Silhouette-Based Method for Object Classification and Human Action Recognition in Video

Yiğithan Dedeoğlu[1], B. Uğur Töreyin[2], Uğur Güdükbay[1], and A. Enis Çetin[2]

[1] Bilkent University, Department of Computer Engineering
{yigithan, gudukbay}@cs.bilkent.edu.tr
[2] Department of Electrical and Electronics Engineering,
06800, Bilkent, Ankara, Turkey
{bugur, cetin}@bilkent.edu.tr

Abstract. In this paper we present an instance based machine learning algorithm and system for real-time object classification and human action recognition which can help to build intelligent surveillance systems. The proposed method makes use of object silhouettes to classify objects and actions of humans present in a scene monitored by a stationary camera. An adaptive background subtraction model is used for object segmentation. Template matching based supervised learning method is adopted to classify objects into classes like human, human group and vehicle; and human actions into predefined classes like walking, boxing and kicking by making use of object silhouettes.

1 Introduction

Classifying types and understanding activities of moving objects in video is both a challenging problem and an interesting research area with many promising applications. Our motivation in studying this problem is to design a human action recognition system that can be integrated into an ordinary visual surveillance system with real-time moving object detection, classification and activity analysis capabilities. The system is therefore supposed to work in real time. Considering the complexity of temporal video data, efficient methods must be adopted to create a fast, reliable and robust system. In this paper, we present such a system which operates on gray scale video imagery from a stationary camera.

In the proposed system moving object detection is handled by the use of an adaptive background subtraction scheme which reliably works both in indoor and outdoor environments [7].

After segmenting moving pixels from the static background of the scene, connected regions are classified into predetermined object categories: human, human group and vehicle. The classification algorithm depends on the comparison of the silhouettes of the detected objects with pre-labeled (classified) templates in an object silhouette database. The template database is created by collecting sample object silhouettes from sample videos and labeling them manually with appropriate categories. The silhouettes of the objects are extracted from the connected foreground regions by using a contour tracing algorithm [11].

T.S. Huang et al. (Eds.): HCI/ECCV 2006, LNCS 3979, pp. 64 – 77, 2006.

The action recognition system also exploits objects' silhouettes obtained from video sequences to classify actions. It mainly consists of two major steps: manual creation of silhouette and action templates offline and automatic recognition of actions in real-time. In classifying actions of humans into predetermined classes like walking, boxing and kicking; temporal signatures of different actions in terms of silhouette poses are used.

The remainder of this paper is organized as follows. Section 2 gives an overview of the related work. In the next two sections we give the details of moving object segmentation and object classification. In the next section, visual action recognition system is explained. Experimental results are discussed in Section 6 and finally we conclude the paper with Section 7.

2 Related Work

There have been a number of surveys about object detection, classification and human activity analysis in the literature [1, 9, 26].

Detecting regions corresponding to moving objects such as people and vehicles in video is the first basic step of almost every vision system because it provides a focus of attention and simplifies the processing on subsequent analysis steps. Due to dynamic changes in natural scenes such as sudden illumination and weather changes, repetitive motions that cause clutter (tree leaves moving in blowing wind), motion detection is a difficult problem to process reliably. Frequently used techniques for moving object detection are background subtraction, statistical methods, temporal differencing and optical flow [10, 12, 17, 22, 23, 26].

Moving regions detected in video may correspond to different objects in real-world such as pedestrians, vehicles, clutter, etc. It is very important to recognize the type of a detected object in order to track it reliably and analyze its activities correctly. Currently, there are two major approaches towards moving object classification which are shape-based and motion-based methods [26]. Shape-based methods make use of the objects' 2D spatial information like bounding rectangle, area, silhouette and gradient of detected object regions; whereas motion-based methods use temporally tracked features of objects for the classification solution.

The approach presented in [15] makes use of the objects' silhouette contour length and area information to classify detected objects into three groups: human, vehicle and other. The method depends on the assumption that humans are, in general, smaller than vehicles and have complex shapes. Dispersedness is used as the classification metric and it is defined as the square of contour length (perimeter) over object's are. Classification is performed at each frame and tracking results are used to improve temporal classification consistency.

