significantly reduced compared to $(M_p \times M_n)$ gradients in pair-wise approaches. For the ranking score, s_i^+ and s_j^-, we used the class separation function in Eq (6) as:

$$s_i = -d(X_i; \Lambda) \tag{10}$$

The model parameters are estimated using a probabilistic gradient descent algorithm [14, 15]. Specifically, we used the inexact line search algorithm to find an appropriate learning rate. A detailed presentation can be found in [16].

2.2.3 Kernelized MFoM approach based on subspace distance minimization

In this section, we pursue a viable solution to fuse feature extraction techniques from the Tokyo Tech side and classifier design techniques from the Georgia Tech side. It is mainly inspired by the idea of encoding pairwise distances between video shots into a kernel matrix K and the idea of directly optimizing a performance metric during the MFoM training phase.

One issue that we had to pursue such a direction was that the MFoM learning approach presented in the previous section was mainly aimed to handle feature vectors based on bag-of-features followed by latent semantic indexing in which mostly linear score functions are used. On the other hand,

the feature extraction technique developed from the Tokyo Tech side constructed a kernel matrix K by encoding pairwise distances between video shots. To take advantage of the kernel matrix K, one typically introduces a Hilbert space $\mathcal{H}$ derived from K such as in SVM [7]. Similarly, we define the score function g for a class j as follows:

$$g(x, \Lambda_j) = \sum_{i=1}^{n} \lambda_{ij} k(x_i, x) + b_j, \tag{11}$$

where $\Lambda_j = [\lambda_{1j}, \ldots, \lambda_{nj}]^T$ is a parameter vector and b_j is a bias for the class j and n is the number of training data samples.

In Eq. (11), one should notice that the dimension of the parameter vector is increased to n multiplied by the number of classes. Recall in this years' TRECVID evaluation, $n \geq 100k$, which makes learning the parameter vectors for all n computationally prohibitive. Therefore, we developed a kernelized MFoM approach based on a subspace distance minimization criterion. Specifically, given a subspace of the Hilbert space $\mathcal{H}$, $\mathcal{U}$ constructed with the entire training samples, another subspace of $\mathcal{H}$, $\mathcal{V}$ was constructed with a subset of training samples in a way that a subspace distance between $\mathcal{U}$ and $\mathcal{V}$ was minimized. Then, only the parameter λ_{kj}'s associated with the training samples that were chosen to construct $\mathcal{V}$ were learned. This way, one can imagine that learning the parameter vector now becomes feasible, while the learned vector is nearly optimal. With the proposed technique, the score function in Eq. (11) is reduced to

$$g(x, \Lambda_j) = \sum_{i=1}^{d} \lambda_{ij} k(x_i, x) + b_j \qquad \text{for } i \in \mathcal{I}_d, \tag{12}$$

where $\mathcal{I}_d$ are an index set with a cardinality of d that indicates the selected training samples. The method to select training samples are based on a particular spectral problem on the kernel matrix K, called the Nystrom extension. Interested readers can refer [8] to see detailed methods along with theoretical background.

2.3 Experiment and Result

Our results in the semantic indexing task are illustrated in Figure 1 and Figure 2. The best result was 7.36% in terms of Mean Inf. AP, which is ranked 10-th in all the full-submission runs, and 5-th among all the participated teams. The followings are details of our four submitted runs.

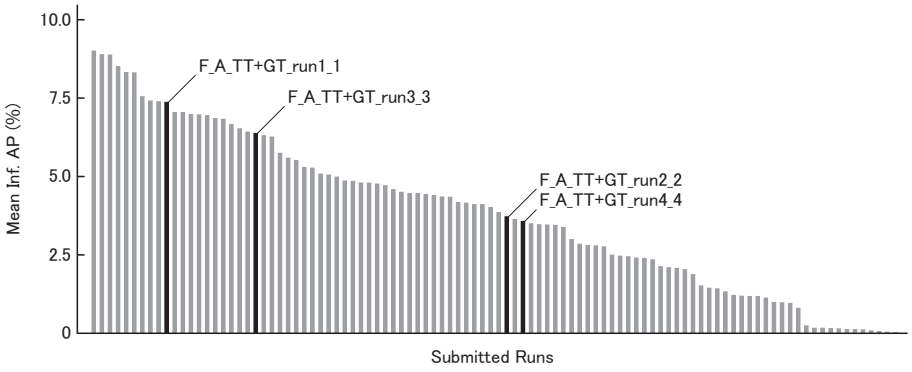

Figure 1: Overview of results of the semantic indexing task in TRECIVD 2010. Mean Inf. AP of our best run using MFCC and SIFT GMM supervector kernels was 7.36%.

