REAL-TIME, LONG-TERM HAND TRACKING WITH UNSUPERVISED INITIALIZATION

Vincent Spruyt[1,2], *Alessandro Ledda*[1] *and Wilfried Philips*[2]

[1]Dept. of Applied Engineering: Electronics-ICT, Artesis University College Antwerp
Paardenmarkt 92, 2000 Antwerpen, Belgium
v.spruyt@ieee.org

[2]Ghent University-TELIN-IPI-IMINDS
St. Pietersnieuwstraat 41, 9000 Gent, Belgium
web: http://telin.UGent.be/
vspruyt@telin.ugent.be

ABSTRACT

This paper proposes a complete tracking system that is capable of long-term, real-time hand tracking with unsupervised initialization and error recovery. Initialization is steered by a three-stage hand detector, combining spatial and temporal information. Hand hypotheses are generated by a random forest detector in the first stage, whereas a simple linear classifier eliminates false positive detections. Resulting detections are tracked by particle filters that gather temporal statistics in order to make a final decision. The detector is scale and rotation invariant, and can detect hands in any pose in unconstrained environments. The resulting discriminative confidence map is combined with a generative particle filter based observation model to enable robust, long-term hand tracking in real-time. The proposed solution is evaluated using several challenging, publicly available datasets, and is shown to clearly outperform other state of the art object tracking methods.

Index Terms— Hand tracking, Particle Filter, Hand detection, Random Forest

1. INTRODUCTION

Human computer interaction by means of hand tracking and gesture recognition has been receiving an increasing amount of attention in both the academic and commercial industry. As low cost depth cameras recently became available, focus has been shifted from monocular computer vision to the interpretation of depth maps. However, due to their dependency on infrared signals, depth sensing devices tend to fail when used in direct sunlight, limiting their applicability. Furthermore, many video sequences that are available today were captured using traditional cameras, and thus lack depth information.

A method for real-time, long-term hand tracking using a simple, low resolution webcam, would overcome these limitations, and could additionally be combined with depth-sensing devices to increase tracking robustness.

In [1], we proposed a real-time hand tracking algorithm, combining a discriminative Hough forest based classifier, with generative cues in a particle filter framework. The algorithm uses a saliency based feature detector to obtain image patches that cast a probabilistic vote for possible locations of the hand centroid. The resulting probability map is combined with color based cues in a particle filter framework that was shown to outperform state-of-the-art object tracking algorithms, such as the well known HandVu hand-tracker [2], and the Predator tracking algorithm [3]. However, a major shortcoming of all these methods, is the need for supervised initialization before tracking can start. Furthermore, manual re-initialization is needed if the tracking algorithm starts to drift or fails.

Research funded by a PhD grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen)

To overcome these limitations, a robust, real-time hand detector is needed that can detect hands, irrespective of their pose, scale and rotation. Due to the high number of degrees of freedom (i.e. 27) in human hands, traditional object detection methods such as Haar based classifiers, fail to capture the full range of hand poses.

Mittal A. *et al.* [4] recently proposed a hand detector capable of detecting hands in unconstrained environments using a parts based deformable model based on HOG (Histogram of Oriented Gradients) features. Detections resulting from this classifier are viewed as hypotheses whose confidence is assessed, based on skin and face detection. While their two-stage algorithm yields impressive results, the whole detection process takes about 2 minutes for an image of size 360×640 pixels on a standard quad-core 2.50 GHz machine and is therefore not suitable for real-time applications. Nevertheless, the work described in this paper is inspired by their results and the idea of cascading a high-recall classifier and a high-precision classifier.

In this paper, we extend the work of [1] by incorporating additional features into the random forest based hand detector, yielding a robust, real-time detector with high recall. Obtained detections are treated as hand hypotheses that are further processed by a second stage linear classifier. Whereas the random forest classifier uses image patches to detect hand hypotheses, the linear classifier simply decides if the obtained bounding boxes actually are hands. Finally, the resulting hand hypotheses are fed to a third stage classifier whose features are uncorrelated with the features used by the previous stages. The third stage classifier is a single decision tree, trained on only temporal information obtained by the particle filter, to eliminate false positive detections resulting from the previous detection stages. This three-stage approach yields a real-time hand detector with high precision and recall, that can automatically initialize the particle filter and re-initialize in case of tracking failure.

