Structured Forests for Pixel-level Hand Detection and Hand Part Labelling

Xiaolong Zhu[a,*], Xuhui Jia[a], Kwan-Yee K. Wong[a]

[a] *Department of Computer Science, Chow Yei Ching Building*
The University of Hong Kong, Pokfulam Road, Hong Kong

Abstract

Hand detection has many important applications in Human-Computer Interactions, yet it is a challenging problem because the appearance of hands can vary greatly in images. In this paper, we present a new approach that exploits the inherent contextual information from structured hand labelling for pixel-level hand detection and hand part labelling. By using a random forest framework, our method can predict hand mask and hand part labels in an efficient and robust manner. Through experiments, we demonstrate that our method can outperform other state-of-the-art pixel-level detection methods in ego-centric videos, and further be able to parse hand parts in details.

Keywords: Hand Detection, Egocentric Vision, Random Forests, Hand Part Labelling

1. Introduction

Hand detection has many important applications in Human-Computer Interactions. It enables computers to consider the flexible movement of human hands in 3D space as a new type of high dimensional user input, and to understand the natural interaction of hands with other objects in various scenarios. However, hand detection is a challenging problem because the appearance of hands can vary greatly in images. For instance, the shape of a hand can change

[*] Corresponding author
Email address: `lucienxlzhu@gmail.com` (Xiaolong Zhu)

dramatically due to the articulation of fingers as well as changes in viewpoint. A hand can be (partially) occluded while interacting with other objects. The colour of a hand can vary greatly under different illuminations, and a hand can even appear to be textureless under extreme illuminations. Traditional Methods [1, 2, 3, 4] based on gradients or skin detection often cannot handle practical unconstrained hand images well due to insufficient training data. Furthermore, ego-centric cameras have become more and more popular. Images captured by such cameras often have a dynamic background, which makes hand detection even more difficult. Nonetheless, hands play a major part in these images, and it is of great interest and importance to detect hands in detail robustly for further higher level analysis.

In this paper, our goal is to improve pixel-level hand detection and hand part labelling within the random forest framework. Rather than predicting per-pixel labels independently as in [5], we aim at exploiting the inherent structure from the label output space and predicting a patch region, which corresponds to a binary shape mask in hand detection and a multi-class label patch in hand part labelling. Technically, our approach is inspired by Semantic Texton Forests [27] and recent work on semantic image labelling [28]. During their training process, only limited number of pixels of a patch were considered in the split function. In order to consider more pixels, we propose to use an intermediate mapping, which groups the training patches for each node into certain amount of clusters by means of unsupervised learning methods. As shown in Figure 1, our method detects hand regions more robustly than previous methods and is able to parse a hand into different parts.

Our proposed approach has the following contributions:

- We explicitly model the labelling of a pixel together with its local neighborhood as a structured output to better utilize the inherent topological information in the training data and enforce such information as constraints during estimation;

- A novel structured split criterion is proposed to enable an efficient train-

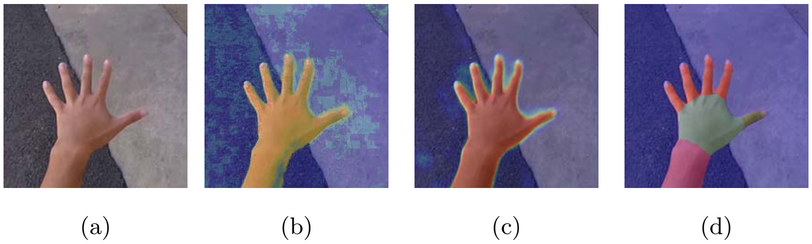


Figure 1: Introduction to our method. (a) Original image. (b) Pixel-level hand detection by single pixel prediction. (c) Pixel-level hand detection by structured mask prediction. (d) Hand part labelling by structured label prediction.

ing and consider more pixels of our structured forests by incorporating unsupervised learning methods;

- We extend the binary hand detection to multi-class hand part labelling within our unified framework to solve these problems in an efficient and robust manner;
- Throughout the experiments, our method outperforms the state-of-the-art methods. We also present a comprehensive analysis on different factors affecting the performance of our method on both tasks.

Next, we briefly review related work on pixel-level hand detection in Section 2. In Section 3, we describe our proposed structured forests for hand detection. In Section 4, we extend our structured forests to handle more general output and apply them to hand part labelling. In Section 5, we show the experimental results for both hand detection and hand part labelling. Finally, we conclude our method in Section 6.

1.1. Literature Review

For many years, hand detection has been studied as a part of gesture analysis and human layout parsing. Early efforts in detecting human hand from a colour image usually considered skin-colour as the major cue [2, 3, 6] to build a model of the hand region in colour space. Mixture of Gaussians [7] was commonly used to model colours of skin and non-skin regions for hand localization [8] and hand tracking [9, 10]. As these methods often require a priori knowledge of skin colour, extracted either from training data or from face detection, to build the skin model, they cannot obtain robust results when they are applied to a novel scene or when illumination changes cast a large variation in colour.

In the mean time, inspired by the great progress in object detection and recognition, a few works directly modeled the appearance of hands with a generic object detection framework. Features could be extracted from a number of training images to train a Viola & Jones-like boosted detector [1, 11, 12] or an HOG-SVM detector [4, 13], which can be viewed as a hand template representation. A hand template could also be learned as an ensemble of edges [14, 15] from a set of $2D$ projections of a $3D$ synthetic hand model. Furthermore, colour information can be used to further create more proposals to improve the detection performance [4]. However, the applications of these methods are limited to a small number of hand configurations. They often need to exploit more training data in order to cover a larger configuration space. Alternatively, hands can be detected as part of a human pictorial structure [16], which may bring more context information and allow inferring hand position via optimization. This is a common practice for still images, but it usually requires at least the upper body being visible for the inference of human layout.

When it comes to videos, motion-based methods can be used for *ad-hoc* applications, such as activity analysis and gesture recognition. They segment foreground hands from background by motion and appearance cues [17, 18, 19]. Hands can usually be tracked easily and they do not require a strong appearance model in most cases. Nevertheless, motion-based methods are often not suitable for moving cameras which produce images with lots of background motion.

Recently, ego-centric cameras, such as Google Glass and GoPro cameras, have become more and more popular. A local-appearance-based pixel labelling method recently proposed by Li and Kitani [5] has shown to be quite successful in dealing with dynamic background and varying appearance of hands in ego-centric videos. However, their method only predicts the label of every pixel independently without considering any shape constraint. To deal with the noisy output, segmentation is required to optimize the shape of hand region [20, 21].

