Appearance Based Behavior Recognition by Event Driven Selective Attention

Toshikazu Wada Takashi Matsuyama

Department of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University
Yoshida Hon-machi, Sakyo, Kyoto, 606-8501, JAPAN

Abstract

Most of behavior recognition methods proposed so far share the limitations of bottom-up analysis, *and* single-object assumption*; the bottom-up analysis can be confused by erroneous and missing image features and the single-object assumption prevents us from analyzing image sequences including multiple moving objects. This paper presents a robust behavior recognition method free from these limitations. Our method is best characterized by 1) top-down image feature extraction by* selective attention mechanism, *2) object discrimination by* colored-token propagation, *and 3) integration of multi-viewpoint images. Extensive experiments of human behavior recognition in real world environments demonstrate the soundness and robustness of our method.*

1 Introduction

Motion understanding is essential for wide varieties of vision applications, such as visual surveillance, human interface and virtual reality. Motion understanding problems can be categorized into the following three levels:

Physical motion analysis: Measure the time sequence of 3D or 2D object locations and shapes.

Object behavior recognition: Classify object motions into a set of behavior patterns (i.e. classes), which emerge from constraints on object properties and surrounding physical environments.

Object action understanding: Reason about the object intention from motions, e.g. gesture, sign language, flag semaphore, and so on.

In this paper, we address the object behavior recognition problem.

In general, the visual object behavior recognition consists of 1) image feature extraction and 2) temporal sequence analysis of extracted features.

Most of the methods proposed so far (e.g. [1], [2], [3], [4]) employ Hidden Markov Model (HMM) for sequence analysis. HMM realizes flexible matching between the detected feature sequence and given model sequences by finding the optimal state transition path which maximizes a matching measure(probability) under the assumption of Markov property. Since the state transitions are obtained by the optimization, current states in HMM are *hidden*.

These systems share the following limitations:

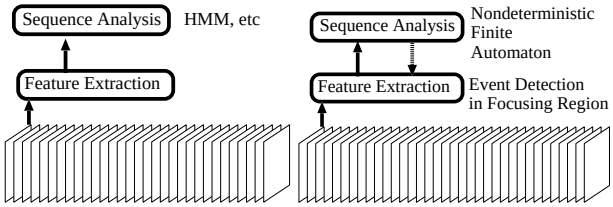

Figure 1: Behavior recognition system. (a): Bottom-up system, (b): Bottom-up and Top-down system.

1 **Bottom-up analysis**: Feature extraction is followed by sequence analysis as shown in Figure.1 (a). This means severe limitation on stability and robustness; erroneous and missing feature will confuse sequence analysis and cause fatal errors.

2 **Single-object assumption**: Number of objects in each image frame is assumed to be one or zero. This assumption limits the applicability; multiple object behaviors cannot be classified simultaneously.

These limitations prevents the systems from recognizing multiple object behaviors in a single image sequence, which is essential for visual surveillance tasks.

To remove these limitations, we propose the following methods:

1 **Selective attention mechanism**: Feature sequence analysis and feature extraction are designed as:

Sequence analysis: Nondeterministic Finite Automaton(NFA: described in section2) is employed as a sequence analyser. NFA is a simple state transition model which allows state transitions from a single state to multiple states for an input. The reason why we use NFA instead of HMM are:

1. Active states of NFA are *not hidden*, i.e., they can be monitored at any time. Based on this property, we can design *top-down feature extraction* referring active states.

2. We can track all possible feature sequences by using NFA, i.e., multi-context behavior analysis can be realized.

Feature extraction: Image features are extracted in specified image regions which varies with active states

of NFA. This can be regarded as a *top-down* feature extraction.

We call the above behavior recognition mechanism *selective attention mechanism*(Figure.1 (b)).

2 **Object discrimination mechanism**: In selective attention mechanism, however, multiple states are simultaneously activated for a single object behavior and the number of objects cannot be recognized. To eliminate this drawback, the following mechanism is also employed:

• The activated states are marked by *colored tokens*, where a color corresponds to an object. By propagating colored tokens on activated states, different object behaviors are discriminated.

Based on these ideas, we can realize a robust multi-behavior recognition system.

However, the appearance based behavior recognition methods share a limitation that 3D object behaviors along viewing direction are degenerated on 2D image plane and hardly recognized. To remove this limitation, we further extend the above method for multi-viewpoint images.

