A View-Based Real-Time Human Action Recognition System as an Interface for Human Computer Interaction

Jin Choi[1], Yong-il Cho[1], Taewoo Han[2], and Hyun S. Yang[1]

[1] AIM Lab., Computer Science Dept., KAIST, Daejeon, South Korea
[2] Dept. of Game & Multimedia, Woo-song University, Daejeon, South Korea
jin_choi, caelus, hsyang@paradise.kaist.ac.kr,
bluebird@paradise.kaist.ac.kr

Abstract. This paper describes a real-time human action recognition system that can track multiple persons and recognize distinct human actions through image sequences acquired from a single fixed camera. In particular, when given an image, the system segments blobs by using the Mixture of Gaussians algorithm with a hierarchical data structure. In addition, the system tracks people by estimating the state to which each blob belongs and assigning people according to its state. We then make motion history images for tracked people and recognize actions by using a multi-layer perceptron. The results confirm that we achieved a high recognition rate for the five actions of walking, running, sitting, standing, and falling though each subject performed each action in a slightly different manner. The results also confirm that the proposed system can cope in real time with multiple persons.

Keywords: view-based action recognition, adaptive background subtraction, motion history image, HCI.

1 Introduction

With the rapid progress of information technology, many researchers are struggling to build smart space where embedded computer systems can perceive the context and provide proper services at the right moment. In this environment, the traditional interfaces of desktop computing such as a keyboard or mouse are inadequate. We need a new type of interface. The visual interface, for instance, has recently gained attention because it is natural and easy to use. However, to use the visual interface, we need a way of effectively recognizing human action in real time.

View-based recognition of human action is comprised of motion analysis involving human body parts, tracking of human motion, and the recognition of human action from image sequences [1]. And this type of research is useful for various applications and especially as an interface for Human Computer Interaction. For example, the function of recognizing human action can be used for an input method for immersive games and visual surveillance systems.

We now present a real-time human action recognition system that can track multiple persons and recognize distinct human actions through image sequences acquired from a single fixed camera.

T.G. Wyeld, S. Kenderdine, and M. Docherty (Eds.): VSMM 2007, LNCS 4820, pp. 112–120, 2008.

In the next section we propose a human action recognition system for multiple persons. We then discuss the experimental results in section3 and summarize our conclusions in the final section.

2 The Proposed Human Action Recognition System

Our proposed real-time system can track people and recognize simple and short human actions such as walking, running, sitting, standing, and falling through image sequences obtained from a single fixed camera. Once we determine how a simple action can be perceived, we can apply this knowledge to the recognition of complex actions such as exercising, fighting, and lurking. For the proposed system to be useful, we incorporated design features that made the system fast and efficient. Figure 1 shows a schema of the proposed system. The proposed system consists of four parts: segmenting foreground, tracking people, modeling action, and classifying action. The details of each part are explained below.

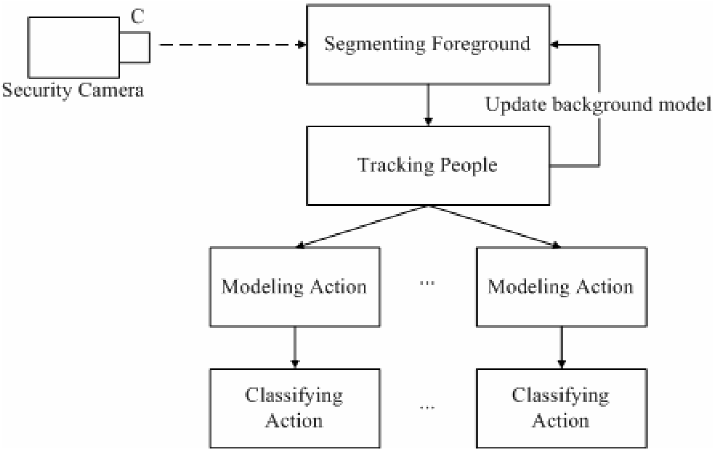


Fig. 1. Schema of the proposed system, which can track multiple targets and also recognize the distinct actions of multiple targets