Furthermore, we propose a normalization scheme to weigh the probabilistic votes obtained by the random forest, depending on the local saliency in the region. This removes the bias in the voting map of [1] due to the non-uniform spatial sampling of feature points, and greatly increases the tracking accuracy.

This paper is outlined as follows. Section 2 describes the first stage of the classifier, comprising spatial feature detection and random forest classification. In section 3, we discuss the second and third stages of the classifier, which eliminate false positive detections based on spatial and temporal information. Section 4 describes the particle filter framework, and section 5 evaluates our approach, and compares the results with other state of the art object tracking and hand detection algorithms.

2. HAND HYPOTHESIS GENERATION

The high number of degrees of freedom in human hands prevent traditional detection algorithms such as Haar-classifiers to accurately learn a general hand shape. However, because of the articulated nature of a hand, several local features, such as fingertips, can be observed in completely different hand pose configurations, making it a perfect candidate for part-based classification.

Gall and Lepitsky [5] proposed the use of a random forest classifier to classify local image patches. In [1], we extended their work to obtain a real-time, scale and rotation invariant classifier, able to adapt online. A random forest is an ensemble classifier containing multiple decision trees, each of which try to accomplish the same task. Each decision tree is trained on a bootstrapped sample of the training data, using randomly selected features at the decision nodes. If an image patch is classified as being part of a hand, it casts a probabilistic vote on the hand's centroid location. In the following paragraphs, we build upon our previous work to obtain a hand detector that is robust enough for unsupervised initialization of a particle filter.

2.1. Image patch description

For each image patch R_i, the appearance $\mathbf{A_i}$ is described by a set of feature descriptors. In [1], five feature descriptors were calculated on a 3×3 grid. Three of these descriptors were color histograms, while the other two were texture descriptors based on rotation normalized local binary patters and orientation histograms.

However, while the resulting voting map improves tracking performance, its discriminative power is too low for robust hand detection. While well known feature descriptors such as SIFT or SURF have great discriminative power, their computational complexity is too high when integrated into a general tracking framework.

Recently, Alahi *et al.* proposed the Fast Retina Keypoint (FREAK) descriptor [6], inspired by the human visual system. This invariant descriptor has been shown to outperform current state-of-the-art descriptors in robustness, while being much faster to calculate as it simply requires a cascade of binary strings to be computed by comparing image intensities over a retinal sampling pattern.

Furthermore, the 512-bit FREAK descriptor implicitly mimics the hierarchical nature of the human retina, as the most significant bits represent fine-grained spatial information, while the least significant bits describe coarser details. By grouping the descriptor in 32-bit cells, our random forest classifier can exploit this spatial topology.

Before feature calculation, image patches are normalized by their scale and orientation. Each image patch, used to train the trees, is then represented as $R_i = \{\mathbf{A_i}, d_i, \theta_i, \mathbf{o_i}, l_i\}$, where $\mathbf{A_i}$ is the region's appearance, d_i is the distance from the patch centroid to the centroid of the hand, normalized by the scale, θ_i is the angular difference between the patch orientation and the offset vector orientation, $\mathbf{o_i}$ is the rotation- and scale-normalized offset vector, and l_i is the class label which indicates if the image patch is part of a hand ($l_i \in \{0, 1\}$).

During training, the number of positive and negative patches, together with a distribution of offset and rotation vectors, are stored in each leaf node of the decision tree.

