*Research Paper*

# A Single Framework for Action Recognition Based on Boosted Randomized Trees

Takayoshi Yamashita,[†1,†2] Yuji Yamauchi[†2] and Hironobu Fujiyoshi[†2]

Human detection and action recognition form the basis for understanding human behaviors. Human detection is used to detect the positions of humans, and action recognition is able to recognize the action of specific humans. However, numerous approaches have been used to handle action recognition and human detection separately. Therefore, three main issues still exist when independent methods of human detection and action recognition are combined, 1) intrinsic errors in object detection impact the performance of action recognition, 2) features common to action recognition and object detection are missed, 3) the combination also has an impact on processing speed. We propose a single framework for human detection and action recognition to solve these issues. It is based on a hierarchical structure called Boosted Randomized Trees. The nodes are trained such that the upper nodes detect humans from the background, while the lower nodes recognize action. We were able to improve both human detection and action recognition rates over earlier hierarchical structure approaches with the proposed method.

## 1. Introduction

Human detection and action recognition form the basis for understanding human behaviors and human detection is an important function for action recognition. By detecting the position of humans, action recognition is able to recognize the actions of specific humans.

Human detection in an image sequence can be achieved by subtracting the background or by sliding a detector into the whole frame [1),4)]. Background subtraction approaches that are robust toward illumination or changes in appearance have been proposed, but they only detect moving objects and cannot differentiate humans from other objects such as vehicles or animals. Detector based approaches are robust toward changes in illumination; however, detection is limited to standing poses [9),13),14)].

Methods of action recognition can be broadly classified into two categories, the first identifies actions by using global changes in the scene and the second by local motion in invariant features. Methods that are based on global changes rely on inter-frame differences [3),10),17),23)]. There are two major problems with existing methods. First, the inter-frame differences are measured as global motion, and detect the positions of humans are not possible. When an object in the background also moves, the method cannot differentiate between motion by the foreground object from that by the background object. Action recognition by local motion extracts invariant features and motion by matching the features between frames [15),16),19),22),26)]. It achieves better performance by implicitly separating humans from the background. It is also difficult for these methods to detect humans in cluttered backgrounds or when there are other moving objects in the background.

As previously mentioned, most of the former approaches handle action recognition and human detection separately. There is a need to combine human detection and subsequent action recognition to recognize action in a particular image. Three of the main issues when human detection and action recognition are combined are: 1) intrinsic errors in object detection impact the performance of action recognition, 2) features common to action recognition and object detection are missed, 3) the combination also has an impact on the processing speed. We propose a single framework for human detection and action recognition to overcome some of these issues. A hierarchical framework accomplishes this within a single framework, where the upper nodes differentiate humans from the background, and lower nodes classify specific actions. A single framework also means that features can be shared across human detection and action recognition, thus resulting in a simple and optimized framework.

All nodes of Randomized Trees are created from weak classifiers that are trained from pre-selected features candidates to achieve the framework. Candidate features in the proposed method are trained by Joint Boosting and effective weak classifiers are selected as the nodes of decision trees by training Randomized

†1 OMRON Corporation
†2 Chubu University

Trees.

Some related work is discussed in Section 2 and the proposed approach is explained in Section 3. We compare the performance of the new method with others for human detection and action recognition in Section 4. There is a brief discussion in Section 5 and we conclude the paper in Section 6.

## 2. Related Works

Many methods of action recognition have been proposed to enable human behaviors to be understood and they can be broadly classified into two categories based on global changes[3),10),17),23)] and local motion with invariant features[15),16),19),22),26)]. Blank et al. proposed a method called ST-Patch for detecting inter-frame spatial and time differences[10)]. Neibles and Fei-Fei proposed a multi-layered model for detecting changes in appearance and motion[15)]. Oshin et al. proposed a model that extracts Speeded Up Robust Features (SURF) for modeling temporal changes[26)]. Almost all the action recognition based on detecting local changes in the target object, used changes in the directions and poses of the target object.

Action recognition is a multi-class classification problem, on the other hand, for which various multi-class classification methods have been proposed. Joint Boosting[5)] can detect the positions and actions of the target object. It trains weak classifiers that are shared between classes, and hence it is unnecessary to train classifiers for specific classes, enabling multi-class classifiers to be trained with fewer weak classifiers. However, it involves huge computational cost because weak classifiers of all classes are necessary to classify specific classes.

This can be solved with a hierarchical structure such as that with decision trees[11),24),25)]. The processing time for a particular class with a tree structure depends on the tree depth. AdaTree[7)] trains classifiers in a hierarchical structure and each node consists of many weak classifiers. Training samples are divided into subsets with samples that are positively classified in the parent nodes. Lower nodes are trained with subsets that contain few samples. Therefore, generalization is affected and it overfits the training samples. Randomized Trees represent a tree structure for multi-class recognition[2),8),12),18)]. An ensemble of decision trees that outputs the likelihood for each of the classes is trained to avoid overfitting. Classification is carried out by summing the likelihoods of all classes from all the decision trees. However, the effectiveness of the features is not guaranteed as the features for each node are selected from those that are randomly prepared.

Candidate features with the proposed method are pre-selected using Joint Boosting, and the features for each of the nodes are randomly selected from the candidates. This enables us to select features from a candidate pool of effectively pre-selected features, while retaining randomness. The background is trained as a particular class that classifies target objects from the background, enabling objects to be detected. This framework enables both human detection and action recognition.

