A Framework for Suspicious Action Detection with Mixture Distributions of Action Primitives

Yoshio Iwai

Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan
iwai@sys.es.osaka-u.ac.jp

Abstract. In this paper, we propose a generic framework for detecting suspicious actions with mixture distributions of action primitives, of which collection represents human actions. The framework is based on Bayesian approach and the calculation is performed by Sequential Monte Carlo method, also known as Particle filter. Sequential Monte Carlo is used to approximate the distributions for fast calculation, but it tends to converge one local minimum. We solve that problem by using mixture distributions of action primitives. By this approach, the system can recognize people's actions as whether suspicious actions or not.

1 Introduction

In recent years, the social environment has become more complex and people's personal lives have become more varied, so the development of security systems that detect hazards and allow us to avoid these has become necessary for us to be safe and secure. However, it is too heavy work for administrators to monitor the environment in 24 hours because they are tied on monitor TVs, and such heavy work causes a mistake. Therefore, a system would need facilities to detect unusual situations automatically and inform system administrators of unusual situations by sensing and recognizing our environment. Such a system would reduce surveillance load, because administrators only pay attention when the system warns.

To detect suspicious actions, the system needs to detect and track people surreptitiously. Cameras have usually been utilized as environmental sensors because they do not make us feel uncomfortable. People are detected and tracked through input images by image processing, and our purpose is to do this and to recognize their actions by using trajectories of movement.

Numerous methods using various features for specific purposes have been proposed to recognize human actions. In general, a system to recognize human actions is consists of two parts: tracking module and recognition module. In tracking people, a human detector is made by Boosting method from the action database[1] or Bayesian approach[2]. Many methods for tracking people have been proposed, but any method is applicable because tracking module and recognition module can be designed separately, so we focused on recognition of human actions and detection of suspicious actions in this paper.

T. Wada, F. Huang, and S. Lin (Eds.): PSIVT 2009, LNCS 5414, pp. 519–530, 2009.

To recognize human actions, many probabilistic approaches have been proposed. Models of action are mainly classified into two models: the first is used to construct continuous human actions and the other is used to define human actions as discrete state transition. Hidden Markov Model (HMM) have frequently been used to model human actions as discrete state transition[3]. To recognize continuous actions, stochastic methods have frequently been used recently, such as the CONDENSATION algorithm[4] and particle filters[5]. These methods require non-linear and non-Gaussian models to distinguish them from methods using a linear model like the Kalman filter. We call these methods Monte Calro method because these methods use Monte Carlo approximation to calculate marginal probability used for Bayesian estimation. Monte Carlo method can recursively calculate the marginal probability at each time step in real time and can calculate the expectation value of the probability distribution by using sample points.

One approach using Monte Carlo method has been proposed[6]. However, there is a problem that they assume that the distribution of human action state would be uniform. In this paper, we reformulate human actions by the posterior probability different from the method proposed by [6], and approximate it by the Monte Calro method. By this reformulation, we can treat human actions within the Monte Calro approximation theory, and also solve the local minimum problem of sequential Monte Calro method described in 3.2.

In this paper, we describe an overview of the proposed system for detecting suspicious actions in the next section, and then we explain the action model and detection method in Sec. 3. In Sec. 4, we describe the implementation of the system, the probability and the likelihood, and we show experimental results in Sec. 5, and summaries in Sec. 6.

2 Action Model

2.1 Representation of Action Models

Human action can be considered as the trajectories of movement in the observation space as shown in Fig. 1. In this work, human actions can be classified into sub trajectories, called action primitive, in the feature space extracted from the observation sensor. Human actions can be modeled by discrete states considered as action primitives m and its transition as shown in Fig. 2. Action primitive m has a trajectory $z = m(f)$ in the feature space, where z is an observation feature and f is a "frame" parameter of trajectory.

