Hough Forests for Object Detection, Tracking, and Action Recognition

Juergen Gall *Member, IEEE*, Angela Yao, Nima Razavi, Luc Van Gool *Member, IEEE*, and Victor Lempitsky

Abstract—The paper introduces *Hough forests* which are random forests adapted to perform a generalized Hough transform in an efficient way. Compared to previous Hough-based systems such as implicit shape models, Hough forests improve the performance of the generalized Hough transform for object detection on a categorical level. At the same time, their flexibility permits extensions of the Hough transform to new domains such as object tracking and action recognition. Hough forests can be regarded as task-adapted codebooks of local appearance that allow fast supervised training and fast matching at test time. They achieve high detection accuracy since the entries of such codebooks are optimized to cast Hough votes with small variance, and since their efficiency permits dense sampling of local image patches or video cuboids during detection. The efficacy of Hough forests for a set of computer vision tasks is validated through experiments on a large set of publicly available benchmark datasets and comparisons with the state-of-the-art.

Index Terms—Hough transform, object detection, tracking, action recognition.

1 Introduction

DETECTING objects like pedestrians in unconstrained images or videos, tracking them over time, and recognizing their actions are challenging tasks due to high intra-class variations in shape, appearance, scale, viewpoint, and pose, but also due to occlusions, illumination changes, and background clutter. Nevertheless, there has been considerable progress over the last years, particularly in the field of object detection in static images, e.g., [1], [2], [3], [4], [5], [6], [7], [8], [9]. Based on the success of such systems, recent works in tracking and in action recognition have extended many successful ideas into the spatio-temporal domain [10], [11], [12], [13], [14], [15]. In this paper, we present a unified framework that can be used for class-level object detection in images, object localization/tracking through videos, and action recognition in videos.

Our work is related to several ideas reoccurring in the literature. Firstly, the idea of *local appearance codebooks* [16] forms the basis of many detection systems, including the bag-of-words approach [17] and approaches that model the geometric relations of object parts [8], [18]. Such codebooks are used to classify the local appearance of interest points into a discrete number of visual words that represent an object class. At test time, the appearances of interest points in image or video are matched to words in the visual codebooks, and the remainder of the detection process is based on the classifier trained on the word representation.

The second idea is the use of the *Hough transform* for object detection. Originally developed for detecting straight lines [19], Hough transforms were generalized for detecting generic parametric shapes [20] and then further for detecting object class instances [21], [22], [23], [24], [25], [26]. These days, "Hough transform" usually refers to any detection process based on additive aggregation of evidence (*Hough votes*) coming from local image/video elements. Such aggregation is performed in a parametric space (*Hough space*), where each point corresponds to the existence of an instance in a particular configuration. The Hough space may be a product set of different locations, scales, aspects, etc. The detection process is then reduced to finding maxima peaks in the sum of all Hough votes in the Hough space domain, where the location of each peak gives the configuration of a particular detected object instance (Fig. 1).

The *Implicit Shape Model* of Leibe et al. [27] serves as a natural baseline for our work as it combines the two ideas of appearance codebooks and Hough transform in a natural way. During training, they augment each visual word in the codebook with the spatial distribution of the displacements between the object center and the respective visual word location. At detection time, these spatial distributions are converted into Hough votes within the Hough transform. Over the years, many adaptations of the implicit shape model have been proposed, focusing on improving voting and hypotheses generation [21], [23], [25], [26], [28].

The *Hough forests* introduced in this work provide an alternative way for the combination of machine learning and Hough transform. Hough forests are sets of decision trees learned on the training data. Each tree in the Hough

• *J. Gall, A. Yao, N. Razavi, and L. Van Gool are with the Department of Information Technology and Electrical Engineering, ETH Zurich, Switzerland. E-mail: {gall,yaoa,nrazavi,vangool}@vision.ee.ethz.ch*
• *L. Van Gool is also with the Department of Electrical Engineering/IBBT, K.U. Leuven, Belgium.*
• *V. Lempitsky is with the Department of Engineering Science, University of Oxford, United Kingdom. E-mail: victorlempitsky@gmail.com*

forest maps local appearance of image or video elements to its leaves, where each leaf is attributed a probabilistic vote in the Hough space. In line with the general random forest paradigm [29], [30], the training process for each tree is governed by a combination of *randomization* and *optimization* strategies.

The set of leaves in the Hough forest can thus be regarded as an implicit appearance codebook that has been directly optimized for Hough-based detection (Fig. 2). This is in contrast to the explicit codebook learned within the ISM approach, which employs a fully unsupervised clustering process that is based solely on appearance. For the whole range of detection tasks, this paper demonstrates the gains in detection accuracy brought by codebook optimization for Hough-based detection.

Similar to general random forests, Hough forests are efficient to learn and to apply. The combination of the tree structure and simple binary tests makes training and matching against the codebook very fast, whereas clustering-based learning of explicit codebooks is considerably more expensive in memory and time. Furthermore, the detection process can afford not to be restricted to sparse interest points, but process local image or video elements densely, so that each image or video element casts a Hough vote. Similar to other studies [31], [32], we found that dense sampling leads to improved detection accuracies and is particularly advantageous in the presence of non-idealities such as low resolution and motion blur. Finally, in line with other kinds of random forests [33], [34], Hough forests allow easy on-line adaptation and we demonstrate how it can be used for object instance tracking in videos.

Preliminary versions of this paper appeared in [35] for object detection, [36] for tracking, and [37] for action recognition. The present paper contains a more general formulation of Hough forests that covers all three applications and a more in-depth discussion on multi-class handling, feature sharing, and on-line adaptation. Finally, we show that the general Hough forest framework is capable of handling multiple viewpoint aspects within the same framework, whereas Leibe et al. [38] construct separate detectors for each aspect and fuse the responses of such detectors in a post-processing step. The resulting single detector is not only more convenient, but also achieves higher accuracy in multi-aspect detection tasks.

2 Related work

Detection. Codebook-based detectors learn the mapping from image features into a Hough space where detection hypotheses are obtained by local maxima. To this end, a codebook of local appearance is trained by clustering a training set of image features and storing their relative location with respect to the object center. The spatial distribution can be estimated by a non-parametric Parzen estimate [21] or a mixture of Gaussians [26]. Given a codebook, a max-margin framework can be used to re-weight the votes for better detection [23]. While [21] clusters the sparse image features only based on appearance, the spatial distribution of the image features is used as cue for the clustering in [24], [39]. Hough forests that were originally presented in [35] use a random forest framework [30] instead of clustering for codebook creation. A similar approach has been independently developed in [40].

Sliding window approaches, in combination with many image features, tend to dominate object detection benchmarks like PASCAL VOC 2007 [41] in terms of accuracy. However, the exhaustive search can be too demanding for some applications with respect to memory or runtime requirements. While there has been significant effort to reduce this burden, e.g., by using cascades [2], [9] or branch-and-bound techniques [8], the Hough-based approaches are still very attractive for fast object detection due to their inherent efficiency, and they can also be further optimized by using similar techniques due to the relation between sliding window and Hough-based object detection [26].

Improvements of the implicit shape models include systems designed for the detection of multiple object aspects. While the segmented training data is used in [42] to cluster the shapes, [43] train a codebook for each aspect-view as in [38] but link the aspects together by appearance. In a separate direction, [44] proposes a non-maxima suppression scheme for Hough-based detection that copes better with multiple occluding instances.

The idea of replacing generative codebooks with random forests has been also investigated in the context of image classification and semantic segmentation in [45], [46], [47], [48]. Most similar to Hough forests are the classification random forests used to obtain the unary potentials within the LayoutCRF method [49].

Tracking. Random forests have also been used for real-time tracking [50] where the forest is trained for a single target object. The approach, however, is instance-specific and does not generalize to other objects of the same class. In [33] an on-line update procedure for random forests has been proposed to segment humans in videos. The on-line random forests have been also combined with optical flow and template matching for object tracking [34].

Codebook-based detectors have the added benefit of robustness to occlusions. Since only a small set of local patches is required to locate the object, the detection is still reliable when the object is partially occluded. The idea of voting has been exploited for tracking in [51] where the template of the object is represented by a set of local patches. Each patch is tracked independently and the patches vote for the center of the object.

Action Recognition. Using spatio-temporal interest points for action recognition has become very popular, e.g., cuboids [10], 3D Harris corners [52], 3D Hessians [53], and 3D salient points [54]. Most of these are extensions of their 2D counterparts used in object detection, and many methods follow a traditional object detection approach. After detecting interest points

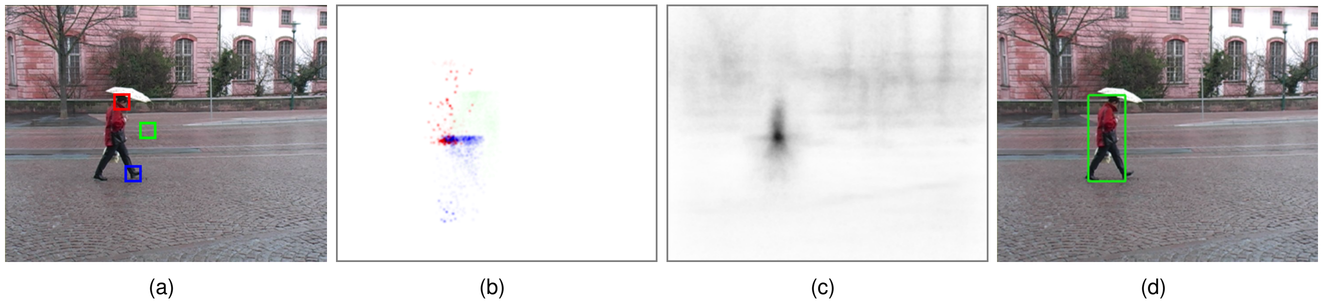


Fig. 1: For each of the three patches emphasized in *(a)*, the pedestrian class-specific Hough forest casts weighted votes about the possible location of a pedestrian *(b)* (each color channel corresponds to the vote of a sample patch). Note the weakness of the vote from the background patch (green). After the votes from all patches are aggregated into a Hough space *(c)*, the pedestrian can be detected *(d)* as a peak in this image.

at multiple scales, feature descriptors are computed, clustered, and assigned to a codebook to be used in some bag-of-words representation [10], [11], [12], [13], [55]. Others have tried to model the spatio-temporal relationships of the features directly [56], [57].

The use of trees and forests for action recognition has been previously explored. In [58], a shape-motion prototype tree is built from shape-motion descriptors; in [59], a vocabulary forest is constructed with local static and flow features, while in [60] a sphere/rectangle tree is built with spatio-temporal interest point features. All three works use trees as indexing structures for performing efficient nearest-neighbor search in either a prototype space [58] or in a feature space in the bag-of-words context [59], [60]. Actions are classified by weighting the n-nearest neighbors and localized either from a foreground segmentation [58] or from the features' spatial information either stored during the training stage [59] or extracted at the test stage [60].