Often hand detection is only the first step in hand gesture analysis. It is of great interest to further recognize the hand parts in detail. Following the great success of image labelling for human pose estimation [22], hand part labelling becomes one of the most investigated fields, especially for depth images. In this line of works [23, 24, 25, 26], recognizing hand parts are considered as an intermediate step for subsequent articulated hand joints estimation. As it is easy to synthesize hand depth images using graphics techniques, there is a lot of priori knowledge, *e.g.*, hand joint position, hand orientation, that can be used to customize the construction of a per-pixel random forest classifier. Such information, however, is not available in conventional colour images. This makes hand part labelling in colour images not well investigated. On the other hand, the progress of semantic labelling [27, 28] enriches us with more possible ways to exploit per-pixel labels for prediction. This encourages us to fill the blank of hand parts labelling in colour images.

2. Random Decision Forests for Hand Detection

In this section, we begin with a review of random decision forests for pixel-level hand detection, and introduce some notations used in pixel-level hand detection settings.

Given an image patch $\mathbf{I_p} \in \mathbb{R}^{w \times w \times 3}$ with a size of $w \times w$ centered at pixel $\mathbf{p} \in \mathbb{Z}^2$ in a colour image $\mathbf{I}$, a feature vector $\mathbf{x_p} \in \mathcal{X}$ is extracted to encode the colour, gradient and texture information of this patch. A binary decision tree $f_{\mathbf{\Theta}}(\mathbf{x_p})$, parameterized by $\mathbf{\Theta}$, is a tree-structured classifier that maps $\mathbf{x_p}$ to a binary label $y_{\mathbf{p}} \in \{0, 1\}$, which indicates whether the pixel $\mathbf{p}$ belongs to a hand (i.e., $y_{\mathbf{p}} = 1$) or not (i.e., $y_{\mathbf{p}} = 0$). The feature sample $\mathbf{x_p}$ is recursively branched left or right down through the tree. In Node j, this process is done according to a split function with parameter $\boldsymbol{\theta}_j$

$$\Phi(\mathbf{x_p}, \boldsymbol{\theta}_j) = \begin{cases} 1, & \text{if } \boldsymbol{\theta}_j^\top [\mathbf{x_p}^\top \ \ 1]^\top \leq 0 \\ 0, & \text{otherwise} \end{cases}, \tag{1}$$

where 1 means $\mathbf{x_p}$ belongs to the left child of Node j while 0 means to right. When the sample reaches a leaf node, the posterior distribution $P(y_\mathbf{p})$ stored in that leaf will be associated to the sample for prediction.

A decision forest is an ensemble of T decision trees, each with independent parameters $\mathbf{\Theta}_i$. Given the feature sample $\mathbf{x_p}$, the output of the decision forest $F(\mathbf{x_p})$ is the final class label $y^*_\mathbf{p}$, which is obtained using an ensemble model of the posterior distributions $P_i(y_\mathbf{p}|\mathbf{x_p})$ in the leaf node of tree i as,

$$y^*_\mathbf{p} = \arg\max_{y_\mathbf{p}} \frac{1}{T}\sum_{i=1}^{T} P_i(y_\mathbf{p}|\mathbf{x_p}). \tag{2}$$

2.1. Training Decision Forests

During the training process, each decision tree is constructed independently from a randomly sampled subset of the training set $S \subseteq \mathcal{X} \times \mathcal{Y}$ in a recursive manner. For a node j with a set of training data $S_j \subset S$, there are several randomly generated candidates of $\boldsymbol{\theta}_j$ and the goal is to find a candidate that maximizes the information gain, $\mathbf{G}(\boldsymbol{\theta}_j)$, of the current split test. The information gain is defined as

$$\mathbf{G}(\boldsymbol{\theta}_j) = H(S_j) - \sum_{k\in\{L,R\}} \frac{|S_j^k|}{|S_j|} H(S_j^k), \tag{3}$$

where $S_j^L = \{(\mathbf{x_p}, y_\mathbf{p}) \in S_j | \Phi(\mathbf{x_p}, \boldsymbol{\theta}_j) = 1\}$ denotes the set of training data to be assigned to its left child and $S_j^R = S_j \setminus S_j^L$ denotes the set of training data to be assigned to its right child, $|\cdot|$ denotes the size of a set and $H(\cdot)$ denotes purity measurement *w.r.t.* $y_\mathbf{p}$. As in our formulation, $y_\mathbf{p}$ is a binary variable, the purity can be measured by Shannon Entropy or Gini impurity [29]. The decision tree is constructed by splitting the training data in its nodes repeatedly until either the minimum number of training data in a leaf node or the maximum depth of a tree is reached. In a leaf node, a posterior distribution $P(y_\mathbf{p})$ is often built by calculating the frequencies of each class and kept in the leaf for future prediction. An alternative is to select the most represented class of the training samples routed to that leaf [30].

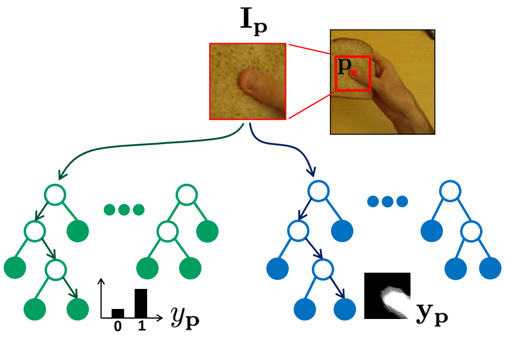


Figure 2: The main difference between random forests and structured forests.

3. Learning Shape Masks in Random Forests

In the traditional random forest framework, each input image patch is assigned with a single label to its central pixel. It does not consider any dependency among all of the pixel in that patch. In fact, if these labels are considered as a whole, they exhibit an inherent structure, which just forms the shape mask of the hand. Nevertheless, the standard random forests cannot fully exploit such interdependencies of the pixels during training and usually lead to a noisy result. It suggests that a better treatment is to learn a shape mask instead of just a single pixel label for an image patch to overcome the limitation of the original random forests, and explicitly utilize the inherent shape information in the training data.

In this section, we propose to augment the original output space to a structured label space and learn these shape masks within the random forest framework. We refer to our method as *Shape-aware Structured Forests* to highlight the major difference compared with standard random forests.

3.1. Shape Masks as Structured Output

Given an image patch $\mathbf{I_p}$, our structured forests again take its feature vector $\mathbf{x_p}$ as input, but the original output space $y_{\mathbf{p}} \in \{0, 1\}$ is extended to a shape mask space $\mathbf{y_p} \in \mathcal{Y} = \{0, 1\}^{w \times w}$. Each decision tree of the structured forests now maps $\mathbf{x_p}$ to a shape mask following the same routing procedure as in standard decision forests. In each leaf node, the posterior distribution of the mask $P(\mathbf{y_p})$ is stored instead of that of the central pixel. However, as the number of the states is exponential to the size of the image patch $\mathbf{I_p}$, we approximate the posterior distribution by the product of its marginal distributions over each pixel. As a result, each leaf node stores the marginal distribution of each pixel for efficiency.