A state of action at time t is denoted by $\boldsymbol{x}_t$. We assume that the transition of action state $\boldsymbol{x}$ is a Markov chain that the transition only depends on the previous state. In short, the following equation is satisfied:

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-k:t-1}) = p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \tag{1}$$

where $\boldsymbol{x}_{t-k:t}$ is a state sequence from time 0 to time t. In the implementation, $\boldsymbol{x}_t$ consists of

1. action primitive at time t: m_t,
2. frame position of action primitive: f_t,
3. velocity of action: v_t.

We denote that $\boldsymbol{x}_t = (m_t, f_t, v_t)$. At this time, we can rewrite:

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = p(m_t, f_t, v_t|\boldsymbol{x}_{t-1}) = p(v_t, f_t|m_t, \boldsymbol{x}_{t-1})p(m_t|\boldsymbol{x}_{t-1}). \quad (2)$$

Especially, we assume that f_t and v_t is independent, and v_t is only dependent on v_{t-1}, we get

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = p(v_t|v_{t-1})p(f_t|m_t, \boldsymbol{x}_{t-1})p(m_t|\boldsymbol{x}_{t-1}). \quad (3)$$

From the above equation, we can obtain probability distributions of each parameters. This equation is used for Monte Carlo approximation to get sample points. The detail method is described in Section 4.3.

2.2 Detecting Action Primitives and Learning Action Models

Before performing the recognition process, action models must be learned from the action database. First, input trajectories of human actions in the feature space are automatically divided into segments separated at stationary points, and the separated segments are candidates as action primitives, but short segments are removed from the candidate set. Next, if a segment in the candidate set have a similar part of a segment in the action database, a similar part is automatically removed from the candidate segment, and the rest of the segment is automatically added to the candidate set. The above process is done until no similar part exists in the action database. After the process ends, if the candidate set is not empty, all segments in the candidate set is added to the action database. At this time, the transition probability of action primitives $p(m_i|m_j)$ is also updated by counting the number of transition in the action database.

Comparison a segment in the candidate set with an action primitive m_i in the action database is automatically performed as follows:

1. Find the nearest points p_s, p_e in a segment to the first and the last frame points in an action primitives m_i.
2. Accumulate the distances between the points on a part of the segment $[p_s, p_e]$ and the corresponding frame points in an action primitive m_i.
3. If the accumulated distance is less than a certain threshold value, we determine that a part of the segment $[p_s, p_e]$ is similar to an action primitive m_i.

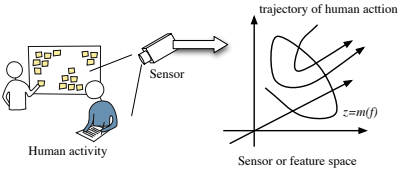


Fig. 1. Trajectories represent human activities

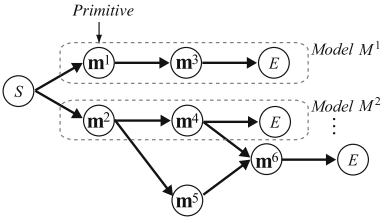


Fig. 2. Action model and action primitive

3 Framework for Action Recognition and Suspicious Action Detection

3.1 Action Recognition

The action recognition is performed by MAP estimation using the posterior probability, $p(M_t|\boldsymbol{z}_{t-k:t})$, of action model M_t at time t given the observation sequence $\boldsymbol{z}_{t-k:t}$, where k is a number of observation frames. The action recognition is formulated by the following equation:

$$M_s = \arg\max_{M_t} p(M_t|\boldsymbol{z}_{t-k:t}). \tag{4}$$

To calculate the posterior probability $p(M_t|\boldsymbol{z}_{t-k:t})$, we use a state of action $\boldsymbol{x}$ as follows:

$$p(M_t|\boldsymbol{z}_{t-k:t}) = \int p(M_t|\boldsymbol{x}_{t-k:t})p(\boldsymbol{x}_{t-k:t}|\boldsymbol{z}_{t-k:t})d\boldsymbol{x}_{t-k:t}, \tag{5}$$

