TREND-SENSITIVE HOUGH FORESTS FOR ACTION DETECTION

Kensho Hara, Takatsugu Hirayama and Kenji Mase

Graduate School of Information Science
Nagoya University, Japan

ABSTRACT

A Hough transform-based method for action detection can achieve robustness to occlusions because the method casts votes for action classes and spatio-temporal action positions based on the visible local features of partially occluded actions. However, each local feature is prone to a false vote. This paper focuses on the trend of past votes to curb the influence of false votes by extending conventional Hough forests to sensing that trend. Our proposed method, called trend-sensitive Hough forests, learns a voting trend model that can be used to discriminate between correct and false votes and calculate the confidence of them. We experimentally confirmed that it outperformed action detection accuracy of conventional Hough forests.

***Index Terms*—** Action detection, Hough transform, Hough forests, Random forests, spatio-temporal features

1. INTRODUCTION

Action recognition is a crucial theme in computer vision because it is helpful in such applications as video surveillance and human computer interaction. There are two approaches for human action recognition. The first recognizes actions after human detection and tracking to localize human positions. A problem is that false human detection and tracking decrease the action recognition accuracy. Actions consist of 3D pose transitions and non-rigid motions, and pose and motion variances complicate both the detection and tracking of 2D images. The second one simultaneously recognizes and localizes actions without prior detection and tracking. This paper focuses on the latter approach which we call action detection.

Some action detection methods are based on Hough transform [1], which casts votes for the information of objects based on local elements. Yu et al. proposed a Hough transform-based method for action detection [2] that casted votes for the spatio-temporal reference positions of actions based on spatio-temporal local features.

One advantage of Hough transform-based methods is robustness to occlusions. They can detect action using the visible local features of occluded actions. However, each local feature is prone to a false vote because of similar local features that belong to different classes. Some research in the field of object detection has resolved this problem. Wohlhart et al. formulated the contribution of voting elements to an object hypothesis as a descriptor for the hypothesis [3]. They used the descriptor to discriminate between correct and incorrect detections. We handle action recognition as a multi-class problem and use such contributions of each class to improve the conventional voting.

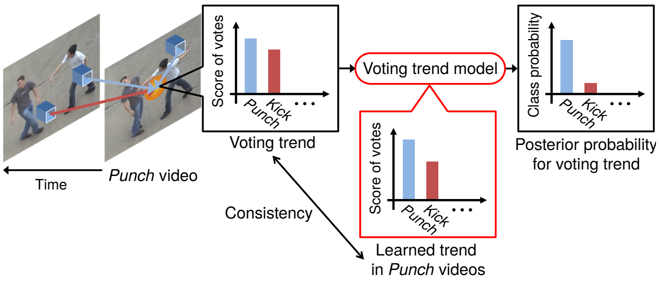

Fig. 1. Idea of voting trends. If votes not only for *Punch* but also for *Kick* are cast in a *Punch* video, posterior probability of *Kick* for the voting trend can be low according to the similarity between the voting trend and the learned trend in *Punch* videos.

Our proposed method is based on the Hough forests [4] proposed by Yao et al. Hough forests learn relations between a representative position of each action on (X, Y, T) space and spatio-temporal local features and vote for action classes and positions based on the local features. Yao et al. recognized actions after human detection and tracking using Hough forests. We confirmed that Hough forests can be used as an action detection method [5]. This paper uses Hough forests as an action detection method.

We assume that the similarity of the local features between each class leads to the false votes of Hough forests. Votes that include both correct and false votes have some trends for each class. Our proposed method learns a voting trend model that can be used to discriminate between correct and false votes and calculate their confidence. It curbs the influence of false votes based on their confidence to improve the detection accuracy. We call our proposed method *Trend-Sensitive Hough Forests*. Figure 1 shows an idea of voting trends. Posterior probability for a voting trend works as confidence of votes.

2. HOUGH FORESTS FOR ACTION DETECTION

This section explains Hough forests for action detection. First, in Section 2.1, we explain conventional Hough forests and then novel trend-sensitive Hough forests in Section 2.2.