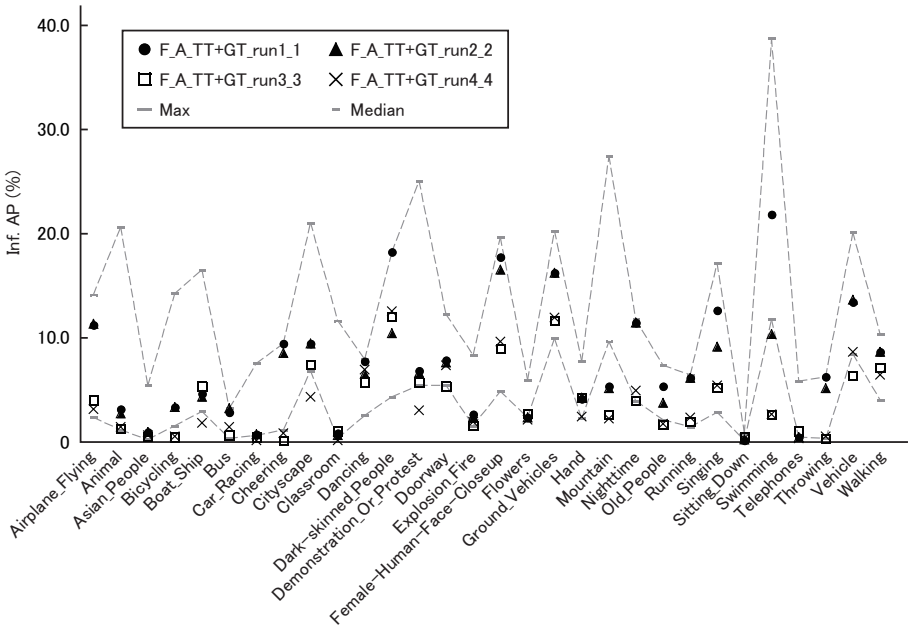

Figure 2: Comparison of performances in the semantic indexing task. Our 4 runs are plotted with the max and median scores in TRECVID 2010.

F_A_TT+GT_run1_1

This run used GMM supervector kernels and SVM classifier with MFCC and SIFT features described in Section 2.1. We used $K = 512$ Gaussians for SIFT and $K = 256$ Gaussians for MFCC. The UBMs were trained using 1,000,000 sample features. The parameter τ for MAP adaptation was set to 20.0 and the parameter γ for the RBF kernel was set to $\tilde{d}^{-1}$ where $\tilde{d}$ is the average distance between two supervectors.

Resulting Mean Inf. AP for this run was 7.36%. The APs for the following concepts improved by more than 10% relative to the run using only visual features (F_A_TT+GT_run3_3): "Singing", "Dancing", "Cheering", "Swimming", "Throwing", "Dark-skinned_People", "Old_Pe ople", "Classroom", and "Animal". According to the results, the audio model helps detect human actions and events, essecially when a person is talking or singing to the camera.

F_A_TT+GT_run2_2

F_A_TT+GT+run2_2 used Latent Semantic Indexing (LSI) features with 12-dimensional color histogram and Gabor texture filter coefficients proposed in [13]. Classifiers were learned for each color and texture features, and fused by the model based transform (MBT) method. *F_A_TT+GT+run2_2* showed much better performance for the *Lite* evaluation in ranking among the submitted models when compared to the *Full* evaluation. Due to the limited time, we carefully adjusted parameters and learning rate only for the 10 proposed concepts of the *Lite* evaluation. Improved performances for the *Full* evaluation are expected if we had used the same procedure for the other 120 concepts as that for the 10 *Lite* version concepts.

F_A_TT+GT_run3_3

This run used the same method as *F_A_TT+GT_run1_1* but no audio (MFCC) features are used. Inf. Mean AP for this run was 6.37%. We found that the number of SIFT samples per shot is the main factor to have a high accuracy, and in fact, although we didn't use the dense SIFT descriptors,

we got good reults with the multi-frame technique and the interest point base region detectors. In our future work, we will analyze at least how many SIFT samples are needed to estimate the GMM parameters.