In the following sections, behavior recognition method, multi-viewpoint extension, practical design and experimental results are described.

2 Behavior Recognition

Here we describe the behavior recognition method consists of selective attention and object discrimination mechanism.

2.1 Selective Attention Mechanism

In image sequences of object behaviors belonging to a class, we can find an image variation sequence specifying the behavior class. For example, when we open and go out through a door, we can find image variations at 1) the door knob, 2) the edge of the door, and so on.

If the camera is fixed, such image variations can be detected in specific image regions[1]. We call this region *focusing region*. If a focusing region sequence specifying a behavior class is given, we can identify behaviors belonging to the class by detecting image variations(*events*) in focusing regions.

In this case, focusing regions should be changed according to the behavior stages. This can be realized by corresponding states in sequence analyser (NFA) to the focusing regions. That is, the detected events activate states in NFA, and the event detector changes its focusing regions according to the activated states. We call this bottom-up and top-down behavior identification mechanism *selective attention mechanism*(Figure.2). The behavior identifier based on selective attention mechanism is described below:

A behavior identifier is a simple classifier which accepts image sequences of a behavior class and rejects those in the others. The behavior identifier consists of 1) a focusing region sequence, 2) a sequence analyzer, and 3) an event detector:

[1]In this paper, we will use a word "region" as a set of pixels.

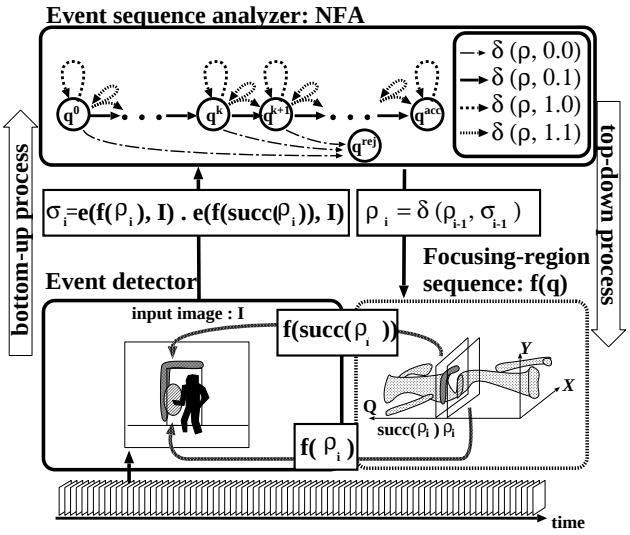

Figure 2: Behavior identifier

Definition 1 (Focusing region sequence)
A focusing-region sequence $f(q)$ is a sequence of image regions where the event detection is performed at state q.

A practical method to acquire a focusing region sequence from training samples is described in section 4.

Definition 2 (Sequence analyzer) *The sequence analyzer (NFA) is represented by $(Q, q^0, \Sigma, \delta, F)$, where*
- *Q : finite set of states*[2]*, $Q = \{q^0, q^1, \cdots, q^m, q^{rej}\}$,*
- *q^0: initial state ,$q^0 \in Q$,*
- *Σ: finite set of event codes,*
- *δ: state transition function, $\delta(q, \sigma)$: $Q \times \Sigma \mapsto Q$,*
- *F: finite set of final states, $F = \{q^m, q^{rej}\}$.*

When an input sequence $\sigma_i \in \Sigma$ $(i = 0, 1, \cdots)$ is given, the active state ρ_i at i-th input is defined by δ as:

$$\begin{cases} \rho_0 & = & q^0, \\ \rho_{i+1} & = & \delta(\rho_i, \sigma_i), \end{cases} \quad (1)$$

Note that:
- *The state transition is not applied to states in F,*
- *δ may have multiple values for a single input. This is called non-deterministic state transition.*
- *A state q^m represents that the input sequence is accepted as the behavior class. Hence, q^m is also represented by q^{acc}.*
- *A state q^{rej} represents that the input is rejected.*

Definition 3 (Event Detector) *An event is a predicate representing the presence of an image feature in a focusing region. The event of an image I with a focusing region f is represented by $e(f, I)$ $(\in \{0, 1\})$.*

[2]A state sequence $(q^1, \cdots, q^m)$ represents an abstracted time axis, i.e., each state in this sequence corresponds to a time interval. See section 4.