where $\boldsymbol{x}_{t-k:t}$ is a state sequence from time 0 to time t. Assuming that state $\boldsymbol{x}_t$ is conditionally independent from the observation sequence and is not affected by future observation, we get

$$p(M_t|\boldsymbol{z}_{t-k:t}) = \int p(M_t|\boldsymbol{x}_{t-k:t}) \prod_{s=t-k}^{t} p(\boldsymbol{x}_s|\boldsymbol{z}_{t-k:s})d\boldsymbol{x}_{t-k:t}. \tag{6}$$

Because of the assumption of Markovian process of state transition, the probability of states, $p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t})$, is calculated recursively by the following equations:

$$p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t-1}) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})p(\boldsymbol{x}_{t-1}|\boldsymbol{z}_{t-k:t-1})\, d\boldsymbol{x}_{t-1}, \tag{7}$$

$$p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}) = \frac{1}{p(\boldsymbol{z}_t|\boldsymbol{z}_{t-k:t-1})} p(\boldsymbol{z}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t-1}). \tag{8}$$

Action recognition is performed by iteratively calculating the posterior probability by using Eqs. 7, 8, and 6 as $p(\boldsymbol{x}_{t-1}|\boldsymbol{z}_{t-k:t-1})$ is known. This calculation can be performed recursively because $p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t})$ is obtained from Eq. 8. The integrals in the above equations are approximated by Monte Carlo method by sampling of a state $\boldsymbol{x}^{(i)}$ from the action state space $\boldsymbol{x}$. The likelihood $p(\boldsymbol{z}_t|\boldsymbol{x}_t)$ is

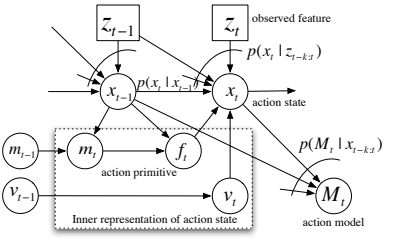


Fig. 3. Probability structure of the proposed framework

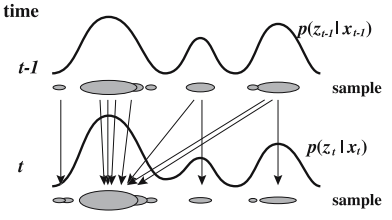


Fig. 4. Convergence to the local minimum

designed to be suit for a given problem. We note that the probability $p(\boldsymbol{z}_t|\boldsymbol{z}_{t-k:t-1})$ in Eq. 8 can be treated as a normalize constant when the probability is calculated by Monte Carlo method, therefore, we do not need the concrete shape of the distribution. Figure 3 shows the probability structure of the proposed framework.

3.2 Resolving Multi-peak Problem

When the above equations are approximated by Monte Carlo method, The samples of states are updated by the probability, $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$, in Eq. 7. In short, the states are updated by $\boldsymbol{x}_{t-1}$ without observations $\boldsymbol{z_t}$, and during sampling, samples are biased by $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$, and then the accuracy of approximation becomes worse. Therefore, new samples of states are made from the old sample sets by weighted resampling to be ubiquity. The method for determining the weight for resampling has been proposed that uses the likelihood[9]. In that case, sampling points have a tendency to converge one local minimum as shown in Fig. 4.

To avoid a local minimum, one solution is that we increase the number of sampling points to approximate the probability distribution accurately, but the computation time would increase. Dispersing Deterministic Crowdings (DDC) method[10] has been proposed to avoid a local minimum in Genetic Algorithm research field, and a method using DDC is applied for Monte Carlo approximation[6]. However, the method does not suit for Monte Carlo approximation because it changes the distribution of weights during resampling. On the other hand, the method using a mixture distribution for modeling the distribution of states[11] has been proposed as follows:

$$p\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}\right) \stackrel{\text{def}}{=} \sum_{m=1}^{N_m} \pi_{m,t} p_m\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}\right), \quad \sum_{m=1}^{N_m} \pi_{m,t} = 1, \tag{9}$$

where N_m is the number of local distribution, $\pi_{m,t}$ is the weight for the local distribution p_m at time t.