## 3. Proposed Method and Related Methods

### 3.1 Randomized Trees

Randomized Trees[2)] is a method of multi-class recognition learning that is used in keypoint detection[8)] and image classification[18)]. It is robust against noise in the training samples, and computational parallelization is possible as all decision trees are independent. It consists of multiple decision trees, $T$, with branch nodes and terminating leaves. When recognizing $C$ individual classes, each leaf has a probability distribution for each of the classes, $c = 1, 2, ..., C$, and branching at each node is based on a split function. The split function determines if feature $I(x)$ on the left child node is less than the threshold, $\theta$, or if that on the right child node is larger, as shown in Eq. (1)

$$I(x) = \begin{cases} < \theta & \textit{branch to the left child node} \\ \geq \theta & \textit{branch to the right child node} \end{cases} \qquad (1)$$

The training consists of three processes: creating subsets, generating nodes, and partitioning the subsets. First, subset $X_s$ of the training sample, $X = x_i, c_j; i \in [1, N], j \in [1, C]$, is created to train the decision trees. The subset is a randomly selected set of $S$ sample images. Nodes are made of a split function, a feature and a threshold. For prepared features, $f_m; m \in [1, M]$ and thresholds, $\theta_{m,k}; k \in [1, K]$, the best combination is selected based on information entropy as in Eq. (2).

$$\triangle E = -\frac{|I_l|}{|I|}E(I_l) - \frac{|I_r|}{|I|}E(I_r) \tag{2}$$

Note that $E(I_l)$ and $E(I_r)$ are the Shannon entropy for the samples in each class when taking the left or right branch for a given combination of features and thresholds. The Shannon entropy is computed as in Eq. (3).

$$E(I) = -\sum_{j=1}^{C} P(c_j)\log P(c_j) \tag{3}$$

$P(c_j)$ is the probability distribution for class $c_j$ at the node.

Subsets are partitioned by using the features that were selected as was described above. Feature values less than the threshold form the subset for the left child node, and values larger than the threshold form the subset for the right child node. This process is repeated on each child node using the new subsets.

Node generation is terminated when the number of training samples is less than a pre-determined depth, or when the training samples comprise only a single class, or when the nodes have reached a certain depth. Terminating leaves have a probability distribution, $P(c)$, for each class. The probability distribution for class $c_j$ can be computed as in Eq. (4).

$$P(c_j|l) = \frac{|I_{c_j}|}{|I|} \tag{4}$$

$|I|$ is the number of samples for all classes, and $|I_{c_j}|$ is the number of samples for class $c_j$.

The input image reaches a single leaf node in each of the decision trees. Then, the probability distributions, $P(C|L_t)$, for each of these leaf nodes, $L = L_t; t \in [1, T]$, are accumulated for each class as in Eq. (5) and the average is obtained. The class with the highest average probability in Eq. (5) is output as the recognition class.

$$P(C|L) = \frac{1}{T}\sum_{t=1}^{T} P(C|L_t) \tag{5}$$

### 3.2 Joint Boosting

Joint Boosting[5] is a multi-class learning algorithm that enables features to be selected shared between classes. As shown in Eq. (6), Joint Boosting trains strong classifiers for the partial sets, $S(n)$, of all classes.

**Algorithm 1** Joint Boosting Algorithm

```
Initialization:
1. Inizialize training sample weight  $w_i^c$
For $i = 1..N$        //No. of samples
   For $c = 1..C$        //No. of classes
      initialize training sample weight$w_i^c$
Training:
2. $m = 1, 2, .., M$        //No. of weak classifier selected
   (a) $n = 1, 2, .., 2^C - 1$        //For all combination of classes
      (i)Compute error for all weak classifier candidates
   (b)Select class combination $S(n)$, with minimal error and the weak classifier candidate
      $h_m^{S(n)}(v, c)$
   (c)Update weight  $w_i^c$
3. Integrate the selected weak classifier with each combination, $n$, to $G^{S(n)}(v)$
```

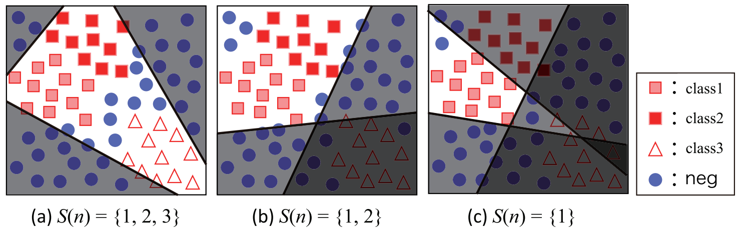


**Fig. 1** Learning example of Joint Boosting.

$$G^{S(n)}(v) = \sum_{m=1}^{M} h_m^{S(n)}(v, c) \tag{6}$$

Here, $h_m^{S(n)}(v, c)$ is the $m$th weak classifier, and $v$ is the feature vector. The training process consists of changing the combinations of positive classes and selecting the best weak classifiers as shown in Algorithm 1. Weak classifier $h_m^{S(n)}(v, c)$ with minimum error from all $2^C - 1$ is selected. For $S(n) = \{1, 2, 3\}$, a weak classifier for all positive classes is trained as shown in **Fig. 1** (a). Similarly, for $S(n) = \{1, 2\}$, a weak classifier for classes 1 and 2 is trained as shown in