2.2. Probabilistic voting

Given the image patch with appearance $\mathbf{A_i}$, we would like to obtain the probability that a certain pixel location (x, y) is the centroid of a hand. By iterating over all pixel locations, the parameters d_i, θ_i and $\mathbf{o_i}$ can be calculated, so that the probability of this location representing the hand's centroid, given the region's appearance, can be written as:

$$p(\mathbf{o_i}, d_i, \theta_i, l_i = 1|\mathbf{A_i}) = p(\mathbf{o_i}, d_i, \theta_i|\mathbf{A_i}, l_i = 1)p(l_i = 1|\mathbf{A_i})$$

which, assuming independence between the arguments, becomes:

$$p(\mathbf{o_i}|\mathbf{A_i}, l_i = 1)p(d_i|\mathbf{A_i}, l_i = 1)p(\theta_i|\mathbf{A_i}, l_i = 1)p(l_i = 1|\mathbf{A_i}) \tag{1}$$

The last factor simply represents the prior probability that an image patch is part of a hand and can be calculated for each decision tree as $P(l_i = 1|A_i) = \dfrac{|\{R_n : l_n = 1\}|}{|\{R_n : l_n = 1\}| + \lambda|\{R_n : l_n \neq 1\}|}$ where λ is the ratio of positive and negative patches in the original training dataset and $\{R_n\}$ is the set of image patches stored in the leaf node that is reached during classification of the patch.

Furthermore, the first three probabilities can be calculated easily from the information stored in the leaf node, because the distributions of offset and rotation vectors are stored in the leaf node during training.

Each detected region is classified by all trees, after which each tree casts a probabilistic vote for each (x, y) location in the image, representing the probability that this location is the hand's centroid. These votes are accumulated in the Hough image which, for each location in the image, represents the likelihood of being the hand's centroid.

However, in practice, this would require the algorithm to iterate over all (x, y) locations for each image patch that is classified by each decision tree. Instead, the same result, apart from a constant multiplicative factor, can be obtained by simply looping over the available training patches in each leaf node, casting a probabilistic vote for the hand centroid corresponding to these training patches, and postprocessing the Hough image by means of a Gaussian filter.

Although this is the approach also used in [1], this assumption is only valid if image patches are sampled on a regular grid, which is not the case, as explained in the next paragraph.

2.3. Likelihood normalization

To focus on salient regions and to allow for real-time detection, we use the rotation and scale invariant feature detector proposed by Kadir and Brady [7] to select regions of interest, instead of simply selecting features on a regular grid.

Image patches are selected such that their entropy, representing the region's unpredictability, is maximized. Furthermore, the scale of the image patch is chosen such that the gradient, and thus the self-similarity, over scale space is maximized. Figure 1 shows the top 20% most salient patches.

If image patches would have been sampled on a regular grid, the probability defined by (1) would be zero for (x, y) locations that correspond to an offset or rotation vector that does not occur in the training data at that leaf node of the tree, and would be non-zero otherwise. Therefore, instead of effectively looping over all (x, y) locations, we could simply loop over the offset and rotation vectors stored in the leaf node, and let these cast a probabilistic vote.

However, since our patches are not sampled on a regular grid, this approach is not valid anymore, because (x, y) locations that have more salient image patches nearby, have a higher prior probability of obtaining a vote than locations that have few salient patches in their neighborhood.

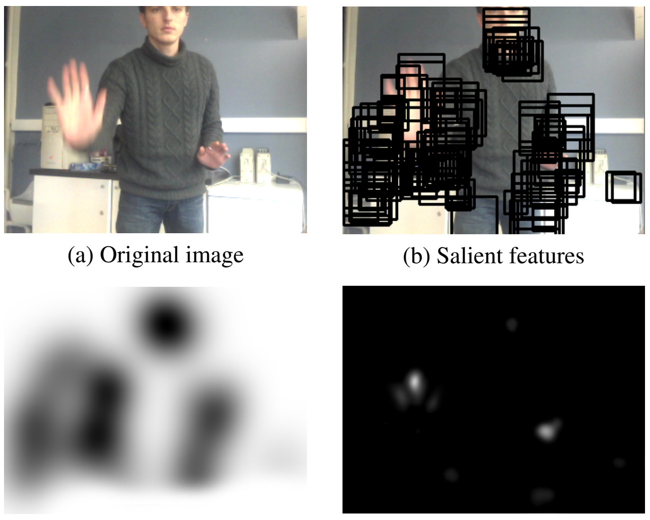
(a) Original image
(b) Salient features
(c) Normalization
(b) Hough voting map

Fig. 1. Illustration of the Hough voting process

The spatial distribution of votes $\mathbf{v}$ cast by the image patches, can be modeled by a mixture of Gaussians, representing $P(\mathbf{v}|R_0,..,R_n)$, where each component of the mixture is centered on the centroid of an image patch R_i that is returned by the saliency detector. This means that the probability of obtaining a vote increases exponentially for locations that are closer to the centroid of on image patch, while similarly the probability of obtaining a vote increases if more image patches are nearby. The variance of the Gaussian components is inversely proportional to the scale of the image patch, since large image patches tend to contain a large part of the object, while small image patches tend to contain only a small part of the object.