Table 1: State transition table at $\rho_i = q^k$ for event code length=2.

$e(f(q^k), I_i) \cdot e(f(q^{k+1}), I_i)$	ρ_{i+1}
$0 \cdot 0$	q^{rej}
$0 \cdot 1$	q^{k+1}
$1 \cdot 0$	q^k
$1 \cdot 1$	q^k or q^{k+1}

Event detector detects events at successive states and combines them into an event code. For example, event code of length 2 at state ρ is represented by:

$$\sigma = e(f(\rho), I) \cdot e(f(succ(\rho)), I), \tag{2}$$

where $succ(\cdot)$ represents the successor function, i.e., if $\rho = q^k$, then $succ(\rho) = q^{k+1}$.

An example of event detection method is described in section 4.

Definition 4 (Behavior identifier) *These components described above are assembled into an identifier in the following manner:*

- *The* **event code** *is used as an input to the sequence analyser,*
- *The* **active states** *of sequence analyser are used to search keys of focusing regions,*
- *The* **focusing region** *is used in the event detector.*

In the case that the event code length is 2, the active state ρ_i at i-th input is defined as:

$$\begin{cases} \rho_0 &= q^0, \\ \rho_{i+1} &= \delta(\rho_i, \sigma_i), \\ \sigma_i &= e(f(\rho_i), I_i) \cdot e(f(succ(\rho_i)), I_i). \end{cases} \tag{3}$$

If a final state q^{acc} is activated, then the input sequence is identified as the behavior.

In the case that the event code length is 2 and $\rho_i = q^k$, ρ_{i+1} can be determined by $e(f(q^k), I_i)$ and $e(f(q^{k+1}), I_i)$, because an event $e(f(q^k), I_i) = 1$ represents the evidence of $\rho_{i+1} = q^k$, and $e(f(q^{k+1}), I_i) = 1$ represents $\rho_{i+1} = q^{k+1}$. Such homogeneous state transition can be described by a state transition table. An example of the state transition table is shown in Table.1.

In Table1, an event code $\sigma_i = 1 \cdot 1$ will cause a nondeterministic state transition from q^k to both q^k and q^{k+1}. This property enables tracking of all possible event sequences.

2.2 Object Discrimination Mechanism

By using selective attention mechanism, multiple object behaviors can be identified from a single image sequence. In this mechanism, however, multiple states are simultaneously activated for a single object. This means that the number of objects cannot be recognized. To solve this problem, we have to correspond the activated states to objects.

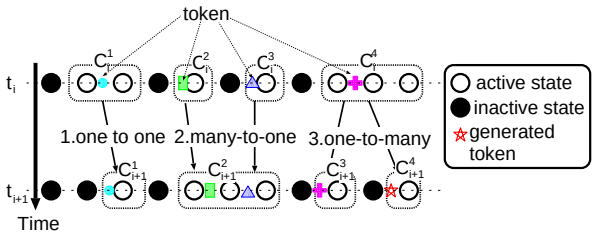

Figure 3: Token propagation patterns

In practice, since the focusing regions at neighboring states are similar with each other for continuous object behaviors, neighboring states are activated for a single object. Hence, the neighboring active states are considered to be activated by an object.

A neighboring active state set C is a subset of active states connected by state transition function δ, that is:

$$^{\forall \sigma \in \Sigma}((\rho \in C \Rightarrow \delta(\rho, \sigma) \in C) \vee (\delta(\rho, \sigma) \in C \Rightarrow \rho \in C)). \tag{4}$$

An active state set P can be decomposed into disjoint sets of neighboring active states C^k, i.e. , $P = \cup C^j$, and $C^k \cap C^j = \phi$ for any $C^k \neq C^j$.

By marking all states in C^k by a token having a color $z^k \in Z$, active states can be corresponded to objects, where Z represents integer set, and $C^k \neq C^j \Rightarrow z^k \neq z^j$.