In this paper, by substituting the above equations for Eq. 7 and 8, we obtain the update equation of $\pi_{m,t}$ and $p_m\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}\right)$ as follows:

$$p_m\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t-1}\right) = \int p\left(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}\right) p_m\left(\boldsymbol{x}_{t-1}|\boldsymbol{z}_{t-k:t-1}\right) d\boldsymbol{x}_{t-1}, \tag{10}$$

$$p_m\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}\right) = \frac{p\left(\boldsymbol{z}_t|\boldsymbol{x}_t\right) p_m\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t-1}\right)}{p_m\left(\boldsymbol{z}_t|\boldsymbol{z}_{t-k:t-1}\right)}, \tag{11}$$

$$\pi_{m,t} = \frac{\pi_{m,t-1} p_m\left(\boldsymbol{z}_t|\boldsymbol{z}_{t-k:t-1}\right)}{\sum_{m=1}^{N_m} \pi_{m,t-1} p_m\left(\boldsymbol{z}_t|\boldsymbol{z}_{t-k:t-1}\right)}. \tag{12}$$

By using the above equation, we can use the weights of local distributions as the weights of samples in the framework of Monte Carlo approximation, and we can also approximate the distribution having local minima by the Monte Carlo method. In this paper, the local distribution p_m is the probability distribution of action primitives. This approach naturally introduces the method[11] in the framework of Monte Carlo approximation.

3.3 Approximation by Sequential Monte Carlo Method

In this section, we describe the resampling method for approximating Eq. 6. An i-th sample of state at time t is denoted by $\boldsymbol{x}_t^{(i)} = (m_t^{(i)}, f_t^{(i)}, v_t^{(i)})$. $\boldsymbol{x}_t^{(i)}$ is generated from probability distribution $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. $w_t^{(i)}$ is the weight of a sample and $\pi_{m,t}$ is the weight of a local distribution p_m.

First, we obtain the sample set $\mathcal{X}_{t-1} = \{\boldsymbol{x}_{t-1}^{(i)} | i = 1...N\}$ and the distribution of weights $W_{t-1} = \{w_{t-1}^{(i)} | i = 1...N\}$ at time $t-1$ in advance. The next sample is generated by the following equation:

$$\boldsymbol{x}_t^{(i)} \sim p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(i)}), \tag{13}$$

however, to avoid a bias, we must resample them before it. As a resampling, we determine the sample points by the distribution of weights W_{t-1}:

$$\boldsymbol{x}_*^{(i)} \sim W_{t-1}, \tag{14}$$

and then, we update the state:

$$\boldsymbol{x}_t^{(i)} \sim p(\boldsymbol{x}_t|\boldsymbol{x}_*^{(i)}). \tag{15}$$

The details of update will be described in Section 4.3. We note that Vermaak's method[11] requires a clustering process to determine which local distribution

the state $\boldsymbol{x}_t^{(i)}$ belongs to, but the proposed method implicitly includes such a clustering process because the state has an information which cluster the state belongs to, i.e., action primitive m_t.