For each tree i, a patch will pass through several binary tests until a leaf node is reached. In the leaf node, a posterior distribution of a mask is stored as a per-pixel posterior $m_i(x, y)$ at (x, y) as illustrated in Figure 2 along with a per-pixel variance $\sigma_i(x, y)$. The output of the structured forest is defined as the weighted average of all these posteriors,

$$m^*(x, y) = \frac{1}{Z}\sum_{i=1}^{T} e^{-k\sigma_i(x,y)} m_i(x, y). \tag{4}$$

where $Z = \sum_i e^{-k\sigma_i(x,y)}$ is a normalization term, and k is a parameter for tuning the weights. If $k = 0$, Z will become the total number of trees T and $m^*(x, y)$ will simply be the average of all the posteriors. Note that each pixel will receive predictions from multiple input patches covering it. The final prediction depends on both the prediction of structured forests and the fusion of predictions from multiple patches. Due to efficiency concern, we turn to a simple fusion method by assuming all the predictions are uncorrelated and calculating the marginal distribution of each pixel from these predictions as the final fusion result. Moreover, it does not need to evaluate every pixel with its local neighborhood image patch because a pixel can receive predictions from its neighboring pixels as long as those corresponding patches cover it. Therefore, a stride-based approach can be also used to accelerate the evaluation on the whole image. For a $w \times w$ patch with stride width d, each pixel receives approximately w^2T/d^2 predictions as compared to T predictions from standard decision forest, which makes prediction more robust in practice.

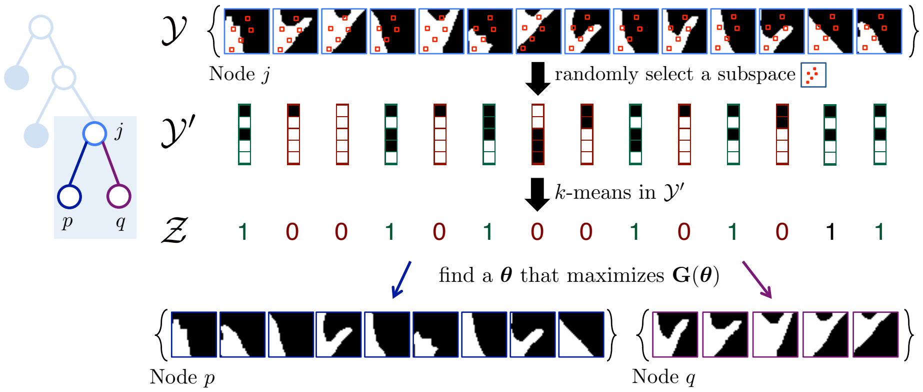

$\mathcal{Y}$
Node j
randomly select a subspace
$\mathcal{Y}'$
k-means in $\mathcal{Y}'$
$\mathcal{Z}$
1 0 0 1 0 1 0 0 1 0 1 0 1 1
find a $\boldsymbol{\theta}$ that maximizes $\mathbf{G}(\boldsymbol{\theta})$
Node p
Node q
j
p
q

Figure 3: Intermediate mapping during splitting in Node j. In the parent node, all mask images are firstly grouped into two clusters. Next, $\boldsymbol{\theta}_j$ for $\mathcal{X}$ is selected to maximize $G(\boldsymbol{\theta}_j)$ calculated from cluster labels in $\mathcal{Z}$.

3.2. Training Forests by Intermediate Mapping

However, it will be extremely time consuming to calculate information gain in the new shape mask space as we need to enumerate all possible states. For instance, there will be $2^{16\times16}$ possible states for a patch with a size of 16×16. In order to speedup the calculation of information gain, an intermediate mapping is used during training to approximate the mask space by a lower dimensional space as illustrated in Figure 3. During node splitting process, a subspace $\mathcal{Y}'$ is first randomly selected from the original mask space $\mathcal{Y}$ so that additional randomness can be injected into forest training to ensure the diversity of trees. This can be done by randomly selecting m elements from the mask $\mathbf{y}_p$. Next, k-means algorithm is performed over the subspace $\mathcal{Y}'$ to group the training masks into 2 clusters, so that either Shannon Entropy or Gini Impurity can be used to compute information gain of a candidate split test. Finally, a standard procedure like in decision forest training is applied to find an optimal split parameter $\boldsymbol{\theta}_j$ for the current node j.

3.3. Multi-scale Hand Detection

Normally, the hand size in training samples does not vary much or is preprocessed to be constrained within a certain range. In order to detect hands of

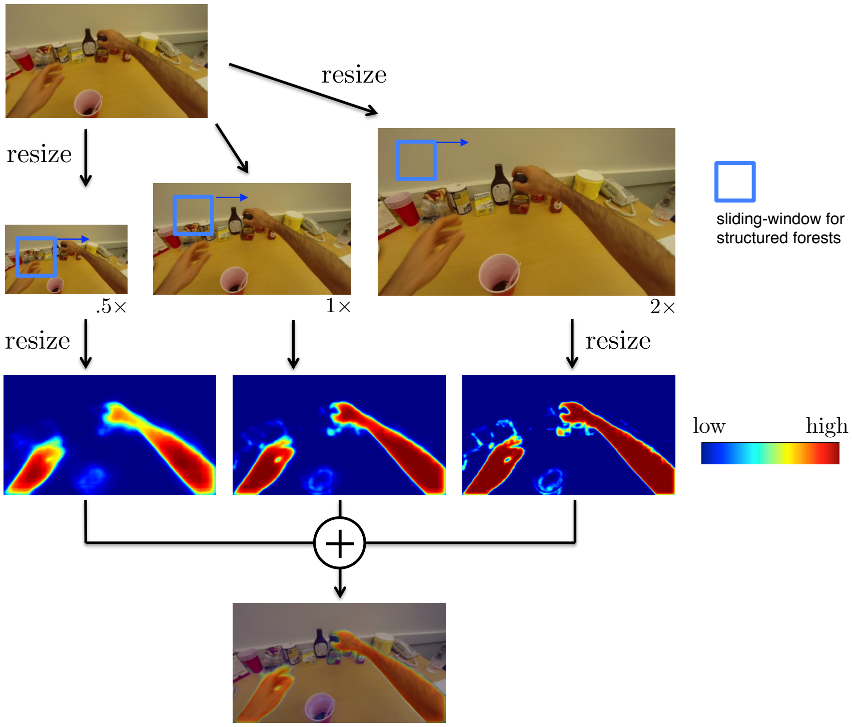


Figure 4: Hand detection in multiple scales. An input image is resized into several copies before a sliding-window of our structured forests predictor is applied to the image in each scale. The subsequent probability maps are then rescaled to the original size of the input image to be aggregated as the final output.