2.1 Segmentation

The ability to rapidly extract correct silhouettes from an image is essential for our system because the modeling action is based on a silhouette. Although there are several popular background subtraction algorithms, such as the Running Average algorithm, and Mixture of Gaussians (MOG) algorithm, we used a modified the MOG algorithm with a hierarchical data structure because the Running Average algorithm can’t cope with multimodal background distribution and the standard MOG takes too long for real-time processing. A MOG algorithm with a hierarchical data structure reportedly enhances the processing speed significantly and yields results that are very similar to the results of the standard MOG algorithm [2]. In Park’s method [2], the hierarchical data structure is constructed in a kind of top-down approach. In this

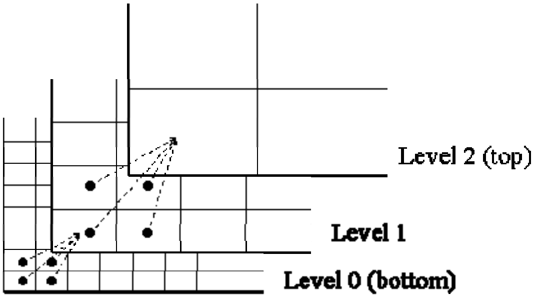


Fig. 2. The three-layer hierarchical data structure formed with a bottom-up approach. The structure is generated from Level 0 to Level 2 in order.

method, however, the user is unable to control the form of the leaf nodes because the image is recursively decomposed into four equal regions. To adjust the form to our needs, we constructed the hierarchical data structure in a bottom-up way (Fig. 2).

Firstly, we group a set of pixels into the form that we wish to detect. We then stack layers that consist of the parent nodes of four children (North West, North East, South West, and South East). The method of searching is the same with the Park's [2]. After stacking the layers to the designed layer, we apply a quadtree-based decomposition to an input image. The searching starts at the top layer. For each node of the top layer, a random pixel is sampled. The pixel is classified as either foreground or background. If the pixel is classified as background, the next node is processed. If not, the node is subdivided into the lower level and this subdivision is repeated until the bottom layer.

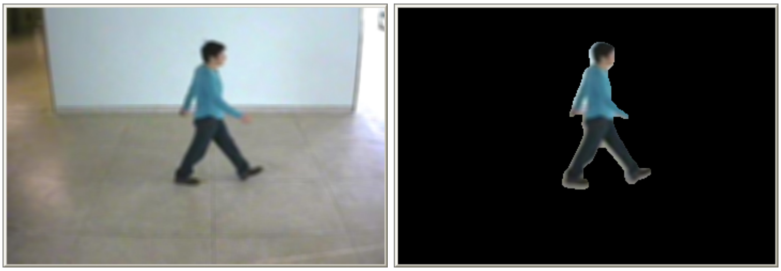


Fig. 3. The results of background subtraction (the left image is an original image and the right image show the foreground image)

The system can rapidly cope with a multimodal background distribution by using the modified MOG algorithm (Fig. 2). When obtained foreground pixels at time t, through connected component analysis, they are grouped into a set of blobs $B_t = \{b_t^i \mid i \text{ is an integer and } 0 \le i\}$, where a blob b_t^i is a set of connected foreground pixels and contains a set of persons as Fig. 4 shows. Let $O_{b_t^i}$ be the set of persons in b_t^i and o^i be a person that is represented as an appearance model.