In this paper, both methods detect space-time interest points [6]. As in previous work [7], we detect these points at multiple spatial and temporal scales. To characterize the appearance and motion of a local feature, we use the histogram of oriented gradients (HOG) and the histogram of optic flow (HOF) [7], which are applied to the space-time volume centered around the detected point.

We define a spatio-temporal center in an action sequence as an action position in common with conventional Hough forests.

2.1. Hough Forests

This section explains the details of Hough forests for action detection. They consist of 2.1.1) *training* and 2.1.2) *detection* phases.

2.1.1. Training

Hough forests, which use random forests [8] that are ensemble classifiers consisting of many decision trees, learn action classes and displacement vectors from the centers of the local feature patches to the action positions. The local feature patches are extracted from the videos of actions as training data. Each tree is constructed from a set of feature patches $\{\mathbf{P} = (\mathbf{I}, c, \mathbf{d})\}$, where $\mathbf{I}$ is a feature vector, c is an action label, and $\mathbf{d} \in \mathbb{R}^3$ is a displacement vector in the (X, Y, T) space. $\mathbf{I}$ can be multi-channeled to accommodate multiple features, i.e., $\mathbf{I} = (\mathbf{I}^1, \mathbf{I}^2, \cdots, \mathbf{I}^F)$, where F is the number of feature channels.

Each non-leaf node of a tree is assigned to the following binary test:

$$b_{f,q,r,\tau}(\mathbf{I}) = \begin{cases} 0 & \text{if } \mathbf{I}^f(q) < \mathbf{I}^f(r) + \tau \\ 1 & \text{otherwise,} \end{cases} \tag{1}$$

where f is a feature channel, q and r are the dimensions of $\mathbf{I}^f$, and τ is offset.

In the training phase, Hough forests are iterated to classify the feature patches into two nodes. This iteration is continued until each one reaches the termination criteria defined by the maximum depth and the minimum number of patches in a node. Each leaf node L stores $p(c|L)$, which is the proportion of feature patches per class label c that reaches the leaf, and $\mathbf{D}_c^L$, which is a set of the displacement vectors of the patches.

Each node generates a set of binary tests $\{b^j\}$ with random values of f, q, r, and τ. Hough forests use two measures to select the most suitable binary test. The first measure is class uncertainty:

$$U_1(\mathbf{A}) = -|\mathbf{A}| \sum_c p(c|\mathbf{A}) \ln p(c|\mathbf{A}), \tag{2}$$

where $\mathbf{A}$ is a set of feature patches and $|\cdot|$ denotes the number of elements in the set. The second measure is the uncertainty of the displacement vectors:

$$U_2(\mathbf{A}) = \sum_c \left(\sum_{\mathbf{d} \in \mathbf{D}_\mathbf{c}^\mathbf{A}} \left\| \mathbf{d} - \overline{\mathbf{d}_c^\mathbf{A}} \right\|^2 \right), \tag{3}$$

where $\overline{\mathbf{d}_c^\mathbf{A}}$ is the mean displacement vectors in $\mathbf{A}$ for class c.

Each node randomly chooses one of these measures and selects binary test b^j that minimizes the uncertainty. This minimization is represented by the following equation:

$$\arg\min_j \left(U_* \left(\{\mathbf{P}|b^j = 0\} \right) + U_* \left(\{\mathbf{P}|b^j = 1\} \right) \right), \tag{4}$$

where U_* is the chosen uncertainty measure for the node and j is the index of the binary tests.

2.1.2. Detection

Hough forests detect actions using the votes cast for the action labels and positions. Consider feature patch $\mathbf{P}_\mathbf{y}$ extracted from position $\mathbf{y} \in \mathbb{R}^3$ and assume that $\mathbf{P}_\mathbf{y}$ ends up in leaf node $L_\mathbf{y}^k$ of tree k. $L_\mathbf{y}^k$ stores $p\left(c_\mathbf{y} = c|L_\mathbf{y}^k\right)$, where $c_\mathbf{y}$ is an unknown class label of $\mathbf{P}_\mathbf{y}$, and $\mathbf{D}_c^{L_\mathbf{y}^k}$. The probability of an action of class c at position $\mathbf{x} \in \mathbb{R}^3$ based on $L_\mathbf{y}^k$ can be defined as