F_A_TT+GT_run4_4

For the model, *F_A_TT+GT+run4_4*, the kernelized MFoM approach described in Section 2.2.3 was used. For K, a kernel matrix based on SIFT descriptors extracted at the interest points where a Harris affine filter detected was chosen, which was the same matrix that Tokyo Tech used for one of the runs. Then, we trained 130 binary-class models through optimizing a F_1 measure. Similar to other runs, parameters were tuned using two-fold cross validation. Among a few parameters, only two were adjusted; a regularization parameter varying from 0.01 to 1e-6 and a slope of a function that determines the number of negative samples, which aimed to ease the unbalanceness between positive and negative classes.

This run achieved the Mean Inf. AP value of 3.56%. The semantic features that this run performed the best and the least were dark-skinned_people with the AP value of 12.5% and asian_people with the AP value of 0.1%, respectively.

Initially, we hoped *F_A_TT+GT+run3_3* and *F_A_TT+GT+run4_4* would return at least similar results, but the kernelized MFoM approach did not provide good results compared to the run from Tokyo Tech side, which used SVM. Our assessments are as follows: a) parameters were not properly adjusted through cross validation (i.e. only two parameters were adjusted), b) a scheme to select negative samples was not effective, and c) unlike our second run, a F_1 measure was optimized. In fact, the kernelized MFoM approach has been updated considerably since submission as an algorithm to provide better initial solutions was developed. So we also hope that we can get much better results for the next year's evaluation.

In addition, we assumed that unlabeled samples are negative samples. After the submission, we found that there exist some positive samples in unlabeled data, which are not negligible. We should have been more careful to deal with the labeled data. Also, we need to find out the reason why the TRECVID evaluation performances were considerably degraded compared to those of our two-fold cross validation experiments.

3 Surveillance Event Detection

3.1 Introduction

In this year's task we selected three events, PersonRuns, PeopleMeet and PeopleSplitUp, out of the seven events TRECVID2010 provided. We chose them for two reasons. First, they appear frequently in the training data set. Second, they are related to the whole-body motion, unlike those related to the motion of body parts, such as Pointing.

Most previous methods first detect a person, then extract local features from the area where a person likely exists. However, it is difficult to detect a person in a crowd. To overcome this problem, we propose an event detection method combining a detector using local features obtained by human tracking and that using global features in a whole image. As for local features, first, we obtain person trajectories by the particle filter, and then, detect events using our event models. As for global features, we detect events by a support vector machine (SVM) whose inputs are the optical flow vectors extracted from a whole image frame. Finally we combine outputs from these two detectors for event detection.

3.2 Local features

3.2.1 Outline

We use person trajectories as local features. First, we calculate the existence probability of a person using a cascade of person detectors [17] at each pixel, and then, cluster those pixels with high probabilities by using the particle filter. This year, we improve the deviation of the existence probability of a person using background subtraction. In addition, we prevent particles from concentrating on one person by placing them on sparse grid points in an image frame. Next, person areas between successive frames are associated to obtain a person trajectory. Finally, we detect each event using the trajectory information. We detail each step below.

3.2.2 Calculation of a existence probability of a person using a cascade of person detectors

For each pixel in an image frame, the existence probability of a person is calculated using a cascade of person detectors. Its input is an image with a given size 8×20, whose center is located at the pixel. This cascade sequentially connects 30 simple detectors, each of which detector uses one Haar-Like feature. Each detector examines the images not rejected by the previous detectors, and rejects the images in which a person is not likely to exist. Therefore, the more detectors an image is processed through, the more likely a person exists in it.

We can calculate the existence probability of a person based on the number of detectors that it passes through. Let A_i be the set of samples accepted by the i-th detector. The existence probability of a person, $E(i)$, is defined as follows:

$$E(i) = \frac{p(i)}{p(i) + n(i)}, \tag{13}$$

where $p(i)$ is the number of positive samples (where a person exists) in A_i, and $n(i)$ is the number of the other samples in A_i. Furthermore, a sigmoid function is used to prevent particles from concentrating on one person whose existence probability is high. The modified existence probability $E'(i)$ is given by

$$E'(i) = \frac{1}{1 + \exp(a(-E(i) + b))}. \tag{14}$$

The parameters a and b are optimized by using training data.

3.2.3 Person detection using a particle filter and clustering

A particle filter [18] detects objects in an image using many particles. We use 800 particles.

1. Set randomly the initial position and velocity of all particles at the first frame.
2. Calculate the existence probability of a person for each particle using the cascade.
3. Move the particles with low probabilities to the region of those with high probabilities.
4. Move all particles according to their velocity for which random noise is added.
5. Go to the next frame and return to 2.