The classification method developed by Collins et al. [7] uses view dependent visual features of detected objects to train a neural network classifier to recognize four classes: human, human group, vehicle and clutter. The inputs to the neural network are the dispersedness, area and aspect ratio of the object region and the camera zoom magnification. Like the previous method, classification is performed at each frame and results are kept in a histogram to improve temporal consistency of classification.

Some of the methods in the literature use only temporal motion features of objects in order to recognize their classes [6, 14, 27]. In general, they are used to distinguish non-rigid objects (e.g. human) from rigid objects (e.g. vehicles). The method proposed in [6] is based on the temporal self-similarity of a moving object. As an object that exhibits periodic motion evolves, its self-similarity measure also shows a periodic motion. The method exploits this clue to categorize moving objects using periodicity.

The systems for action recognition using video can be divided into three groups according to the methods they use: general signal processing techniques to match action signals, template matching and state-space approaches.

The first group treats the action recognition problem as a classification problem of the temporal activity signals of the objects according to pre-labeled reference signals representing typical human actions [26]. For instance Kanade et al. makes use of the signals generated by the change of the angle between the torso and the vertical line that passes through a human's body to distinguish walking and running patterns [7]. In another work Schuldt et al. make use of a local SVM approach to define local properties of complex motion patterns and classify the patterns using well known popular classifier Support Vector Machine [21]. General methods such as Dynamic time warping, Hidden Markov models and Neural Networks are used to process the action signals.

Second group of approaches converts image sequences into static shape patterns and in the recognition phase compares the patterns with pre-stored ones. For instance by using PCA, Chomat et al. created motion templates and a Bayes classifier was used to perform action recognition [4].

The last group considers each pose of the human body as a state and calculates a probability density function for each different action sequences [24]. A sequence can be thought of as a tour between different states. Hence the probability density function can be calculated from different tours of the same action. The probability functions than can be used to recognize test sequences.

3 Learning Scene Background for Segmentation

We use a combination of a background model and low-level image post-processing methods to create a foreground pixel map and extract object features at every video frame. Our implementation of background subtraction algorithm is partially inspired by the study presented in [7] and works on grayscale video imagery from a static camera. Background subtraction method initializes a reference background with the first few frames of video input. Then it subtracts the intensity value of each pixel in the current image from the corresponding value in the reference background image. The difference is filtered with an adaptive threshold per pixel to account for frequently changing noisy pixels. The reference background image and the threshold values are updated with an IIR filter to adapt to dynamic scene changes.

Let $I_n(x)$ represent the gray-level intensity value at pixel position (x) and at time instance n of video image sequence I which is in the range [0, 255]. Let $B_n(x)$ be the corresponding background intensity value for pixel position (x) estimated over time from video images I_0 through I_n-1. As the generic background subtraction scheme

suggests, a pixel at position (x) in the current video image belongs to foreground if it satisfies:

$$|I_n(x) - B_n(x)| > T_n(x)$$

where $T_n(x)$ is an adaptive threshold value estimated using the image sequence I_0 through I_n-1. The above equation is used to generate the foreground pixel map which represents the foreground regions as a binary array where a 1 corresponds to a foreground pixel and a 0 stands for a background pixel. The reference background $B_n(x)$ is initialized with the first video image I_0, $B_0 = I_0$, and the threshold image is initialized with some pre-determined value (e.g. 15).

Since this system will be used in outdoor environments as well as indoor environments, the background model needs to adapt itself to the dynamic changes such as global illumination change (day night transition) and long term background update (parking a car in front of a building). Therefore the reference background and threshold images are dynamically updated with incoming images. The update scheme is different for pixel positions which are detected as belonging to foreground ($x \in FG$) and which are detected as part of the background ($x \in BG$):

$$B_{n+1}(x) = \begin{cases} \alpha B_n(x) + (1-\alpha) I_n(x), & x \in BG \\ \beta B_n(x) + (1-\beta) I_n(x), & x \in FG \end{cases}$$

$$T_{n+1}(x) = \begin{cases} \alpha T_n(x) + (1-\alpha)(\gamma \times |I_n(x) - B_n(x)|), & x \in BG \\ T_n(x), & x \in FG \end{cases}$$

where α, β and γ ($\in$ [0.0, 1.0]) are learning constants which specify how much information from the incoming image is put to the background and threshold images.