$$\begin{aligned} & p(\mathbf{h}(c, \mathbf{x})|L_\mathbf{y}^k) \\ &\quad = p\big(\mathbf{d}_\mathbf{y} = \mathbf{y} - \mathbf{x}|c_\mathbf{y} = c, L_\mathbf{y}^k\big) p\left(c_\mathbf{y} = c|L_\mathbf{y}^k\right) \\ &\quad = \left(\frac{1}{\left|\mathbf{D}_c^{L_\mathbf{y}^k}\right|} \sum_{\mathbf{d} \in \mathbf{D}_c^{L_\mathbf{y}^k}} G\left((\mathbf{y} - \mathbf{x}) - \mathbf{d}\right) \right) p\big(c_\mathbf{y} = c|L_\mathbf{y}^k\big), \end{aligned} \tag{5}$$

where $\mathbf{h}(c, \mathbf{x})$ is the hypothesis that an action of class c occurs at position $\mathbf{x}$ and $\mathbf{d}_\mathbf{y}$ is the displacement vector of $\mathbf{P}_\mathbf{y}$. Here G is a 3D Gaussian Parzen window function. Using Eq. (5), the voting score of an action of class c at position $\mathbf{x}$ on video $\mathbf{V}$ can be defined as

$$s\left(\mathbf{h}(c, \mathbf{x})|\mathbf{V}\right) = \sum_{\mathbf{y} \in \mathbf{Y}} \left(\frac{1}{K} \sum_{k=1}^{K} p\left(\mathbf{h}(c, \mathbf{x})|L_\mathbf{y}^k\right) \right), \tag{6}$$

where $\mathbf{Y}$ is the set of the positions of the feature patches extracted from video $\mathbf{V}$ and K is the number of trees. The local maximum of Eq. (6) specifies class c and position $\mathbf{x}$ of an action.

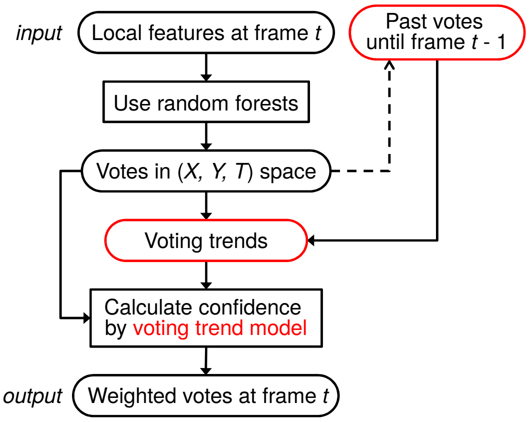
input
Local features at frame *t*
Past votes until frame *t* - 1
Use random forests
Votes in (*X*, *Y*, *T*) space
Voting trends
Calculate confidence by voting trend model
output
Weighted votes at frame *t*

Fig. 2. Flow of proposed voting process at frame *t*.

2.2. Trend-Sensitive Hough Forests

Trend-sensitive Hough forests add robustness to the false votes in conventional Hough forests using the trend of past votes. We assume that the similarity of the local features between each class leads to the false votes of Hough forests. Votes that include both correct and false votes have trends for each class. The proposed method learns the voting trend model that can discriminate between correct and false votes and calculate their confidence. Figure 2 shows the flow of our proposed voting process. The votes are cast at each frame. We consider the temporal variation of the trend. The proposed method gives weights to votes based on the consistency between the trends of past votes and the voting trend model at each frame. In Section 2.2.1, we explain the voting trends in detail, and in Section 2.2.2, we explain the weighting based on the voting trends.