3 HOUGH FORESTS

Hough forests consist of a set of random trees [30] that are trained to learn a mapping from densely-sampled D-dimensional feature *cuboids* to their corresponding votes in a Hough space $\mathcal{H} \subseteq \mathbb{R}^H$. The Hough space encodes the hypothesis $\mathbf{h}$ for an object/action position in scale(time)-space and class. The term *cuboid* below is used in a generalized sense and refers to a local image patch ($D = 2$) or video spatio-temporal neighborhood ($D = 3$) depending on the task.

Let $\mathcal{I}$ further denote the mapping from the input domain $\mathbf{y} \in \Omega \subseteq \mathbb{R}^D$ to the features $\big(I^1(\mathbf{y}), I^2(\mathbf{y}), ..., I^F(\mathbf{y})\big) \in \mathbb{R}^F$, i.e., $\mathcal{I}$ denotes the appearance of an image/video. Here, each I^f is a feature channel and F is the total number of feature channels. The leaves of the trees, $\{L\}$, model the mapping from the appearance of the cuboid centered at $\mathbf{y}$ to the probabilistic Hough vote:

$$\mathcal{L} : (\mathbf{y}, \mathcal{I}) \mapsto p\big(\mathbf{h} \big| \, L(\mathbf{y})\big). \quad (1)$$

Here, $p\big(\mathbf{h} \big| \, L(\mathbf{y})\big)$ is the distribution of Hough votes within the Hough space $\mathcal{H}$. Learning the mapping $\mathcal{L}$ is described in Section 3.1 and using it for detection in Section 3.2. In case of images, i.e., when $D = 2$, and a 2D Hough space, i.e., when $\mathbf{h}$ encodes only the image position $\mathbf{x}$, the detection is illustrated in Fig. 1 and the leaves of the Hough forest in Fig. 2.

3.1 Training

For training, we assume that for each class $c \in C$, a set of training examples is available. For the positive classes, we additionally assume that a D-dimensional bounding box is provided to determine the center and the size of the positive examples. Each tree T in the Hough forest $\mathcal{T} = \{T_t\}$ is then constructed from a set of feature cuboids $\{\mathcal{P}_i = (\mathcal{I}_i, c_i, \boldsymbol{d}_i)\}$ that are randomly sampled from the examples where,

- $\mathcal{I}_i$ are the extracted features for a cuboid of fixed size in $\mathbb{R}^D$,
- c_i is the class label for the exemplar, the cuboid is sampled from,
- $\boldsymbol{d}_i$ is a displacement vector from the cuboid center to the center of the training exemplar.

The negative instances have their own class label and a pseudo displacement $\boldsymbol{d}_i = 0$. We scale the positive examples to a unit size, so that the longest spatial dimension is about $s_u = 100$. Without loss of generality, we assume in this section that the aspect ratio for a class is fixed and that the size of an object can be represented by a scale factor s/s_u. In the experiments, we will give an example where the aspect ratio is an additional dimension of the Hough space. The dimensions of the cuboids that we use are 16×16 for object detection and $16 \times 16 \times 5$ for action detection. This size of cuboids provides a good balance between discriminability of their appearance, allowing inference of the relative location with low ambiguity, and repeatability, allowing good generalization during learning.

Each leaf node L stores the probability of the cuboids belonging to the object class $p\big(c \big| \, L\big)$, estimated by the proportion of feature cuboids per class label reaching the leaf after training, and $D_c^L = \{\boldsymbol{d}_i\}_{c_i = c}$, the cuboids' respective displacement vectors. Each non-leaf node of a tree is assigned a binary test in relation to the cuboid appearance $\mathcal{I}$ during training. The binary test is defined by a comparison of two feature values at locations

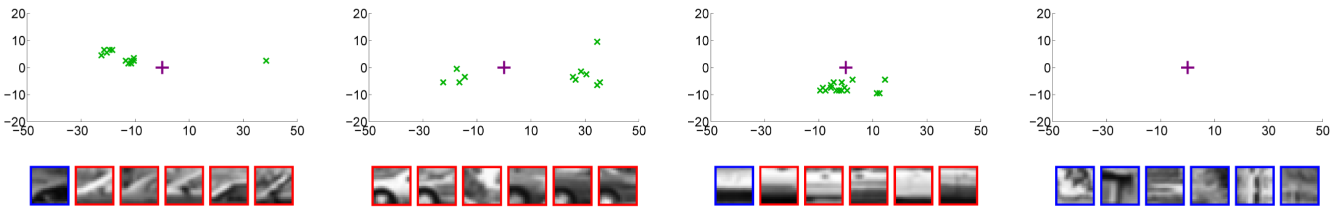


Fig. 2: Visualization of some leaves of a tree for detecting cars (side-view; two classes). Each leaf node L stores the probability of a patch belonging to the object class $p(c|\,L)$, estimated by the proportion of patches from the positive (*red*) and negative examples (*blue*) reaching the leaf after training. For the positive class, the displacement vectors $d \in D_c^L$ are shown (*green*). The leaves of the Hough forest form a discriminative class-specific codebook: the positive training examples falling inside each of the first three leaves can be associated with different parts of a car.

$\boldsymbol{p} \in \mathbb{R}^D$ and $\boldsymbol{q} \in \mathbb{R}^D$ in feature channel f with some offset τ. The binary test at a non-leaf node can be defined as

$$t_{f,\boldsymbol{p},\boldsymbol{q},\tau}\left(\mathcal{I}\right) = \begin{cases} 0 & \text{if } I^f\left(\boldsymbol{p}\right) < I^f\left(\boldsymbol{q}\right) + \tau \\ 1 & \text{otherwise.} \end{cases} \qquad (2)$$

The random trees in Hough forests are constructed according to a standard random forest framework [30]. Construction begins at the root by choosing a binary test, splitting the training cuboids according to the test results and then constructing children nodes. At each subsequent child node, the same procedure continues recursively, with each node being designated as a non-leaf node until the termination criteria is met, i.e., the child node is of a maximum depth, or there are less than a minimum number of cuboids remaining. Upon termination as a leaf, the remaining cuboids' information, $\left(p\left(c|\,L\right), D_c^L\right)_{c\in C}$, is stored (Fig. 2); otherwise, another binary test is chosen and the cuboids are split again. Since the cuboids from all classes $c \in C$ that pass the binary tests and arrive at a certain leaf share the same appearance, the probabilities $p\left(c|\,L\right)$ represent the degree of sharing between the classes.

The ideal binary test should split the cuboids in such a way as to minimize the uncertainty of their class label and displacement vectors. To do this, we use two measures to evaluate the uncertainty for a set of cuboids $A = \{\mathcal{P}_i = (\mathcal{I}_i, c_i, \boldsymbol{d}_i)\}$. The first measure aims to minimize class uncertainty

$$U_1\left(A\right) = -\left|A\right| \cdot \sum_{c \in C} p\left(c|\,A\right) \ln\left(p\left(c|\,A\right)\right), \qquad (3)$$

where $|A|$ is the number of cuboids in set A and $p\left(c|\,A\right)$ is the proportion of cuboids with label c in set A. Note that minimzing this expression for a node corresponds to maximizing the information gain. The second measure aims to minimize the uncertainty of displacement vectors:

$$U_2\left(A\right) = \sum_{c \in C}\left(\sum_{\mathbf{d}\in D_c^A}\left\|\mathbf{d} - \frac{1}{|D_c^A|}\sum_{\mathbf{d}'\in D_c^A}\mathbf{d}'\right\|^2\right). \qquad (4)$$

Note that the displacement vectors of the negative class ($\mathbf{d} = 0$) have no impact on the measure.

At each node during training, a pool of binary tests $\left\{t^k\right\}$ is generated with random values of f, $\boldsymbol{p}$, $\boldsymbol{q}$, and τ. Then, either class or displacement uncertainty is chosen at random to be minimized at that given node. The set of cuboids arriving at the node is evaluated with all binary tests in the pool and the binary test satisfying the following minimization objective is chosen:

$$\underset{k}{\operatorname{argmin}}\left(U_\star\left(\left\{\mathcal{P}_i\,\middle|\,t^k = 0\right\}\right) + U_\star\left(\left\{\mathcal{P}_i\,\middle|\,t^k = 1\right\}\right)\right), \qquad (5)$$

where $\star$ indicates the chosen uncertainty measure for the node (U_1 or U_2). By randomly selecting the uncertainty measure, nodes decreasing both class and displacement uncertainty are interleaved throughout the tree. As such, cuboids being stored at the leaves tend to have low variation *both* in class label and in displacement; hence, they vote with low uncertainty into the Hough-space.

In [40] a weighted objective function is proposed to minimize both uncertainties. Although the impact of the two objective functions (3) and (4) can be controlled by an additional weighting parameter, we show in the experiments that, in general, this does not result in better performance. Furthermore, replacing (4) by a more expensive information-theoretic displacement uncertainty measure as in [40] also does not improved the performance significantly. We finally remark that the trees are not guaranteed to be balanced and are not perfectly balanced in practice. There is, however, a bias towards balanced trees, as both splitting criteria (3) and (4) have biases towards equal-size partitions.

3.2 Detection

For detection, extracted D-dimensional feature cuboids are passed through each tree in the Hough forest; the leaves that the cuboids arrive in are then used to cast votes to the Hough space $\mathcal{H} \subset \mathbb{R}^H$. Fig. 1 illustrates the voting for object detection in an image. To begin with, consider a cuboid $\mathcal{P}(\mathbf{y}) = (\mathcal{I}(\mathbf{y}), c(\mathbf{y}), \mathbf{d}(c(\mathbf{y})))$ located at position $\boldsymbol{y} \in \mathbb{R}^D$, where $\mathcal{I}\left(\boldsymbol{y}\right)$ are the extracted features for the cuboid, $c(\mathbf{y})$ the unknown class label, and $\mathbf{d}(c(\mathbf{y}))$ the displacement of the cuboid from the unknown object's center. Based on the appearance $\mathcal{I}(\mathbf{y})$, the cuboid ends in a leaf $L(\mathbf{y})$. Let $\mathbf{h}(c, \mathbf{x}, s)$ be the hypothesis for the object belonging to class $c \in C$ with size s and centered at $\boldsymbol{x} \in \mathbb{R}^D$. We are interested in the conditional probability $p\big(\mathbf{h}|\,L\big)^1$, which can be decomposed as follows:

$$\begin{aligned}
&p\big(\mathbf{h}(c,\mathbf{x},s)|\,L(\mathbf{y})\big)\\
&=\sum_{l\in C}\big(p\big(\mathbf{h}(c,\mathbf{x},s)|\,c(\mathbf{y})=l,L(\mathbf{y})\big)\;p\big(c(\mathbf{y})=l|\,L(\mathbf{y})\big)\big)\\
&=p\big(\mathbf{h}(c,\mathbf{x},s)|\,c(\mathbf{y})=c,L(\mathbf{y})\big)\;p\big(c(\mathbf{y})=c|\,L(\mathbf{y})\big) \qquad (6)\\
&=p\Big(\mathbf{x}=\mathbf{y}-\frac{s}{s_u}\mathbf{d}(c)|\,c(\mathbf{y})=c,L(\mathbf{y})\Big)p\big(c(\mathbf{y})=c|\,L(\mathbf{y})\big),
\end{aligned}$$

where s_u is the unit size from the training data.