In this method, however, the number of token colors z^k may vary with time even if the number of objects is constant. To count the correct number of objects, token colors must be consistently assigned to C^k. This can be realized by propagating tokens along the time axis. Since state transitions represent progress of object behaviors, tokens should also be propagated by state transition function δ. That is, tokens are propagated from C_i^j to C_{i+1}^k satisfying:

$$\left(\bigcup_{\sigma \in \Sigma} \bigcup_{\rho \in C_i^j} \delta(\rho, \sigma) \right) \cap C_{i+1}^k \neq \phi, \tag{5}$$

where C_i^j represents a neighboring active state set at i-th input. In this case, C_i^j has a *link* to C_{i+1}^k.

But, when C_i^j has multiple links, we should not propagate the copies of tokens through those links, because any active states in different C^ks should not have the same token simultaneously.

Based on the discussion above, tokens should be propagated as follows (Figure.3):

- When C_i^j has no links, tokens are discarded.
- When C_i^j has a single link, tokens are propagated through the link.
- When C_i^j has multiple links, tokens are divided into disjoint sets, which are distributed through the links. If the number of links is greater than that of tokens, new tokens should be generated after the propagation. Any division minimizing the token generation is feasible.

By incorporating this token propagation, behavior identification procedure can be described as:

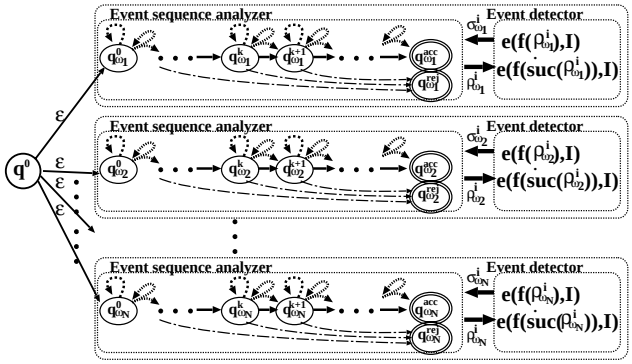

Figure 4: Behavior classifier

1 For each $C_i^j \in P_i$, if no token is assigned, then new token is generated and assigned to it.

2 Compute state transitions for I_i and form $\{C_{i+1}^k\}$ from activated states ρ_{i+1}.

3 Propagate tokens from $\{C_i^j\}$ to $\{C_{i+1}^k\}$ through links among them.

4 Count the token colors at q^{acc}. Increment i and goto 1.

By using this procedure, we can identify object behaviors.

2.3 Behavior Classification

To recognize object behaviors, we have to classify behavior patterns. The classifier consists of independent behavior identifiers. By introducing an initial state q^0 and ϵ-state transitions[3] from q^0 to initial states of these identifiers, the classifier can easily be realized as shown in Figure.4.

3 Multi-Viewpoint Behavior Recognition

Appearance based behavior recognition methods share a limitation that 3D object behaviors along viewing direction are degenerated on 2D image plane and hardly recognized. To remove this limitation, we further extend the above method for multi-viewpoint images.

This extension is to integrate multi-viewpoint information, which can be categorized into the following three levels(Figure.5):

Image-level integration : Multi-viewpoint images are pasted into an image and the recognition system takes it as input.

Event-level integration : Independently detected event codes from multi-viewpoint images are integrated into an event code, which drives a sequence analyzer.

State-level integration : Independent behavior identification systems mutually interact to inhibit redundant state transitions.

Image-level integration cannot verify the cooccurence of image features in multi-viewpoint images, i.e., this method neglects the difference of viewpoints.

In event-level integration, event codes detected at different viewpoints are integrated by bitwise 'AND' operation,

[3]The ϵ-state transition means a state transition caused by null input.