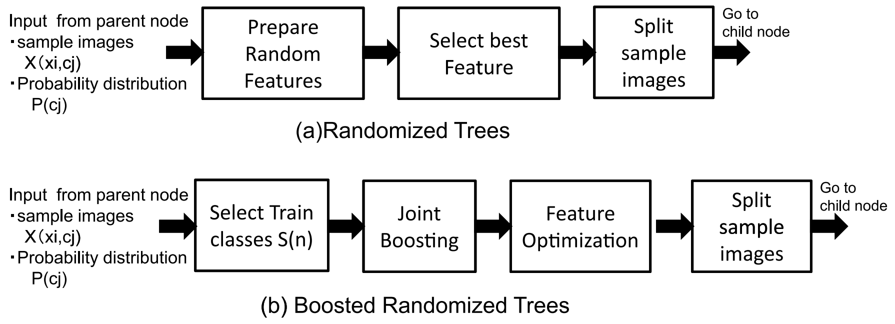


**Fig. 2** Training of node of decision tree.

Fig. 1 (b). For $S(n) = \{1\}$, a weak classifier that classifies class 1 is trained as seen in Fig. 1 (c). The weight of samples in S(n) are updated with Eq. (7).

$$w_i^c = w_i^c e^{-z_i^c h_m^{S(n)}(v,c)} \tag{7}$$

Note that $z_i^c \in \{+1, -1\}$ represents the labels of class $c$. The response of $h_m^{S(n)}(v,c)$ for classes not included in S(n) is 0; hence, the weight is updated for samples in these classes.

### 3.3 Boosted Randomized Trees

The flow for generating the nodes of Randomized Trees and Boosted Randomized Trees is outlined in **Fig. 2**. As shown in Fig. 2 (a), the generating the nodes for Randomized Trees consists of three steps: preparing random features and the threshold, selecting the best combination of features and thresholds, and evaluating sample images to generate child node subsets. Child nodes are generated in four steps with the proposed method, first by defining training classes, second by preselecting features through Joint Boosting, third by optimizing features, and fourth by generating child node subsets. Node generation for the proposed method is shown in Algorithm 2. Each of the steps is discussed in the following sections.

#### 3.3.1 Defining Class Set

Joint Boosting selects a weak classifier for a specific class subset, hence it is not possible to pre-define class sets. The decision trees in the proposed method have a hierarchical structure, with upper nodes handling multiple classes and lower nodes handling specific classes. Thus, weak classifiers for class sets with multiple classes are selected for the upper nodes, and class sets with a specific class are selected for the lower nodes. The best class sets for each node are selected based on the class likelihood as shown in step-2 of Algorithm 2. The class likelihoods for all classes are computed from the probability for each class as shown in Eq. (4). For a given node, we define the class set, $S(n)$, as a combination of classes $n$ in all class combinations with a total of class likelihoods $L$, as in Eq. (8), greater than threshold $\tau$.

$$S(n) \in \{n_1, n_2, ..., n_i : L(n_i) > \tau, i \in I\} \tag{8}$$

Note that $n_i$ is a class element, $I$ is the number of total class elements, and $\tau$ is the threshold. The combination with the least number of classes is selected. Upper nodes have more classes with the likelihood for each class being lower and lower nodes tend toward a specific class with the likelihood for the specific class being higher. This removes the need to consider classes with low likelihood for recognition, and class sets only consist of specific classes.

**Algorithm 2** Node Generation Process

```
Initialization:
1. Inizialize training sample weight $w_i^c$
For $i = 1..N$        //No. of samples
    For $c = 1..C$        //No. of classes
        initialize training sample weight$w_i^c$
Training:
2. Defining a set of classes
    $S(n) \in \{n_1, n_2, ..., n_i : L(n_i) > \tau, i \in I\}$
3. Preliminary feature selection
  3.1. $m = 1, 2, .., M$      //No. of weak classifier selected
    (a) combination of classes : S(n)
        (i)Compute error for all weak classifier candidates
    (b)Select the weak classifier candidate of S(n) with minimal error
    (c)Update the weight of samples $w_i^c$
  Repeat to obtain weak classifiers, $M$,
4. Feature Optimization
  Optimize the size and position of weak classifier,
  and select best one, $h_m^{S(n)}(v,c)$, with random threshold as the node
```

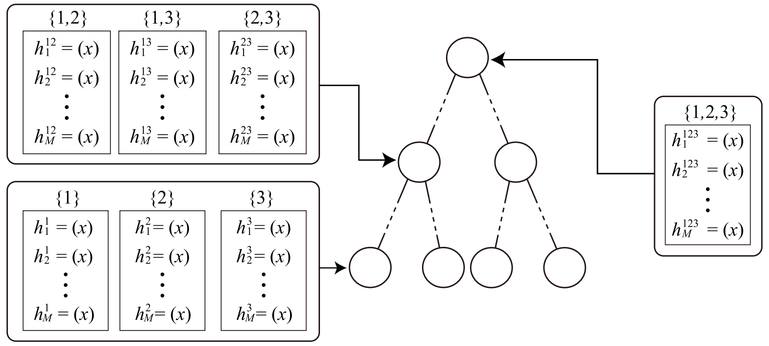


**Fig. 3** Feature selection by Joint Boosting.

### 3.3.2 Feature Pre-selection using Joint Boosting

The best features in Joint Boosting are trained from all combinations of given classes as described in Section 3.2. Because of this, features specific to a particular class might be selected for lower nodes rather than features that are common to multiple classes. In step-3 of Algorithm 2 with the proposed method, feature candidates are trained for limited class sets pre-defined based on class likelihoods as described in Section 3.3.1. The case of three classes in **Fig. 3** in the upper nodes as the training dataset includes many classes, because each class has a high class likelihood. Therefore, features related with many classes are selected, such as $S(n) = \{1, 2, 3\}$. Lower nodes, on the other hand, are trained for a specific class and features specific to a class are selected such as $S(n) = \{1\}$. Thus, by constraining the method of feature selection in Joint Boosting using class sets, hierarchical features can be efficiently selected.