different sizes in a new image, we need to apply our model in multiple scales. Similar to [31], we adopt an image pyramid-based detection framework as illustrated in Figure 4. We first rescale the input image $\mathbf{I}$ to form an image pyramid. In each level, colour and gradient features are extracted as channel features [32]. In particular, we extract *CIELUV* channels as per-pixel colour features, which have been shown to be the most discriminative colour features in many applications [33, 5, 34]. As for per-pixel gradient features, we simply extract the magnitude and the orientation for each pixel and split its orientation into 9 channels followed by Gaussian smoothing among all bins. In order to describe the texture of a hand, we also include self-similarity features [35], which are pairwise differences among cells that subdivide a patch into tiles. This will also help to differentiate hand and non-hand regions in the patches. As all features are either channel features of order 1 or pairwise features of order 2, a lookup table can be built so that feature extraction can be done very efficiently during the test phase.

After the features are extracted, a sliding window of size $w \times w$ with stride width d is used to apply our structured forests and the final probability map of pixel-level hand detection is obtained by averaging the results over different scales and overlapping windows.

4. Extending Shape Masks to Hand Parts

The hand shape mask $\mathbf{y_p}$ consists of a set of binary labels indicating whether a pixel belongs to a hand or not. In this section, we further extend these binary labels to multi-class labels indicating which hand part a pixel belongs to.

More formally, a shape mask $\mathbf{y_p}$ in original mask space $\mathcal{Y}$ can be further extended to a label patch $\mathbf{l_p}$ in the new structured label space $\mathcal{L} = \{0, ..., C\}^{w \times w}$, where every pixel is assigned with a class label $l \in \{1, ..., C\}$ or a background label 0. As a consequence, the problem can be further extended to a semantic image labelling of hand parts. For a given training set $S \subset \mathcal{X} \times \mathcal{L}$, we can construct structured forests to map a feature vector $\mathbf{x_p}$ to a label patch $\mathbf{l_p}$ by making some customization.

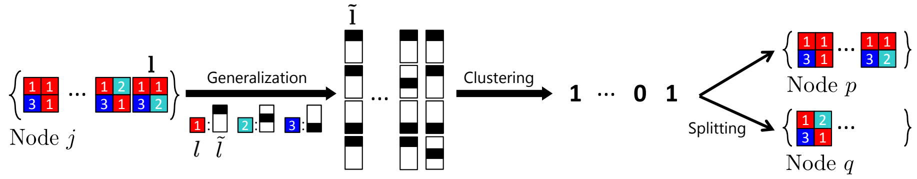


Figure 5: Intermediate mapping for hand part patches. During node splitting, linear generalization for categorical labels is done before these sample patches are clustered in the promoted output space, and serve for information gain calculation.

4.1. Intermediate Mapping for Hand Part Labels

Unlike in [28], where a k-label joint distribution is directly used for information gain calculation, we still use a two-step intermediate mapping in order to better utilize the output space. However, as the hand part label $l_{\mathbf{p}}$ of pixel $\mathbf{p}$ becomes a categorical variable, k-means algorithm cannot be directly applied during the clustering step. Instead, inspired by generalizing linear binary classifiers [36], we loose the class label $l \in \{0, 1, ..., C\}$ to a $(C+1)$-dimensional binary vector $\tilde{l}$ with only one dimension being 1 as illustrated in Figure 5. The label space $\mathcal{L}$ is promoted to a higher dimensional space $\tilde{\mathcal{L}} = \{0, 1\}^{(C+1)\times w\times w}$ so that k-means algorithm can be applied to find the "mean" joint class probability of the pixels in the patch. After the clustering is done, the information gain is calculated over subsequent two clusters in test function similar to previous section.

Compared with [28], we do not need to specify k and select optimal k-labels to separate the training samples at current node. Instead, unsupervised learning is used to split current subspace of $\mathcal{L}$ and fit the training subsets autonomously. In this way the samples can be coarsely separated in higher levels with more labelling information considered and finely clustered in deeper nodes. In Figure 6, we show that the training samples that reaches the leaves are semantically coherent by our method.

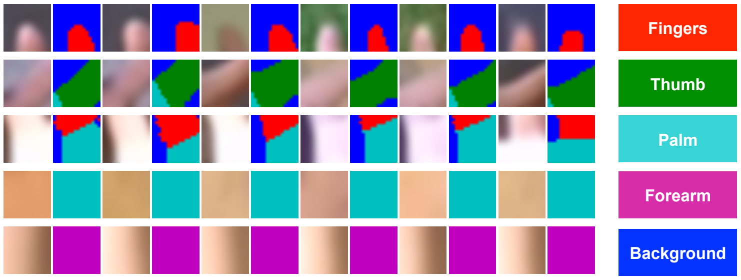


Figure 6: The examples of image patches with corresponding label patches collected from different leaf nodes (each row) in EDSH Dataset.

4.2. Selecting Prototypes in Leaf Nodes

When a tree grows to its leaf node t by reaching the stopping criteria with $D_t \subseteq \mathcal{L}$ as the set of hand label samples in the training data, we store a single label patch as the most represented patch of D_t to approximate the true posterior like in [28]. This can be done by choosing the label patch $\mathbf{l}_{\mathbf{p}}^{t*}$ that maximizes the joint class probability $P(\mathbf{l})$ at node t,

$$\mathbf{l}_{\mathbf{p}}^{t*} = \arg\max_{\mathbf{l}_{\mathbf{p}}^{t} \in D_t} P(\mathbf{l}_{\mathbf{p}}^{t}) \approx \arg\max_{\mathbf{l}_{\mathbf{p}}^{t} \in D_t} \prod_{\mathbf{u} \in \mathcal{N}(\mathbf{p})} P(l_{\mathbf{u}}^{t}), \qquad (5)$$

where $\mathcal{N}(\mathbf{p})$ is the local neighborhood of pixel $\mathbf{p}$ and the marginal distribution $P(l_{\mathbf{u}}^{j})$ is obtained by calculating the frequencies of each semantic class of pixel $\mathbf{u}$. As a consequence, we use a prototype to represent all samples in current leaf node in order to minimize computation cost while enforcing a proper constraint for prediction.

Once a forest is trained, each pixel label can be predicted by aggregating the multiple predictions in its local neighborhood and choosing its most probable class by the forest.

5. Experimental Results

We first tested our structured hand detector on ego-centric videos and depth images. Next we explored the hand part labelling in manually labelled ego-centric images. In ego-centric videos, fine shape is always needed for further high-level analysis, such as action analysis or object recognition. We compared our methods with state-of-the-art methods on these images and analyzed different factors that may affect the detection performance, and then evaluated the performance of our method on hand part labelling.