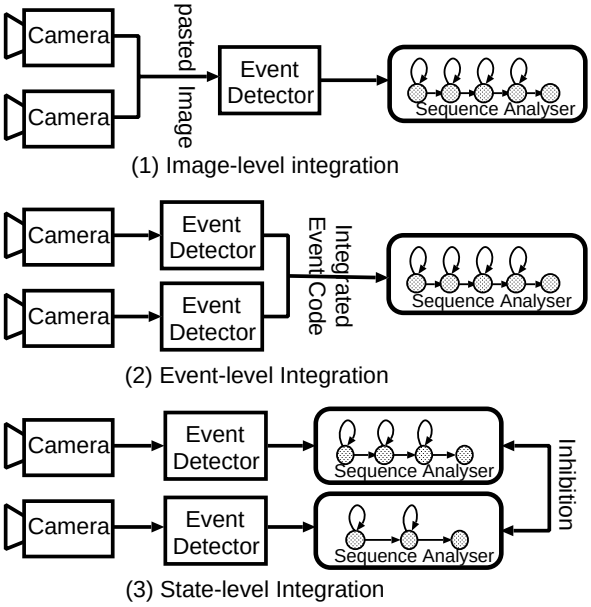

Figure 5: Three Levels of Integration

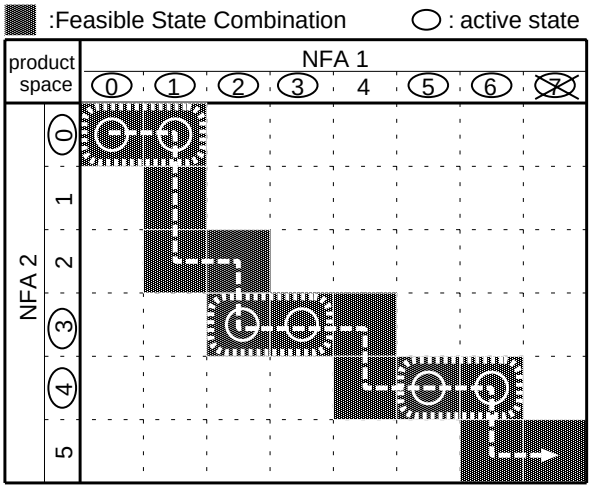

Figure 6: State Product Space

which checks the cooccurence of events corresponding to an event in 3D space[4].

State-level integration inhibits redundant active states according to *feasible state combinations*, which represent the valid combinations of active states in identifiers at different viewpoints. Feasible state combinations can be learned from training samples as shown in section 4. When the time intervals corresponding to states are synchronized among the identifiers at different viewpoints, this integration is equivalent to the event-level integration. That is, this integration includes the event-level integration as a special case.

The feasible state combinations can be represented by a set of points in state product space of identifiers at different viewpoints (Figure.6). This integration can be regarded as

[4]"An event in 3D space" does not mean 3D object location but the occurrence of an event when the cameras are not calibrated.

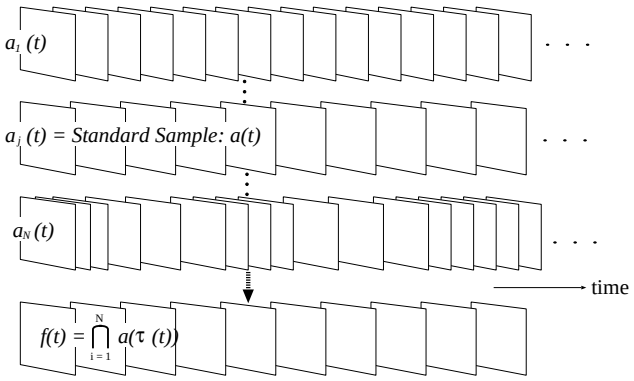

Figure 7: Learning common anomalous region sequence

an augmentation of domain of state transition and token propagation from states to feasible state combinations.

4 A Practical Design

By using background subtraction, anomalous region at time t can be obtained as:

$$a(t) = \{(x, y) \mid |I(x, y; t) - I_{bg}(x, y)| > \nu\}, \quad (6)$$

where $I(\cdot, \cdot; t)$ represents input image at time t, $I_{bg}(\cdot, \cdot)$ background image, and ν threshold. Here we describe a practical event detection and learning method using anomalous features.

Event Detection

An event can be detected when anomalous pixels fill the focusing region, which is defined as:

$$e(f, I(t)) = \begin{cases} 1, \ f = \phi \ or \ \frac{|f \cap a(t)|}{|f|} > \theta \\ 0, \ otherwise \ , \end{cases} \quad (7)$$

where θ ($0 < \theta \leq 1$) represents a threshold, f the focusing region, and ϕ the null focusing region.

Focusing Region at Initial state

We assign null focusing region ϕ to state q^0. This guarantees that the current state set always includes the initial state q^0 as a gatekeeper. Hence, we can recognize behaviors successively by using this configuration.