### 3.3.3 Local Features

We extract features based on a histogram of gradients, which is effective for detecting human bodies. There is an outline of the features in **Fig. 4**. As seen in Eq. (9), the feature is the difference between the values, $g_{r_1,t_1}(i)$ and $g_{r_2,t_2}(j)$, which are bins in the gradient histograms of two localized regions, i.e., $r_1$ in the $t_1$ frame and $r_2$ in the $t_2$ frame.

$$F = g_{r1,t1}(i) - g_{r2,t2}(j) \quad (9)$$

We can capture both changes in gradient differences in a single frame and differences across frames.

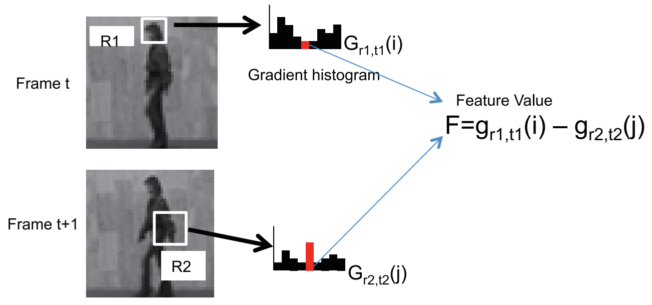


**Fig. 4** Features expanded to image sequences.

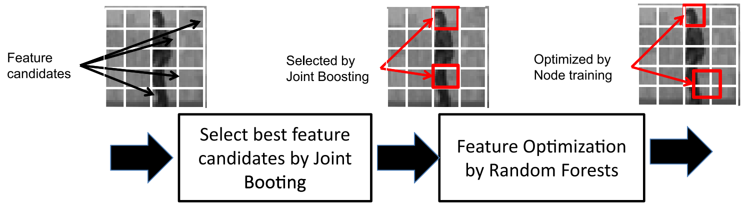

Feature candidates
Selected by Joint Boosting
Optimized by Node training
Select best feature candidates by Joint Booting
Feature Optimization by Random Forests

**Fig. 5** Optimization of features by Joint Boosting and Randomized Trees.

### 3.3.4 Optimization of Features

Let us focus on the differences between two local regions, and there are a very large number of combinations of regions. Features are selected in the two steps shown in **Fig. 5** by pre-selecting feature candidates trained by Joint Boosting, and optimizing features for a node. First, candidate features are trained in step-3 of Algorithm 2 with features generated by grid sampling. Then, the size and position of the features pre-selected with Joint Boosting are randomly adjusted. The threshold for the split function of nodes is also randomly set. The best combination of features with random adjustment and threshold are selected by using Eq. (3) in step-4 of Algorithm 2.

### 3.3.5 Human Detection by BRTs

When Boosted Randomized Trees are trained, the training samples have a separate class label for the background class apart from the classes for each of the actions. Each one of the decision trees output likelihoods for each action and also for the background. This indicates that a high likelihood for the background class is recognized as a background region, and a high likelihood for a particular action is detected as human position. Hence, the proposed method not only recognizes human actions but also detects human position. The BRTs search window is made to slide over the image to detect humans at any position in the image.

## 4. Experiments

### 4.1 Experiment Overview

Experiments to compare the performance of human detection and action recognition were carried out. The existing public database for action recognition contains several actions and scenarios. Even though the database includes many actions, it can be confined to a few based on the application. For example, "walking", "bending" and "picking" are important actions for analyzing the purchasing behavior of customers, where "jogging", "boxing" or "jumping" are not necessary. Hence, we evaluated performance in a specific scenario. In these experiments, 1) the performance of action recognition was compared with popular methods using the KTH database, 2) human detection and action recognition were compared by using our specific database, which was for purposes of analyzing customer behaviors, 3) and part of the Weizmann database was used to compare performance within the context of analyzing customer behaviors. The performance of human detection with the proposed method was compared with Joint Boosting, Randomized Trees, and AdaTree. Human detection was based on a sliding window approach to search within frames. False positive rates included false detection of backgrounds and misclassifications of actions.

Ten decision trees were trained for Randomized Trees and Boosted Randomized Trees until the training samples were exhausted or the tree depth reached 15. Randomized Trees prepared 100 candidate features randomly in the whole human region and 100 thresholds were also prepared randomly for all candidate features. Boosted Randomized Trees prepared 100 candidate features from 10 pre-selected features trained by Joint Boosting with random shift and scaling. Three frame snippets normalized to $48 \times 48$ pixels were used to train each of the classes.

**Table 1** Action recognition in KTH.

| Method | Training Method | Accuracy |
|---|---|---|
| Kim et al. | LOOCV | 95.33% |
| Wong et al. | LOOCV | 91.60% |
| Fathi et al. | Splits | 90.50% |
| Gilbert et al. | Splits | 89.92% |
| Nowozin et al. | Splits | 87.04% |
| Niebles et al. | LOOCV | 81.50% |
| Dollar et al. | LOOCV | 81.20% |
| Schuldt et al. | Splits | 71.72% |
| Ke et al. | Splits | 62.97% |
| Oshin | Splits | 89.10% |
| Our Method | Splits | 88.50% |
| Our Method (Action Recognition only) | Splits | 90.80% |

### 4.2 Experimental Results in KTH Database

Experiments to compare the performance of action recognition were undertaken using the KTH database, which contained the six actions of boxing, clapping, waving, jogging, running, and walking in four different scenarios. All actions were performed by 25 different people. There were two training methods of Leave-one-out Cross Validation (LOOCV) and Splits. We employed the Splits training method as Shuldt et al. had done[6], using a training set with eight people, a validation set with eight people, and the remaining nine people for testing. Two hundred snippets were prepared for each class. The action recognition results are summarized in **Table 1**. The methods based on LOOCV were better than Splits based methods, because only one test dataset was evaluated by the rest of similar training datasets. Compared with methods that employed the Split training method, ours performed comparably with state-of-the-art methods. In addition, it only recognized action for detected humans. When we tested all human regions in the test data, our method outperformed the others.