We cluster the particles every five frames (200ms duration). The resulting clusters are identified as a person.

In order to improve a particle filter, we added the following two new processes this year.

Background subtraction

A still person is often detected as a person, but we would like to detect only "moving" persons. To realize this, we decrease the existence probability of a person at the pixel whose color is close to the background color. Specifically, we multiply the existence probability of a person by a factor β:

$$\beta = 1 - \alpha \exp\left(-\frac{d^2}{2\sigma^2}\right), \tag{15}$$

where

$$d = \sqrt{(r_a - r_b)^2 + (g_a - g_b)^2 + (b_a - b_b)^2} \tag{16}$$

is the difference between color (r_a, g_a, b_a) of the pixel and background color (r_b, g_b, b_b) of the same pixel. We set $\alpha = 0.93$ and $\sigma = 10$.

The initial background image and the initial background candidate image are both the image of the first frame. We set the counters t for all pixels to zero at first. Let c be the image frame color, d be the background color, and e be the background candidate color. Then we perform the following update process for each pixel images every frame.

If $|c - d| \leq T$,

1. $d = 0.05c + 0.95d$
2. $t = t - 10$
3. If $t \leq 0$, then $d = c$

Else if $|c - e| \leq T'$,

1. $e = 0.1c + 0.9e$
2. $t = t + 1$
3. If $t \geq 100$, then $d = e$ and $e = c$

Otherwise,

1. $t = t - 10$
2. If $t \leq 0$, then $e = c$

T and T' are different for different pixels.

We increased the person detection accuracy by 3.8%, by using this background subtraction.

Decentration of particles
Particles are often concentrated on a few clusters, even when there exist many persons. We place each particle on one of the 5×5 grid points to prevent these particles from concentrating on one cluster. This decentration increased the person detection accuracy by 1.6%.

3.2.4 Human tracking

In order to obtain the trajectory of a particle cluster identified as a person, each cluster in the i-th frame is associated with its corresponding cluster in the $(i+1)$-th frame using the positions and velocities of these clusters. Here we set the frame period to 200 ms. First we define position $\mathbf{p}_c(i)$ and velocity $\mathbf{v}_c(i)$ of cluster c at time frame i as follows.

$$\mathbf{p}_c(i) = \frac{1}{N_c}\sum_{k=1}^{N_c}\mathbf{p}^k(i),\ \mathbf{v}_c(i) = \frac{1}{N_c}\sum_{k=1}^{N_c}\mathbf{v}^k(i), \tag{17}$$

where N_c is the number of particles in cluster c, and $\mathbf{p}^k(i)$ and $\mathbf{v}^k(i)$ are the position and velocity of k-th particle, respectively.

We predict position $\mathbf{p}_c(i+1)$ of the same cluster at the next frame as

$$\mathbf{p}_c(i+1) = \mathbf{p}_c(i) + \mathbf{v}_c(i). \tag{18}$$

Next we associate this cluster c with the cluster nearest to $\mathbf{p}_c(i+1)$. If there is no cluster to be associate with, or, the distance between two associated clusters is longer than a predetermined threshold $d_f = 4|\mathbf{v}_c|$, we assume that the trajectory ends at the present frame.

3.2.5 Event detection using event model

We describe how to make an event model utilizing th trajectory information. Here the input features of each trajectory are the positions and velocities of the associated clusters for five frames. Let $l_c(i,j)$ be a distance of cluster c from the i-th frame to the j-th frame. Then, we define the score for each of the three events S_L as follows:

- PersonRuns: $S_L = \max_{0\le i\le 4}|\mathbf{v}_c(i)|$
- PeopleMeet: $S_L = \sum_{i=0}^{3}(|\mathbf{v}_c(i)| - |\mathbf{v}_c(i+1)|)$
- PeopleSplitUp: $S_L = (l_c(2,3) + l_c(3,4)) - (l_c(0,1) + l_c(1,2))$

We set the score for PeopleMeet to zero when there is at most one cluster.

3.3 Global features

As global features, we use optical flow vectors [19] extracted in 10(col) $\times$ 8(row) = 80 grid points in an image. Their components are concatenated to make one feature vector. We use support vector machines (SVMs) to detect event frames from the extracted feature vectors. For each of the three events, we train one SVM. We use feature vectors extracted from the frames in which the event occurs as positive samples, and use feature vectors extracted from the other frames as negative samples. The output score from SVM, S_G, shows the degree how likely the event occurs.

