DRAFT Notebook Paper

Brno University of Technology at TRECVid 2012
Interactive Surveillance Event Detection Pilot

Petr Chmelar, Jozef Mlich, Martin Pesek, Tomas Volf, Pavel Zemcik, Jaroslav Zendulka

IT4Innovations Centre of Excellence
Brno University of Technology
Faculty of Information Technology
Bozetechova 2
Brno, 612 66
Czech Republic

Abstract

In the paper, we describe our experiments in the interactive surveillance event detection pilot (SED) of the 2009 TRECVid evaluation. Our approach inherits functionality of the Surveillance Network Augmented by Retrieval (SUNAR) system, which is an information retrieval based wide area (video) surveillance system being developed at FIT, Brno University of Technology. It contains both standard and experimental techniques evaluated at the AVSS 2009/10 Multi-Camera Tracking Challenge. We have deployed active learning functionality based on moving objects' trajectory statistics and shape classification using VTApi (Video Terror Application programming interface), which was created to unify and accelerate the intelligent vision applications development.

The paper is organized as follows. The first section provides a motivation and a brief introduction. Section 2 is dedicated to the active learning approach used and some other theoretical aspects of the work. Details on the technology is presented in Section 3. Section 4 shows some experimental results achieved during the training. Finally, Section 5 discusses achieved results and concludes the paper. Evaluation results are attached at the end of the paper.

1. We have submitted the following SED runs:
   - BrnoUT_2012_**retro**ED_EVAL12_ENG_s-camera_p-SUNAR_1 – contains 1000 “best” shots classified for each event based on the object shape and trajectory using active-learning-based annotations.
   - BrnoUT_2012_**interactive**ED_EVAL12_ENG_s-camera_p-SUNAR_1 – includes only shots classified as good as annotated during the interactive period.

2. The mayor difference between the runs is the Round #3 active learning step based on 25 min. annotating of results of the retrospective task. The retrospective task maximized recall, while the interactive task maximized precision.

3. The mayor contribution was the semi-automatic annotation using active-learning, classification of object description using trajectory and shape features and the tracker able to handle multiple occlusions.

4. The challenge of the TRECVid SED pilot and the video surveillance event detection in general is the ability to learn from annotations provided and to improve the classifiers by providing more accurate samples and higher-quality features extracted.

Acknowledgements

This work has been supported by the research project Security-Oriented Research in Information Technology CEZ MSM0021630528, grant VG20102015006 of the Ministry of the Interior of the Czech Republic, the European Regional Development Fund in the IT4Innovations Centre of Excellence (CZ.1.05/1.1.00/02.0070) and with a financial support from the Czech Republic state budget through the Ministry of Industry and Trade. Thanks to Janicka Ch.

1. Introduction

In 2006, we have started to develop an IR-based multi-camera tracking system to be at the top of the state of the art. The idea was to create an automated system for visual features detection, indexing and analysis, which can reduce the burden of continuous concentration on monitoring and increase the effectiveness of information reuse by security, police, emergency, firemen or armed services.

Brno University of Technology, Faculty of Information Technology has taken part at TRECVid since 2007. In the past, we have taken part in various tasks, but SED in 2008 only. Our attempt was based on advanced masking, background subtractions and extracted trajectories. This time, we have avoided the masking approach used for ObjectPut and focused more on other moving object based features.

The challenge of the TRECVid pilot and a better video surveillance event detection in general are high-quality annotations. There is only temporal localization ground-truthed by the University of Pennsylvania Linguistic Data Consortium. And this is not much helpful for a surveillance task of this kind. The manual labeling of objects taking part in the event annotated is really expensive in such amounts of video (a hour of video takes about 5 hours of burden work), therefore we need a more optimal strategy. For that purpose we have developed two modes of annotation interfaces – the first of them uses output of our vision module (so that annotating 1 hour of video takes about 1 hour). The second approach sorts the classified outputs and only two keys are necessary to mark the event shown – positive (1+) or negative (0, Enter) and thus the annotation process may be further optimized. This is accomplished by active learning, as described further.

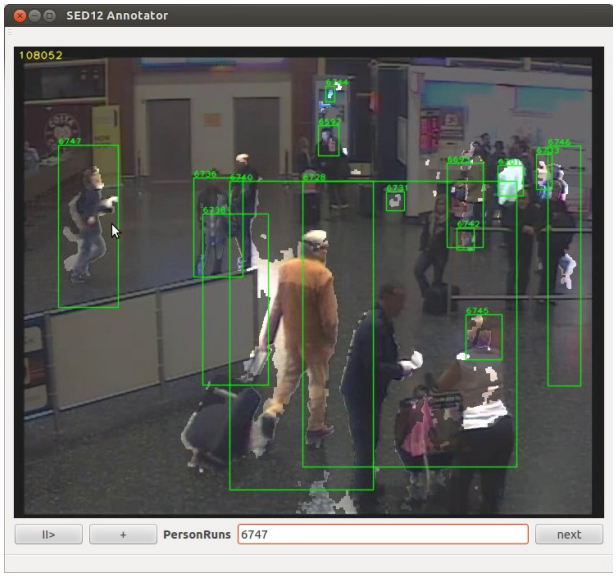


Fig. 1. SED12 Annotator in the initial Round #1 mode. A human annotator is supposed to click the running object or type the proper number when occluded.