In order to normalize the Hough voting map, we multiply the probability, as calculated by (1), with the spatial distribution of expected voting behavior $P(\neg v|R_0,R_1,..R_n)$. This is illustrated in figure 1 (a), while the resulting Hough voting map is shown in figure figure 1 (b).

3. HAND HYPOTHESIS CLASSIFICATION

While the random forest classifier yields a recall (i.e., true positive rate) that is high enough to detect all hands in a video sequence within milliseconds after its appearance, it also shows a low precision as discussed in section 5. This can simply be explained by the lack of discriminative texture information and the high number of degrees of freedom in a human hand, allowing large shape and appearance variations.

Inspired by Mittal A. *et al.* [4], we propose a cascade of our high recall detector, with a simple linear classifier that is able to remove most of the false positive detections. The linear classifier operates on the bounding box of the obtained hand hypotheses and uses the same features as used by the random forest detector. While the random forest detector solves the most difficult problem of finding the hands in the video frame, the linear classifier solves the easier problem of removing noise from the detections.

Finally, each remaining hand hypothesis is used to initialize a particle filter that starts tracking the object in the background. Each

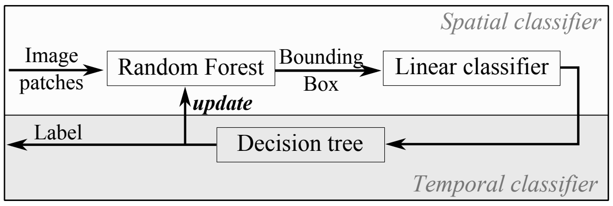

Fig. 2. Three-stage cascade classifier

particle filter gathers simple temporal statistics about the behavior of the tracked object, in order to make a final decision about its classification after N video frames.

The first feature used by the third-stage classifier, is simply the average number of times that the previous stage classifier classified the tracked object as being a hand. The second feature is the average Frobenius norm of the covariance matrix of the particle filter's state vector. This feature thus represents the average uncertainty during object tracking. The third feature is the Frobenius norm of the temporal variance of the particle filter's state vector, which gives an indication about the amount of motion.

A simple decision tree was trained using these features, resulting in the final three-stage classifier. Once the final label is obtained, the Hough forest classifier is updated to allow for online adaption, as schematically illustrated in figure 2 and further explained below.

By keeping record of the image patches that voted during the first stage of the classification, we can update the prior probabilities of the random forest probabilistic voting scheme, in order to allow it to actively learn and adapt. In a specific environment $\mathcal{C}$, some leaf nodes will be reached more often than others. For instance, image patches containing only background pixels will be detected in several frames and always end up in the same leaf node. Learning is accomplished by simply counting the number of times T_p a leaf node is reached by an image patch that turned out to be part of a hand, and by counting how many times T_n the node is reached by a patch that turned out not to be part of a hand. The probabilistic vote for an image patch R_i, is then multiplied by $P(R_i \in \mathcal{C}|\mathbf{A_i}, l_i = 1) = \dfrac{|T_p|}{|T_p| + \eta|T_n|}$ where η is the ratio of the number of positive and negative patches detected over time.

4. PARTICLE FILTER INTEGRATION

Once the temporal classifier decides that a hand hypothesis actually is a hand, the observation model of the particle filer is continuously adapted, to cope with changing illumination and cluttered backgrounds. A Bayesian skin classifier is trained offline and used to generate a likelihood map, as described in [8].