Table 2: T_C values

	Camera1	Camera2	Camera3	Camera4	Camera5
PersonRuns	177	100	143	150	231
PeopleMeet	57	28	45	30	30
PeopleSplitUp	10	13	12	18	20

Table 3: Results of TT+GT group's run for Surveillance Event Detection

	Ref	Sys	CorDet	FA	Miss	ANDCR
PersonRuns	107	7	0	7	107	1.002
PeopleMeet	449	8	0	8	449	1.003
PeopleSplitUp	187	43	1	42	186	1.008

3.4 Combining local features and global features

The score S_L was obtained using the event models based on local features. And the score S_G was obtained using the SVMs based on global features. Here we define a new score S_C by combining these two scores with a weight ω as follows:

$$S_C = (1 - \omega)S_L + \alpha\omega S_G, \tag{19}$$

where α is a constant number to adjust the difference of dynamic ranges of S_L's and S_G's. We set $\alpha = 50$. The frames whose S_C's are larger than the predetermined threshold T_C are detected as the event frames. T_C values are shown in Table 2. The weight ω in Eq. (19) was determined by our preliminary experiments using training data; PersonRuns: $\omega = 0.25$, PeopleMeet: $\omega = 0.81$, and PeopleSplitUp: $\omega = 0.79$. In this preliminary experiments, we tried to reduce the false detection rate and to improve ANDCR, by controlling the weights.

3.5 Results

Table 3 shows our detection scores for the three events. Our ANDCR scores were still larger than 1.000. It may be because the background detector didn't work well and because the convergence speed of the particle filter was slow. We found that some persons were often misrecognized as the background. Our assumption that there are no persons exist in the first frame of a video clip were not satisfied in the testing data. Secondly, the particles often failed to converge to persons since the video lengths of the test data were too short. Our future work will focus on improving our person detector by estimating the existence probability more correctly.

References

[1] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," In *IJCV*, vol.60(2), pp.91-110, Jan. 2004.

[2] K. Mikolajczyk, and C. Schmid, "Scale & Affine Invariant Interest Point Detectors," In *IJCV*, vol.60(1), pp.63-86, Jan. 2004.

[3] A. F. Smeaton, *et al.*, "Evaluation Campaigns and TRECVid," In *Proc. of MIR*, pp.321-330, Oct. 2006.

[4] W. Campbell, *et al.*, "SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation," In *Proc. of ICASSP*, vol.1, pp.97-100, 2006.

[5] X. Zhou, *et al.*, "SIFT-Bag Kernel for Video Event Analysis," In *Proc. of ACM Multimedia*, pp.229-238, 2008.

[6] N. Inoue, *et al.*, "High-Level Feature Extraction using SIFT GMMs and Audio Models," In *Proc. of ICPR*, pp.3220-3223, 2010.

[7] C. J. C. Burges. "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[8] C.-H. L. Byungki Byun. "A kernelized maximal-figure-of-merit learning approach based on subspace distance minimization," In *ICASSP*, 2010. Submitted.

[9] C. Cortes and M. Mohri. "AUC optimization vs. error rate minimization," In *NIPS*, 2004.

[10] J. Davis and M. Goadrich. "The relationship between precision-recall and ROC curves," In *ICML*, 2006.

[11] S. Gao, C.-H. Lee, and J. H. Lim. "An ensemble classifier learning approach to ROC optimization," In *ICPR*, 2006.

[12] S. Gao and Q. Sun. "Improving semantic concept detection through optimizing ranking function," In *IEEE Transactions on Multimedia*, volume 9, pages 1430–1442. IEEE, 2007.

[13] S. Gao, D.-H. Wang, and C.-H. Lee. "Automatic image annotation through multi-topic text categorization," In *ICASSP*, 2006.

[14] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua. "A maximal figure-of-merit approach to text categorization," In *ACM SIGIR*, 2003.

[15] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua. "A MFoM learning to robust multiclass multi-label text categorization," In *ICML*, 2004.

[16] I. Kim and C.-H. Lee. "Optimization of average precision using maximal figure-of-merit learning algorithm," submitted to *ICASSP*, 2011.

[17] P. Viola and M. Jones. "Rapid object detection using a boosted cascade of simple features," *Proc. CVPR*, vol. 1, 2001.

[18] M. Isard and A. Blake. "Condensation: Unifying low-level and high-level tracking in stochastic framework," *Proc. ECCV*, 1998

[19] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artifical Intelligence*, vol. 17, no. 1, pp. 185-203, 1981.