### 4.3 Experimental Results from Our Database

The proposed method, which was a combination of Randomized Trees and Joint Boosting, efficiently selected features and trained classifiers. First, the performance of human detection was compared with Joint Boosting and Randomized Trees based approaches for various human poses to demonstrate improvements in detection. Action recognition was compared with AdaTree, which is a hier-

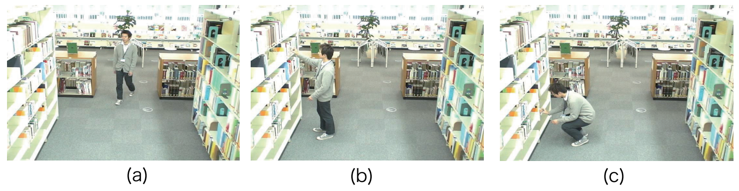


**Fig. 6** Example of our database: (a) walking, (b) taking object down from high shelf, and (c) picking object up from low shelf.

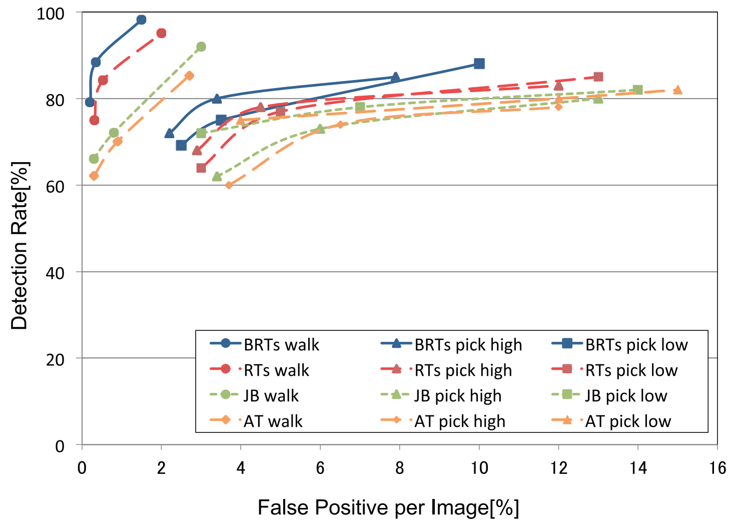


**Fig. 7** ROC curve of human detection results.

archical tree-based approach. Action recognition in customer behavior analysis was chosen for the evaluation. The action set consisted of walking, taking objects down from an upper shelf, and picking objects up from a lower shelf in the cluttered background shown in **Fig. 6**. Each action was a 4-sec video sequence for 10 different people. Three hundred snippets were selected for each class, and the background class was trained using 1,000 images selected at random from a 10,000-image background image database. The rates for human detection by all the methods are plotted in **Fig. 7**. Compared to Joint Boosting (JB), Randomized Trees (RTs) and AdaTree (AT), Boosted Randomized Trees (BRTs) achieved better detection rates for all poses.

The walking data set contained frontal and side view poses, and even though there were differences in appearance, there were few variations in poses, hence detection rates were high for all the methods. However, subjects picking up objects from the lower shelf had a variety of body postures, creating greater variations in poses. Similarly, subjects taking objects down from the upper shelf also had wider variations in arm inclinations. Joint Boosting selected features that were biased towards a particular class since combinations of classes could not be pre-defined; hence, detection rates for these poses were lower. Randomized Trees selected features at random and features that could best detect these pose variations might have been ignored. However, as class combinations for the nodes were pre-defined with the proposed method, effective features common between classes were preselected. Also, since it had a tree structure, there could have been multiple leaf nodes for a given class thus allowing variations within a class. The proposed approach was able to improve detection rates for walking, taking down, and picking up actions using Boosted Randomized Trees in this way.

We also conducted experiments comparing the performance of action recognition, and the results for our database are summarized in **Tables 2**, **3** and **4**. Clearly, Boosted Randomized Trees achieved better recognition rates than Randomized Trees and AdaTree. The other approaches recognized some actions incorrectly, but the proposed classified un-recognizable actions as "not-recognized" and not as incorrect classification. The proposed method was able to efficiently pre-select features that captured changes between frames, which may be related to the improved action recognition.

### 4.4 Action Recognition in Part of Weizmann Database

We selected actions from the Weizmann database for purposes of analyzing customer behaviors. Three actions of "walk", "bend", and "wave" were similar to taking objects down from a higher shelf and picking them up from a lower shelf as shown in **Fig. 8**. We broke down "bend" and "wave" into more specific actions, recognizing "bend" as either "bend down" or "bend up", and "wave" as either up or down hands. Two hundred forty snippets in each class and 2,000 snippets from background images were trained. The snippets of eight people were trained and

**Table 2** Action recognition rate for RTs.

| | walk | pick from high | pick from low | False Negative |
|---|---|---|---|---|
| walk | 94.5 | 2.7 | | 2.8 |
| pick from high | 5.0 | 86.7 | | 8.3 |
| pick from low | 1.3 | 1.4 | 84.3 | 13.0 |
| average | 88.5% | | | |