Both factors in (6) are estimated during training. While $p\big(c|\,L\big)$ is estimated by the proportion of feature cuboids per class label reaching the leaf after training, the distribution $p\big(\mathbf{h}|\,c,L\big)$ can be approximated by a sum of Dirac measures $\delta_{\mathbf{d}}$ for the displacement vectors $d\in D_c^L$:

$$p\big(\mathbf{h}(c,\mathbf{x},s)|\,L(\mathbf{y})\big)= \qquad (7)$$
$$\frac{p\big(c(\mathbf{y})=c|\,L(\mathbf{y})\big)}{\Big|D_c^{L(\mathbf{y})}\Big|}\left(\sum_{\mathbf{d}\in D_c^{L(\mathbf{y})}}\delta_{\mathbf{d}}\left(\frac{s_u(\mathbf{y}-\mathbf{x})}{s}\right)\right).$$

For the entire forest $\mathcal{T}$, we pass the appearance of the cuboid $\mathcal{I}(\mathbf{y})$ through all trained trees and average the probabilities (7) coming from the different leaves [30]:

$$p\big(\mathbf{h}|\,\mathcal{I}(\mathbf{y})\big)=\frac{1}{T}\sum_{t=1}^{T}p\big(\mathbf{h}|\,L_t(\mathbf{y})\big), \qquad (8)$$

where $L_t(\mathbf{y})$ is the corresponding leaf for tree T_t. To integrate the votes coming from all extracted cuboids of the input domain $\Omega\subseteq\mathbb{R}^D$, we accumulate them into the Hough image $\mathcal{H}$:

$$p\big(\mathbf{h}|\,\mathcal{I}\big)\propto\sum_{\mathbf{y}\in\Omega}p\big(\mathbf{h}|\,\mathcal{I}(\mathbf{y})\big). \qquad (9)$$

Note that the sum is not a probability measure since it does not integrate to one. However, we are only interested in the modes of $p\big(\mathbf{h}|\,\mathcal{I}\big)$ that can be obtained by searching for local maxima without estimating the normalization factor. The maxima can be searched by applying a Parzen estimator with a Gaussian kernel K:

$$\hat{p}\big(\mathbf{h}|\,\mathcal{I}\big)=\sum_{\mathbf{h}'\in\mathcal{N}(\mathbf{h})}w_{\mathbf{h}'}\cdot K\left(\mathbf{h}-\mathbf{h}'\right),\text{ where} \qquad (10)$$
$$w_{\mathbf{h}'}=\sum_{y\in\Omega}\sum_{t=1}^{T}\sum_{\mathbf{d}\in D_c^{L_t(\mathbf{y})}}\frac{p\big(c(\mathbf{y})=c|\,L_t(\mathbf{y})\big)}{T\Big|D_c^{L_t(\mathbf{y})}\Big|}\delta_{\mathbf{d}}\left(\frac{s_u(\mathbf{y}-\mathbf{x})}{s}\right).$$

The weight of a hypothesis $w_{\mathbf{h}'}$ accumulates all votes that support the same hypothesis $\mathbf{h}'(c,\mathbf{x},s)\in\mathcal{H}$. After all votes are cast, $\hat{p}\big(\mathbf{h}|\,\mathcal{I}\big)$ represents the sum of the weights of the hypotheses in the neighborhood of $\mathbf{h}$ weighted by a Gaussian kernel K. While the location of a local maximum $\hat{\mathbf{h}}(c,\mathbf{x},s)$ encodes class, position, and size of the object, the value $\hat{p}\big(\hat{\mathbf{h}}|\,\mathcal{I}\big)$ serves as confidence measure for each hypothesis.

The accumulation of the probabilities in (9) is non-probabilistic, however, the summation is preferred over multiplication due to better stability in practice. A more probabilistic treatment, corresponding to multiplying the robust estimates of the probabilities and allowing principled recovery of multiple detections in the same image/video is also possible [44].

4 Applications

We present three applications of the Hough forests, namely object detection (Section 4.1), tracking (Section 4.2), and action recognition (Section 4.3).

4.1 Object Detection

For object detection, the input data is an image, i.e., $\Omega\subseteq\mathbb{R}^2$, the estimated parameters are pixel location and size, i.e., $\mathcal{H}\subseteq\mathbb{R}^3$, and there are two classes, a positive and a negative one. In our particular training setup, the positive examples were rescaled, so that the size of the largest bounding box dimension $s_u=100$. $20\,000$ random binary tests were considered for each node. Each tree was trained on about $25\,000$ positive and $25\,000$ negative patches. To bias our training to work better on hard examples, we used the following boosting-like procedure. For the first 5 trees, the patches were randomly sampled from all available examples. Then the constructed Hough forest was applied to the training data and the 400 positive and negative instances that were harder to classify were acquired. These were used to construct the next 5 trees and added to the previous 5. We applied this procedure once more, resulting in a forest of 15 trees. For detection, we used a Gaussian kernel with $\sigma^2=9$ (10). In a multi-scale setting, the additional third dimension was filtered with $\sigma^2=1$. At test time, 4–5 scales with equal spacing were used to handle the variety of object sizes in the test data.

The performance curves were generated by changing the acceptance threshold on the hypotheses vote strength $p(\hat{\mathbf{h}}|\,\mathcal{I})$. We rejected the detection hypotheses with centers inside the bounding boxes detected with higher confidence in order to avoid multiple detections of the same instance. We adhered to the experimental protocols and detection correctness criteria established for each of the datasets in previous works.

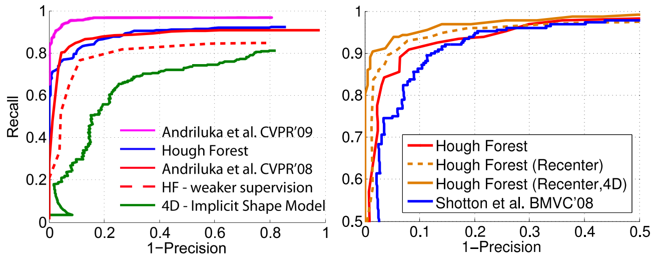
UIUC cars. The UIUC car dataset [61] contains images of side views of cars. *UIUC-Single* contains 210 cars of approximately same scale and *UIUC-Multi* 139 cars at multiple scales. For patch appearance, 3 channels were used (intensity, absolute value of x- and y-derivatives). Applying this forest for the detection achieved an impressive 98.5% EER for UIUC-Single and 98.6% for UIUC-Multi, thus exactly matching the state-of-the-art performance reported recently in [62] (Table 1).

More importantly, the Hough forest considerably outperformed the Hough-based implicit shape model [21]

1. In the text, we use the abbreviated forms $p\big(\mathbf{h}|\,L\big)$, $p\big(\mathbf{h}|\,c,L\big)$, and $p\big(c|\,L\big)$ for $p\big(\mathbf{h}(c,\mathbf{x},s)|\,L(\mathbf{y})\big)$, $p\big(\mathbf{h}(c,\mathbf{x},s)|\,c(\mathbf{y})=c,L(\mathbf{y})\big)$, and $p\big(c(\mathbf{y})=c|\,L(\mathbf{y})\big)$, respectively.

Methods	UIUC-Single	UIUC-Multi
Hough-based methods		
Implicit Shape Model [21]	91%	–
ISM+verification [21]	97.5%	95%
Boundary Shape Model [24]	85%	–
Max-margin HT+verif. [23]	97.5%	–
Random forest based method		
LayoutCRF [49]	93%	–
State-of-the-art		
Mutch and Lowe CVPR'06 [63]	99.9%	90.6%
Lampert et al. CVPR'08 [62]	98.5%	98.6%
Karlinsky et al. CVPR'10 [64]	99.5%	–
Hough Forests		
Hough Forest	98.5%	98.6%
HF - Weaker supervision	94.4%	–
HF - Sparse	95.5%	–
HF - Weighted [40]	98.5%	–

TABLE 1: Performance of different methods on the two UIUC car datasets at recall-precision equal error rate (EER). The Hough forest outperforms the previous Hough-based and random forest based methods and achieves the state-of-the-art.



(a) TUD Pedestrian dataset (b) Weizmann Horse dataset

Fig. 3: Hough forests demonstrate a competitive performance with respect to the previous state-of-the-art methods on two challenging datasets.

(even with an additional MDL verification step) and boundary-shape model approach [24] as well as the random-forest based LayoutCRF method [49]. It has to be mentioned, however, that these related methods used smaller subsets of the provided training data. In the case of the ISM and the LayoutCRF, this is due to the necessity of obtaining pixel-accurate annotations. Additionally, in the case of ISM and the boundary-shape model [24] this might be due to the computational burden of constructing and processing generative codebooks. As Hough forests are not limited by these factors, we used the complete set of provided training data, possibly accounting for some part of the improvement.

TUD pedestrians, multi-scale Weizmann Horses. To assess Hough forests performance on more challenging articulated classes, we evaluated our method on the TUD pedestrian datasets [65]. The dataset contains partial occlusions and variations in scales, poses, clothing styles, and weather conditions. In addition to the 400 positive training images with pedestrians, we used training background images from the INRIA dataset [3]. Otherwise, we followed the experimental protocol of [65] and tested on 250 images with 311 pedestrians in it. We have also considered the Weizmann Horses dataset [66] containing the near-side views of horses in natural environments under varying scale and strongly varying poses. We used the training-testing split (100 horse images and 100 background image for training, 228 horse images and 228 background images for testing) as suggested in [39].