2.2.1. Voting Trends

This paper defines the normalized voting scores of all classes at a spatio-temporal position $\mathbf{x} \in \mathbb{R}^3$ as the voting trend. The trend at each frame t is based on votes until frame $t-1$. The score of the past votes of video $\mathbf{V}$ can be defined as

$$s_i^{t-1}(\mathbf{x}) = \frac{s\left(\mathbf{h}(c_i,\mathbf{x})|\mathbf{V}_{1:t-1}\right)}{\sum_{j=1}^{N} s\left(\mathbf{h}(c_j,\mathbf{x})|\mathbf{V}_{1:t-1}\right)}, \qquad (7)$$

where $\mathbf{V}_{1:t-1}$ is a sub-sequence of $\mathbf{V}$ from the start frame to $t-1$, s is the left side of Eq. (6), and N is the number of action classes. The trend of past votes at position $\mathbf{x}$ until frame $t-1$ is $\mathbf{S}(\mathbf{x}, t-1) = (s_1^{t-1}(\mathbf{x}), \ldots, s_N^{t-1}(\mathbf{x}))$.

2.2.2. Weighting Based on Voting Trends

Figure 3 shows the proposed weighting framework based on the voting trends, which are different at each position. Our proposed method gives weight to each vote based on the voting trend at each position.

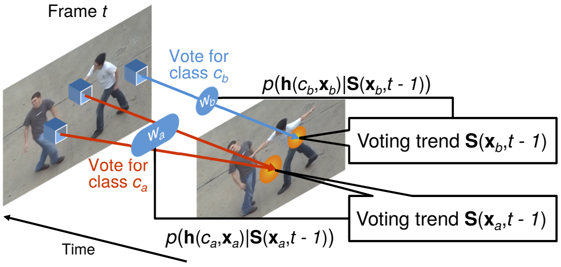
Frame *t*
Vote for class c_b
$p(\mathbf{h}(c_b,\mathbf{x}_b)|\mathbf{S}(\mathbf{x}_b,t - 1))$
w_b
w_a
Voting trend $\mathbf{S}(\mathbf{x}_b,t - 1)$
Vote for class c_a
Voting trend $\mathbf{S}(\mathbf{x}_a,t - 1)$
Time
$p(\mathbf{h}(c_a,\mathbf{x}_a)|\mathbf{S}(\mathbf{x}_a,t - 1))$

Fig. 3. Our proposed weighting framework. w denotes the weighting based on the voting trend.

The weights denote the confidence of the votes. To give weights to votes, we train a multi-class classifier that inputs voting trends and outputs the posterior probability of a class. We use random forests as the classifier. To generate the training data, we use videos annotated with action classes and action positions. Conventional Hough forests cast votes at each frame of the videos to train the classifier. This voting process generates voting trend $\mathbf{S}(\mathbf{x}_\mathrm{g}, t-1)$, where $\mathbf{x}_\mathrm{g} \in \mathbb{R}^3$ is the action position of a ground truth. A feature vector for the classifier is $\mathbf{f} = (t - t_\mathrm{g}, \mathbf{S}(\mathbf{x}_\mathrm{g}, t-1))$, where t_g is a temporal position that corresponds to $\mathbf{x}_\mathrm{g}$. Each supervisory signal of the feature vectors is the action label. The learned classifier works as a model of the voting trend and outputs the posterior probability of a class for the voting trend as the weight.

As in Section 2.1.2, we consider feature patch $\mathbf{P}_\mathbf{y}$ extracted from position $\mathbf{y} \in \mathbb{R}^3$ and its temporal position $t_\mathbf{y}$. The probability of class c at position $\mathbf{x} \in \mathbb{R}^3$ based on $\mathbf{S}(\mathbf{x}, t_\mathbf{y} - 1)$ and $L_\mathbf{y}^k$ can be defined as

$$\begin{aligned} & p\left(\mathbf{h}(c,\mathbf{x})|\mathbf{S}(\mathbf{x}, t_\mathbf{y}-1), L_\mathbf{y}^k\right) \\ &\quad = p\left(\mathbf{h}(c,\mathbf{x})|\mathbf{S}(\mathbf{x}, t_\mathbf{y}-1)\right) p\left(\mathbf{h}(c,\mathbf{x})|L_\mathbf{y}^k\right), \end{aligned} \qquad (8)$$