5.1. Hand Detection in Ego-centric Videos: GTEA and EDSH dataset

We first evaluated our detector on the Geogia Tech Ego-centric Activity dataset (GTEA) [18]. The GTEA dataset involves little camera motion and is taken under the same environment as it is primarily recorded for activity

	Baseline	Li & Kitani [5]	Serra *et al.* [38]	Ours
GTEA-Coffee	78.05	88.4	-	**90.19**$_{\pm 1.07}$
GTEA-Tea	72.53	**87.3**	-	84.30$_{\pm 1.12}$
GTEA-Peanut	74.71	81.5	-	**84.37**$_{\pm 2.11}$
EDSH2	72.31	78.1	78.9	**80.43**$_{\pm 3.10}$
EDSH-Kitchen	74.37	80.8	83.1	**92.11**$_{\pm 1.41}$

Table 1: Comparison on different pixel-level hand detection methods in GTEA and EDSH dataset. The F-scores of the baseline random forests, Li and Kitani's approach with feature selection, Serra *et al.*'s approach before temporal smoothing and our approach are listed in Column 2-5 respectively.

recognition. Similar to the experimental setup in [5], all video clips were firstly down-sampled to 640 × 360. There are 7 actions for 4 subjects, one of which, Subject 1 (S1), the ground truth labels are available for evaluation. The original hand masks are quite noisy and sometimes confused with the objects in hand due to unsatisfactory segmentation. We turned to the masks made available in [5] obtained using GrabCut [37]. We performed three experiments where we used Coffee sequence for training and Tea and Peanut sequences respectively for testing, and used Tea sequence for training and Coffee sequence for testing.

We also compared our approach on a publicly available EDSH dataset [1], which involves more illumination changes and camera motion. EDSH1 and EDSH2 recorded both hands of a subject walking through different indoor and outdoor scenes in order to capture the changes in skin colour. EDSH-Kitchen recorded a subject performing different activities in a kitchen, where there were great ego-motion and hand deformations. These are typical scenarios for hand detection in daily life and all these videos were recorded in 640 × 360, and 442 labelled frames were used for training our shape-aware structured forests.

[1] http://www.cs.cmu.edu/~kkitani/perpix/

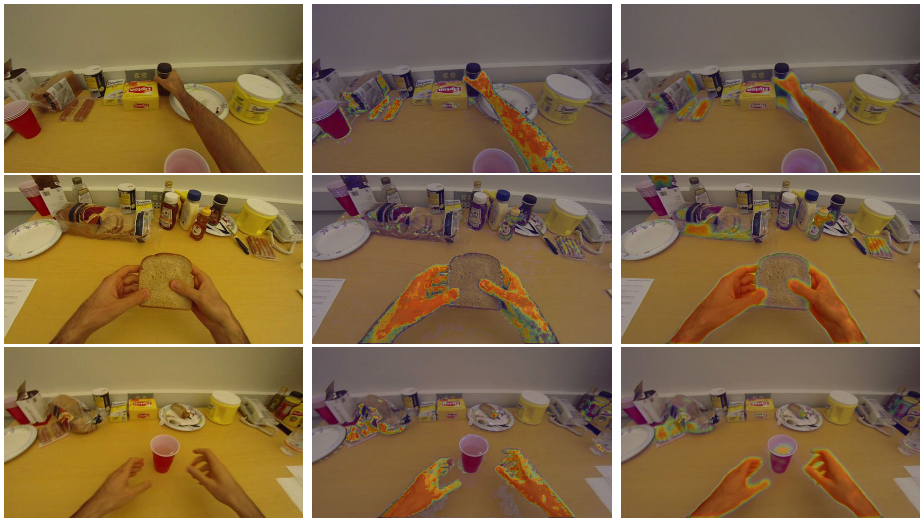


Figure 7: Sample images of GTEA dataset: (left column) original images, (second column) results of Li and Kitani's Method [5], and (last column) our results. From top to bottom: GTEA-Coffee, GTEA-Peanut and GTEA-Tea. Best viewed digitally at high zoom.

5.1.1. Comparison

We performed the same experiment using standard random forests for single pixel prediction based on 9×9 patches using colour and HOG features as baseline method. Furthermore, we also included Li and Kitani's results after feature selection for all 5 experiments [5] and Serra *et al.*'s results before using temporal information [38]. The average F-score, *i.e.*, harmonic mean of precision-recall rate, over all test images was used to measure the detection performance. The results are shown in Table 1. Our approach outperformed the baseline method in all 5 experiments. The improvement mainly happened in some cluttered regions because our method can filter out the noise and also smooth the prediction of hand region by averaging.

Figure 7 and Figure 8 show some sample images overlaid by per-pixel probabilities produced by the public code in [5] and our method in GTEA and EDSH dataset.

In Figure 7, we find that single pixel hand prediction always fails at the edge of hand and fingers. This is because the local neighborhood of these pixels

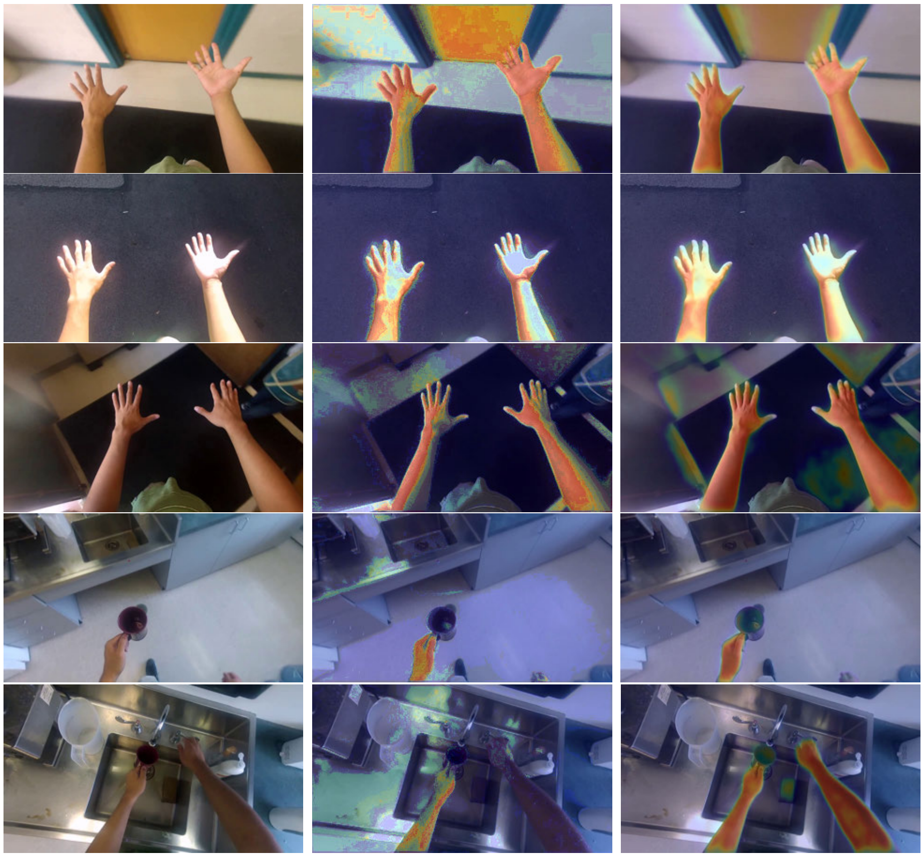
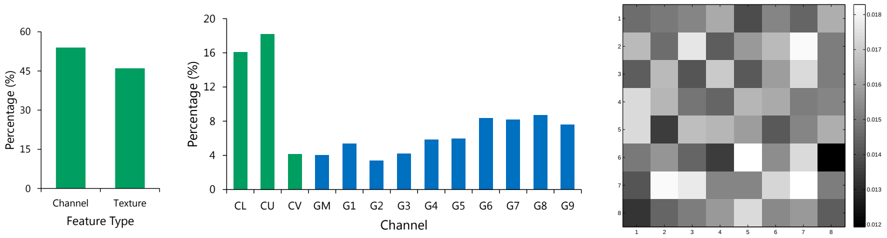