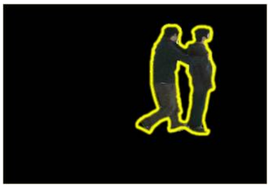
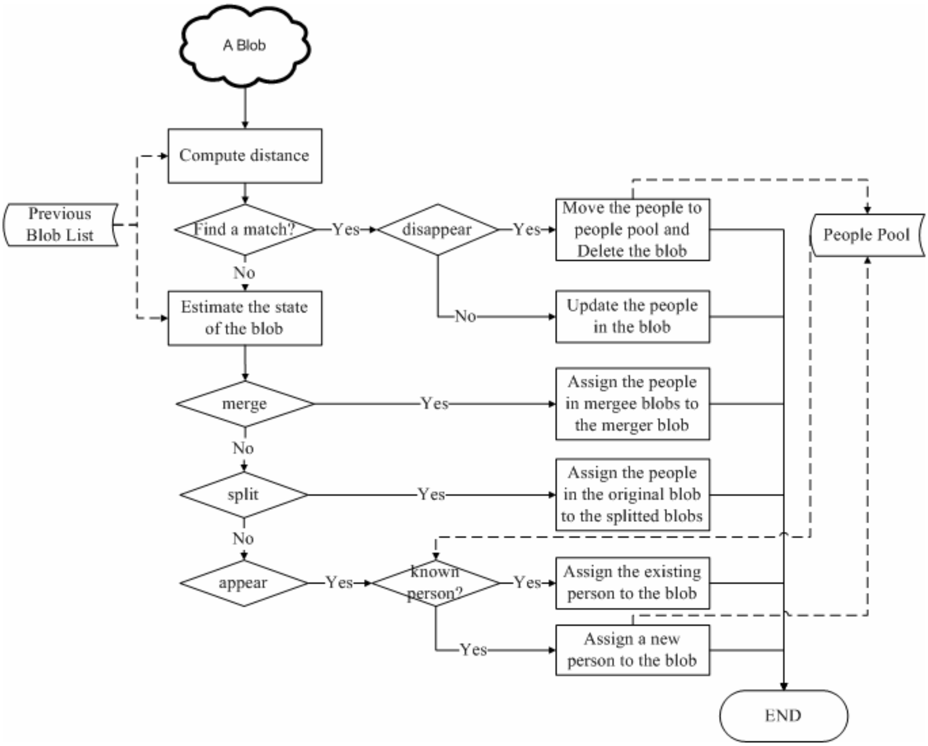


Fig. 4. A blob is containing two persons

2.2 People Tracking

The ability to track people is necessary for recognizing the distinct actions of multiple persons because an action of a person is modeled with a sequence of person's silhouettes [4][5]. To simplify the problem, we assume that a person appears near the entrance and disappears at the exit. No consideration is given to the possibility that people appear or disappear in a group. To identify a person without additional information such as tag, we use an appearance model.

Given B_t through segmentation, we estimate the state to which each blob belongs and localize people according to its state. And Fig. 5 shows a flow chart for the tracking people. We assume that every blob has one of the following states.



A Blob
Compute distance
Previous Blob List
Find a match?
Yes
No
disappear
Yes
No
Move the people to people pool and Delete the blob
Update the people in the blob
People Pool
Estimate the state of the blob
merge
Yes
Assign the people in mergee blobs to the merger blob
No
split
Yes
Assign the people in the original blob to the splitted blobs
No
appear
Yes
known person?
Yes
Assign the existing person to the blob
Yes
Assign a new person to the blob
END

Fig. 5. Flow chart for the tracking people

Appearing. A new blob appears in the frame at time t
Disappearing. An existing blob in the frame at time t is disappearing.
Continuation. A blob continues from the frame at time t-1 to time t.
Merging. Two blobs in the frame at time t-1 merge into one blob in the frame at time t.
Splitting. One blob in the frame at time t-1 separates into two blobs in the frame at time t.

In particular, we compute the distance between b_t^i and every blob of B_t in order to establish correspondence. Let d_{hue} be the normalized Bhattacharyya distance between hue histograms and d_{size} be the normalized size difference and d_{dis} be the normalized Euclidean distance between centroids. We can then define the distance $d(b_t^i, b_{t-1}^j)$ between b_t^i and b_{t-1}^j as follows:

$$d(b_t^i, b_{t-1}^i) = \frac{1}{3}(d_{hue}(b_t^i, b_{t-1}^i) + d_{size}(b_t^i, b_{t-1}^i) + d_{dis}(b_t^i, b_{t-1}^i)) \tag{1}$$