2. Active learning

Active Learning (AL) systems attempt to overcome the labeling bottleneck by asking queries in the form of unlabeled instances to be labeled by an oracle (e.g. a human annotator). In this way, the active learner aims to achieve high accuracy using as few labeled instances as possible, thereby minimizing the cost of obtaining labeled data. Active learning is an optimal experimental design strategy. The hypothesis is that if the learning algorithm is allowed to choose the data from which it learns, it will perform better with less training [1].

An example of active learning is the pool-based active learning cycle. A learner may begin with a small number of instances in the labeled training set L and request labels for one or more carefully selected instances, learn from the query results, and then leverage its new knowledge to choose which instances to query next. An illustration of such process is in figure 2.

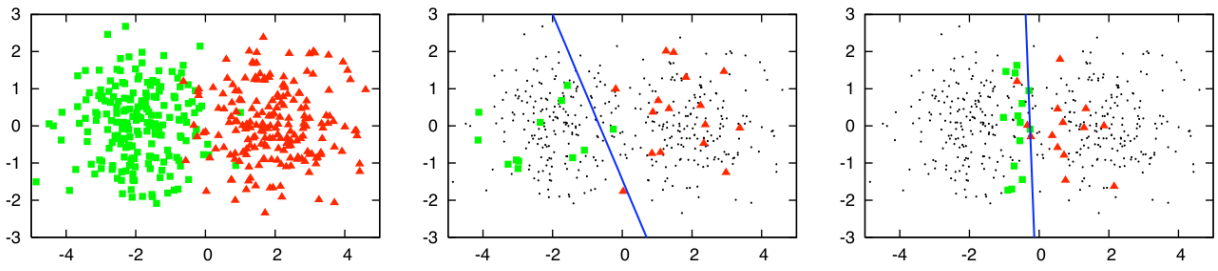


Fig. 2: Illustration of a pool-based active learning. It shows the advantage to the learning performance when annotated the same amount of samples in (b) and (c) out of (a) [Settles, 2009].

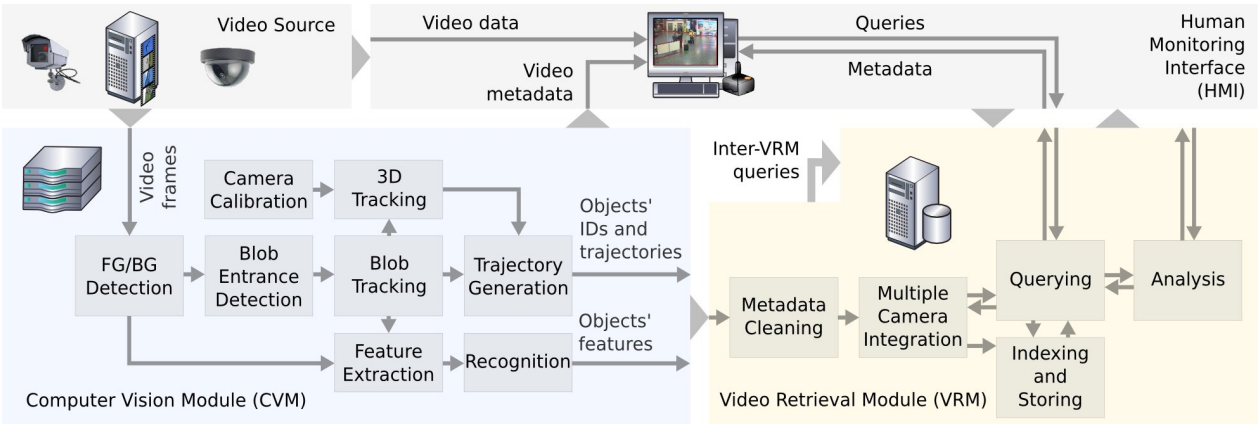


Fig. 3. Illustration of the SUNAR architecture.

There are several approaches to the active learning. One of the first active learning scenarios to be investigated is learning with membership queries. In this setting, the learner may request labels for any unlabeled instance in the input space, including (and typically assuming) queries that the learner generates de novo, rather than those sampled from some underlying natural distribution [1]. The idea of synthesizing queries has also been extended to regression learning tasks, which is similar to the stream-based selective sampling . Other query strategies aim to the metric that should be minimized (or maximized) by the learner. It is for instance the entropy, expected model change, density weight, error rate or variance. For more detailed information see [1].