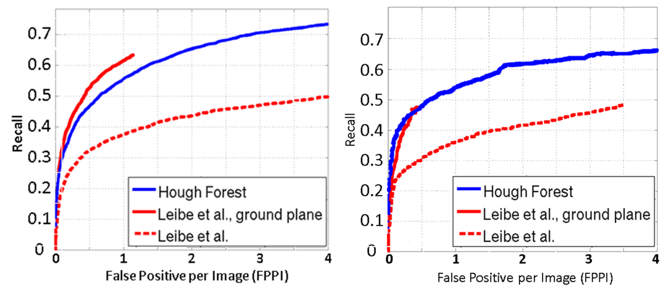
We have considered the following 16 feature channels: 3 color channels of the *Lab* color space, the absolute values of the first and second-order derivatives in x- and y-direction and nine HOG [3] channels. Each HOG channel was obtained as the soft bin count of gradient orientations in a 5×5 neighborhood around a pixel. To increase the invariance under noise and articulations of individual parts, we further processed the above-introduced 16 channels by applying the min and the max filtration with 5×5 filter size, yielding $C = 32$ feature channels (16 for the min filter and 16 for the max filter).

The performance of different methods including ours is shown in Fig. 3*(a)*. For TUD pedestrians, our method (recall-precision EER = 86.5%, AUC = 0.87, recall at 90% precision = 85%) achieves competitive results compared to the state-of-the-art methods [65], [67] and performs significantly better than the implicit shape model-based method [68] (reproduced from [65]). It should be noted that the competing methods require additional annotation for training. While [65], [67] are based on an explicit model with joint annotation, the ISM-based approach [68] relies on silhouettes. Again, these systems were trained on a subset of the training data. For an image from the TUD dataset, our system requires 6 seconds (720×576 pixel resolution; 4 scales $(0.3, 0.4, 0.5, 0.6)$).

For the multi-scale Weizmann Horse dataset, the performance of the Hough forest was clearly better than the related work of Shotton et al. [69] (Fig. 3*(b)*). Nevertheless, we have tried two more improvements addressing the two challenges of this dataset. Firstly, the positions of the bounding box centers are not stable with respect to the horse bodies, which leads to a certain smearing of votes. To address this, we ran our detector on the positive training images and recentered the bounding boxes to the peaks of the response. After that the forest was retrained. Secondly, the aspect ratios of the boxes varied considerably due to the articulations and variations in the viewpoint. To address this, we performed voting in a 4D Hough space, where the 4th dimension corresponded to the aspect ratio multiplier. As can be seen from Fig. 3*(b)*, both improvements increased the performance considerably (recall-precision EER went from 91% to 93.9%, AUC from 0.96 to 0.98, recall at 90% precision from 91.5% to 95.1%). For comparison, the recursive compositional model [70] reports AUC = 0.98[2].

PASCAL VOC 2007. We have tested the Hough forests on the categories "car" and "tvmonitor" of PASCAL VOC'07 [41] where an average precision of 0.166 and 0.215 has been achieved for object detection, which is considerably lower than the state-of-the-art. As other Hough transform-based approaches, the method struggles with the variation of the data that contains many truncated examples. However, techniques described in Section 2 like non-maxima suppression [44] or an addi-

2. Note that the performance numbers recently reported in [64] are not comparable to ours as they correspond to a simpler single-scale version of the Weizmann Horse dataset.



(a) Leuven car (b) Amsterdam

Fig. 4: The Hough forest outperforms the multi-view approach of Leibe et al. [38] on two challenging car datasets. The boost in performance is comparable to the use of geometric scene information (*ground plane*).

tional verification step as in [23] can be combined with Hough forests to improve the detection results.

Impact of displacement supervision and feature density. Several previous approaches have used random forests as discriminative codebooks [45], [46], [47], [48], [50]. Hough forests differ from them as they store the displacement vectors at leaves and use them at runtime to perform voting. Furthermore, the displacement information is used as supervision during training since half of the binary tests are chosen to minimize the displacement uncertainty (4). We therefore addressed the question whether such additional supervision matters. To this end, we built forests where all binary splits were chosen to minimize the class uncertainty (3), a commonly used criteria for building random forests. The leaf information and the detection procedure remained as before. The performance of the new forests form the *HF-weaker supervision* entries in Table 1 (UIUC-Single) and Fig. 3*(a)* (TUD). A considerable drop in performance compared to fully-supervised Hough forests is observed, suggesting that displacement vectors were a valuable supervision during training.

When the randomized selection between the two uncertainty measures for training is replaced by a weighted objective function of both measures (*HF-weighted* [40]), a similar performance is achieved (Table 1). Although an optimal setting of the weighting parameter might improve the results for some datasets, the additional parameter needs to be either manually set as in [40] or estimated from training data.

For evaluating the impact of the feature density, we trained and tested the Hough forests only on interest points extracted by a Hessian-Laplace detector (*HF-sparse*) as in [21]. Although the performance decreases when sparse features are used, it is still better than ISM without verification (Table 1). This shows that using dense features improves the performance, but also that Hough forests outperform ISM even for sparse features.

Multiple aspect views. Multiple aspect views, where the views are annotated in the training data, can be handled by assigning each view a class label, i.e., $C = \{0, v_1, \dots, v_n\}$ where 0 is the label for the negative class and v_i the label for the viewpoints. Hence, the estimated parameters are pixel location, scale, and viewpoint, i.e., $\mathcal{H} \subseteq \mathbb{R}^4$. The viewpoint annotation is necessary to obtain an accurate bounding box estimation for the detection without additional postprocessing. For testing, we use the Leuven and Amsterdam car dataset [38]. For training, 1471 cars annotated with 7 different view aspects are provided. The Leuven dataset consists of 1175 images and is very challenging due to low resolution, strong partial occlusion between parked cars, motion blur, and contrast changes between brightly lit areas and dark shadows. The Amsterdam sequence consists of 290 images.

We report the results of the Hough forest in Fig. 4. As previously, the Hough forest outperforms the cluster-based codebook approach of Leibe et al. [38] (ISM). The detection performance almost matches the performance of ISMs when geometric scene information (ground plane) is provided. Additionally, the Hough forest leads to an arguably more "compact" system since it trains one codebook for all views, whereas [38] constructs a codebook for each view and fuses the responses of the detectors in a post-processing step.

Note that neither [38] nor the Hough forests are specific to multi-aspect view detection but treat the views as different classes. In general, better detection performances can be achieved by taking additional extensions into account as mentioned in Section 2. For instance, the non-maxima suppression scheme proposed in [44] improves the results for the TUD pedestrian dataset (Fig. 3*(a)*) and the multi-view detector can be made more efficiently by exploiting back-projection [71]. Multi-class handling is evaluated more in-depth in Section 4.3, in the context of action recognition.

4.2 Tracking

An object can be tracked by assembling the detections to tracks, e.g., by using the confidence $\hat{p}(\mathbf{h}|\mathcal{I})$ (10) as observation for a particle filter [72], [73]. Besides the position $\mathbf{x}$ and scale s of the object, dynamic parameters like velocity $\mathbf{v}$ and acceleration $\mathbf{a}$ are also estimated. We denote the state vector by $\mathbf{e} = (\mathbf{x}, s, \mathbf{v}, \mathbf{a})$. For tracking, one seeks the posterior distribution for a current frame $\mathcal{I}_t$, i.e., $p(\mathbf{e}_t|\, H_t, \dots, H_0)$ where $H_t = p(\mathbf{h}|\mathcal{I}_t)$. The posterior is approximated by a set of particles $\{\mathbf{e}_k\}$ and is estimated by the recursive equation:

$$p(\mathbf{e}_t|\, H_t, \dots, H_0) \propto p\big(H_t|\, \mathbf{e}_t\big) p\big(\mathbf{e}_t|\, \mathbf{e}_{t-1}\big) p\big(\mathbf{e}_{t-1}|\, H_{t-1}, \dots, H_0\big), \quad (11)$$

i.e., after predicting the particles $\{\mathbf{e}_k\}$ according to the dynamical model $p\big(\mathbf{e}_t|\, \mathbf{e}_{t-1}\big)$, the particles are weighted by the likelihood $w_k = \hat{p}(\mathbf{h}(\mathbf{x}_k, s_k)|\mathcal{I}_t) \propto p\big(H_t|\, \mathbf{e}_k\big)$ (10). Since the normalization factor is unknown, the weights are normalized such that $\sum_k w_k = 1$ before the resampling [73]. An example is given in Fig. 6*(a)*.

4.2.1 On-line Adaptation

This, however, is not the most efficient way since an off-line trained detector as in Section 4.1 tries to solve a

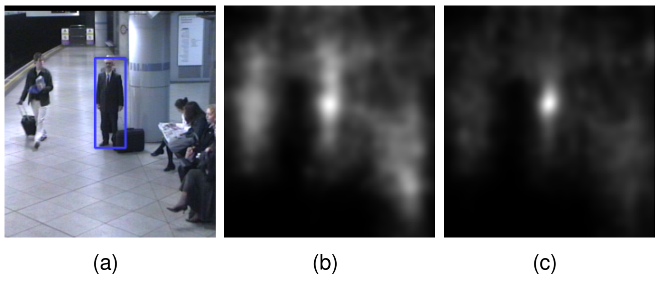


Fig. 5: In order to track an instance of a class, like a certain person from the class pedestrians, we adapt on-line a class-specific Hough forest to the instance. *(a)* Blue box indicates the instance of interest. *(b)* Voting image obtained by an off-line trained Hough forest for pedestrians. *(c)* Voting image obtained by the instance-specific Hough forest. The peak at the center of the instance is better visible than in *(b)*.

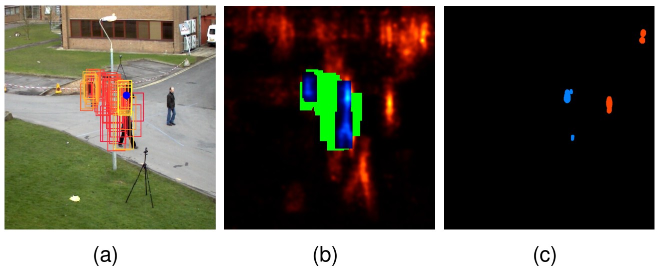


Fig. 6: On-line adaptation of the Hough forest. *(a)* After updating the particles, the multi-modal posterior distribution is approximated. The weights of the particles are indicated by color (*yellow: high, red: low*). The target is marked by a *blue dot*. *(b)* Based on the posterior, the voting space is labeled (*blue: foreground, red: background, green: uncertain*). The intensity of the background (*red*) has been increased for a better visibility. In reality, the maximum of the background is much lower than for the foreground. *(c)* Votes that contributed to the detected local maxima are used to update the instance-specific statistics. In this example, there are two strong foreground maxima (*blue*) such that the votes for the other instance are also taken into account for the update.

much more difficult task than object tracking, namely identifying any instance of the class in any image. For tracking, the statistics of the target object E and the background are very similar between successive frames. By updating the statistics stored in the leaves of the Hough forest (7), it can be adapted to the target, which is a specific instance of the class.