This skin likelihood is combined with motion detection and probabilistic background subtraction. Furthermore, color distributions for hands and background are updated online, and a linear combination of the offline trained skin classifier and the adaptive, online skin classifier is used in order to avoid drifting while being able to cope with changing illumination. Offline color statistics are calculated in RGB color space, whereas online statistics are calculated in the HSV color space. This allows us to incorporate multiple color spaces into our algorithm, yielding better illumination independence [8].

The resulting likelihood image is probabilistically combined with the normalized Hough image. As both are generated using different features, a weak form of independence can be assumed such that the likelihood maps can simply be combined multiplicatively.

Optical flow is used in combination with a constant velocity motion model in the prediction stage of the particle filter. Furthermore, to avoid particle depletion and degeneracy, a mean-shift iteration is embedded into the particle filter. This iteration effectively moves particles to local peaks in the likelihood [9] and reduces the number of resampling rounds needed.

Partitioned sampling is used, with state partitions S_1 and S_2, defined as $S_1 = \{x, y\}$ and $S_2 = \{width, height\}$. This allows the state to be estimated accurately using only 50 particles, since after state space partitioning, only two two-dimensional problems need to be solved, instead of a single four-dimensional problem.

The label of the tracked object is constantly re-evaluated, and particle filters are automatically initialized and re-initialized when needed, as described in the previous section.

5. EVALUATION

The classifier was trained using the same dataset as used in [1], containing over 9000 bounding box annotated hand images. Our random forest consists of fifteen trees, each grown without pruning, to a maximum depth $d_{max} = 18$.

We evaluate the results of the first two stages of our classifier with two publicly available datasets, namely the Mittal dataset [4] containing 660 challenging images, and the Signer dataset [10] containing five news sequences with different signers. Our results on the Mittal dataset are compared with the state of the art detection algorithm proposed by Mittal A. *et al.*, while results on the Signer dataset are compared with the results obtained by both Mittal A. *et al.* and Karlinsky *et al.* [11]. Results for the Mittal dataset are illustrated in table 1.

Table 1. Results obtained on the Mittal dataset

	Precision (%)	Recall (%)	Execution time
Mittal	**48.2**	**85.3**	2 min
Ours	31.7	75.6	**23 ms**

The results on the Mittal dataset show that the first two stages of our algorithm, when used as a static classifier, perform worse than the system proposed by Mittal A. *et al.* On the other hand, average processing time is only 23 milliseconds per image, compared to 2 minutes per image needed by the Mittal algorithm.

For evaluation on the Signer dataset, the same performance measure is used as proposed by Karlinsky *et al.* They fit a chain model to the body of the signer in order to detect hands. Chain fitting starts from the ground truth bounding box of the face and thus only works if such ground truth is available and if the face is visible. The detected hand is considered to be correct if it is within half face width from the ground truth location of the hand. Detection performance is reported within the top *k* detections per ground truth hand instance.

Since the Signer dataset contains video sequences, we evaluate both our two-stage and our three-stage hand tracker on this dataset. For the three-stage classifier, new hand hypotheses originating from the previous stages are constantly evaluated and tracked by the final stage. Table 2 shows the recall percentage for different values of *k*.

These results again show that the two-stage classifier on itself is weaker than other state-of-the-art methods for static hand detection. The reason for this is that we chose to use very simple features in order to meet the real-time constraint. However, when combined with

Table 2. Results obtained on the Signer dataset

	2 max	3 max	4 max
Mittal	90.0	**95.64**	**97.44**
Karlinsky	**92.8**	95.4	96.7
Ours (2-stage)	88.57	94.06	95.1
Ours (3-stage)	**98.7**	**98.7**	**98.7**

the final stage that incorporates temporal information, performance clearly improves, and our detector outperforms the others.

In order to evaluate the complete tracking system, we use the publicly available dataset that was proposed in our earlier paper [1], containing fast motion, changing illumination, camera motion and cluttered backgrounds. Because detailed results on this dataset, obtained by the current state-of-the-art tracking algorithms such as Predator [3] and HandVu[2], have already been reported in our previous work, and since the algorithm proposed there, clearly outperformed these methods on the available dataset, we only report our results compared to the results of our previous solution as illustrated in table 3. Since the same evaluation metric and test data is used, results reported here can directly be compared to the results of the algorithms listed in [1].