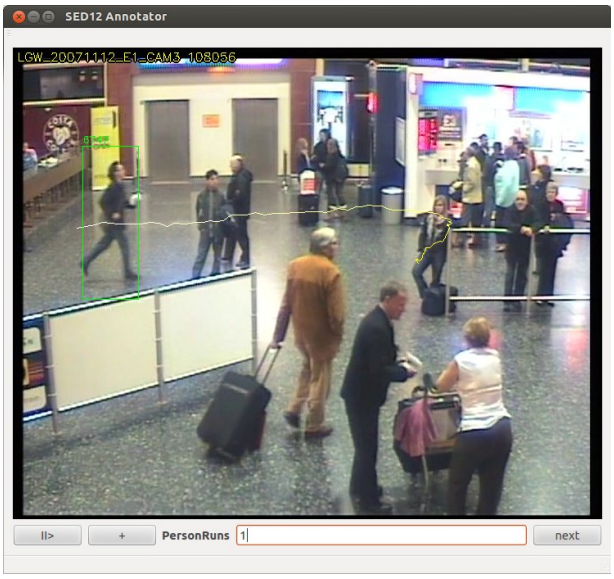


Fig. 4. Same scene as in figure 1 at the Round #2 of active learning process – a human annotator is supposed press “1” if the object highlighted is really running.

The approaches of AL may thus iterate to achieve a higher learner performance. Moreover, it can be supplemented by an unsupervised (clustering) or semi-supervised learner [2]. In this way, the annotator can mark only well discriminative centers of clusters (making sense) according to the requirements for instance. A survey on other semi-supervised learning methods can be found in [3].

The more detailed information on how the active learning concept was applied on moving objects can be found in the following chapter and the experiments are presented in chapter 4.

3. The technology

The goal was simply to perform online tracking, object and event detection to produce the metadata; and to clean, integrate, index and store the metadata to be able to query and analyze it. Upon our knowledge, SUNAR has been the first implementation of a surveillance system, which functionality is based on querying. The queries are of two types – online used mainly for identity preservation; and offline to query the metadata of the camera records in the wide area when an accident, crime, a natural or human disaster occurs.

3.1 SUNAR

In brief, SUNAR is composed of three basic modules – video processing, retrieval, the monitoring interface and the video source (server or camera network). Computer Vision Modules are based on the OpenCV Library for object tracking extended by feature extraction and network communication capability similar to MPEG-7. Information about objects and the area under surveillance is cleaned, integrated, indexed and stored in Video Retrieval Modules. They are based on the PostgreSQL database extended to be capable of similarity and spatio-temporal information retrieval, which is necessary for both non-overlapping surveillance camera system as

well as information analysis and mining in a global context.

The video source might be e.g. a camera or a video server. It is not a generic part of the system, but it must be capable of a standard TCP/IP communication used to communicate between all of described modules. Each module except the Human Monitoring Interface is responsible for capturing, analysis and retrieval in appropriate part of the wide area under surveillance. Modules communicate basically only with their neighborhoods. In this way, we can build a considerably large system, because no special central unit is necessary.

Computer vision

The input of the Computer Vision Module (CVM) is a video stream. We use OpenCV [4] for tracking and 3D calibration especially (if feasible). We have extended the OpenCV Blobtrack demo to be capable of feature extraction, object (and event) recognition and IP based video stream capture using all capabilities of customized built-in FFMPEG (www.ffmpeg.org) server and client.

Object tracking [5] is a complex problem and it is hard to make it working well, upon to our experience with real (crowded) scenes as illustrated in figure 2. Discussed approach is based mainly on proved methods of object tracking implemented in the Open Computer Vision Library [4]. These methods are illustrated in figures 1 and 5; and the schema in figure 3.

Foreground is derived from background, which is modeled using Gaussian Mixture Models (GMM, [6]) as an average value of color in each pixel of video and the foreground is a value different to the background. We have also been inspired by the approach developed by Carmona et al. [7], which is based on segmentation of the color in RGB color space into background, foreground and noise (reflection, shadow, ghost and fluctuation) using a color difference cone with vertex located in the beginning of the RGB coordinate system. In this way, the illumination can be separated from the color more easily. However the selection of appropriate parameters is a burden task, which is usual in unsupervised learning [8].

The other two modules – blob entrance and tracking are standard OpenCV Blobtrack [4] functions with appropriate parameters. Blob entrance detection is done by tracking connected components of the foreground mask. The Blob tracking algorithm is based again on connected components tracking and Particle filtering based on Means-shift resolver for collisions. We plan to extend the entrance algorithm, because it doesn't track the object from its early appearance. There was used also the trajectory refinement using the (inverted) Kalman filter as described in [9].

The trajectory generation module has been completely rewritten to add the feature extraction and TCP/IP network communication capability. The protocol is based on SQL, rather than XML (in previous versions) similarly to MPEG-7 [10].