As in [21], [71], one can collect the *entries* of the leaves, $\mathbf{d} \in D_c^L$, that voted for a given object hypothesis; these entries can be regarded as a signature for the target of interest. Since a change of pose and appearance can lead to an activation of very different tree leaves, we learn the statistics for the target and the background over time, i.e., we update on-line the probability of each entry of the leaves for belonging to the target. By taking the target-specific statistics into account during voting, the target can be distinguished from other instances in the background yielding a higher detection confidence for the target, see Fig. 5.

To this end, we estimate $p\big(\mathbf{h}_E \big| L(\mathbf{y})\big)$, i.e., the probability that a hypothesis is caused by the target object E. Similar to (7), we get

$$p\big(\mathbf{h}_E \big| L(\mathbf{y})\big) = \frac{1}{\left|D_c^{L(\mathbf{y})}\right|} \left(\sum_{\mathbf{d} \in D_c^{L(\mathbf{y})}} \delta_{\mathbf{d}} \left(\frac{s_u(\mathbf{y}-\mathbf{x})}{s} \right) \cdot \qquad (12)$$

$$\ldots p\big(\mathbf{h}_E = \mathbf{h}(\mathbf{d},\mathbf{y}) \big| \, c(\mathbf{y}) = c, L(\mathbf{y})\big) \cdot p\big(c(\mathbf{y}) = c \big| \, L(\mathbf{y})\big) \right),$$

where $p\big(\mathbf{h}_E = \mathbf{h}(\mathbf{d},\mathbf{y}) \big| \, c(\mathbf{y}) = c, L(\mathbf{y})\big)$ is the probability that the vote cast by $\mathbf{d}$ belongs to the target. For the adaptation only $p\big(\mathbf{h}_E = \mathbf{h} \big| \, c, L\big)$ needs to be estimated since the other terms are already computed off-line (7).

For estimation, we count the number of times an entry of a leaf $\mathbf{d} \in D_c^L$ votes for the target instance $\Omega_{\mathbf{d},E} = \big\{\mathbf{y} \big| \, \mathbf{h}(\mathbf{d},\mathbf{y}) = \mathbf{h}_E\big\}$ and the number of times it votes for other objects $\Omega_{\mathbf{d},\complement E} = \big\{\mathbf{y} \big| \, \mathbf{h}(\mathbf{d},\mathbf{y}) \neq \mathbf{h}_E\big\}$:

$$p\big(\mathbf{h}_E = \mathbf{h}(\mathbf{d},\mathbf{y}) \big| \, c(\mathbf{y}) = c, L(\mathbf{y})\big) = \frac{|\Omega_{\mathbf{d},E}|}{|\Omega_{\mathbf{d},E}| + |\Omega_{\mathbf{d},\complement E}|}. \qquad (13)$$

When the entry has not been previously activated for voting, we assume a 0.5 chance that the patch belongs to E, see Fig. 7.

In order to compute (13), we assign a label to each $\mathbf{h}$ based on the posterior distribution (11) as illustrated in Fig. 6. Namely 1 (*blue*) or -1 (*red*) if we are confident that it either belongs to the instance or it does not. When the posterior is greater than zero but relatively low, we assign the label 0 (*green*) to it. After labeling the elements in the Hough space, we search for strong local maxima in the positive and the negative cluster. The elements of the cluster labeled with 0 are discarded. Finally, we collect the votes that contributed to the local maxima and add them to the corresponding sets $\Omega_{\mathbf{d},E}$ and $\Omega_{\mathbf{d},\complement E}$.

Note that the update performs only a re-weighting of the entries in the Hough forest. It neither changes the stored displacement vectors d nor does it add new displacements to the leaves. On the one hand, the localization accuracy does not suffer from the updates as it might happen for other on-line learning approaches. On the other hand, instances that are not localized a-priori by the detector cannot be tracked since new observations are not added. In the worst case, a target with an a-priori weak confidence is confused by another hypothesis with a-priori higher confidence.

4.2.2 Experiments

For a quantitative evaluation, we use two standard datasets i-Lids [75] and PETS09 [76] that have been recorded in an underground station and a public place. The sequences contain several instances (persons) of the class (pedestrians). For comparison, we apply the tracker with on-line adaptation and a particle filter without any adaptation (*No Update*), i.e., using the class-specific Hough forest only. For the Hough forest, we used only the first 5 trained trees for the TUD pedestrian dataset due to efficiency. The depth of each tree is 15 and the average number of entries per leaf is 11.4. All trackers run with 50 particles and are initialized by a given bounding box. As an estimate, we take the strongest mode of the posterior. The accuracy is measured by taking the intersection-union ratio of the estimated and ground-truth bounding box for each frame. The results in Fig. 8 show the benefit of the on-line adaptation of a

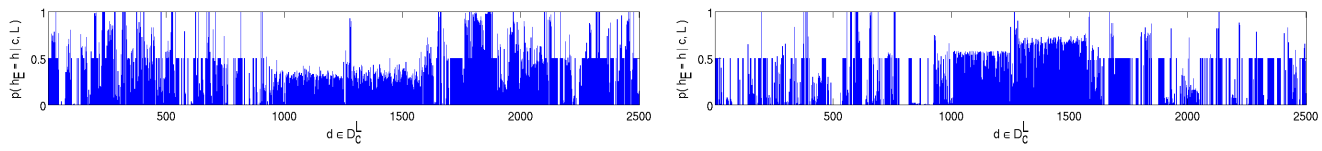
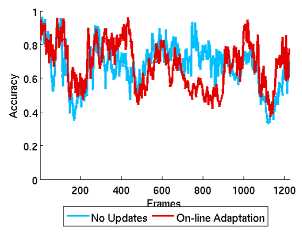


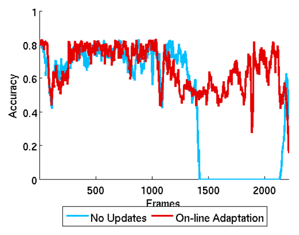
Fig. 7: The probabilities $p(\mathbf{h}_E = \mathbf{h}(d)|\, c, L)$ for the first 2500 entries $\mathbf{d} \in D_c^L$ of the leaves L of a single tree. The probabilities are estimated on-line by (13) for two different persons after 100 frames. They give an instance-specific signature that can be used to improve tracking, see Fig. 5. While entries with probability > 0.5 are specific to the instance, probabilities < 0.5 indicate entries specific to the background. Entries with probability equal to 0.5 belong mainly to leaves that have not been activated during tracking.

Accuracy (%)	On-line Adaptation	No Update	[51]	[74]	[14]	[15]
i-Lids(easy)	$\mathbf{67.4} \pm 13.5$	66.9 ± 12.8	42.9 ± 18.1	25.0 ± 21.2	0.9 ± 8.8	28.5 ± 18.9
i-Lids(medium)	65.4 ± 12.2	45.9 ± 33.9	$\mathbf{73.7} \pm 8.0$	23.1 ± 31.0	6.7 ± 21.1	35.9 ± 36.6
i-Lids(hard)	$\mathbf{65.9} \pm 15.0$	53.2 ± 15.2	28.5 ± 33.9	21.0 ± 31.9	6.7 ± 21.4	34.8 ± 37.7
PETS09	$\mathbf{60.3} \pm 15.3$	31.3 ± 29.9	8.7 ± 20.5	9.6 ± 24.3	13.2 ± 26.9	8.4 ± 22.2

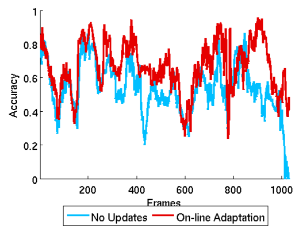
TABLE 2: Mean and standard deviation of the tracking accuracy.



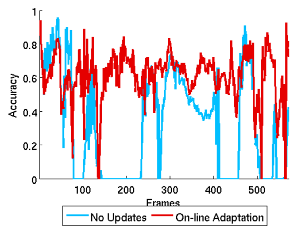
(a) i-Lids *easy* [75]



(b) i-Lids *medium* [75]



(c) i-Lids *hard* [75]



(d) S2.L1 PETS09 [76]

Fig. 8: Tracking accuracy for 4 sequences over time. Mean and standard deviation are given in Table 2. On the *easy* sequence, the class-specific Hough forest and the on-line adaptation perform well. In sequences *medium*, *hard*, and S2.L1, the scene is more crowded. This is a situation where on-line adaptation outperforms the class-specific Hough forest.

Feature	Particle filter	Voting	On-line Adaptation
180*msec.*	0.3*msec.*	235*msec.*	63*msec.*

TABLE 3: Since votes with zero probability are not cast, voting with on-line adaptation is 2.8 times faster than voting with the class-specific Hough forest (*851msec.*).

class-specific Hough forest to the target instance. While simple sequences without ambiguities can be handled by a class-specific Hough forest, more complex scenes with several instances cannot be tracked without the on-line adaptation. Note that the on-line adaptation reduces the computation time, see Table 3. The results for the fragment tracker [51] and some on-line boosting approaches [14], [15], [74] are given in Table 2. However, we have to emphasize that the publicly available implementations neither handle scale nor make use of any off-line training.

4.3 Action Recognition

Hough forests can also be applied to localize and recognize human actions in unconstrained video, i.e., $\Omega \subseteq \mathbb{R}^3$.

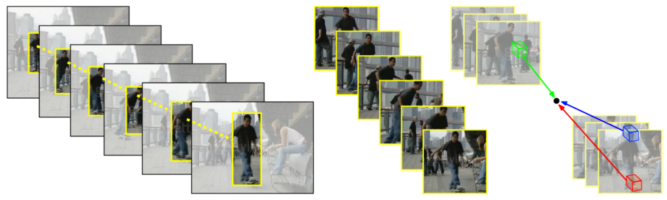


Fig. 9: After tracking, the video data is normalized by scale and position of the human. The normalized *action tracks* are used as input data for the action recognition.

To facilitate the recognition, we first track the human as described in Section 4.2 without on-line adaptation unless otherwise noted. Since humans show a large variation in pose and appearance, particularly for sports clips, the likelihood (11) can be enriched with some additional color and texture information [77]. After tracking the human, the video data is normalized into spatial- and scale-invariant action tracks as illustrated in Fig. 9.

For training, we assume that for all action classes $C = \{c_1, \ldots, c_n\}$ a set of training sequences is available. Each training example is annotated such that it can be transformed into a normalized action track and contains roughly one action cycle, i.e., it is annotated by a 2D bounding box for each frame, the action label, and the temporal boundaries of the action cycle. For testing, the trained Hough forest is applied to a normalized action track to obtain the class label c and the spatio-temporal location $(\mathbf{x}, t)$ of the action, i.e., $\mathcal{H} \subseteq \mathbb{R}^4$. Despite having spatially localized tracks, we vote in the spatial dimensions as well, in order to enforce spatio-temporal consistency of the Hough votes for hypotheses generation.