If b_{t-1}^j with the smallest distance is less than a threshold value, we regard b_t^i as continuation of b_{t-1}^j , and we obtain $O_{b_t^i}$ by updating $O_{b_{t-1}^j}$ with current tracks. Otherwise we can infer that b_t^i have one of three states such as merging, splitting, and appearing. The stat of b_t^i can be estimated based on the relation between B_{t-1} and b_t^i (Fig. 6). Let b^x be an element of $B_{t-1} \cap b_t^i$. If $b^x \approx b_{t-1}^i$, merging might occur. Thus, we simply assign the people in two mergee blobs to the merger blob. And if $b^x \approx b_t^i$, splitting might occur. In this case, to assign the people in the original blob to the splitted blobs, we make all hypotheses and evaluate each hypothesis by using appearance model of people. We then assign the people according to the hypothesis with highest probability. Finally if there is no relation between B_{t-1} and b_t^i, we can think that b_t^i is in the appearing state. It means that the known person reappears or a stranger appears. If the person is not known, we just create a new person instance o_i and assign it to b_t^i.

2.3 Action Modeling

In contrast to the process of posture recognition, which involves a specific image, action recognition involves consideration of a sequence of images. Given a sequence of images, we adapt a representation of motion history image (MHI) for the purpose of modeling an action. The MHI collapses an image sequence into a 2-D image that captures spatial and temporal information about motion [3]. The MHI is known for its fast processing speed and ability to represent short-duration movement.

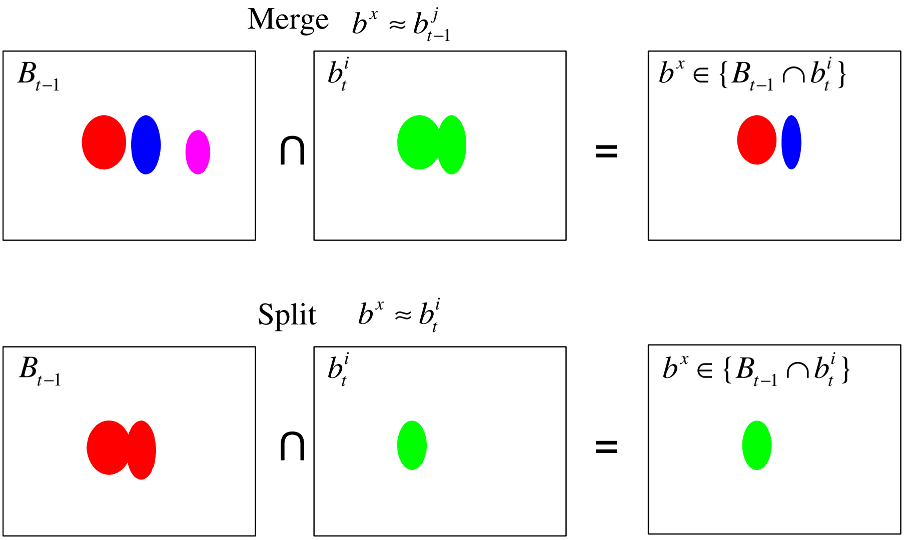

Merge $b^x \approx b^j_{t-1}$
B_{t-1}
b^i_t
$b^x \in \{B_{t-1} \cap b^i_t\}$
∩
=
Split $b^x \approx b^i_t$
B_{t-1}
b^i_t
$b^x \in \{B_{t-1} \cap b^i_t\}$
∩
=

Fig. 6. The relation between B_{t-1} and b^i_t when merging or splitting occurs

An MHI at time t is updated as

$$\mathrm{MHI}^t_\delta(x,y) = \begin{Bmatrix} t/\delta & \text{if } \Psi(I^t(x,y)) \neq 0 \\ \mathrm{MHI}^{t-1}_\delta(x,y) & \text{otherwise} \end{Bmatrix} \tag{2}$$

where δ is the number of images used for the collapse, $I^t(x,y)$ is the current image and Ψ signals the presence of a blob at pixel (x,y). Fig. 7 shows an example of an MHI of a person falling. The first four images from the left of Fig. 7 show extracted silhouettes, and the image on the right-hand side is the corresponding MHI. For multiple people, we maintain an MHI for each person by using previous tracking outputs.