The output of the CVM module is metadata of objects and the environment. It includes local identification of objects, its spatio-temporal location and its changes (speed) in the monitored area and a description of such objects – its dimensions, shape, color, texture or other special features (e.g. state plate or face descriptor) similarly to MPEG-7 [10]. The description is complemented with recognition of basic object classes (e.g. cars, trolleys, people or groups) and events (opposite way, left luggage) as described in the theoretical chapter.

Video retrieval

The main idea of the proposed wide area system is however implemented in the Video Retrieval Module (VRM). The input of the module is metadata produced by CVMs. This metadata is cleaned and normalized in time and space (lighting, color bias and estimated 3D parameters) and stored in the PostgreSQL database. The primary function of the VRM was the object identification – to integrate identifiers (IDs) of objects in the wide area, based on the previous occurrence of the object and its appearance. This has been evaluated at the AVSS 2009/10 Multi-Camera Tracking Challenge. Although, this functionality was omitted for the purpose of SED 2012 evaluation, we have used the feature extraction and classification capabilities of SUNAR to perform the event classification.

Analysis and classification

As stated above, the Analysis submodule is rather complex – it uses OLAP–based functionality for providing statistics on different granularities and views and it supports many machine-learning methods as Bayes classifiers, SVM [11], EM/GMM [6], HMM some other and time-series variants, frequent pattern analysis and various clustering algorithms. More detailed information can be found in [12]. We use appropriate libraries for this purpose.

For the purpose of SED pilot we have employed parameter selection search using 5-fold cross-validation SVM [11] based on transformed features extracted from the moving objects and their trajectories. We refer this classification scheme as “Track”. It contains features:

1. Camera (1-5).
2. Position – trajectory start (x_1, y_1), end (x_2, y_2), mean (μ_x, μ_y), standard deviation (σ_x, σ_y) and sum (Σdx, Σdy).
3. Trajectory duration (t).
4. Speed at trajectory start (dx_1, dy_1, v_1) end (dx_2, dx_2,

v_2) mean (μ_{dx}, μ_{dy}, μ_v) and standard deviation (σ_{dx}, σ_{dy}, σ_v).

5. Object size at first occurence (w_1, h_1), last one (w_2, h_2), mean (μ_w, μ_h) and standard deviation (σ_w, σ_h).
6. Average color (layout) based on JPEG compression technique of 8x8 pixel object resampled in Y'CbCr color space, from which are zig-zag extracted DCT coefficients. We use 15 (Y) + 2*10 (Cb and Cr) coefficients ($c_{1..35}$).
7. Object shape using central normalized moments up to the third order (η_{20}, η_{11}, η_{02}, η_{30}, η_{21}, η_{12}, η_{03}) computed [4] from segmented image (alpha channel).

Because the shape moments do not give good classification results when aggregated (average), we have created a separate training case. The trajectory is split into 4 segments and their border shapes are extracted and concatenated into a feature vector. We refer this classification scheme as “Shape”.

Because the VRM Analysis module uses various classifiers, we have adopted the fully-probabilistic combination of their results. For SED 2012, we used simple naïve Bayes combination.

The interactive interface

Human Monitoring Interface (HMI) is capable not only of simple monitoring the area, but also querying monitored object(s) based on its previous occurrences, visual properties and behavior. The behavior is either a detected event or (statistical) analysis of the objects' spatio-temporal properties in the global context, such as who met who, where was who when something happened or some other nontrivial analysis based on statistics and data mining – using VRM.

For SED 2012 we have simplified the user interface to make the annotations as simple as possible. We have two modes of the SED12 Annotator as illustrated in figures 1 and 4.

The first mode (annotator) was used for Round #1 annotations. The GUI shows the output of blob-track algorithm and cuts the shots where an event is expected accordingly to the LDC's annotations. The goal of the #1 is to match the event and the exact object for the learner. It can be done in two ways – either a human annotator can click the objects (subjects) that are concerned in the event or type their trajectory numbers when occluded. Most events, however need just a single number, but Embrace, ObjectPut, PeopleMeet and PeopleSplitUp have eventually two or more objects involved. In 2012, we haven't investigated their mutual relationships, because the lack of annotations.

After around 1000 events were annotated during Round #1 – see table 1 for details. This took about 20 hours. We have performed the learning and classification of other objects within intervals specified in LDC's annotations then. Accordingly to their probability (and grouped by videos for performance reasons), they were presented to the human annotator in the second mode (validator). In this mode (figure 4), a human annotator is supposed press “1” (1+) if the event belongs to the object highlighted or “0” (Enter) else. We have annotated about 1400 events in less than 6 hours.

Because of the simplicity, Round #2 annotations/validations were considered “extremely boring” in contrast to “just boring” Round #1 annotation. Thus, for the purpose of evaluations - the 25 min. “interactive” annotations (Round #3) the evaluation videos were played faster (150%) and because there was just one object marked including the whole trajectory, it was still well decidable (she considered it “high-dynamically boring”). See the “interactiveED” attachment for details.