To achieve time-scale invariance, the action tracks can in theory be either up- or down-sampled accordingly and the same Hough forest can then be applied to label actions at differing speeds. We note, however, that action speeds typically do not vary more than by a factor of two (disregarding framerate variations). Furthermore, the system has some tolerance built in through variation in speed of the training data. Therefore, in our current work, we found it unnecessary to apply the Hough forest at multiple time scales.

4.3.1 *Experiments*

We evaluated our system on six datasets, covering a variety of action recognition scenarios. The first two, Weizmann [78] and KTH [79], are popular benchmarks used in action recognition and consist of single persons performing actions in front of static backgrounds. Current state-of-the-art recognition systems have saturated the performance on these two datasets, but we include their evaluation for comparison purposes against other systems. We also evaluate our system on four more challenging datasets: the UCF sports dataset [80], the UCR Videoweb Activities Dataset [81], the UT-Tower Dataset [82], and the TUM Kitchen Dataset [83].

We evaluated our system's ability to apply the correct action label to a given video sequence and call this classification. Classification was measured with three variations of training and testing data: (*A*) training and testing on tracks generated from ground-truth annotations (*B*) training on tracks from ground truth and testing on automatically extracted tracks and (*C*) training and testing on automatically extracted tracks. We refer to these as data variations *A*, *B*, and *C*, respectively.

For the KTH and UCF sports dataset, we also evaluate the accuracy of detections in the automatically extracted action tracks and call this localization. The localization evaluation is the same as [59]; a detection is considered correct if (1) the action track that it belongs to was correctly classified and (2) the intersection-union ratio of the detection and ground truth bounding box is greater than 0.5. As a measure of localization, we present the average precision. For action recognition, we used the same six feature channels in all datasets: intensity, absolute value of x-, y- and time derivatives and the absolute value of the optical flow in the x- and y-direction.

Weizmann and KTH. The Weizmann dataset consists of 90 videos of nine actors performing ten different actions. Evaluations were done with a leave-one-out cross-validation. The KTH dataset consists of 599 videos of 25 actors performing six actions. Evaluations were done with a five-fold cross-validation, using 20 actors for training and five for testing. As each sequence lasts several hundred frames, we limited each sequence to only one or two cycles of the action in our evaluation.

Classification results for the three variations *A*, *B*, and *C* are shown in Table 4 and compared with the state-of-the-art. In addition, the confusion matrices for variation *B* are shown in Fig. 10. In this case, we report an average classification of 95.6% for Weizmann and 92.0% for KTH, both of which are comparable with state-of-the-art action recognition systems. The closest comparison is that of [60], in which a random forest of 50 trees were trained as a comparison against the "sphere/rectangle" trees; our performance, with only 5 trees in the random forest, is significantly higher and highlights the strength of the Hough-voting framework.

Localization results of each action in the dataset are presented in Table 5; both our method and the vocabulary forest method [59] achieve an average precision of

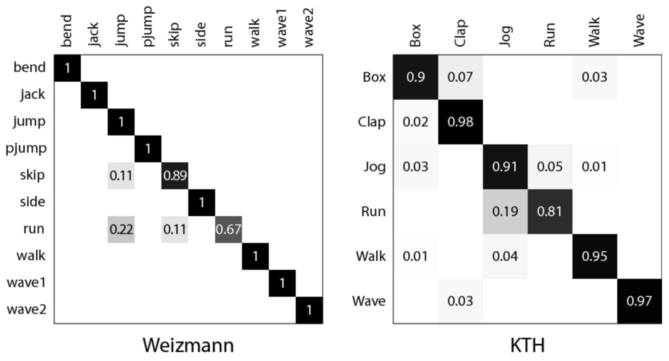


Fig. 10: Confusion matrices for Weizmann and KTH dataset using ground truth action tracks for training and automatically extracted action tracks for testing (data variation *B*).

Method	Weizmann	KTH
Hough forest (*A*)	97.8%	93.5%
Hough forest (*B*)	95.6%	92.0%
Hough forest (*C*)	92.2%	93.0%
voc. forest [59]	-	93.2%
SR tree [60]	-	90.3%
random forest [60]	-	72.9%
prototype tree [58]	100%	93.4%
temp. segment [84]	-	81.2%
Niebles et al. [13]	90.0%	83.3%
Schindler et al. [85]	100%	92.7%
Laptev et al. [12]	-	91.8
Liu et al. [11]	-	93.8%
Ommer et al. [86]	97.2%	87.9%

TABLE 4: Comparison of KTH and Weizmann classification with other methods. Results of all other methods presented are comparable with data variation *B*, with the exception of [85] (*A*) and [86] (*C*).

0.89 over all classes in the KTH dataset.

Broadcast Sports: UCF Sports. The UCF sports dataset is a collection of 150 broadcast sports sequences from network news videos. Evaluations were done with a five-fold cross-validation. Due to an unequal number of sequences in each action category, each fold consisted of approximately one-fifth of the total number of sequences per category.

Classification results over the three variations of training and testing data are shown in Table 6 and compared with the results reported from other methods. We outperform [80] and [87], and have comparable results with [32] and [57] despite our use of much simpler features (their best results were achieved using 3D-HOG descriptors). While the differences between the variants *B* and *C* are not significant, *B* and *C* perform worse than *A*. The difference can be explained by the accuracy of the tracker on UCF that is worse than on other datasets. The confusion matrix for variation *B* is shown in Fig. 11*(a,c)* along with some example classification results. There is some confusion between running and kicking, since kicking sequences often open with an individual running before kicking a ball. Similarly, walking and golfing sequences are also confused since several walking sequences are drawn from individuals walking on a golf course, suggesting that the trees take some context into account when splitting the cuboids.

Localization results for the UCF dataset are presented in Table 7; no other works at this time have published a

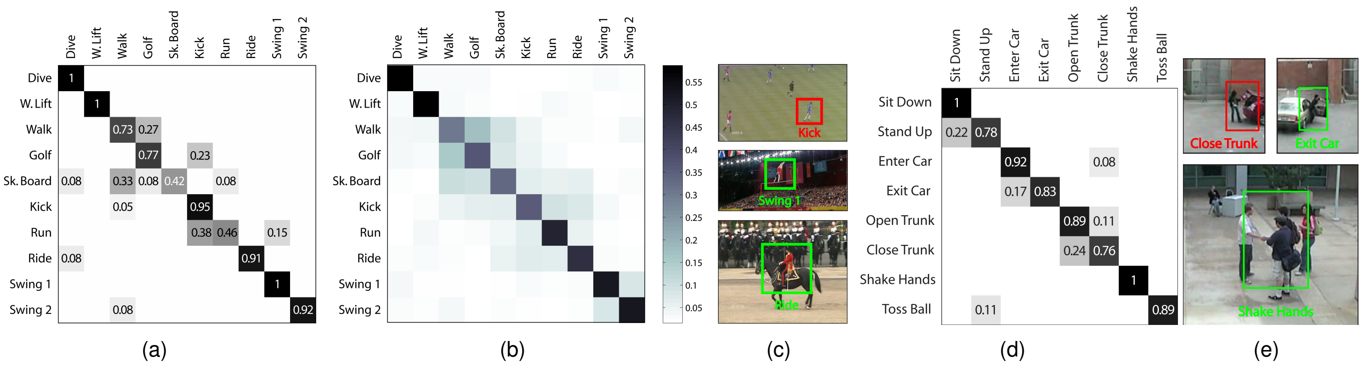


Fig. 11: *(a)* Confusion matrix for UCF sports dataset using data variation *B*. *(b)* Probability of feature sharing between action classes. *(c)* Example classifications. *(d)* Confusion matrix for Videoweb Activities dataset with some example classifications *(e)*.

Method	Box	Clap	Jog	Run	Walk	Wave
Hough forest	0.88	0.96	0.84	0.72	0.95	0.98
voc. forest [59]	0.98	0.97	0.79	0.78	0.86	0.96

TABLE 5: Comparison of KTH localization results

Method	Mean Performance
Hough forest (*A*)	86.6%
Hough forest (*B*)	81.6%
Hough forest (*C*)	79.0%
Rodriguez et al. [80]	69.2%
Yeffet & Wolf [87]	79.2%
Wang et al. [32]	85.6%
Kovashka & Grauman [57]	87.3%

TABLE 6: Comparison of UCF classification with other methods. Results of all methods presented are comparable with *B*, with the exception of Rodriguez et al. [80], which is comparable with *A*.

Class	Precision	Class	Precision
Dive	0.52	Kick	0.28
W.Lift	1	Run	0.37
Walk	0.67	Ride	0.66
Golf	0.77	Swing 1	0.44
Sk. Board	0.39	Swing 2	0.26

TABLE 7: UCF localization results

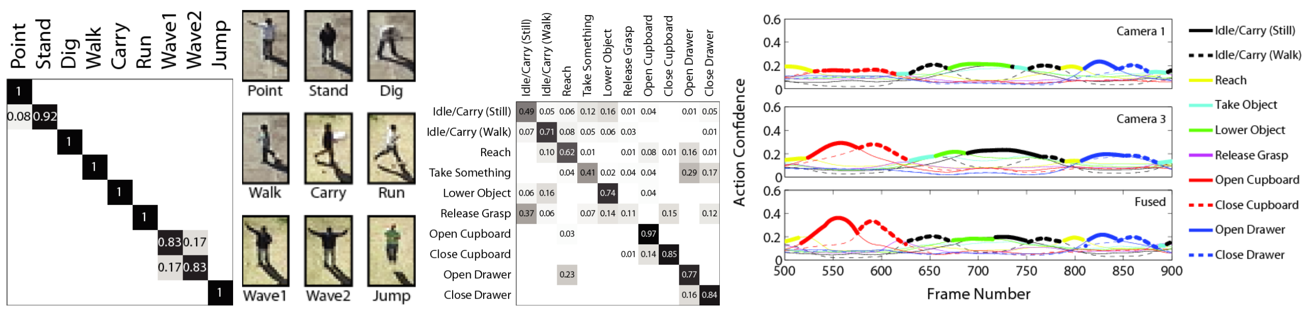
similar evaluation for comparison. Over all classes, we achieve an average precision of 0.54. The low average precision can be attributed to the fact that the ground truth annotations have changing aspect ratios, while we assumed a fixed aspect ratio when generating the action tracks. This is particularly relevant for the sports in which people have irregular and rapidly changing articulations, such as diving, kicking, and the swinging classes. Classification performance in these classes, however, are still very high as the fixed aspect ratio is sufficient to capture the action.