Next, we determine the weights of samples and local distributions by the following equations:

$$w_*^{(i)} = \frac{\tilde{w}_t^{(i)}}{\tilde{w}_{m_{t-1}^{(i)},*}}, \quad \pi_{m,*} = \frac{\tilde{w}_{m,*}}{\sum_{n=1}^{N_m} \tilde{w}_{n,*}}, \tag{16}$$

where

$$\tilde{w}_t^{(i)} = p(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)}), \quad \tilde{w}_{m,*} = \sum_{\forall j \text{ s.t. } m_{t-1}^{(j)}=m} \tilde{w}_t^{(j)}. \tag{17}$$

These equations approximates Eqs. 11 and 12, but resampling and reclustering are performed, so we calculate the weights as follows:

$$\pi_{m,t} = \sum_{\forall j \text{ s.t. } m_t^{(j)}=m} \pi_{m_{t-1}^{(j)},*} w_*^{(j)}, \quad w_t^{(i)} = \frac{\pi_{m_{t-1}^{(i)},*} w_*^{(i)}}{\pi_{m_t^{(i)},t}}. \tag{18}$$

Finally, Eq. 9 is approximated by the following equation:

$$p(\boldsymbol{x}_t|\boldsymbol{z}_{t-k:t}) = \sum_{j=1}^{N} \pi_{m_t^{(j)},t} w_t^{(j)} \delta(\boldsymbol{x}_t, \boldsymbol{x}_t^{(i)}), \tag{19}$$

where N is the total number of samples, and δ is the Kronecker delta function. After all, we get the following equation and we can perform action recognition by the following equation:

$$p(M_t|\boldsymbol{z}_{t-k:t}) = \sum_{i=1}^{N} p(M_t|\boldsymbol{x}_{t-k:t}^{(i)}) \prod_{s=t-k}^{t} p(x_s^{(i)}|z_{t-k:s}), \tag{20}$$

where $\boldsymbol{x}_{t-k:t}$ is a transition path of i-th particle from time $t-k$ to time t. Particle $x_t^{(i)}$ is resampled by the distribution $p(x_t^{(i)}|x_{t-1}^{(i)})$, and weight $w_t^{(i)}$ is calculated as follows:

$$\tilde{w}_t^{(i)} = w_{t-1}^{(i)} p(z_t|x_t^{(i)}), w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j \in \mathcal{I}_m} \tilde{w}_t^{(j)}}, \tag{21}$$

where $\mathcal{I}_m$ is the particle set whose member has the same action primitive as $m_t^{(i)}$ of particle $x_t^{(i)}$.

3.4 Initialization of Sample Set

The initial sample set $\mathcal{X}_{t=0}$ must be given in advance because sample $\boldsymbol{x}_t^{(i)}$ is generated from probability distributions $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(i)})$. Initial sample $\boldsymbol{x}_{t=0}^{(i)} =$

$(m_{t=0}^{(i)}, f_{t=0}^{(i)}, v_{t=0}^{(i)})$ is determined randomly like that; action primitive $m_{t=0}^{(i)}$ is randomly determined at first, and then frame position $f_{t=0}^{(i)}$ is randomly selected from the several first frames of learning data of $m_{t=0}^{(i)}$. Movement velocity $v_{t=0}^{(i)}$ is randomly chosen between the range $[v_{\min}, v_{\max}]$.

3.5 Suspicious Action Detection

We consider an action not in the action database as a suspicious action. However, all sample points are generated from the action database and no sample points are generated from a suspicious actions. Therefore, we need some criteria to detect suspicious actions. In this research we use the probability $p(\boldsymbol{z}_t|\boldsymbol{x}_t)$ that means the accuracy of the prediction of actions in the action database, and we determined an action as a suspicious action when all likelihood $p(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)})$ of all samples $\boldsymbol{x}_t^{(i)}$ becomes less than a certain threshold value ε_L as follows:

$$\forall i \;\; p(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)}) < \varepsilon_L, \tag{22}$$

which means all samples fail to predict the next position of the action.

In addition, when a suspicious action is detected, all samples are discarded and then all samples are initialized by the method describe in the above section for further detection process.