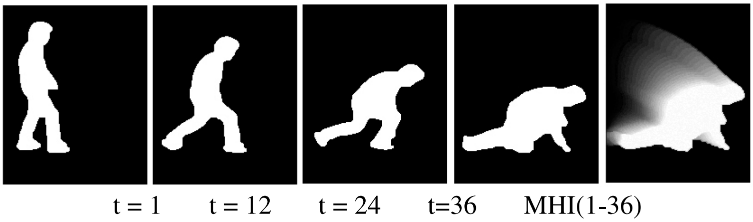


Fig. 7. Selected frames of a person falling and a corresponding MHI

2.4 Classification

When given an MHI, we search it for a bounding box to extract the various features. The bounding box is then normalized at 256 features. Additionally, we add the width and height of the bounding box to a feature vector in order to classify actions such as walking and running which are similar to the normalized MHI.

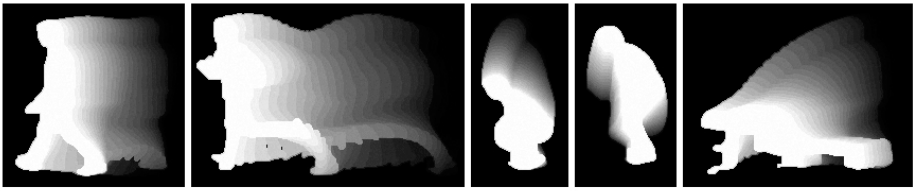


Fig. 8. Sample MHIs (from the left: walking, running, sitting, standing, and falling) for the training of a multi-layer perceptron

The task of formalizing actions is difficult because people rarely act in the same way. Hence, to classify actions, we use a multi-layer perceptron (MLP), which is a sort of robust neural network. We define five classes: walking, running, sitting, standing, and falling. Fig. 8 shows examples of MHIs that are used to train an MLP.

3 Experimental Results

In this experiment, we asked seven subjects to perform the five distinct actions of walking, running, sitting, standing, and falling, and to perform each action 10 times

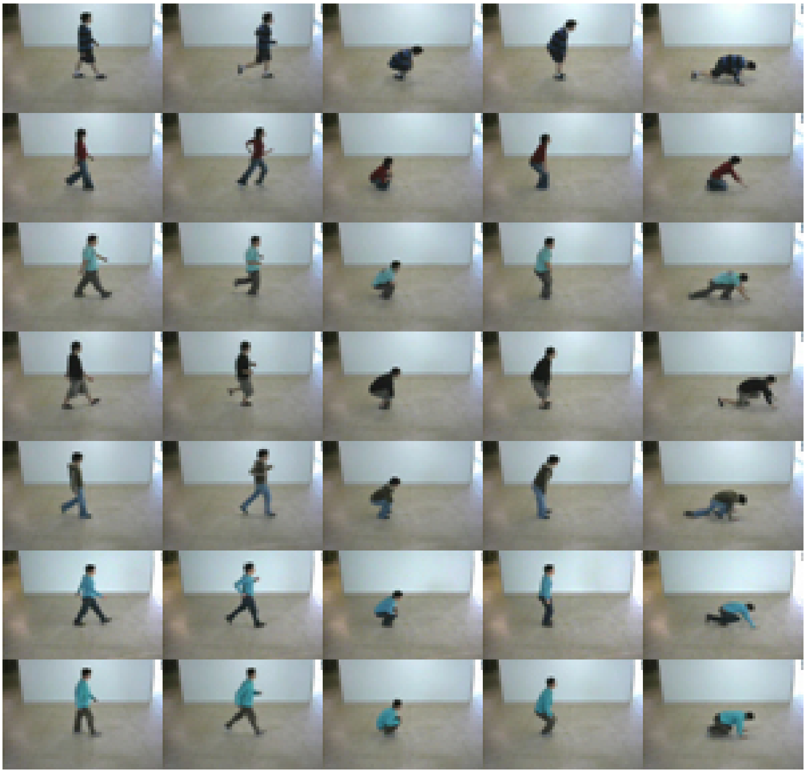


Fig. 9. The five distinct actions performed by the seven subjects

(Fig. 9). Three subjects were included in the training and the other four subjects were not. Table 1 shows the number of correct results.