To illustrate the amount of feature sharing among classes, we passed the training cuboids $\mathcal{P}_i$ for a class c_i through the trees and averaged the probabilities for all classes c_j, i.e., $\frac{1}{N}\sum_{\mathcal{P}_i} p\big(c_j|\, L(\mathcal{P}_i)\big)$. The obtained matrix is shown in Fig. 11*(b)*. The diving and weightlifting classes are very distinct and share little to no features with other actions. On the other hand, the two gymnastics swing classes are very similar to (and only with) each other, and as such, share features with each other. There are also less distinct groupings, such as walking, golfing, skateboarding, and kicking, suggesting that both body position and context are accounted for in the feature sharing. For example, walking, golfing, and skateboarding all involve upright individuals with legs in relatively straight alignment with the body. On the other hand, several walking sequences are drawn from people walking on golf courses with green fields, which also resemble the soccer fields in the kicking sequences.

Surveillance: UCR Videoweb Activities. The UCR Videoweb Activities Dataset consists of about 2.5 hours of video from four to eight cameras in various surveillance scenarios. From this footage, we selected 110 sequences of eight actions that not only illustrate changes in body configuration (sitting down, standing up), but also interaction with the environment (entering and exiting a car, opening and closing a trunk), interaction with other people (shaking hands) and interaction with objects (tossing a ball). Evaluations were done with a five-fold cross-validation in the same manner as the UCF sports dataset. As our system handles only monocular views, we treat the same action instance recorded by different cameras as different sequences. As this is a newly released dataset, there are no other works with comparable results that we know of.

We achieve an average performance of 91.2%, 88.4%, and 92.2% for variations *A*, *B*, and *C*. The confusion matrix for the variation *B* is shown in Fig. 11*(d,e)*, together with some example classification results. As expected, there are some confusions between action pairings such as sit down/stand up, enter/exit car, and open/close trunk. This performance is remarkable considering the small size of the people in the surveillance sequences (typically 40 to 60 pixels high).

Aerial Footage: UT-Tower Dataset. The UT-Tower Dataset [82] is a collection of videos taken from the top of a 90 meter tall tower. There are 12 actors performing 9 actions and, due to the distant view, the average height of the actors is only around 20 pixels. Due to the low resolution, we tracked the individuals based on foreground masks included in the dataset and train and test as per variation *C*. We achieved an overall classification performance of 95.4%. The confusion matrix and sample images are shown in Fig. 12*(a,b)*. There is some confusion between similar actions such as *standing* and *pointing* or *wave1* and *wave2* but all other actions are classified correctly.



(a) (b) (c) (d)

Fig. 12: *(a,b)* Confusion matrix and sample images from UT-Tower Dataset. *(c)* Confusion matrix for fused results according to the max-rule for TUM Kitchen Dataset. *(d)* Normalized action confidences for two camera views as well as fused confidences for frames 500-900 of episode *0-11*.

In-house monitoring: TUM Kitchen Dataset. Although the action recognition system as described in 4.3 is meant only for monocular videos, we extended it for a multi-view scenario. A separate Hough forest is trained for each of the cameras in the multi-view setup; the output per view is a confidence score of each action class over time, normalized such that the confidences over all classes at any time point sum up to 1 (see Fig. 12*(d)*). A classifier combination strategy is then used to combine the outputs from the multiple views [88]. The motivation for fusing the single views is that actions which are ambiguous in one view, e.g., due to self-occlusion, may be more distinguishable from another view.

We apply the extended multi-view algorithm to the TUM Kitchen Dataset [83] that contains 20 episodes of recordings from 4 views of 4 subjects setting a table. The dataset is particularly challenging for action recognition as the actions are more subtle than those of KTH, Weizmann, UCF Sports, etc. In this dataset, the cameras are fixed and background subtraction was used to generate silhouettes of the person performing the action. Bounding boxes are then extrapolated around the silhouette and the trajectory of the bounding boxes is smoothed to build the track.

Training was done on episodes *1-0* to *1-5*, all of which are recorded from subject 1 and testing was done on episodes *0-2*, *0-4*, *0-6*, *0-8*, *0-10*, *0-11*, and *1-6*, which are recorded from all 4 subjects. For the action recognition, we use the 9 labels that are annotated for the 'left hand' [83] and further split the idle/carry class according to whether the subject is walking or standing.

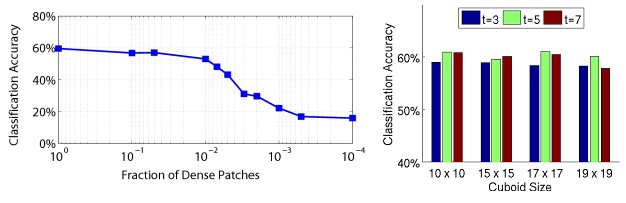
Results of the action recognition for the individual cameras as well as the fused results are shown in Table 8. For classifier fusion, we use the max-rule that gave the best performance compared to other standard ensemble methods [88], though results were similar for all the methods. The confusion matrix for the fused classifier is shown in Fig. 12*(c)*. Fig. 12*(d)* shows examples of action confidences for two single views and for the fused views.

Dense cuboid sampling. We have investigated our system's performance with respect to decreased cuboid sampling rates on the TUM dataset. In Fig. 13*(a)*, the average classification performance is plotted with respect to sampling density. Since we are using dense sampling in three dimensions, there is considerable overlap between cuboids. Performance does not drop until around one percent of the original sampling density but from this point onwards, the performance decrease is graceful. This shows that the amount of data processing can be reduced by a factor up to 100 for time-critical applications. The classification performance with respect to the cuboid size is shown in Fig. 13*(b)*.

On 100 frames of the KTH dataset, it takes around 10s to classify pre-existing action tracks with dense sampling and 170s to generate an action track.

	C1	C2	C3	C4	Fused
Subject 1	54.2%	49.3%	56.9%	56.4%	57.4%
Subject 2	53.2%	50.1%	45.6%	56.0%	58.5%
Subject 3	69.0%	71.8%	65.2%	66.6%	74.0%
Subject 4	61.9%	52.9%	61.0%	61.0%	70.6%
Average	59.6%	56.0%	57.2%	60.0%	65.1%

TABLE 8: Individual camera and fused action recognition performance for subjects 1-4; fused performance is higher than any individual camera view for each subject.



(a) Sampling density (b) Cuboid size

Fig. 13: *(a)* Performance decreases when cuboid sampling is reduced. At 1 (dense), 10^{-2}, 10^{-3}, and 10^{-4}, the average overlaps of two nearest cuboids are 91%, 56%, 39%, and 7%, respectively. *(b)* The performance is not very sensitive to the cuboid size.

5 Conclusion

We have presented a general *Hough forest* framework that can be applied to object detection, tracking, and action recognition. In our experiments, we have evaluated the performance on 6 datasets for object detection, 4 datasets for tracking, and 6 datasets for action recognition[3]. While the performance for detection and tracking is mainly compared to related methods, a thorough evaluation with comparison to the state-of-the-art is given for action recognition. Furthermore, we have shown that Hough forests handle multi-class/multi-aspect view problems,

3. All experiments are based on the available source code: www.vision.ee.ethz.ch/~gallju/projects/houghforest/houghforest.html.

share features among classes, and can be easily adapted on-line to a specific instance, which are generally desirable properties.

We conclude that Hough forests are a simple yet efficient tool for the three applications. Although they are not very specific to one single task, they perform well compared to the state-of-the-art for all three tasks. In particular, the performance on action recognition is impressive since many object detectors designed for images like sliding window cannot be easily extended to the spatio-temporal domain. Compared to cluster-based codebooks like [21], a significant improvement of the performance has been observed on all datasets. The boost in performance can be explained by the ability of the Hough forests to process a larger amount of training examples and sampling them densely. Another improving factor is the use of the displacement distribution of the sampled cuboids for supervision during training. More importantly, Hough forests not only outperform cluster-based codebooks for object detection, but also make new applications like action recognition feasible for Hough-based methods.

The present work has addressed the creation of codebooks for Hough-based detection, but not the detection scheme or any kind of post-processing. However, it is obvious that techniques like non-maxima suppression [44] can be combined with Hough forests to improve the detection results. Exploiting the relations between sliding window and Hough-based object detection [26] is another promising approach for improving the detection accuracy. One of the most important limitations of the current implementation seems to be the star-voting model, i.e., voting for the center of the object. The star model is very good for rigid objects but parts of deformable or articulated objects in rare poses are treated as parts with rare appearance. While the star model works still well for objects with limited pose variations like pedestrians, objects with high pose variations are difficult to detect without separating pose and appearance in a more principled way.

In general, Hough forests provide an excellent balance between high detection accuracy and time efficiency both at training and test time. Similar to random forests, it is expected that an implementation of a Hough forest on a GPU [34], [89] would give an extra significant speed-up.

Acknowledgments

This research has been supported by funding from the Swiss National Foundation NCCR project IM2, the SNF projects CASTOR (200021-118106) and Vision-supported Speech-based Human Machine Interaction (200021-130224), and the EC projects IURO and RADHAR. A. Yao is supported by funding from NSERC Canada. V. Lempitsky is supported by EU ERC grant VisRec no. 228180 and Microsoft Research projects in Russia. Part of this work was done while J. Gall and V. Lempitsky were with Microsoft Research Cambridge.

References

[1] H. Schneiderman and T. Kanade, "Object detection using the statistics of parts," *Int'l J. Computer Vision*, vol. 56, no. 3, pp. 151–177, 2004.

[2] P. Viola and M. Jones, "Robust real-time face detection," *Int'l J. Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2005, pp. 886–893.

[4] V. Ferrari, F. Jurie, and C. Schmid, "Accurate object detection with deformable shape models learnt from images," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[5] S. Maji, A. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1627–1645, 2010.

[7] P. Schnitzspan, M. Fritz, S. Roth, and B. Schiele, "Discriminative structure learning of hierarchical representations for object detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[8] C. Lampert, M. Blaschko, and T. Hofmann, "Efficient subwindow search: A branch and bound framework for object localization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2129–2142, 2009.

[9] P. Felzenszwalb, R. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[10] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *VS-PETS*, 2005.

[11] J. Liu, J. Luo, and M. Shah, "Recognizing realistic actions from videos 'in the wild'," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[12] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[13] J. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int'l J. Computer Vision*, vol. 79, no. 3, pp. 299–318, 2008.

[14] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *European Conf. Computer Vision*, 2008.

[15] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[16] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Int'l Conf. Computer Vision*, 2003, pp. 1470–1477.

[17] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on Statistical Learning in Computer Vision*, 2004, pp. 1–22.