3.2 VTApi

VTApi is an application programming interface designed to fulfill the needs of specific distributed computer vision systems and to unify and accelerate their development. It is oriented towards processing and efficient management of image and video data and related metadata for their retrieval, characterization and intelligent analysis with the special emphasis on their spatio-temporal nature in real-world conditions. VTApi is a free extensible framework based on progressive and scalable open source software as OpenCV for high-performance computer vision and pattern recognition, PostgreSQL for efficient data management, indexing and retrieval extended by similarity search and geography/spatio-temporal data manipulation.

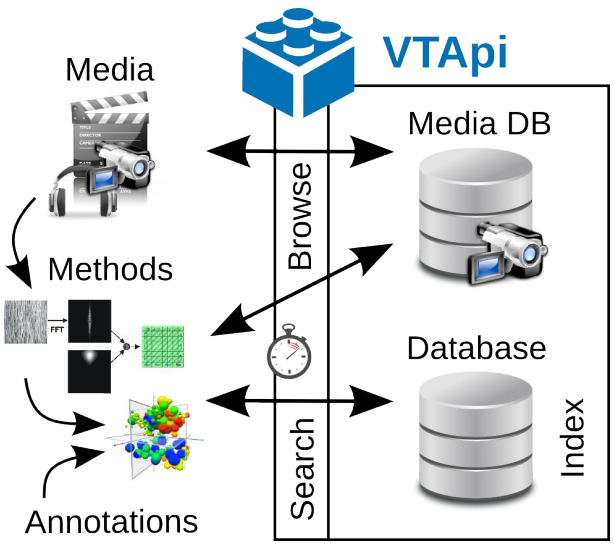


Fig. 5. The illustration of a position of the VTApi and a concept of methods’ chaining.

```
Mat samples; // cv::Mat of training feature vectors
VTApi* vtapi = new VTApi(argc, argv);
Dataset* dataset = vtapi->newDataset("train"); // training dataset
dataset->next();
Sequence* sequence = dataset->newSequence();
 while (sequence->next()) { // for each video
    Interval* track = new Interval(*sequence, "tracks");
    while (track->next()) { // for each trajectory
        Mat sample; // cv::Mat for feature vector of trajectory
        float feature = track->getFloat("feature");
        // ... read features and fill feature vector
        samples.push_back(sample);
    }
}
CvEM model, labels; // GMM-EM model and cluster labels
CvEMParams params; // EM parameters
// ... set EM parameters including number of clusters
model.train(samples, Mat(), params, labels);
// ... choose dataset and sequences according to sample code above
while (track->next()) {
    Mat sample;
    // ... read features and fill feature vector
    int cluster = (int) model.predict(sample); // get cluster label
    track->setInt("cluster", cluster); // store cluster label
}
```

Fig. 6. The illustration of a position of the VTApi and a concept of methods' chaining (a). Sample code of reading trajectories, preparing training samples, GMM training and storing cluster labels into the database (b).

The main objective of the VideoTerror (VT) project is to define, explore and create a prototype of a system warehousing image and video accomplished with computer vision and analytics based on a computer cluster. The basic requirements include image and video feature extraction, storage and indexing to enable (content-based) retrieval, summarization and characterization together with video analytics in the meaning of object detection and recognition in an interactive and iterative process.

In addition to the technology, we also target usual aspects of the research – to unify and accelerate it by choosing an appropriate design methodology and architectural framework for the composition of domain and application specific tools focusing on open source software. In particular, we propose a solution that will enable the development and adaptation of a complex computer vision application at a reduced cost in terms of time and money. We target this goal by (re)using and integrating tool chains of (CV) methods and (multimedia) data and metadata in an arbitrary combination as simple and versatile as possible.

The VT methodology is based on the fact, that most methods of the same purpose have similar types of inputs and outputs, so there may be a base class for each of them. Moreover, the input of a process (a running instance of a method) can be seen as another process's output (e.g., annotation, feature extraction, classification) including media data creation, which has the null input. Methods are generally not included in the API self, they are created by developers using the API. In this way, methods and their chains can be created and reused on multimedia data and metadata for specific applications by providing a unified platform (API) for processing and management of both video and still image sets using custom methods. This is illustrated in figure below.

The VT project is not limited to a specific kind of data as Internet archives of image and video or closed-circuit television, so the framework can be used to support any CV evaluation campaign. In the surveillance video, it is important to be able to track moving objects and to extract their visual and spatio-temporal features. Such extracted metadata then should be cleaned, stored and indexed to be able to query and analyze it.

Object tracking is considered difficult especially in crowded scenes. The outputs of such methods include spatio-temporal locations in the form of trajectories, blobs and other features of moving objects, which can be aggregated, summarized, analyzed and enriched with additional features such as annotations, tags or classes. We transform the blobtrack trajectories and store them in the database as vectors capable of the first order temporal interpolation even stored as discrete points, and we index them using 3D bounding-cube and GiST (bitmap index), so that they can be retrieved very efficiently (in fact, it is an array of points).