In Table 1, T1, T2, and T3 refer to the subjects who were included in the training and P1, P2, P3, and P4 refer to the subjects who were not included in the training. A recognition rate of 92% was attained by T1, T2, and T3, and a recognition rate of 90% was attained by P1, P2, P3, and P4. The actions that were recognized most easily were the common actions of falling, running, and sitting. However, the action of standing was not easily recognized due to incorrect segmentation. In summary, although each subject performed the same action in a slightly different manner, the seven subjects attained a high recognition rate of 90.9%. We expect the recognition rate to be raised with more accurate segmentation.

Table 1. Experimental results of the proposed human action recognition system

	walking	running	Sitting	standing	falling	Recognition rate (%)
T1	10	10	10	10	10	100.0
T2	10	10	10	7	10	94.0
T3	8	10	8	5	10	82.0
P1	10	9	10	10	10	98.0
P2	7	10	10	8	10	90.0
P3	4	10	10	10	10	88.0
P4	9	10	10	3	10	84.0
Recognition rate (%)	82.9	98.6	97.1	75.7	100.0	90.9

We performed an additional experiment to evaluate the processing speed of the proposed system. While three persons were performing actions, we recorded video footage at 30 fps in a format of 352 pixels by 240 pixels. When we inputted the recorded video into the proposed system, which was running on a 3.0 GHz computer, we were able to achieve recognition results in real time. Fig. 10 shows the output of the proposed system in that situation.

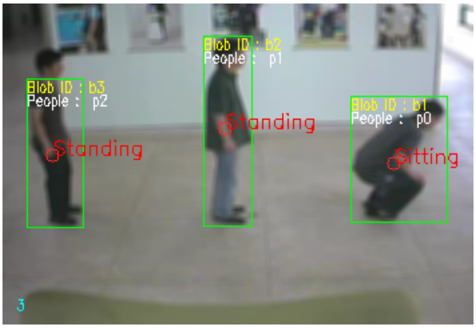


Fig. 10. The results of the proposed system when the persons on the left and in the middle are standing and the person on the right is sitting

4 Conclusion

We present a real-time human action recognition system that can track people and recognize distinct human actions through image sequences acquired from a single fixed camera. In particular, when given an image, the system can segment silhouettes by means of a MOG algorithm with a hierarchical data structure. The system can also track people by estimating the state to which each blob belongs and assigning people according to its state. Next, the system makes MHIs of tracked people and can recognize distinct actions by using an MLP. The results of an experiment on the five actions of walking, running, sitting, standing, and falling showed a high recognition rate, even though each subject performed the same action in a slightly different manner. The results also confirm that the proposed system copes in real time with multiple persons.

To enhance the proposed system, we plan to study how to extract accurate silhouettes and to make view-independent motion representations with the aid of multiple cameras.

Acknowledgments

This research is supported by the Ubiquitous Computing and Network Project of the Ministry of Information and Communication 21st Century Frontier R&D Program in Korea. It is also supported by the Korea Science and Engineering Foundation through the Advanced Information Technology Research Center.

References

1. Aggarwal, J.K., Cai, Q.: Human Motion Analysis: A Review. Computer Vision and Image Understanding: CVIU 73(3), 428–440 (1999)
2. Park, J., Tabb, A., Kak, A.C.: Hierarchical Data Structure for Real-Time Background Subtraction. In: IEEE International Conference on Image Processing (2006)
3. Bobick, A., Davis, J.: The recognition of human movement using temporal templates. IEEE Trans. Patt. Analy. And Mach. Intell. 23(3), 257–267 (2001)
4. Yilmaz, A., Javed, O., Shah, M.: Object Tracking: A Survey. In: ACM Computing Surveys, December 2006, vol. 38(4) (2006)
5. Huang, Y., Essa, I.: Tracking Multiple Objects Through Occlusions. In: CVPR 2005, San Diego, CA (June 2005)