**Table 3** Action recognition rate for AT.

| | walk | pick from high | pick from low | False Negative |
|---|---|---|---|---|
| walk | 90.5 | 1.8 | 2.2 | 5.5 |
| pick from high | 7.1 | 73.9 | 4.2 | 14.8 |
| pick from low | 10.3 | 5.0 | 72.8 | 11.9 |
| average | 79.0% | | | |

**Table 4** Action recognition rate for BRTs.

| | walk | pick from high | pick from low | False Negative |
|---|---|---|---|---|
| walk | 98.5 | | | 1.5 |
| pick from high | | 92.4 | | 7.6 |
| pick from low | | | 94.1 | 5.9 |
| average | 95.0% | | | |

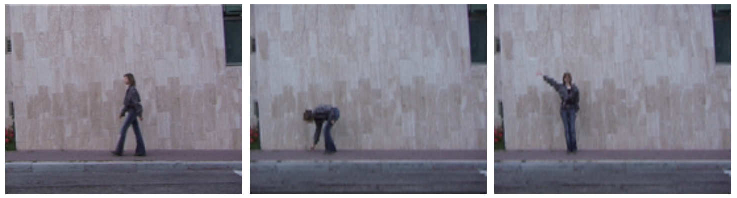


**Fig. 8** Example of experiment images; (a) walk, (b) bend, (c) wave.

**Table 5** Action recognition rate for RTs.

| | walk | bend-up | bend-down | Wave-up | Wave-down | False Negative |
|---|---|---|---|---|---|---|
| walk | 95.5 | | | 2.0 | | 2.5 |
| bend-up | | 91.1 | 1.0 | 3.2 | | 4.7 |
| Bend-down | | 2.1 | 89.5 | | 3.0 | 5.4 |
| wave-up | | 5.5 | | 80.1 | 1.3 | 13.1 |
| wave-down | | | 4.2 | 1.8 | 84.7 | 9.3 |
| average | 89.4% | | | | | |

**Table 6** Action recognition rate for AT.

| | walk | bend-up | bend-down | Wave-up | Wave-down | False Negative |
|---|---|---|---|---|---|---|
| walk | 90.8 | | | | | 9.2 |
| bend-up | | 86.4 | | | | 13.6 |
| Bend-down | | | 85.8 | | | 14.2 |
| wave-up | | | | 72.3 | | 27.7 |
| wave-down | | | | | 69.2 | 30.8 |
| average | 80.9% | | | | | |

**Table 7** Action recognition rate for BRTs.

| | walk | bend-up | bend-down | Wave-up | Wave-down | False Negative |
|---|---|---|---|---|---|---|
| walk | 99.5 | | | | | 0.5 |
| bend-up | | 95.1 | | | | 4.9 |
| Bend-down | | | 95.6 | | | 4.4 |
| wave-up | | | | 85.1 | | 14.9 |
| wave-down | | | | | 90.3 | 9.7 |
| average | 93.9% | | | | | |

the remaining one person was used for evaluating cross validation. The action recognition results are listed in **Tables 5**, **6** and **7**. The results revealed that Boosted Randomized Trees performed better than the other methods.

## 5. Discussion

### 5.1 Relation to Number of Trees

We investigated what effect the number of decision trees would have on performance, and the ROC curve for the number of decision trees is plotted in **Fig. 9**. The rate of human detection in our database was calculated as the average of three actions with a 2% false positive rate, where detection error included errors

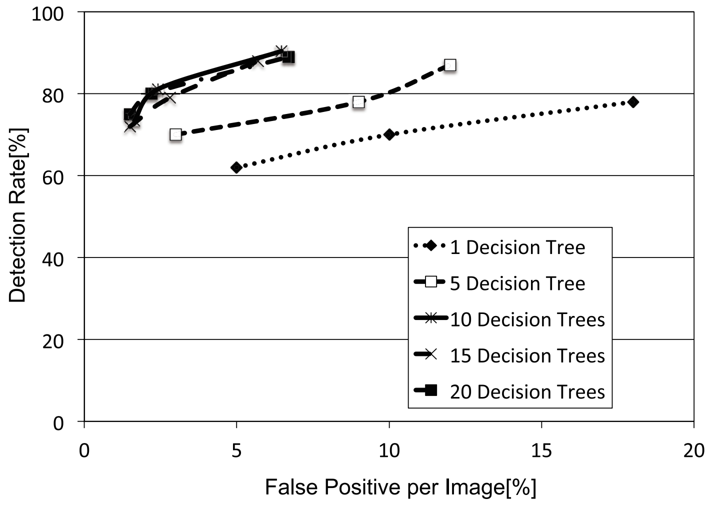


**Fig. 9** Performance of human detection with numbers of trees.

in action recognition. This indicated that the detection rate improved with increased numbers of decision trees. As the detection rate saturated at around a tree count of 10, this indicated that the optimal tree count was around 10 for the present scenario.

### 5.2 Relation to Number of Features

Features in the proposed method were randomly pre-selected with Joint Boosting and candidate features were generated from them. The thresholds for all generated candidate features were also randomly prepared. Performance with our database was based on the number of candidates features and thresholds. Candidate features were generated from 10 pre-selected features from Joint Boosting. The results of human detection for 50, 100 and 150 candidate features are plotted in **Fig. 10**. The thresholds were changed to 50, 100, and 150 and the number of decision trees was set to 10. As a result, the number of candidate features and thresholds were directly proportional to performance, but there were no marked improvements in performance between 100 and 150 features.