where $p(\mathbf{h}(c,\mathbf{x})|\mathbf{S}(\mathbf{x}, t_\mathbf{y}-1))$ is the posterior probability for voting trend $\mathbf{S}(\mathbf{x}, t_\mathbf{y}-1)$, which is the output by the classifier and works as the weight, and $p(\mathbf{h}(c,\mathbf{x})|L_\mathbf{y}^k)$ is the posterior probability for feature patch $\mathbf{P}_\mathbf{y}$, which is calculated by Eq. (5). The weighted voting score of class c at position $\mathbf{x}$ on video $\mathbf{V}_{1:t}$ can be defined as

$$\begin{aligned} & s'\left(\mathbf{h}(c,\mathbf{x})|\mathbf{V}_{1:t}\right) \\ &\quad = \sum_{t'=1}^{t} \sum_{\mathbf{y}\in\mathbf{Y}_{t'}} \left(\frac{1}{K} \sum_{k=1}^{K} p\left(\mathbf{h}(c,\mathbf{x})|\mathbf{S}(\mathbf{x}, t_\mathbf{y}-1), L_\mathbf{y}^k\right) \right), \end{aligned} \qquad (9)$$

where $\mathbf{Y}_{t'}$ is the set of the positions of the patches extracted from video $\mathbf{V}$ at frame t' and $t_\mathbf{y} = t'$. The local maximum of Eq. (9) specifies class c and position $\mathbf{x}$ of an action.

3. EXPERIMENTS

We evaluated trend-sensitive Hough forests using the UT-Interaction dataset [9] based on precision, recall, and f-scores.

Table 1. Precision, recall, and f-scores for trend-sensitive Hough forests (TSHF) and Hough forests (HF) on the UT-Interaction dataset. These results were calculated using the best threshold γ for the f-scores.

	Precision (TSHF, HF)			Recall (TSHF, HF)			F-scores (TSHF, HF)		
	Set 1	Set 2	Avg. per set	Set 1	Set 2	Avg. per set	Set 1	Set 2	Avg. per set
Shake hands	(**0.80**, 0.71)	(**1.00**, 0.92)	(**0.90**, 0.82)	(**0.67**, 0.42)	(0.75, 0.75)	(**0.71**, 0.58)	(**0.73**, 0.53)	(**0.86**, 0.83)	(**0.79**, 0.68)
Hug	(**1.00**, 0.91)	(0.82, **0.90**)	(**0.91**, 0.90)	(0.83, 0.83)	(0.81, 0.81)	(0.83, 0.83)	(**0.91**, 0.87)	(0.82, **0.86**)	(**0.87**, 0.86)
Kick	(**0.88**, 0.54)	(**0.57**, 0.45)	(**0.72**, 0.50)	(0.58, 0.58)	(0.33, **0.42**)	(0.46, **0.50**)	(**0.70**, 0.56)	(0.42, **0.44**)	(**0.56**, 0.50)
Point	(0.36, 0.36)	(0.89, 0.89)	(0.62, 0.62)	(**0.6**, 0.33)	(0.80, 0.80)	(**0.70**, 0.57)	(**0.45**, 0.34)	(0.84, 0.84)	(**0.65**, 0.59)
Punch	(0.50, **0.75**)	(0.29, **0.40**)	(0.40, **0.58**)	(**0.33**, 0.25)	(**0.45**, 0.36)	(**0.39**, 0.31)	(**0.40**, 0.38)	(0.36, **0.38**)	(0.38, 0.38)
Push	(**0.90**, 0.75)	(0.88, 0.88)	(**0.89**, 0.81)	(0.69, 0.69)	(0.88, 0.88)	(0.78, 0.78)	(**0.78**, 0.72)	(0.88, 0.88)	(**0.83**, 0.80)
Avg. per class	(**0.74**, 0.67)	(0.74, 0.74)	(**0.74**, 0.71)	(**0.62**, 0.52)	(0.67, 0.67)	(**0.65**, 0.59)	(**0.66**, 0.57)	(0.70, 0.70)	(**0.68**, 0.64)

We compared our proposed method with conventional Hough forests. Both methods detected actions by finding the local maxima of Eqs. (6) and (9) using mean-shift [10].