4 Design of Probability and Likelihood

4.1 State of Actions

State of action $\boldsymbol{x}_t$ expresses a point in the action space spanned by the state trajectories. We use center points $(X_{\boldsymbol{x}_t}, Y_{\boldsymbol{x}_t})$ of human regions in input images as feature vectors in this paper. Therefore, human actions are modeled by trajectories $\{(X_{\boldsymbol{x}_t}, Y_{\boldsymbol{x}_t})\}_{t=0}$ of center points and action primitives are modeled by segments of the trajectories. State of action $\boldsymbol{x}_t^{(i)}$ indicates a center point $(X_{\boldsymbol{x}_t}, Y_{\boldsymbol{x}_t})$ determined by its action primitive $m_t^{(i)}$ and its frame position $f_t^{(i)}$. We denotes the center point $(X_{\boldsymbol{x}_t}, Y_{\boldsymbol{x}_t})$ indicated by $\boldsymbol{x}_t^{(i)}$ as follows:

$$\left(X_{\boldsymbol{x}_t^{(i)}}, Y_{\boldsymbol{x}_t^{(i)}}\right) = (X_\tau^m, Y_\tau^m)\,. \tag{23}$$

4.2 Likelihood

Likelihood of a sample $p(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)})$ is calculated from the Mahalanobis distance between a center position indicated by a sample and observation $\boldsymbol{z}_t = (X_t^{\boldsymbol{z}}, Y_t^{\boldsymbol{z}})$ as follows:

$$p(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)}) \propto \exp\left(-\frac{1}{2}\frac{\sum_{l=-L/2}^{L/2} D_{t+l,\tau+\kappa l}}{L+1}\right), \tag{24}$$

where

$$D_{t,\tau} = \sqrt{S_{t,\tau}^T \Sigma^{-1} S_{t,\tau}}, \quad S_{t,\tau} = \begin{bmatrix} X_t^{\boldsymbol{z}} - X_\tau^m \\ Y_t^{\boldsymbol{z}} - Y_\tau^m \end{bmatrix}, \quad \Sigma = \begin{bmatrix} \sigma_X^2 & 0 \\ 0 & \sigma_Y^2 \end{bmatrix}, \tag{25}$$

and $\sigma_X^2 \sigma_Y^2$ are scaling parameters.

The proposed framework uses the Monte Carlo approximation. The accuracy of approximation increases in proportion to the number of particles and reaches the upper limit when the number of particles is infinity. In a real system it is impossible to prepare infinite particles, therefore prior knowledge of the target is incorporated into the likelihood for better calculation using finite particles.

4.3 State Transition

State transition formula, Eq. 15 is calculated by using Eq. 3. First, an action primitive is selected according to $p(m_t|\boldsymbol{x}_{t-1})$. This can be done by using the transition probability of action primitives expressed by the following equation:

$$p(m_t|\boldsymbol{x}_{t-1}) = r(x_{t-1})p(m_t|m_{t-1}) + (1 - r(x_{t-1}))\delta(m_t, m_{t-1}), \tag{26}$$

where $\delta(m_t, m_{t-1})$ is the Kronecker delta function that becomes 1 when $m_t = m_{t-1}$, otherwise 0, and $r(\boldsymbol{x}_t)$ is a transition probability that the transition of action primitives occurs or not. In this paper, $r(\boldsymbol{x}_t)$ is defined as follows:

$$r(\boldsymbol{x}_t) = \begin{cases} 1 - \varepsilon & f_{m_t} < f_t + v_t, \\ 0 & \text{no transition exists in the action database,} \\ \varepsilon & \text{otherwise,} \end{cases} \tag{27}$$

where f_{m_t} is the end frame of action model m_t.

Next, frame position f_t is updated. If $m_t = m_{t-1}$, the frame position f_t is updated as follows:

$$f_t = f_{t-1} + v_{t-1} + \mathcal{N}(0, \Sigma_f), \tag{28}$$

where v_{t-1} is velocity of state $\boldsymbol{x}_{t-1}$, and $\mathcal{N}(0, \Sigma_f)$ is an system noise assumed to be a Gaussian noise. Otherwise, it means that action primitive is changed, therefore f_t is chosen from several first frames of action primitive m_t. Finally, velocity v_t is updated by the following equation:

$$v_t = v_{t-1} + \mathcal{N}(0, \Sigma_v), \tag{29}$$

where $\mathcal{N}(0, \Sigma_v)$ is also a system noise assumed to be a Gaussian noise.