The average rate for action recognition for the number of candidate features is listed in **Table 8**. There were 10 decision trees and 100 thresholds. As a

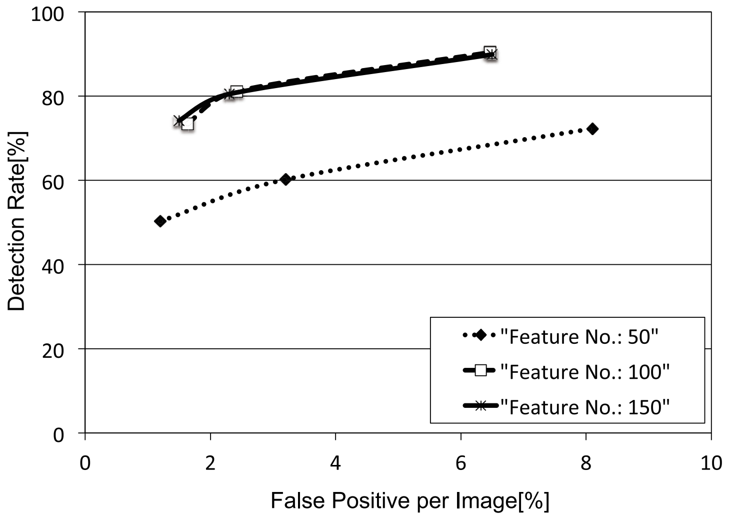


**Fig. 10** Performance of human detection with numbers of features.

**Table 8** Performance of action recognition with numbers of features.

| Candidate Feature No. | Recognition Rate |
|---|---|
| 50 | 80.5% |
| 100 | 95.0% |
| 150 | 95.3% |
| RTs (Feature No. = 100) | 88.5% |

result, performance with fewer features was worse than that with Randomized Trees, where there was little randomness and poor generalization because few candidate features were randomly generated from effective pre-selected features. Conversely, performance with many features that were randomly generated was better than with Randomized Trees. This means many candidate features generated from effective pre-selected features yielded better generalization.

### 5.3 Tree Visualization

The tree was visualized in the part of the Weizmann database shown in **Fig. 11** to analyze how effective the proposed method was. Joint Boosting at the root node selected a feature shared by three of the poses, capturing the foot region (a). The training samples were divided into two subsets by the feature selected for

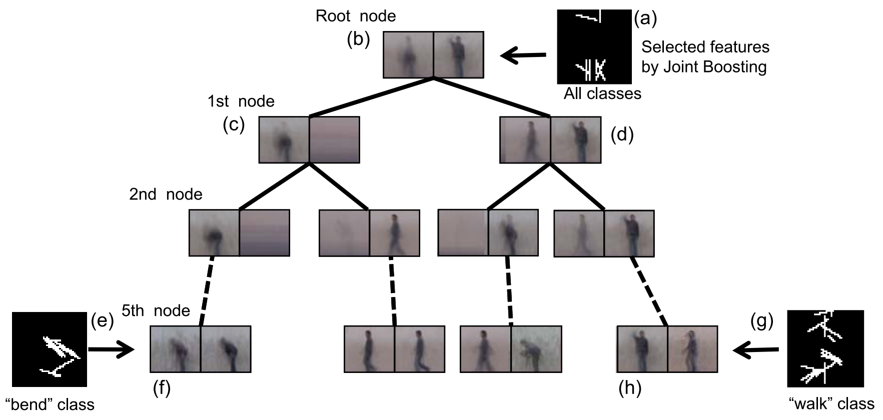


**Fig. 11** Visualization of Leaf node of Boosted Randomized Trees.

the root node (b). One partition included bending and background classes, while another included mainly walk and wave classes. Samples at the root node were split into walk and wave, which appeared similar. Bend and background classes have been selected at the left node in the first layer (c), while on the right side (d), walk and wave classes have been selected. Almost all the classes have been separated from the background in the second layer. This means that human bodies in various poses have been separated from the background in the first few layers, which is equivalent to human detection. As various poses have also been differentiated, we can also say that overall poses have simultaneously been estimated. Classes that recognize actions have been separated into finer detail by conducting further training for several layers. The node at (f) accounts for most of the bending samples, and features common to the bending class, which represent the bent areas of the subject's back (e), were pre-learned through Joint Boosting. The node at (h) has classified most of walk class, and walk and wave have been differentiated by pre-learned features (g) representing the arm and leg positions that are common to walk class. Many features that differentiated various action classes were selected in the lower nodes in this way. Our pre-selection of features allowed them to be selected hierarchically, with features common to many classes selected for the upper nodes and features specific to individual classes selected for the lower nodes, thus achieving a single hierarchical framework from human detection to action recognition.

## 6. Conclusion

We proposed Boosted Randomized Trees for action recognition, which defined recognition classes based on the likelihood of each class when the nodes of a decision tree were generated. By pre-selecting the effective features of these classes by Joint Boosting, shared features were selected for upper nodes, and specific class features were selected for lower nodes. As a result, the nodes were trained such that the upper nodes detected humans from the background, while the lower nodes recognized specific actions from human detection to action recognition within a single framework. The experimental results demonstrated action recognition matched some of the earlier methods that have been proposed. We also achieved better performance when we compared our approach with similar methods such as AdaTree and Randomized Trees. We also investigated what effect the number of trees and candidate features had on performance. We found that many candidate features generated from effective pre-selected features yielded better generalization.