All the above features might be used and searched for similarity by VTApi. A trajectory query may relate either to relationships between moving objects or a specific spatio-temporal region. Such an analysis can be performed both on VTApi clients and server, because we have adopted the OpenGIS GEOS library, that has been adopted by PostGIS. In order to perform these operations efficiently, VTApi adds a binary access to geometry types and n-dimensional cubes that are used as spatio-temporal minimum bounding boxes of (moving) objects (as described in ???).

We support trajectory clustering, classification, object recognition, outliers detection and so on. The following example shows a clustering of trajectories using VTApi and an OpenCV implementation of Expectation-maximization (EM) algorithm, which estimates parameters of a Gaussian mixture model (GMM) \cite{Bishop}. First, feature vectors representing trajectories are read from the database and training samples for the EM algorithm are prepared (see figure above). Suppose that trajectories are stored in "tracks" in this example. Second, GMM is trained by the EM algorithm and appropriate cluster labels are stored in the database see figure *above*.

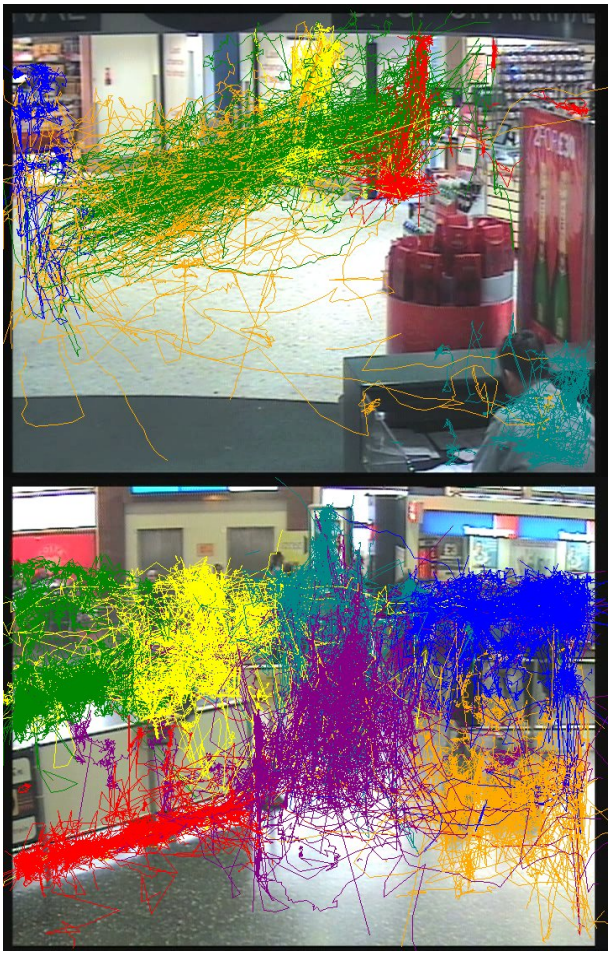


Fig. 7. Examples of trajectory clustering results obtained by EM algorithm on trajectories from the first camera (top) and by k-means algorithm on trajectories from the third camera (bottom).

We performed the trajectory clustering on a set of trajectories extracted from the the i-LIDS dataset of five cameras at the LGW airport. An example of visualization of some obtained results is shown in Fig. below. Different colors of trajectories refer to different clusters. On the left, there is a result of clustering trajectories from the first camera using the EM algorithm mentioned above. On the right, there is a result of clustering trajectories from the third camera by the K-means clustering algorithm to show the easy changeability of methods of the same purpose. We have prepared also an outliers analysis within the Video Terror project.

4. Experiments

We have performed two rounds of the active learning process during the development and training. They are described in section 3.1 – The interactive interface. The table 1 presents the numbers of theoretical (LDC), Round #1 and Round #2 annotations after 20 and 6 hour of continuous burden, which is the reason we haven't used all the annotations suggested (coping with the unsatisfactory tracker results of heavily occluded objects and the overall quality of data, because some events take just a few pixels).

Table 1. Numbers of annotated objects.

Event	#LDC	#1	#2
CellToEar	828	80	120
ElevatorNoEntry	12	4	5
Embrace	940	75	138
ObjectPut	3172	181	422
OpposingFlow	34	1	4
PeopleMeet	2718	282	717
PeopleSplitUp	1571	122	441
PersonRuns	673	59	153
Pointing	4095	235	478
TakePicture	30	0	0
Sum (distinct)	<< 14073	**944**	**2280**

Table 2 presents the SVM – based classification accuracy of optimized classification schemes “Track” and “Shape”, as described in section 3.1 - Analysis and classification. Note, that Round #3 classification data is of about 1GB and the whole database is about 20GB (compared to 300GB video data). A single learning process of an average classification model is about 20 seconds. We performed 5 (fold) * 100 (parameter selection) learning processes, which takes about 30 minutes, performed 9 times in parallel. We have considered 9 distinct events – omitting the TakePicture, because we were unable to asses who is taking the picture in the devel recordings.