The output of foreground region detection algorithm generally contains noise and therefore is not appropriate for further processing without special post-processing. Morphological operations, erosion and dilation [11], are applied to the foreground pixel map in order to remove noise that is caused by the first three of the items listed above. Our aim in applying these operations is to remove noisy foreground pixels that do not correspond to actual foreground regions and to remove the noisy background pixels near and inside object regions that are actually foreground pixels.

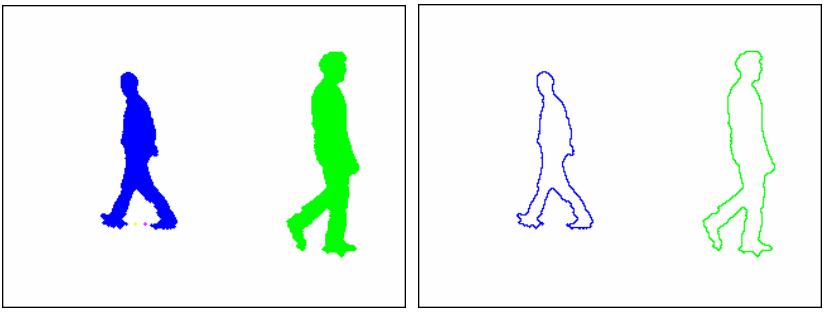

Fig. 1. Sample objects and their silhouettes

3.1 Calculating Object Features

After detecting foreground regions and applying post-processing operations to remove noise and shadow regions, the filtered foreground pixels are grouped into connected regions (blobs) and labeled by using a two-level connected component labeling algorithm presented in [11]. After finding individual blobs that correspond to objects, spatial features like bounding box, size, center of mass and silhouettes of these regions are calculated.

In order to calculate the center of mass point, $C_m = (x_{Cm}, y_{Cm})$, of an object O, we use the following equation [18]:

$$x_{C_m} = \frac{\sum_i^n x_i}{n}, \quad y_{C_m} = \frac{\sum_i^n y_i}{n}$$

where n is the number of pixels in O.

Both in offline and online steps of the classification algorithm, the silhouettes of the detected object regions are extracted from the foreground pixel map by using a contour tracing algorithm presented in [11]. Figure 1 shows sample detected foreground object regions and the extracted silhouettes. Another feature extracted from the object is the silhouette distance signal. Let $S = \{p_1, p_2, \ldots, p_n\}$ be the silhouette of an object O consisting of n points ordered from top center point of the detected region in clockwise direction and C_m be the center of mass point of O. The distance signal $DS = \{d_1, d_2, \ldots, d_n\}$ is generated by calculating the distance between C_m and each p_i starting from 1 through n as follows:

$$d_i = Dist(C_m, p_i), \quad \forall\ i \in [1 \ldots n]$$

where the *Dist* function is the Euclidian distance.

Different objects have different shapes in video and therefore have silhouettes of varying sizes. Even the same object has altering contour size from frame to frame. In order to compare signals corresponding to different sized objects accurately and to make the comparison metric scale-invariant we fix the size of the distance signal. Let N be the size of a distance signal DS and let C be the constant for fixed signal length. The fix-sized distance signal $\widehat{DS}$ is then calculated by sub-sampling or super-sampling the original signal DS as follows:

$$\widehat{DS}[i] = DS[i * \frac{N}{C}], \quad \forall\ i \in [1 \ldots C]$$

In the next step, the scaled distance signal $\widehat{DS}$ is normalized to have integral unit area. The normalized distance signal $\overline{DS}$ is calculated using the following equation:

$$\overline{DS}[i] = \frac{\widehat{DS}[i]}{\sum_1^n \widehat{DS}[i]}$$

Figure 2 shows a sample silhouette and its original and scaled distance signals.