4.4 Transition Probability of Action Primitives

The transition probability of action primitives varies a frame position f_t of action state $\boldsymbol{x}_t$ because transition probability would increase when a frame position f_t draw near the last frame of a action primitive. Therefore, the transition probability of action primitive must be calculated from the current action state $\boldsymbol{x}_t$.

In this research, we calculate the transition probability ,$r(\boldsymbol{x}_t) = r(m_t, v_t, f_t)$, for simplicity by the following equation:

$$r(m_t, v_t, f_t) = \begin{cases} \varepsilon_r & f_m - f_t + v_t > \varepsilon_f \\ 1 - \varepsilon_r & \text{otherwise} \end{cases}, \tag{30}$$

where f_m is number of frames of action primitive m_t indicated by action state $\boldsymbol{x}_t$. f_t and v_t are a frame position and velocity of action primitive m_t indicated by action state $\boldsymbol{x}_t$, respectively. ε_r and ε_f are small positive constants. The above equation expresses that the transition might not occur until one action primitive would finish.

5 Experiments

50 indoor scenes captured with an omnidirectional image sensor[7] were used to test the proposed method. The size of images was 512×440 pixels and the depth of images was 8 bits gray. The room was 7×7 m. The camera was fixed at the center of the room at height of 140 cm. Trajectories of human movement extracted with a tracking module were shown in Fig. 5. We use 5 routes of trajectories and 10 trajectories of each route, and total 50 trajectories were used in the experiments. Each route is shown in Table 1.

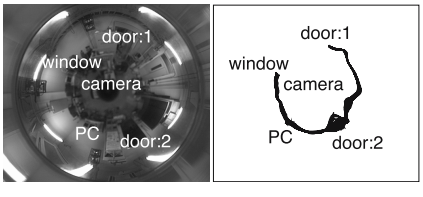


Fig. 5. Input image and trajectories

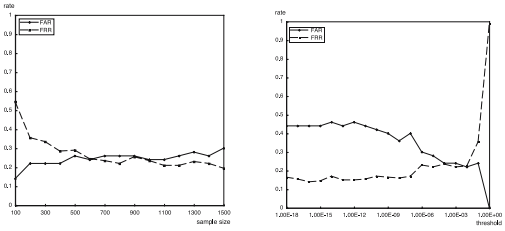
Table 1. Routes

Route	Trajectory
1	door:1 → door:2
2	door:2 → door:1
3	door:1 → PC → door:1
4	door:2 → window → door:2
5	door:2 → PC → door:2

5.1 Suspicious Action Detection

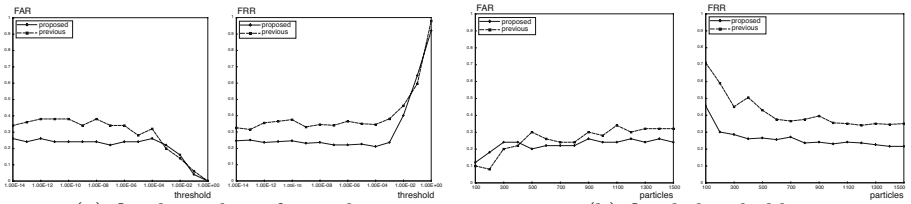
Detection rate is affected by parameters such as a threshold value and number of sampling points. Therefore, we examine the detection rate by fixing one parameter and changing another parameter. Experiment is performed by the jackknife test.