Figure 8: Sample images of EDSH data set: (left column) original images, (second column) results of Li and Kitani's Method [5], and (last column) our results. From top to bottom: confusion with the door in EDSH1; shadow on the hand in outdoor in EDSH2; poor light condition in EDSH2; motion blur in EDSH-Kitchen; confusion with sink in EDSH-Kitchen. Best viewed digitally at high zoom.

varies a lot when the hand is moving and deforming. Therefore it cannot collect a strong evidence saying that the central pixel belongs to a hand. On the contrary, our method can provide partial support from the neighborhood of edge pixels via structured label predictions because these partial contributions will aggregate into the edge pixel such that the ambiguity along the edge can be removed.

In Figure 8, both our method and single pixel prediction might cause con-



(a) (b) (c)

Figure 9: Usage of features in the structured forest. (a) Feature of channel and similarity features. (b) Feature in different channels. The first 3 channels are *CIELUV* colour channels, the 4th are magnitude of gradient and the rest are gradients in different orientations. (c) Spatial distribution of the features rasterized in a 8×8 grid. Higher intensity indicates more usage in corresponding area.

fusion with certain textureless objects, *e.g.*, doors in 1^{st} row. However, our method smoothed these regions so that they could be easily removed by post-processing. Meanwhile, our method was more robust to the incorrect labels in training samples caused by improper segmentation. These labels often appear along edges of the hand as well as on the objects in the hand. For single pixel prediction, these labels will be incorrectly treated during training, so they will affect the prediction in a fundamental way. However, it is not common in our approach as ours is based on patch observation that is robust to pixel-level noise.

5.1.2. Feature

We first investigated the contribution of different features by checking its usage in the structured forests. All selected features are aggregated from all non-leaf nodes in the forests. First in Figure 9(a), we show the ratio of channel features and texture features. They are almost equally important so the pairwise texture features are essential in mask prediction. Figure 9(b) shows that the colour features (first 3 channels) are the mostly used ones among all channel features. This means that colour is still the most discriminative feature for hand detection. Moreover, the orientations of the gradients are more often used than their magnitudes, which suggests that the edge orientation of a hand is

	Colour	Gradient	C+G	C+G+T
EDSH2	72.86	52.62	77.05	**80.43**
EDSH-Kitchen	83.35	55.32	87.14	**92.11**

Table 2: F-score (100×) of different feature settings.

more informative in determining its shape mask. Figure 9(c) shows the spatial distribution of selected channel features in a 32×32 patch rasterized in a 8×8 grid, we can see that most of the places are used for predicting the hand shape mask.

We also trained several models of various settings on EDSH dataset in order to validate the importance of different feature types. All models took 16×16 colour patch as input and their settings of feature combinations are shown in Table 2. The performance of only using colour channels on EDSH2 dataset is relatively lower than that of EDSH-Kitchen mainly because of the great illumination changes. The model of only using gradient channels may introduce a lot of false positives along long edges. The performances were improved on both datasets by combining colour and gradient channels. With texture descriptor extracted, they were further improved as the pairwise pixel differences are more robust to the change of lighting.

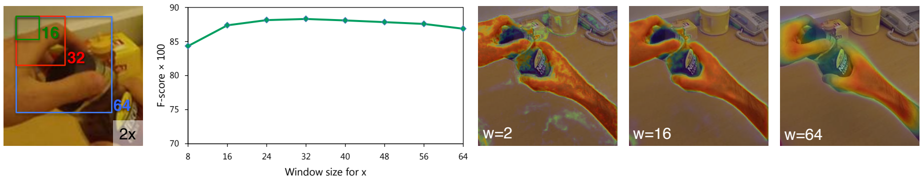


Figure 10: Performance of our structured forests of different patch sizes for GTEA dataset.

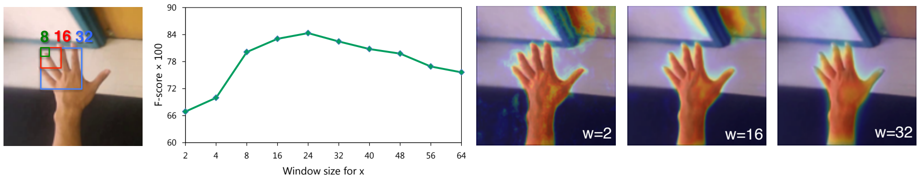


Figure 11: Performance of our structured forests of different patch sizes for EDSH dataset.

5.1.3. Size

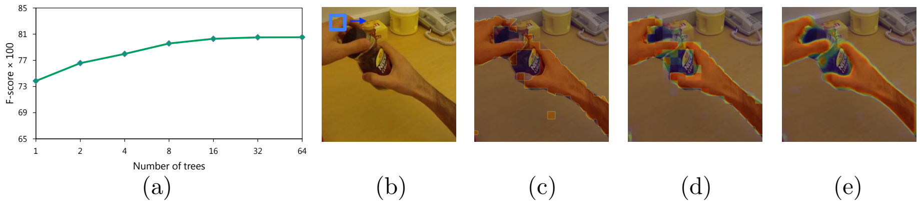
We next examined the effect of different sizes of patches used for our structured forests. In order to examine the hand in different resolution for ego-centric videos, we down-sampled all the images in EDSH dataset from 640 × 360 to 320 × 180. For both datasets, we increased the size of training patches from half the finger size to twice the palm size. Both in Figure 10 and Figure 11, there are two phases in the F-score curve. During the increasing phase, it brings more spatial context for shape mask prediction so the F-score will increase dramatically. In the decreasing phase, the structured forests will suffer from two limitations. First, it will over-smooth along the hand contour which makes it sensitive to the detection threshold. Second, there will not be sufficient training samples for the exponentially increased output space. Thus the forests will probably overfit the training data. From our observation, more than half the palm size is suitable for a robust hand detector.