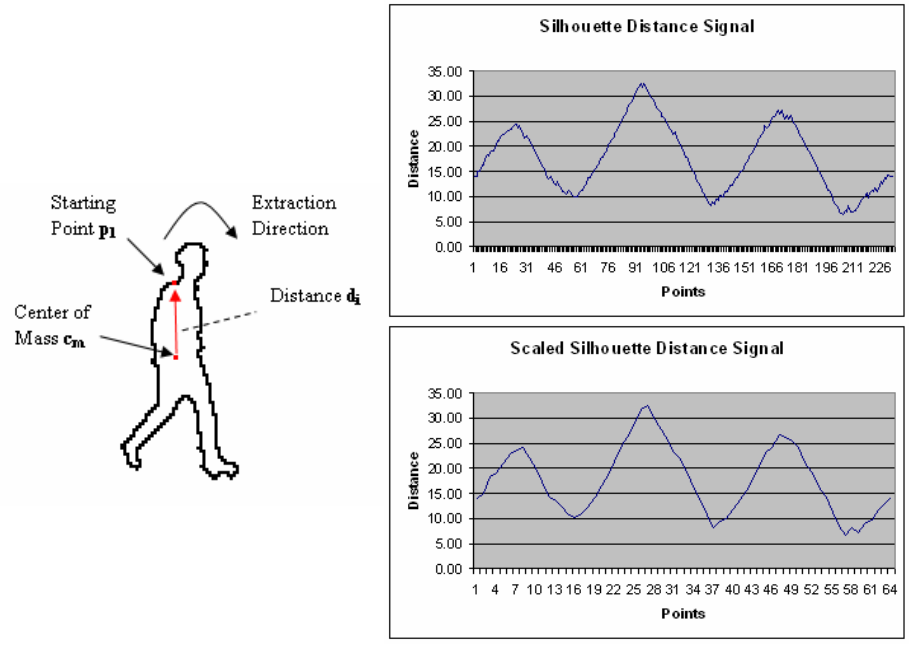

Fig. 2. Sample distance signal calculation and normal and scaled distance signals

4 Classifying Objects

The ultimate aim of different smart visual surveillance applications is to extract semantics from video to be used in higher level activity analysis tasks. Categorizing the type of a detected video object is a crucial step in achieving this goal. With the help of object type information, more specific and accurate methods can be developed to recognize higher level actions of video objects. Hence, we present a video object classification method based on object shape similarity to be used as a part of a "smart" visual surveillance system.

Typical video scenes may contain a variety of objects such as people, vehicles, animals, natural phenomenon (e.g. rain, snow), plants and clutter. However, main target of interest in surveillance applications are generally humans and vehicles.

The classification metric used in our method measures object similarity based on the comparison of silhouettes of the detected object regions extracted from the foreground pixel map with pre-labeled (manually classified) template object silhouettes stored in a database. The whole process of object classification method consists of two steps:

- Offline step: A template database of sample object silhouettes is created by manually labeling object types.
- Online step: The silhouette of each detected object in each frame is extracted and its type is recognized by comparing its silhouette based feature with the

ones in the template database in real time during surveillance. After the comparison of the object with the ones in the database, a template shape with minimum distance is found. The type of this object is assigned to the type of the object which we wanted to classify.

The template silhouette database is created offline by extracting several object contours from different scenes. Since the classification scheme makes use of object similarity, the shapes of the objects in the database should be representative poses of different object types. Figure 3 shows the template database we use for object classification. It consists of 24 different poses: 14 for human, 5 for human group and 5 for vehicles.

In classification step, our method does not use silhouettes in raw format, but rather compares converted silhouette distance signals. Hence, in the template database we store only the distance signal of the silhouette and the corresponding type information for both computational and storage efficiency.

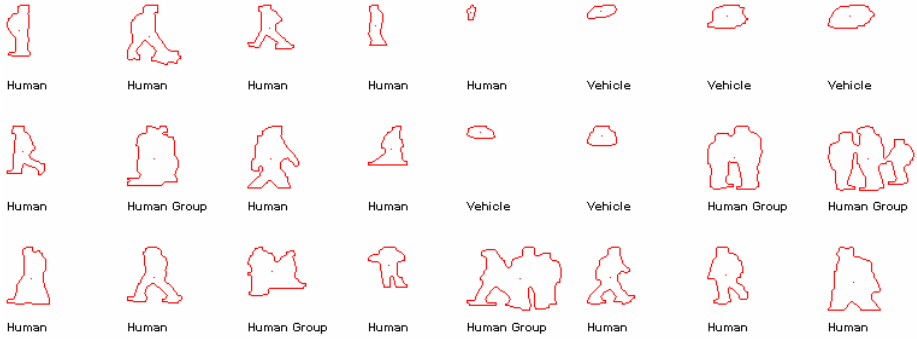

Fig. 3. Sample object silhouette template database

4.1 Classification Metric

Our object classification metric is based on the similarity of object shapes. There are numerous methods in the literature for comparing shapes [20, 5, 18, 2, 13]. The reader is especially referred to the surveys presented in [25, 16] for good discussions on different techniques.