3.1. UT-Interaction Dataset

The UT-Interaction dataset contains videos of the continuous executions of six action classes: shaking hands, pointing, hugging, pushing, kicking, and punching. The dataset provides ground truth labels that include time intervals and bounding boxes. The total number of action executions is 162. The dataset provides two sets, each of which consists of ten video sequences. *Set 1* was recorded with a static background, and *set 2* was recorded with a slightly moving background and camera jitter. We defined the spatio-temporal center of the time intervals and the bounding boxes as the ground truth label of the action position. The resolution and the frame rate of the videos are 720 × 480 pixels and 30 fps, respectively.

3.2. Evaluation Method

Both Hough forests and our proposed method detected action positions. We separately calculated the spatial and temporal Euclidean distances between the detected action and ground truth positions. If both the spatial and temporal distances were lower than the thresholds, we defined the detection as a success. The spatial and temporal thresholds were 50 pixels and 30 frames, respectively. We calculated the precision, recall, and f-scores for each class by thresholding the local maxima of Eqs. (6) and (9) with threshold γ.

We employed leave-one-sequence-out cross-validation that used the data of one sequence as the test data and the remainder as the training data. The dataset contains motions that are not labeled. We used the motions as the *others* class in the training. In the detection, we did not cast votes for any class if the feature patches were classified as the *others* class.

3.3. Results

Table 1 shows the precision, recall, and f-scores using our proposed method and Hough forests. Compared with Hough forests, our proposed method achieved higher f-scores averaged over all the sets for all the action classes. Our proposed weighting method improved the detection accuracy. The proposed method also achieved both higher precision and recall averaged over all sets and classes than the Hough forests. When the proposed method worked correctly, the weights of the false votes were low, and the scores of the ground truth positions were high relatively. The weights enabled us to reduce not only the false positives but also the false negatives.

Improvements of the f-score of *shake hands* class by our proposed method is the highest of all classes. Compared with other actions, actors of the UT-Interaction dataset shook hands by similar motion. The low within-class variation of *shake hands* class led to the highest improvements.

The f-scores of the proposed method only increased in *set 1*. Some bounding boxes of the UT-Interaction dataset contained noise motions that did not belong to the actors. The noise motions of *set 2* were larger than those of *set 1* and might prevent the proposed method from extracting any trends in the training. These results suggest that our proposed method is not robust to training data noise.

4. CONCLUSION

We extended conventional Hough forests to curb the influence of false votes. We assume that the similarity of the local features between each class leads to the false votes of Hough forests. Votes that include both correct and false votes have trends for each class. Our proposed method learned a voting trend model that can be used to discriminate between correct and false votes and calculate their confidence. Our proposed method curbed the influence of false votes based on this confidence to improve the detection accuracy. We experimentally confirmed the effectiveness of our method. Future work will evaluate robustness to occlusions.

Acknowledgment

This research was supported by a grant from the NICT.

5. REFERENCES

[1] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[2] G. Yu, J. Yuan, and Z. Liu, "Propagative Hough voting for human activity recognition," in *Proc. European Conference on Computer Vision*, 2012, pp. 693–706.

[3] P. Wohlhart, S. Schulter, M. Köstinger, P. Roth, and H. Bischof, "Discriminative Hough forests for object detection," in *Proc. British Machine Vision Conference*, 2012, pp. 40.1–40.11.

[4] A. Yao, J. Gall, and L. V. Gool, "A Hough transform-based voting framework for action recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2061–2068.

[5] K. Hara, T. Hirayama, and K. Mase, "Simultaneous action recognition and localization based on multi-view Hough voting," in *Proc. Second IAPR Asian Conference on Pattern Recognition*, 2013, pp. 1–5.

[6] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[7] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[8] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[9] M. S. Ryoo and J. K. Aggarwal, "UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA)," http://cvrc.ece.utexas.edu/SDHA2010/Human_Interaction.html, 2010.

[10] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

978-1-4799-5751-4/14/$31.00 ©2014 IEEE