Table 2. Prediction accuracy of 5-fold cross-validation on training data.

	Round #1		Round #2	
Event	Tracks	Shapes	Tracks	Shapes
CellToEar	91.52	91.53	94.78	94.74
ElevatorNo...	99.79	99.68	99.86	99.78
Embrace	92.05	92.06	94.04	93.99
ObjectPut	81.25	81.03	81.71	81.54
OpposingFlow			99.82	
PeopleMeet	71.93	70.55	69.69	68.59
PeopleSplitUp	87.29	87.08	81.23	80.75
PersonRuns	94.28	93.75	95.31	93.29
Pointing	76.27	75.21	80.88	79.04
TakePicture				
Average	86.80	86.36	88.59	86.47

Evaluation results

To be updated. The retrospective task maximized recall, while the interactive task maximized precision.

Not bad for the first time && processing just 10% of annotations (see SED slides).

5. Conclusions

In the paper, we present an open source single-camera computer vision based surveillance event detection system SUNAR-ED (see sourceforge.net/p/sunar-ed). We have selected, integrated and extended a set of a state of the art progressive and robust open source tools efficient for multimedia data and related metadata storage, indexing, retrieval and analysis. It contains both standard and experimental techniques evaluated at the AVSS 2009 Multi-Camera Tracking Challenge.

SUNAR is composed of three basic modules - video processing, retrieval and the monitoring interface. Computer Vision Modules are based on the OpenCV library for object tracking providing cleaned and transformed trajectory and shape-based features based on tracking data (objects subtracted from background) – color and shape descriptors similarly to MPEG-7.

Information about objects and the area under surveillance is cleaned, integrated, indexed and stored in Video Retrieval Modules. They are based on the PostgreSQL database extended to be capable of similarity and spatio-temporal information retrieval. We have integrated many machine-learning methods as Bayes classifiers, SVM, EM/GMM, HMM, frequent pattern analysis and various clustering algorithms.

We have focused mainly on active learning and semi-automatic annotation generation for future evaluations. We have developed a simple (yet boring) user interface, which can reduce the burden of continuous concentration on monitoring and increase the effectiveness of security personnel.

Together with SUNAR-ED, we offer to the public data and metadata management framework – VTApi (application programming interface, see gitorious.org/vtapi). The main advantages of the API is the reduction of effort and time to produce quality intelligent vision applications by unified and reusable both methods and data sets of video, image, metadata and features on all levels. We offer data and methods interfaces and a methodology to be used by researchers and developers of both academic and commercial sectors to collaborate and chain their efforts. Using VTApi, we have developed tools to be (re)used in future. At the moment, they serve as technology demonstrators only.

We have submitted two SED runs – retroED and interactiveED. The retrospective task tried maximized recall, while the interactive task focused on maximizing precision. The results are not comparably bad in spite of processing about 10% of annotations.

Overall conclusions

We have to thank all the people in NIST and groups providing data, annotations, evaluation metrics, all the human and computer power. We think that this is the real force of TRECVid, together with the inspiration from and of all the participants and groups.

References

[1] B. Settles, *Active Learning*. Morgan & Claypool Publishers, 2012.

[2] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques: Concepts and Techniques*. Elsevier, 2011.

[3] X. Zhu and A. B. Goldberg, “Introduction to Semi-Supervised Learning,” *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–130, Jan. 2009.

[4] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, 1st ed. O’Reilly Media, 2008.

[5] W. Hu, T. Tan, L. Wang, and S. Maybank, “A survey on visual surveillance of object motion and behaviors,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334 –352, Aug. 2004.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Springer, 2007.

[7] E. J. Carmona, J. Martínez-Cantos, and J. Mira, “A new video segmentation method of moving objects based on blob-level knowledge,” *Pattern Recogn. Lett.*, vol. 29, no. 3, pp. 272–285, Feb. 2008.

[8] P. Chmelar, I. Rudolfova, and J. Zendulka, “Clustering for Video Retrieval,” in *Data Warehousing and Knowledge Discovery*, 2009, pp. 390–401.

[9] P. Chmelar and J. Zendulka, “Visual Surveillance Metadata Management,” in *Eighteenth International Workshop on Database and Expert Systems Applications*, 2007, pp. 79–83.

[10] B. S. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, 2002.