Our classification metric compares the similarity between the shapes of two objects, A and B, by finding the distance between their corresponding distance signals, DS_A and DS_B. The distance between two scaled and normalized distance signals, DS_A and DS_B is calculated as follows:

$$Dist_{AB} = \sum_{i=1}^{n} \left| \overline{DS}_A[i] - \overline{DS}_B[i] \right|$$

In order to find the type T_O of an object O, we compare its distance signal DS_O with all of the objects' distance signals in the template database. The type T_P of the

template object P is assigned as the type of the query object O, $T_O = T_P$ where P satisfies the following:

$$Dist_{OP} \leq Dist_{OI}, \quad \forall \text{ object } I \text{ in the template database}$$

Figure 4 shows the silhouettes, silhouette signals and signal distances of a sample query object and template database objects for type classification.

Distance between two objects can be computed using more sophisticated methods such as dynamic programming providing a nonlinear warping of the horizontal axis [3] instead of the linear warping used in the calculation $Dist_{AB}$. However, a straightforward implementation of dynamic programming increases computational complexity and may not be suitable for the purposes of a real-time system.

In order to reduce noise in object classification a maximum likelihood scheme is adopted. The assigned object types are counted for a window of k (= 5) frames and the maximum one is assigned as the type. This reduces false classifications due to errors in segmentation.

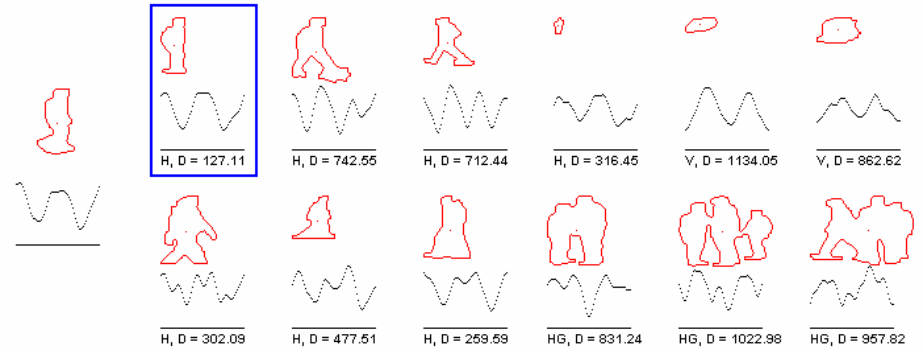

Fig. 4. Sample query object and its distances (D) to several objects in the silhouette template database. Object types are Human (H), Human Group (HG) and Vehicle (V). The matching object is shown with the bounding rectangle.

5 Recognizing Human Actions

After detecting the type of an object, if it is a human, its actions can be recognized. The action recognition system can recognize six different human actions which are: walking, boxing and kicking. Figure 6 shows video frames from sample sequences for these action types. The whole process of human action recognition method consists of two steps:

- Offline step: A pose template database by using human silhouettes for different poses is created. The silhouettes in this database are used to create a pose histogram which is used as an action template. An action template database is created by using these histograms calculated from sample action sequences.

- Online step: The silhouette of each detected human in each frame is extracted and its pose is matched with one in the pose template database. Then a histogram of the matched poses is created at each frame by using a history window of the matched human poses. Then the calculated histogram is matched against the ones in the template action database, and the label of the action with minimum distance is assigned as the current action label.

5.1 Creating Silhouette-Based Pose Template Database

A typical human action such as walking involves repetitive motion. Although throughout a video sequence several hundreds of silhouettes can be extracted for a subject, the shapes of the silhouettes will exhibit an almost periodic similarity. Hence, the basic set of shapes for a full period can represent an action. Furthermore, the key poses in the basic motion set show differences from action to action. For instance, the silhouettes of a walking person from side view can be represented with three key poses corresponding to the cases of full stretched legs, closed legs and partially stretched legs. Similarly, the boxing action again can be represented with two key poses: (i) one arm is stretched and (ii) both arms are near the chest. Some of the possible poses that can be seen during walking action are shown in Figure 5 with an ID number beneath.