[18] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman, "Learning object categories from google's image search," in *Int'l Conf. Computer Vision*, 2005, pp. 1816–1823.

[19] R. Duda and P. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.

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

[21] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *Int'l J. Computer Vision*, vol. 77, no. 1-3, pp. 259–289, 2008.

[22] J. Liebelt, C. Schmid, and K. Schertler, "Viewpoint-independent object class detection using 3d feature maps," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[23] S. Maji and J. Malik, "Object detection using a max-margin hough transform," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[24] A. Opelt, A. Pinz, and A. Zisserman, "Learning an alphabet of shape and appearance for multi-class object detection," *Int'l J. Computer Vision*, vol. 80, no. 1, pp. 16–44, 2008.

[25] B. Ommer and J. Malik, "Multi-scale object detection by clustering lines," in *Int'l Conf. Computer Vision*, 2009.

[26] A. Lehmann, B. Leibe, and L. Van Gool, "Fast prism: Branch and bound hough transform for object class detection," *Int'l J. Computer Vision*, 2010.

[27] B. Leibe and B. Schiele, "Interleaved object categorization and segmentation," in *British Machine Vision Conf.*, 2003, pp. 759–768.

[28] P. Yarlagadda, A. Monroy, and B. Ommer, "Voting by grouping dependent parts," in *European Conf. Computer Vision*, 2010.

[29] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Computation*, vol. 9, no. 7, pp. 1545–1588, 1997.

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

[31] F. Jurie and B. Triggs, "Creating efficient codebooks for visual recognition," in *Int'l Conf. Computer Vision*, 2005, pp. 604–610.

[32] H. Wang, M. Ullah, A. Kläser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *British Machine Vision Conf.*, 2009.

[33] H. Chen, T. Liu, and C. Fuh, "Segmenting highly articulated video objects with weak-prior randomforests," in *European Conf. Computer Vision*, 2006, pp. 373–385.

[34] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "PROST Parallel Robust Online Simple Tracking," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[35] J. Gall and V. Lempitsky, "Class-specific hough forests for object detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[36] J. Gall, N. Razavi, and L. Van Gool, "On-line adaption of class-specific codebooks for instance tracking," in *British Machine Vision Conf.*, 2010.

[37] A. Yao, J. Gall, and L. Van Gool, "A hough transform-based voting framework for action recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[38] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool, "Dynamic 3d scene analysis from a moving vehicle," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[39] J. Shotton, A. Blake, and R. Cipolla, "Multiscale categorical object recognition using contour fragments," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1270–1281, 2008.

[40] R. Okada, "Discriminative generalized hough transform for object dectection," in *Int'l Conf. Computer Vision*, 2009.

[41] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007," http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[42] E. Seemann, B. Leibe, and B. Schiele, "Multi-aspect detection of articulated objects," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[43] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, , and L. Van Gool, "Towards multi-view object class detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[44] O. Barinova, V. Lempitsky, and P. Kohli, "On the detection of multiple object instances using hough transforms," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[45] R. Marée, P. Geurts, J. Piater, and L. Wehenkel, "Random sub-windows for robust image classification," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2005, pp. 34–40.

[46] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in *Neural Information Processing Systems*, 2006.

[47] F. Schroff, A. Criminisi, and A. Zisserman, "Object class segmentation using random forests," in *British Machine Vision Conf.*, 2008.

[48] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[49] J. Winn and J. Shotton, "The layout consistent random field for recognizing and segmenting partially occluded objects," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2006, pp. 37–44.

[50] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2005, pp. 775–781.

[51] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2006, pp. 798–805.

[52] I. Laptev and T. Lindeberg, "Space-time interest points," in *Int'l Conf. Computer Vision*, 2003.

[53] G. Willems, J. Becker, T. Tuytelaars, and L. Van Gool, "Exemplar-based action recognition in video," in *British Machine Vision Conf.*, 2009.

[54] K. Rapantzikos, Y. Avrithis, and S. Kollias, "Dense saliency-based spatiotemporal feature points for action recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[55] L. Cao, Z. Liu, and T. Huang, "Cross-dataset action detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[56] P. Matikainen, M. Hebert, and R. Sukthankar, "Representing pairwise spatial and temporal relations for action recognition," in *European Conf. Computer Vision*, 2010.

[57] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[58] Z. Lin, Z. Jian, and L. Davis, "Recognizing actions by shape-motion prototype trees," in *Int'l Conf. Computer Vision*, 2009.

[59] K. Mikolajczyk and H. Uemura, "Action recognition with motion-appearance vocabulary forest," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[60] K. Reddy, J. Liu, and M. Shah, "Incremental action recognition using feature-tree," in *Int'l Conf. Computer Vision*, 2009.

[61] S. Agarwal, A. Awan, and D. Roth, "Learning to detect objects in images via a sparse, part-based representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1475–1490, 2004.

[62] C. Lampert, M. Blaschko, and T. Hofmann, "Beyond sliding windows: Object localization by efficient subwindow search," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[63] J. Mutch and D. Lowe, "Multiclass object recognition with sparse, localized features," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2006, pp. 11–18.

[64] L. Karlinsky, M. Dinerstein, H. Daniel, and S. Ullman, "The chains model for detecting parts by their context," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[65] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking." in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[66] E. Borenstein and S. Ullman, "Class-specific, top-down segmentation," in *European Conf. Computer Vision*, 2002, pp. 639–641.

[67] M. Andriluka, S. Roth, and B. Schiele, "Pictorial structures revisited: People detection and articulated pose estimation," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[68] E. Seemann and B. Schiele, "Cross-articulation learning for robust detection of pedestrians," in *Symp. Pattern Recognition*, 2006, pp. 242–252.

[69] J. Shotton, A. Blake, and R. Cipolla, "Efficiently combining contour and texture cues for object recognition," in *British Machine Vision Conf.*, 2008.

[70] L. Zhu, Y. Chen, A. Torralba, W. Freeman, and A. Yuille, "Part and appearance sharing: Recursive compositional models for multi-view multi-object detection," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[71] N. Razavi, J. Gall, and L. Van Gool, "Backprojection revisited: Scalable multi-view object detection and similarity metrics for detections," in *European Conf. Computer Vision*, 2010.

[72] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *European Conf. Computer Vision*, 1996, pp. 343–356.

[73] A. Doucet, N. De Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*. New York: Springer, 2001.

[74] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *British Machine Vision Conf.*, 2006, pp. 47–56.

[75] "Imagery library for intelligent detection systems i-lids," http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html.

[76] J. Ferryman, J. Crowley, and A. Shahrokni, "Pets 2009 benchmark data," http://www.cvg.rdg.ac.uk/PETS2009/a.html.

[77] A. Yao, D. Uebersax, J. Gall, and L. Van Gool, "Tracking people in broadcast sports," in *Symp. Pattern Recognition*, 2010.

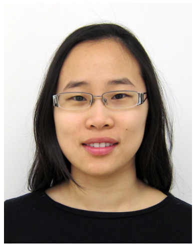
[78] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *Int'l Conf. Computer Vision*, 2005.

[79] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Int'l Conf. Pattern Recognition*, 2004.

[80] M. Rodriguez, J. Ahmed, and M. Shah, "Action mach a spatio-temporal maximum average correlation height filter for action

recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[81] "Ucr videoweb activities dataset," http://vwdata.ee.ucr.edu.
[82] C.-C. Chen, M. Ryoo, and J. Aggarwal, "UT-Tower Dataset: Aerial View Activity Classification Challenge 2010," http://cvrc.ece.utexas.edu/SDHA2010/Aerial_View_Activity.html.
[83] M. Tenorth, J. Bandouch, and M. Beetz, "The TUM kitchen data set of everyday manipulation activities for motion tracking and action recognition," in *IEEE Workshop on THEMIS*, 2009.
[84] A. Oikonomopoulos, I. Patras, and M. Pantic, "An implicit spatiotemporal shape model for human activity localization and recognition," in *Human Communicative Behavior Analysis*, 2009.
[85] K. Schindler and L. Van Gool, "Action snippets: How many frames does human action recognition require?" in *IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[86] B. Ommer, T. Mader, and J. M. Buhmann, "Seeing the objects behind the dots: Recognition in videos from a moving camera," *Int'l J. Computer Vision*, vol. 83, pp. 57–71, 2009.
[87] L. Yeffet and L. Wolf, "Local trinary patterns for human action recognition," in *Int'l Conf. Computer Vision*, 2009.
[88] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, 1998.
[89] T. Sharp, "Implementing decision trees and forests on a GPU," in *European Conf. Computer Vision*, 2008, pp. 595–608.

Juergen Gall obtained his B.Sc. and his Master's degree in mathematics from the University of Wales Swansea (2004) and from the University of Mannheim (2005). He was an intern with the Machine Learning and Perception group at Microsoft Research Cambridge (2008). In 2009, he obtained a Ph.D. in computer science from the Saarland University and the Max-Planck-Institut für Informatik. Since 2009, he is a postdoctoral researcher at the Computer Vision Laboratory, ETH Zurich. His research interests include interacting particle systems, markerless human motion capture, object detection, and action recognition.



Angela Yao received the BASc degree in Engineering Science from the University of Toronto in 2006 and the MSc degree in Biomedical Engineering from ETH Zurich in 2008. She is currently a PhD candidate at the Computer Vision Laboratory at ETH Zurich, where she works on topics related to human motion analysis, such as tracking, pose estimation and action recognition.

Nima Razavi received his BSc from the Sharif University of Technology (2006) and MSc from ETH Zurich (2008) where he is currently working towards his PhD. His research interests include object detection and tracking, scene understanding, and neuropsychology.

Luc Van Gool got a degree in electromechanical engineering at the Katholieke Universiteit Leuven in '81. Currently, he is professor at the Katholieke Universiteit Leuven in Belgium and the ETH in Zurich, Switzerland. He leads computer vision research at both places, where he also teaches computer vision. He has authored over 200 papers in this field. He has been a program committee member of several major computer vision conferences. His main interests include 3D reconstruction and modeling, object recognition, tracking, and gesture analysis. He received several Best Paper awards. He is a co-founder of 5 spin-off companies.

Victor Lempitsky is a postdoc researcher in computer vision at the Visual Geometry Group at the University of Oxford. Prior to that he was a postdoc with the Computer Vision Group at Microsoft Research Cambridge. Victor holds a PhD degree in applied mathematics (2007) from Moscow State University. His research interests are in visual recognition and analysis applications for photographs and biomedical images. He has published over 15 papers at major computer vision conferences and journals. He also received a best paper award at the international symposium on Functional Imaging and Modelling of the Heart (FIMH) in 2009.