Table 3. Average VOC score and variance for each video sequence.

Sequence	[1]		Ours	
	Average	Variance	Average	Variance
1	0.69	0.031	**0.78**	0.010
2	0.7	0.017	**0.77**	0.034
3	0.68	0.02	**0.78**	0.010
4	0.64	0.028	**0.74**	0.011
5	0.73	0.016	**0.78**	0.025
6	0.69	0.025	**0.74**	0.018
7	0.66	0.019	**0.76**	0.011
8	0.66	0.026	**0.76**	0.018

These results clearly show the benefits of the likelihood normalization step which greatly increases the tracking accuracy. The proposed algorithm outperforms the algorithm used in [1], while the tracker is now initialized automatically. Because of the unsupervised re-initialization in case the tracker starts to drift, robust long-term hand tracking is achieved as each of the video sequences tested are 1.5 to 2 minutes in length.

On average, the first hand tracker is started automatically after 2.4 seconds and both hands are tracked after 3.7 seconds. The complete system is able to process 26 frames of size 320×240 per second on a 2.1 Ghz Quad core CPU with 8 Gigabyte of memory.

6. CONCLUSION

We proposed a complete hand tracking solution, capable of unsupervised initialization and automatic recovery from failure. We showed that robust hand detection in video is possible by a cascade of weak classifiers combining spatial and temporal information, in combination with a particle filter tracker. The discriminative classifier cascade used for hand detection results in a confidence map that is further used to enhance the observation model of a particle filter. Our solution operates in real-time, and is shown to outperform state of the art tracking algorithms when applied to hand tracking in unconstrained environments.

7. REFERENCES

[1] V. Spruyt, A. Ledda, and W. Philips, “Real-time hand tracking by invariant Hough forest detection,” in *Proceedings of the IEEE International Conference on Image Processing ICIP12*, 2012, pp. 149–152.

[2] M. Kölsch, “An appearance-based prior for hand tracking,” in *Proceedings of the International Conference on Advanced concepts for Intelligent Vision systems ACIVS 2010*, 2010, pp. 292–303.

[3] Z. Kalal, J. Matas, and K. Mikolajczyk, “P-n learning: Bootstrapping binary classifiers by structural constraints,” in *Proceedings of the Computer Vision and Pattern Recognition conference CVPR10*, june 2010, pp. 49 –56.

[4] A. Mittal, A. Zisserman, and P. H. S. Torr, “Hand detection using multiple proposals,” in *Proceedings of the British Machine Vision Conference BMVC11*, 2011.

[5] J. Gall and V. Lempitsky, “Class-specific Hough forests for object detection,” in *Proceedings of the Computer Vision and Pattern Recognition conference CVPR09*, june 2009, pp. 1022 –1029.

[6] A. Alahi, R. Ortiz, and P. Vandergheynst, “Freak: Fast retina keypoint,” in *Proceedings of the Computer Vision and Pattern Recognition conference CVPR12*, 2012, pp. 510–517.

[7] T. Kadir and M. Brady, “Saliency, Scale and Image Description,” *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, Nov. 2001.

[8] V. Spruyt, A. Ledda, and S. Geerts, “Real-time multi-colourspace hand segmentation,” in *Proceedings of the IEEE International Conference on Image Processing ICIP10*, sept. 2010, pp. 3117 –3120.

[9] C. Shan, T. Tan, and Y. Wei, “Real-time hand tracking using a mean shift embedded particle filter,” *Pattern Recognition*, vol. 40, no. 7, pp. 1958–1970, July 2007.

[10] P. Buehler, M. Everingham, D. P. Huttenlocher, and A. Zisserman, “Long term arm and hand tracking for continuous sign language TV broadcasts,” in *Proceedings of the British Machine Vision Conference BMVC08*, 2008.

[11] L. Karlinsky, M. Dinerstein, D. Harari, and S. Ullman, “The chains model for detecting parts by their context,” in *Proceedings of the Computer Vision and Pattern Recognition conference CVPR10*, june 2010, pp. 25 –32.