Learning a Focusing Region Sequences

A focusing-region sequence can easily be acquired from the anomalous region $a(t)$. When n samples of anomalous region sequences $a^i(t)$ $(i = 1, 2, \cdots, n)$ in a class are given, time axes of these samples can be normalized so as to maximize the following matching measure by the dynamic programming:

$$\int \frac{|a(t) \cap a^i(\tau_i(t))|}{|a(t) \cup a^i(\tau^i(t))|} dt, \quad (8)$$

where $a(t)$ is the standard sample of the class, τ^i is a monotone increasing function, and $|\cdot|$ represents the number of pixels.

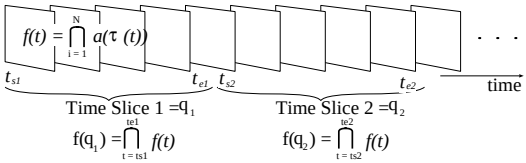

Figure 8: Learning a focusing region sequence

From the normalized anomalous region sequences, the common anomalous region sequence among the training samples $f(t)$ can be extracted as:

$$f(t) = \bigcap_{i=1}^{n} a^i(\tau^i(t)). \quad (9)$$

The common anomalous region sequence is represented in the normalized time axis t. Intervals along this time axis corresponds to the states of an NFA. If a state q corresponds to a given time interval $t_s \leq t < t_e$, focusing region $f(q)$ at the state q is computed as:

$$f(q) = \bigcap_{t=t_s}^{t_e - 1} f(t) \quad (10)$$

Learning Feasible State Combinations

For the multi-viewpoint behavior recognition, feasible state combinations must be learned from training samples. By using standard samples of $a^{cam1}(t), \cdots, a^{camN}(t)$ for the same behavior where the time axes are synchronized, we obtain common anomalous region sequences $f^{cam1}(t), \cdots, f^{camN}(t)$. Note that these sequences share a common time axes. When we generate focusing region sequences $f^{cami}(q)$, we can refer the corresponding time intervals, and hence, the feasible state combinations can be obtained.

5 Experiments

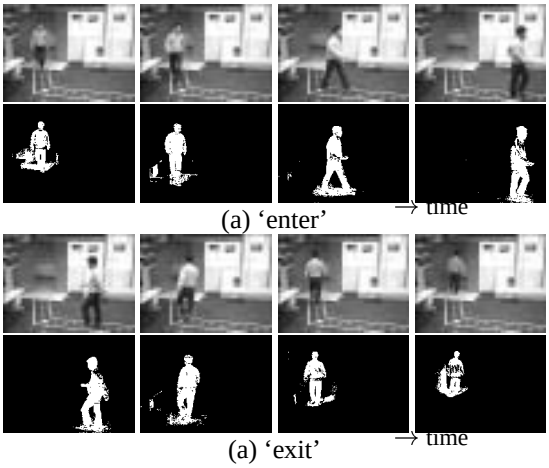

Figure 9: Examples of training data (gray-level image, anomalous region)

Here we show experimental results of recognizing human behaviors incoming and outgoing through a door

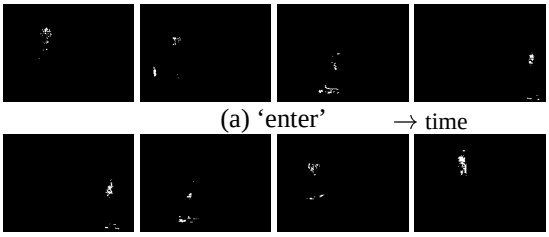

Figure 10: Focusing region sequences

(Figure.9 (a),(b)) by using 2 cameras. These behaviors are named "enter" and "exit". For each behavior class, 20 training data, i.e., 40 image sequences of single object behaviors, are used for acquiring focusing regions. The acquired focusing regions are shown in Figure.10.

Possible methods are non-integrated classifications using (a) camera1, (b) camera2, and integrated classifications at (c) image level, (d) event level, (e) state level. These methods are applied to 60 test data, i.e., 120 image sequences of multiple object behaviors. Examples of test data and state transitions in independent identifiers (state-level integration) are shown in Figure.11 and 12. In this case, 2 'enter's and 2 'exit's are correctly recognized.