Figure 6 (a) shows the detection rate that the threshold value is fixed at 1.0×10^{-4} and Fig. 6 (b) shows the detection rate that the number of sample points is fixed at 1000. From Fig. 6 (a), the accuracy of approximation becomes better when the number of samples is increased, therefore, false rejection rate (FRR) becomes smaller, but it takes much time that all samples are below the threshold and false accept rate (FAR) becomes worse. In the range of number of samples, $600 \sim 1000$, the both error are equal. The system performs detection process in 60 fps when the system uses 1000 samples. From Fig. 6 (b), when the



(a) fixed threshold (b) fixed number of samples

Fig. 6. Detection rate of suspicious action



(a) fixed number of samples (b) fixed threshold

Fig. 7. Comparative experiment: detection rate

threshold value becomes larger, normal actions are misclassified to suspicious actions, so FRR becomes worse, vice versa, when the threshold value becomes smaller, suspicious actions are accepted as normal actions, and FAR becomes worse. As a result, we must carefully choose the threshold value and the number of samples. In this experiment, if these parameters are selected carefully, equal error rate (ERR) is about 20%. In future work, we will improve the detection rate.

We also conducted a comparative experiments with the proposed method and the previous method[6]. Figure 7(a) shows the detection rate that the number of sample points is fixed at 1000, and Fig. 7(b) show the detection rate that the threshold value is fixed at 1.0×10^{-4}. The left column in the both figure shows FAR and the other side shows FRR. The proposed and previous method have the same tendency, but the FAR and FRR in the proposed method are better than those of the previous method.

6 Conclusions and Future Work

We have proposed a generic framework for detecting suspicious actions with mixture distributions of action primitives, of which collection represents human actions. We used an Bayesian approach to recognize human actions and approximate the probability by the Monte Carlo method. We also applied Sequential Monte Calro method for fast calculation. Sequential Monte Calro has the disadvantage that it converges one local minimum. We have solved that problem

by using mixture distributions of action primitives. The EER of the proposed method is 20 % and 80 % of suspicious actions are successfully detected in the experiment. In addition, it is clarified that the number of samples for Monte Carlo approximation and the threshold value for detection of suspicious actions must carefully be selected. In future work, we will investigate the effects of these parameters and improve the recognition rate.

References

1. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), vol. 1, pp. 511–518 (2001)
2. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based probabilistic tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 661–675. Springer, Heidelberg (2002)
3. Oliver, N.M., Rosario, B., Pentland, A.P.: A Bayesian computer vision system for modeling human interactions. IEEE Trans. on PAMI 22(8), 831–843 (2000)
4. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. IJCV 29, 5–28 (1998)
5. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit probabilistic models of human motion for synthesis and tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 784–800. Springer, Heidelberg (2002)
6. Matsumura, A., Iwai, Y., Yachida, M.: Stochastic action recognition from omnidirectional images. In: Proc. of Asian Conf. on Computer Vision, vol. 1, pp. 120–125 (2004)
7. Yamazawa, K., Yagi, Y., Yachida, M.: Omnidirectional imaging with hyperboloidal projection. In: Proc. of the Int. Conf. on Intelligent Robots and Systems(IROS 1993), vol. 2, pp. 1029–1034 (1993)
8. Mituyoshi, T., Yagi, Y., Yachida, M.: Real-time human feature acquisition and human tracking by omnidirectional image sensor. In: Proc. IEEE Conf. on Multisensor Fusion and Integration for Intelligent Systems, pp. 258–263 (2003)
9. Black, M.J., Jepson, A.D.: A probabilistic framework for matching temporal trajectories: CONDENSATION-based recognition of gestures and expressions. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 909–924. Springer, Heidelberg (1998)
10. Himeno, M., Himeno, R.: The effect of crossover and mutation to DC in early generations for multimodal function optimization. IEICE (D-I) J85-D-I(11), 1015–1027 (2002)
11. Vermaak, J., Doucet, A., Pérez, P.: Maintaining multi-modality through mixture tracking. In: Proc. 9th ICCV, vol. 2, pp. 1110–1116 (2003)