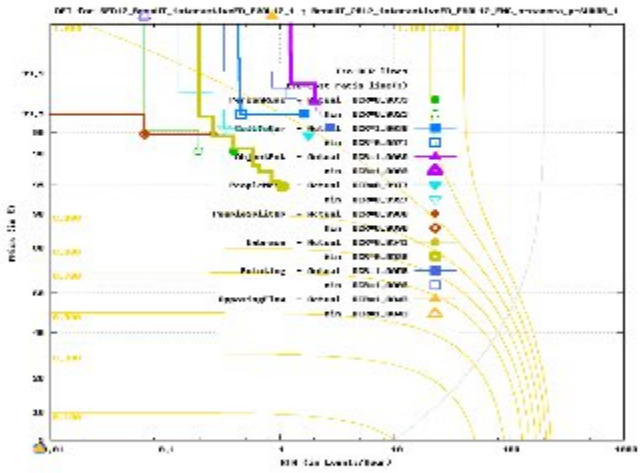
[11] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[12] P. Chmelar, A. Lanik, and J. Mlich, “SUNAR: Surveillance Network Augmented by Retrieval,” in *ACIVS 2010*, 2010, pp. 155–166.

SITEID : BrnoUT (Brno University of Technology)
Experiment ID : BrnoUT_2012_interactiveED_EVAL12_ENG_s-camera_p-SUNAR_1 (Primary interactiveED)
Events Processed : CellToEar, ElevatorNoEntry, Embrace, ObjectPut, OpposingFlow, PeopleMeet, PeopleSplitUp, PersonRuns, Pointing

Scoring Parameters	Duration	Cost False Alarm	Cost Miss	Rate of Target	*BETA*
Value	54890	1	10	20	0.005

	Inputs			Actual Decision DCR Analysis								Minimum DCR Analysis			
Title	#Targ	#NTarg	#Sys	#CorDet	#Cor!Det	#FA	#Miss	RFA	PMiss	DCR	Dec. Tresh	RFA	PMiss	DCR	Dec. Thresh
CellToEar	194	25	26	1	0	25	193	1.63964	0.995	1.0030	0.1870	0.45910	0.995	0.9971	0.190
Embrace	175	17	26	9	0	17	166	1.11496	0.949	0.9541	0.2600	1.04937	0.949	0.9538	0.261
ObjectPut	621	31	33	2	0	31	619	2.03316	0.997	1.0069	0.6930	0.06559	1.000	1.0003	0.907
OpposingFlow	1	13	13	0	0	13	1	0.85261	1.000	1.0043	0.0000	0.85261	1.000	1.0043	0.000
PeopleMeet	449	27	32	5	0	27	444	1.77081	0.989	0.9977	0.8010	0.32793	0.991	0.9927	0.935
PeopleSplitUp	187	4	6	2	0	4	185	0.26234	0.989	0.9906	0.4880	0.06559	0.989	0.9896	0.521
PersonRuns	107	6	8	2	0	6	105	0.39351	0.981	0.9833	0.3450	0.19676	0.981	0.9823	0.420
Pointing	1063	42	51	9	0	42	1054	2.75460	0.992	1.0053	0.7150	0.06559	1.000	1.0003	0.869



SITEID : BrnoUT (Brno University of Technology)
Experiment ID : BrnoUT_2012_retroED_EVAL12_ENG_s-camera_p-SUNAR_1 (Primary retroED)
Events Processed : CellToEar, ElevatorNoEntry, Embrace, ObjectPut, OpposingFlow, PeopleMeet, PeopleSplitUp, PersonRuns, Pointing

Scoring Parameters	Duration	Cost False Alarm	Cost Miss	Rate of Target	*BETA*
Value	54890	1	10	20	0.005

	Inputs			Actual Decision DCR Analysis								Minimum DCR Analysis			
Title	#Targ	#NTarg	#Sys	#CorDet	#Cor!Det	#FA	#Miss	RFA	PMiss	DCR	Dec. Tresh	RFA	PMiss	DCR	Dec. Thresh
CellToEar	194	989	1000	11	0	989	183	64.86427	0.943	1.2676	0.1870	0.59027	0.995	0.9978	0.204
Embrace	175	976	1000	24	0	976	151	64.01166	0.863	1.1829	0.2490	0.06559	1.000	1.0003	1.224
ObjectPut	621	969	1000	31	0	969	590	63.55256	0.950	1.2678	0.6900	0.13117	1.000	1.0007	0.976
OpposingFlow	1	1000	1000	0	0	1000	1	65.58572	1.000	1.3279	0.0000	65.58572	1.000	1.3279	0.000
PeopleMeet	449	959	1000	41	0	959	408	62.89670	0.909	1.2232	0.7920	0.00000	0.998	0.9978	1.148
PeopleSplitUp	187	967	1000	33	0	967	154	63.42139	0.824	1.1406	0.4880	0.06559	1.000	1.0003	0.756
PersonRuns	107	982	1000	18	0	982	89	64.40517	0.832	1.1538	0.3100	5.11569	0.972	0.9975	0.668
Pointing	1063	946	1000	54	0	946	1009	62.04409	0.949	1.2594	0.7140	0.06559	1.000	1.0003	0.962