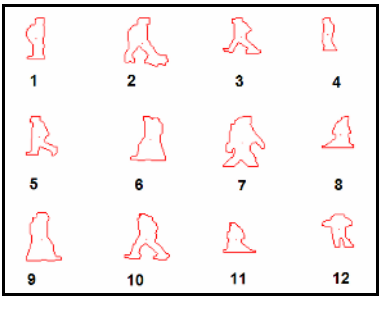

Fig. 5. Sample silhouettes from a walking sequence

The template pose database is manually created with extracted object silhouettes as shown in Figure 5 and contains key poses for all of the actions that can be recognized by the system. The pose silhouettes are labeled with integer IDs in the range [1, ID_{MAX}]. The template database which we used in our tests contains 82 poses for different actions.

5.2 Creating Action Template Database

After creating the pose database the next step is to create action templates. Actions can be represented with a histogram of key poses (pose IDs) it matches. In other words, if we create a histogram of the size of the total number of key silhouettes in the silhouette template database, and match generated silhouettes at each frame of the

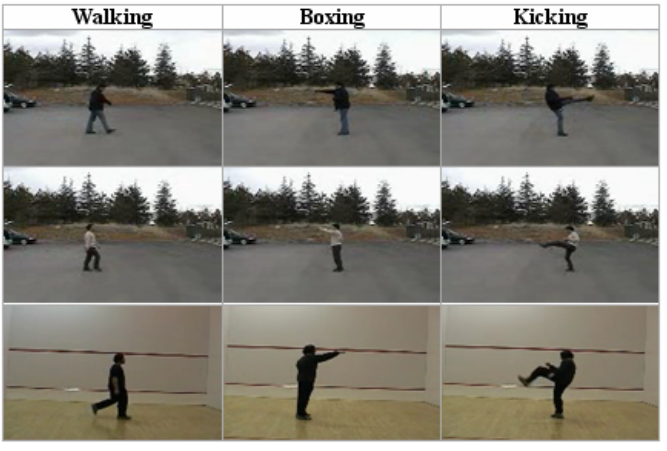

Fig. 6. Sample video frames for different action types

training action sequence, to a key pose in the template silhouette database and increase the value of the corresponding bin (key pose's ID) in the histogram, we can create a histogram for the action. Formally, let $A = \{S_1, S_2, \ldots, S_i, \ldots, S_N\}$ be a sequence of silhouettes extracted from a sample motion of a human subject at each video frame $i \in [1, N]$.

Then for each S_i a corresponding pose match P_i is found in the silhouette pose template database by using the distance metric explained in Section 3.1. Let $L = \{P_1, P_2, \ldots, P_N\}$ represent the list of matched poses, where $P_i \in [1, ID_{MAX}]$. Then the list L can be used to create a histogram H (with ID_{MAX} bins) of IDs. After the histogram is created, it is normalized to have unit area and made ready to represent an action template like a signature. A histogram H_j is created in this manner for each action j, $j \in \{Walking, Boxing, Kicking\}$, and these histograms form the action template database. Figure 7 shows sample action histograms for each action.

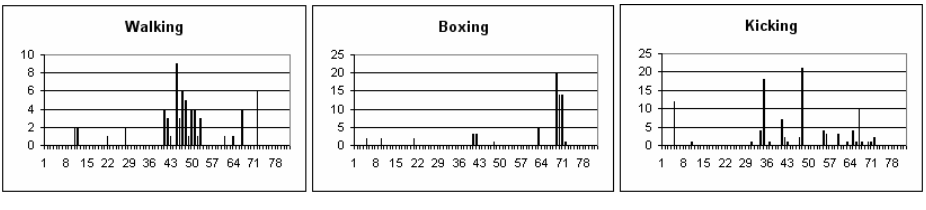

Fig. 7. Un-normalized histograms for actions in the template database

5.3 Recognizing Actions in Real-Time

After creating action template database with histograms for distinct actions, test actions are recognized in real-time.

In order to recognize an action, we keep a circular list of the IDs of the matching silhouettes in the template pose database for the subject's silhouette.