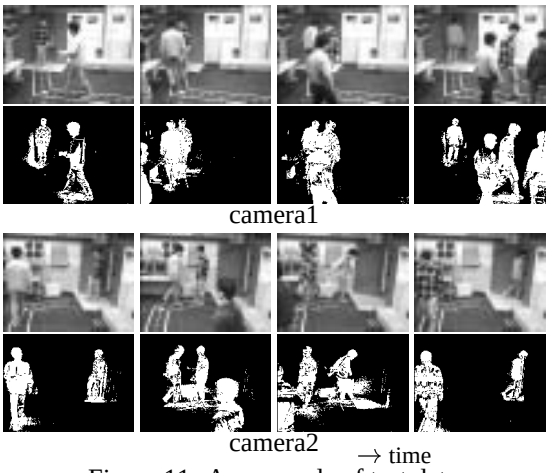

Figure 11: An example of test data

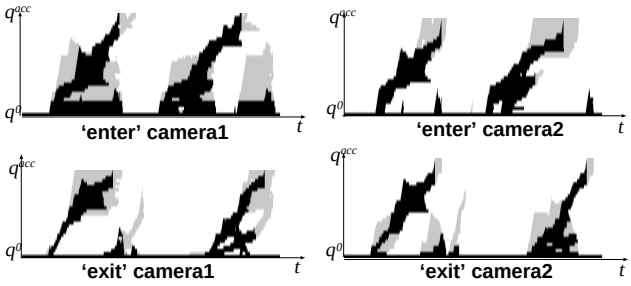

Figure 12: State Transitions for Fig. 11 (gray: inhibited states. black: activated states).

The classification results are shown in Figure.13. In each graph, vertical axis represents the number of data which are correctly recognized in both senses of 'classification' and

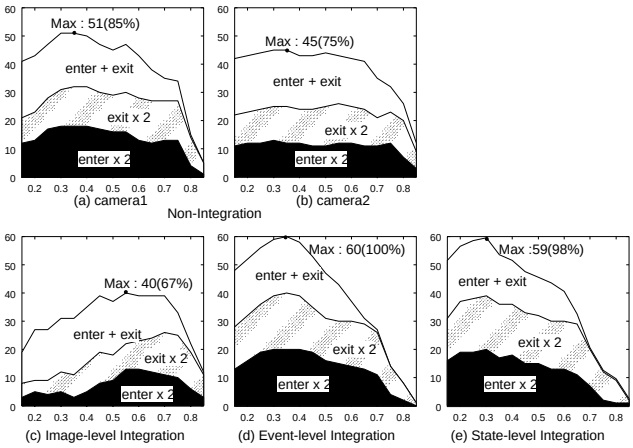

Figure 13: Recognition results (Horizontal: threshold θ, Vertical:Number of correct recognition).

'number of objects'. Horizontal axis represents the threshold θ for event detection. From this figure, we can notice that event and state level integrations are effective than the others. State-level integration includes event-level integration as a special case. However, the latter is slightly superior to the former in this result. This can be supposed that each class has been excessively specialized at the learning stage in state-level integration because of its superiority. In other word, 20 training data are insufficient for generalizing these behavior classes.

6 Conclusion

In this paper, we proposed a behavior recognition method for multiple objects. It is designed based on *selective attention* and *object discrimination* mechanisms. Also, integrated recognition methods for multi-viewpoint images are proposed and examined by extensive experiments of human behaviors. In this experiment, we have confirmed that *event* and *state* level integration methods correctly recognize even for complex behaviors. Currently this method can be applied only to the position dependent behaviors. This limitation will be removed in the future work.

Acknowledgment

Authors are thankful to Mr. Masayuki Sato in Kyoto University for his assistance. This work was supported by the Research for the Future Program of the Japan Society for the Promotion of Science (JSPS-RFTF96P00501) and Grant-in-Aid for Scientific Research ((A)(2)08408010).

References

[1] Yamamoto J., Ohya J., and Ishii K., "Recognizing human action in time-sequential images using hidden markov model", Proc. of **CVPR**, pp. 664-665, (1992)
[2] Starner T. and Pentland A., "Real-time American sign language recognition from video using hidden markov models", Proc. of **ISCV**, pp. 265-270, 1995.
[3] Bregler C. and Omohundro S.M., "Nonlinear manifold learning for visual speech recognition", Proc. of **ICCV**, pp.494-499, 1995.
[4] Wilson A. and Bobick A., "Learing Visual Behavior for Gesture Analysis", M.I.T. Media Laboratory Perceptual Computing Section Technical Report No.337. 1995.