5.1.4. Number of trees

As for the common smoothing effect introduced by our structured forests, we further examined the contribution of different number of trees to shape detection. Figure 12(a) shows the performance of structured forests under different number of trees. We used a forest trained from 16 × 16 patches to observe the shape mask prediction. In Figure 12(c), we can see that a single tree can outline the shape but the shape contour is not smooth enough. This can be improved either by increasing the number of trees T as shown in Figure 12(d) or reducing the stride width d as shown in Figure 12(e). Both can accumulate more spatial context in order to obtain a better shape mask.

5.1.5. Timing

Table 3 records the time cost for evaluating a 720×405 image on a Macbook Pro with Core i7 2.5 GHz CPU and 16 GB Memory. We compared the public code provided by the author [5] with our `MATLAB` implementation. We used 9 × 9 patch to train the single pixel predictor, and used the same size in our



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

Figure 12: Performance under different number of trees and stride width. (a) Overall F-score *w.r.t.* different numbers of trees. (b) Original image. (c) Prediction by forest ($d = 16$, $T = 1$). (d) Prediction by forest ($d = 16$, $T = 16$) (e) Prediction by forest ($d = 1$, $T = 1$).

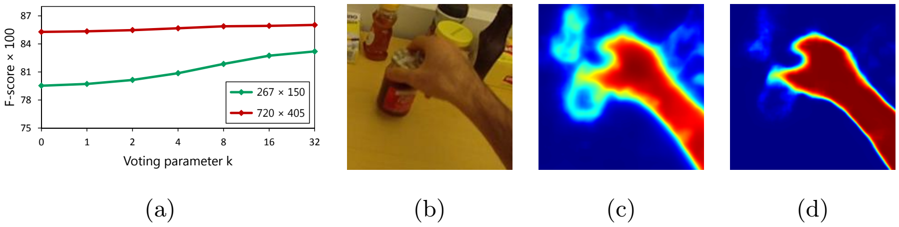
	Li *et al.* [5]	Ours (s)			Ours (m)		
		$d = 1$	$d = 2$	$d = 8$	$d = 1$	$d = 2$	$d = 8$
Time (ms)	1167	576	214	71	2956	1038	387
F-Score (100×)	86.37	89.21	89.25	87.81	88.91	89.06	89.35

Table 3: Comparison on time cost to evaluate a 720×405 colour image using different methods with input patch of size 9×9. s and m stand for single and multiple scale detection, and d is the stride width.

implementation. s and m stands for single and multiple scale detection. d is the stride width. Most of their time is spent on feature extraction compared with ours. Moreover, we can reduce the time cost for prediction by increasing the stride width with only a little performance drop. This is extremely helpful when it comes to multiple scale implementation, where image copies of different scales are considered.

5.1.6. Fusion Parameter k

We performed two experiments to evaluate different choices of the fusion parameter k in Eq. 4. Firstly we trained a set of forests from 16×16 patches using original 720×405 images of GTEA dataset and trained another one set using their down-sampled 267×150 images. From Figure 13(a), we found that there was a great improvement in F-score for the latter one while that for the former set was not improved much. This is because for the images of larger resolution the confusion often happened along the contour of the hand where the detection results were good enough. Therefore larger k did not contribute much although it made the boundary clearer as shown in Figure 13(c-d). When it comes to smaller resolution, the confusion also existed in finger region during fusion process and better results could be obtained with a larger k.



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

Figure 13: Performance under different choices of k. (a) Overall F-score of two experiments. (b) Original image. (c) Prediction by forest ($k = -1$). (d) Prediction by forest ($k = 32$).

5.1.7. Generalization Capability

It is also interesting to investigate the generalization capability of our method among different subjects and datasets.

We first tested the our approach by training on one subject (S1) in GTEA dataset and tested on all subjects. For S2-S4, we manually lablled 20 images from different activities and used them for evaluation. The result are shown in Figure 14. In general our model can achieve good performance among all 4 subjects. When more training samples are used, the performance can be improved accordingly.

Next we showed the performance of hand detection in GTEA dataset using the model trained from EDSH dataset (E2G) and that of EDSH dataset with the model from GTEA dataset (G2E) in Figure 15 respectively. It is interesting to see that the performances were quite low probably because our model uses raw channel features which are not robust to illumination changes.

We observe that this is limited by the random forest framework as it usually requires a lot of training data in order to cover input space sufficiently. As our method uses channel features as input, it can be generalized to a new scene if its lighting condition is similar to original one.

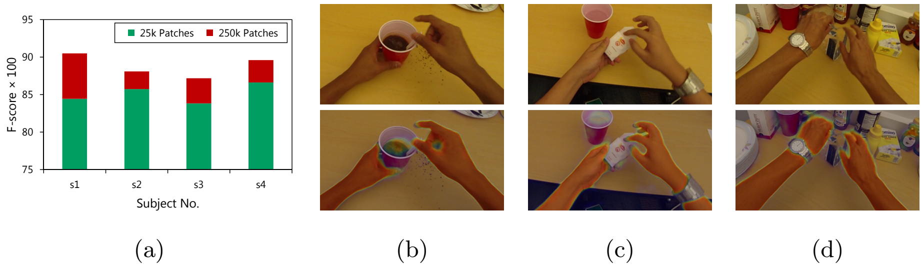


Figure 14: Performance on each subjects. (a) Overall F-score on each subjects with different amount of training samples. (b-d) Sample images for Subject 2-4: original images (top) and their corresponding prediction (bottom).

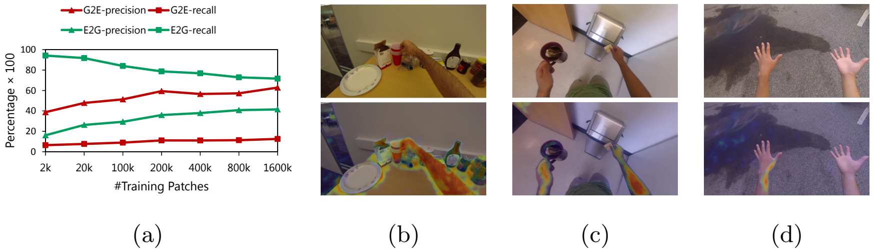


Figure 15: Performance on GTEA and EDSH datasets. (a) Average precision and recall of two settings. (b) Sample image from GTEA dataset (top) and its prediction (bottom). (c) Example from EDSH-kitchen dataset. (d) Example from EDSH1 dataset.