Let $A_T = \{S_{i-(w-1)}, S_{i-(w-2)}, \ldots, S_i\}$ be the fixed length list of the silhouettes of a test subject in the last w frames of video. For each S_i, a corresponding pose template match P_i is found in the silhouette pose template database by using the same distance metric used in training. Let $L_T = \{P_1, P_2, \ldots, P_N\}$ represent the list of matched pose IDs, where $P_i \in [1, ID_{MAX}]$. After this step, like in the training phase, a normalized histogram H_T of IDs is created by using the IDs in L_T.

In next step, the distance between H_T and each action template histogram H_j in the action database is calculated. The distance metric in this calculation is Euclidian distance and defined similar to the $Dist_{AB}$ as explained in Section 4.1. The action type label of the action histogram Hj, which has the minimum distance with H_T is assigned as the label of the current test action A_T. Figure 8 shows a sample history of poses for a window size of $w = 4$, in the actual implementation we use $w = 25$.

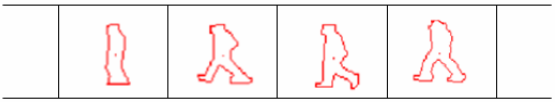

Fig. 8. Sample history window for a test sequence

6 Experimental Results

All of the tests are performed by using a video player and analyzer application that we implemented for developing our computer vision algorithms, on Microsoft Windows XP Professional operating system on a computer with an Intel PIV-2600 MHz CPU and 512 MB of RAM.

In order to test the object classification algorithm we first created a sample object template database by using an application to extract and label object silhouettes. We used four sample video clips that contain human, human group and vehicle samples. We used the template object database to classify objects in several movie clips containing human, human group and vehicle. We prepared a confusion matrix to measure the performance of our object classification algorithm. The confusion matrix is shown in Table 1. The confusion matrix is for the following object types: Human, Human Group and Vehicle.

We performed our action recognition experiments with three human subjects. One subject is used to create the template pose and template action databases and the other subjects are used in recognition tests.

Table 1. Confusion matrix for object classification

	Human	Human Group	Vehicle	Success
Human	175	13	20	84.13%
Human Group	12	52	14	66.67%
Vehicle	38	22	238	79.86%
Average Success Rate				76.88%

We created action templates for the following actions: Walking, Boxing and Kicking. Below, the confusion matrix for the cumulative action recognition results is shown:

Table 2. Confusion matrix for action recognition

	Walking	Boxing	Kicking	Success
Walking	12	1	1	85.71%
Boxing	0	4	0	100.00%
Kicking	0	1	3	75.00%
Average Success Rate				86.94%

7 Discussion

In this paper, we proposed a novel system for real-time object classification and human action recognition using object silhouettes. The test results show that the presented method is promising and can be improved with some further work to reduce false alarms. The proposed methods can also be utilized as part of a multimedia database to extract useful facts from video clips [19].

A weakness of the proposed methods is that they are view dependent. If the camera setup is different in training and testing, the success rate will be too low. Automating the template database creation steps will help to obtain a self calibrating object classification and human action recognition system.

Acknowledgement

This work is supported in part by European Commission Sixth Framework Program with Grant No: 507752 (MUSCLE Network of Excellence Project).

References

[1] J.K. Aggarwal and Q. Cai. Human motion analysis: a review. Computer Vision and Image Understanding, 73(3):428–440, March 1999.

[2] E. M. Arkin, L.P. Chew, D.P. Huttenlocher, K. Kedem, and J.S.B. Mitchell. An efficiently computable metric for comparing polygonal shapes. IEEE Transactions on Pattern Recognition and Machine Intelligence, 13:209–216, 1991.

[3] A. Enis Cetin, Report on progress with respect to partial solutions on human detection algorithms, human activity analysis methods, and multimedia databases. WP-11 Report, EU FP6-NoE: MUSCLE (Multimedia Understanding Through Computation and Semantics), www.muscle-noe.org, May 2005.

[4] O. Chomat, J.L. Crowley, Recognizing motion using local appearance, International Symposium on Intelligent Robotic Systems, University of Edinburgh, pages 271-279, 1998.