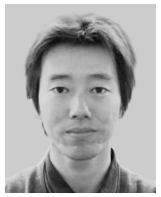
## References

1) Stauffer, C. and Grimson, W.E.L.: Adaptive background mixture models for real-time tracking, *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp.1491–1498 (1999).
2) Breiman, L.: Random forests, *Machine Learning*, Vol.45, No.1, pp.5–32 (2001).
3) Efros, A., Berg, A., Mori, G. and Malik, J.: Recognizing action at a distance, *IEEE International Conference on Computer Vision*, Vol.2, pp.726–733 (2003).
4) Cucchiara, R., Grana, C., Piccardi, M. and Prati, A.: Detecting moving objects, ghosts, and shadows in video streams, *IEEE Trans. Pattern Recognition and Machine Intelligence*, Vol.25, No.10, pp.1337–1342 (2003).
5) Torralba, A., Murphy, K.P. and Freeman, W.T.: Sharing features: Efficient boosting procedures for multiclass object detection, *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.2, pp.762–769 (2004).
6) Schuldt, C., Laptev, I. and Caputo, B.: Recognizing human actions: A local SVM approach, *IEEE International Conference on Pattern Recognition*, Vol.3, pp.32–36 (2004).
7) Grossmann, E.: AdaTree: Boosting a Weak Classifier into a Decision Tree, *Confer-*

*ence on Computer Vision and Pattern Recognition Workshop* (*CVPRW'04*), Vol.6, p.105 (2004).
8) Lepetit, V., Lagger, P. and Fua, P.: Randomized Trees for real-time keypoint recognition, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.775–781 (2005).
9) Dalal, N. and Triggs, B.: Histograms of oriented gradients for human detection, *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp.886–893 (2005).
10) Blank, M., Gorelick, L., Shechtman, E., Irani, M. and Basri, R.: Actions as space-time shapes, *IEEE International Conference on Computer Vision*, Vol.2, pp.1395–1402 (2005).
11) Tu, Z.: Probabilistic boosting-tree: Learning discriminative models for classification, recognition and clustering, *IEEE International Conference on Computer Vision*, Vol.2, pp.1589–1596 (2005).
12) Geurts, P., Ernst, D. and Wehenkel, L.: Extremely Randomized Trees, *Machine Learning*, No.36, Vol.1, pp.3–42 (2006).
13) Zhu, Q., Avidan, S., Yeh, M.C. and Cheng, K.T.: Fast human detection using a cascade of histograms of oriented gradients, *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.2, pp.1491–1498 (2006).
14) Wu, B. and Nevatia, R.: Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet Based Part Detectors, *International Journal of Computer Vision*, Vol.75, pp.247–266 (2007).
15) Niebles, J.C. and Fei-Fei, L.: A hierarchical model of shape and appearance for human action classification, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2007).
16) Kim, T., Wong, S. and Cipolla, S.: Tensor canonical correlation analysis for action classification, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2007).
17) Nowozin, S., Bakir, G. and Tsuda, K.: Discriminative subsequence mining for action classification, *IEEE International Conference on Computer Vision*, pp.1–8 (2007).
18) Shotton, J., Johnson, M. and Cipolla, R.: Semantic Texton Forests for Image Categorization and Segmentation, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2008).
19) Schindler, K. and Van Gool, L.: Action snippets: How many frames does human action recognition require?, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2008).
20) Rogez, G., Rihan, J., Ramalingam, S., Orrite, C. and Torr, P.H.S.: Randomized Trees for Human Pose Detection, *IEEE Conference on Computer Vision and Pattern Recognition,* pp.1–8 (2008).
21) Fathi, A. and Mori, G.: Action recognition by learning midlevel motion features, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2008).
22) Gilbert, A., Illingworth, J. and Bowden, R.: Scale invariant action recognition using compound features mined from dense spatio-temporal corners, *European Conference on Computer Vision*, pp.222–233 (2008).
23) Liu, J. and Shah, M.: Learning human actions via information maximization, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2008).
24) Gall, J. and Lempitsky, V.: Class-specific hough forests for object detection, *IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–8 (2009).
25) Okada, R.: Discriminative and Generalized Hough Transform for Object Detection, *IEEE International Conference on Computer Vision*, pp.1–8 (2009).
26) Oshin, O., Gilbert, A., Illingworth, I. and Bowden, R.: Action Recognition using Randomized Ferns, *IEEE Workshop on Video-Oriented Object and Event Classification*, pp.530–537 (2009).

(Received November 9, 2010)
(Accepted August 29, 2011)
(Released December 28, 2011)

(Communicated by *Hiroaki Kawashima*)



**Takayoshi Yamashita** received his M.S. degree in engineering from the Nara Institute of Science and Technology (NAIST), Japan, in 2002 and the Ph.D. degree from the Graduate School of engineering, Chubu University, Japan in 2011. He has been working in OMRON Corporation since 2002. His current research interests include object detection, object tracking, human activity understanding, and pattern recognition. He is a member of IEEE and IEICE.

**Yuji Yamauchi** received his B.S. and M.S. degrees in computer science from Chubu University, Japan, in 2009. He is currently pursuing his Ph.D. and has been in the Graduate School of engineering, Chubu University since 2009. He has been a JSPS research fellow since 2010. His research interests include computer vision and pattern recognition. He is a member of IEICE.

**Hironobu Fujiyoshi** received his Ph.D. in Electrical Engineering from Chubu University, Japan, in 1977. From 1997 to 2000 he was a post-doctoral fellow at the Robotics Institute of Carnegie Mellon University, Pittsburgh, PA, USA, working on the DARPA Video Surveillance and Monitoring (VSAM) effort and the humanoid vision project for the HONDA Humanoid Robot. He is now a professor of the Department of Computer Science, Chubu University, Japan. From 2005 to 2006, he was a visiting researcher at Robotics Institute, Carnegie Mellon University. His research interests include computer vision, video understanding and pattern recognition. He is a member of IEEE, IEICE, IPSJ, and IEE.