[5] R. T. Collins, R. Gross, and J. Shi. Silhouette-based human identification from body shape and gait. In Proc. of Fifth IEEE Conf. on Automatic Face and Gesture Recognition, pages 366–371, 2002.
[6] R. Cutler and L.S. Davis. Robust real-time periodic motion detection, analysis and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence, 8:781–796, 2000.
[7] R. T. Collins et al. A system for video surveillance and monitoring: VSAM final report. Technical report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, May 2000.
[8] Y. Dedeoglu, Moving object detection, tracking and classification for smart video surveillance, Master's Thesis, Dept. of Computer Eng. Bilkent University, Ankara, 2004.
[9] D. M. Gavrila. The analysis of human motion and its application for visual surveillance. In Proc. of the 2nd IEEE International Workshop on Visual Surveillance, pages 3–5, Fort Collins, U.S.A., 1999.
[10] I. Haritaoglu, D. Harwood, and L.S. Davis. W4: A real time system for detecting and tracking people. In Computer Vision and Pattern Recognition, pages 962–967, 1998.
[11] F. Heijden. Image based measurement systems: object recognition and parameter estimation. Wiley, January 1996.
[12] J. Heikkila and O. Silven. A real-time system for monitoring of cyclists and pedestrians. In Proc. of Second IEEE Workshop on Visual Surveillance, pages 74–81, Fort Collins, Colorado, June 1999.
[13] H.Ramoser, T.Schlgl, M.Winter, and H.Bischof. Shape-based detection of humans for video surveillance. In Proc. of IEEE Int. Conf. on Image Processing, pages 1013-1016, Barcelona, Spain, 2003.
[14] A. J. Lipton. Local application of optic flow to analyse rigid versus non-rigid motion. Technical Report CMU-RI-TR-99-13, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, December 1999.
[15] A. J. Lipton, H. Fujiyoshi, and R.S. Patil. Moving target classification and tracking from real-time video. In Proc. of Workshop Applications of Computer Vision, pages 129–136, 1998.
[16] S. Loncaric. A survey of shape analysis techniques. Pattern Recognition, 31(8):983–1001, August 1998.
[17] A. M. McIvor. Background subtraction techniques. In Proc. of Image and Vision Computing, Auckland, New Zealand, 2000.
[18] E. Saykol, U. Gudukbay, and O. Ulusoy. A histogram-based approach for object-based query-by-shape-and-color in multimedia databases, Image and Vision Computing, vol. 23, No. 13, pages 1170-1180, November 2005.
[19] E. Saykol, U. Gudukbay, O. Ulusoy. A Database Model for Querying Visual Surveillance by Integrating Semantic and Low-Level Features. In Lecture Notes in Computer Science (LNCS), (Proc. of 11th International Workshop on Multimedia Information Systems-MIS'05), Vol. 3665, pages 163-176, Edited by K. S. Candan and A. Celentano, Sorrento, Italy, September 2005.
[20] E. Saykol, G. Gulesir, U. Gudukbay, and O. Ulusoy. KiMPA: A kinematics-based method for polygon approximation. In Lecture Notes in Computer Science (LNCS), Vol. 2457, pages 186-194, Advances in Information Sciences (ADVIS'2002) Edited by Tatyana Yakhno, Springer-Verlag, 2002.
[21] C. Schuldt, I. Laptev and B. Caputo, Recognizing human actions: a local SVM approach, In Proc. of ICPR'04, Cambridge, UK.

[22] C. Stauffer and W. Grimson. Adaptive background mixture models for realtime tracking. In Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 246-252, 1999.

[23] B. U. Toreyin, A. E. Cetin, A. Aksay, M. B. Akhan, Moving object detection in wavelet compressed video. Signal Processing: Image Communication, EURASIP, Elsevier, vol. 20, pages 255-265, 2005.

[24] B. U. Toreyin, Y. Dedeoglu, A. E. Cetin, HMM based falling person detection using both audio and video. IEEE International Workshop on Human-Computer Interaction, Beijing, China, Oct. 21, 2005 (in conjunction with ICCV 2005), Lecture Notes in Computer Science, vol. 3766, pages 211-220, Springer-Verlag GmbH, 2005.

[25] R.C. Veltkamp and M. Hagedoorn. State-of-the-art in shape matching, In Principles of Visual Information Retrieval, pages 87–119, Springer, 2001.

[26] L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. Pattern Recognition, 36(3):585–601, March 2003.

[27] L. Wixson and A. Selinger. Classifying moving objects as rigid or non-rigid. In Proc. of DARPA Image Understanding Workshop, pages 341–358, 1998.