5.2. Hand Detection in Depth Images

As our method is not limited to ego-centric images, we further explored its ability to segment hands in depth images on *NYU Hand Pose Dataset* [39]. The dataset consists of 6736 depth frames of a subject doing various hand gesture and their ground truth per-pixel hand labels. As the number of training images are larger than the previous datasets and the hand size varies a lot as well, we down-sampled each image from the resolution 640×480 to 320×240 in order to use more hand patches for training and cover more hand samples. Accordingly, we extracted pixel-wise depth comparison as channel features for each pixel in order to achieve depth and scale invariance following the success in [22].

5.2.1. Comparison with Benchmark

We compared with classical random decision forest approach, which has been shown successful in [22, 39]. As the feature used in our approach and theirs are the same, we mainly discuss the number of candidate features used for pixel-wise comparison and the size of patches.

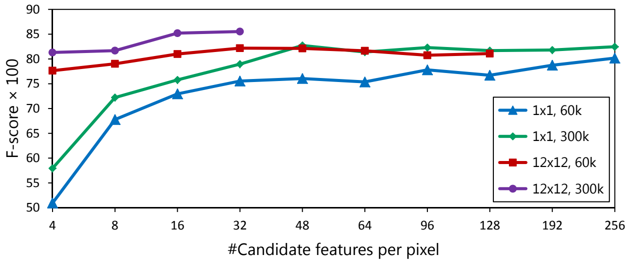


Figure 16: Performance of our structured forests under different number of candidate features on NYU Hand Pose dataset. 1×1 and 12×12 indicate patch size w. 60k and 300k stand for the numbers of training samples.

Number of Candidate Features of Pixel-wise Comparison. For single pixel classification, it usually requires a large amount of pixel-wise comparison to achieve high accuracy. As shown in Figure 16, the performance increases as the number of candidate features increases, because it can capture more information of its local neighborhood. On the contrary, our structured forests can predict the hand region well with less training samples even if there is limited number of candidate features per pixel.

Size of Patches. Similar to previous observation, there are also two phases in F-score curve in Figure 17. 12×12 is optimal in our setting, which corresponds to two-finger sizes when a hand is 600mm away from the depth camera. Figure 18 shows some examples predicted by the benchmark model (1×1) and ours with 64 candidate features and probe offset 64 pixel meters. We can see that our method can eliminate the confusion in the head and arm given a small set of candidate features.

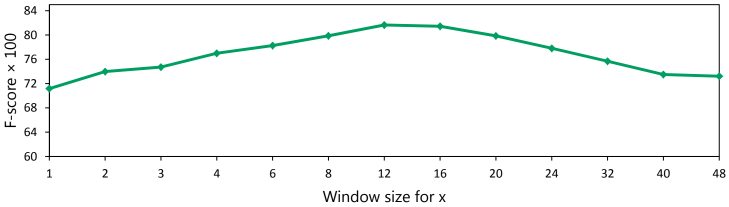


Figure 17: Performance of our structured forests under different patch sizes for NYU Hand Pose dataset.

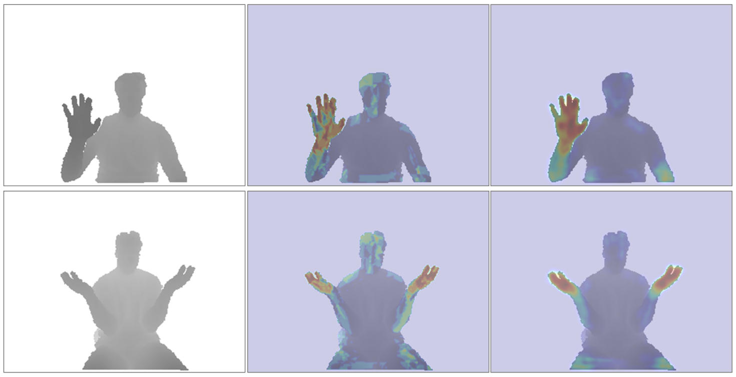


Figure 18: Sample results on NYU Hand Pose Dataset: (left column) original images, (second column) results of benchmark method [39], and (last column) our results.

5.3. Labelling Hand Parts: EDSH dataset

For hand parts labelling, we continued to label hand parts on EDSH dataset. We defined 4 regions of a hand, namely as *thumb*, *finger* (for the rest of the fingers), *palm* and *forearm* as shown in Figure 6. Because detailed hand segmentation maskes have already been made available in the public dataset, we further developed a labelling tool to manually label each part and perform an `AND` operation to obtain a precise segmentation of each hand part. 437 images in EDSH1 were used for training and 104 images in EDSH2 were used for testing.

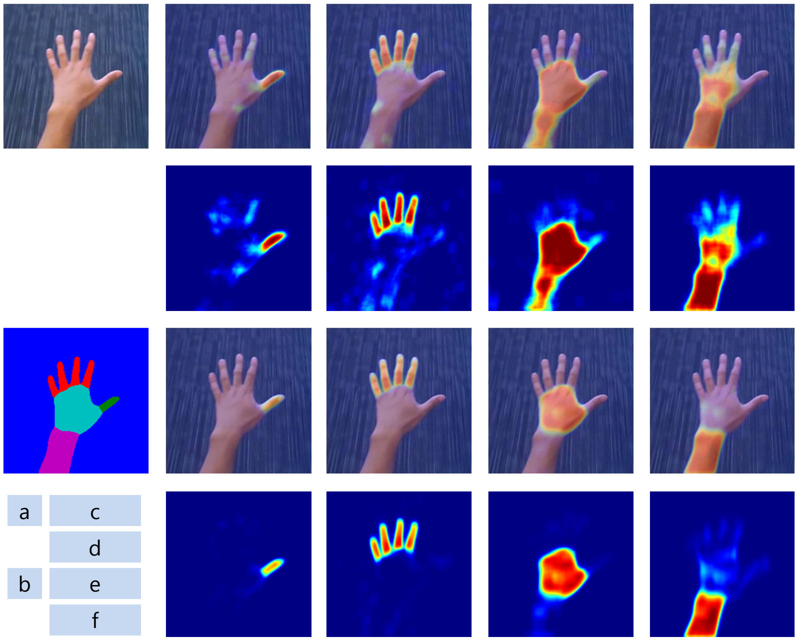


Figure 19: Sample images for comparison with hand part detection. (a) Original image. (b) Predicted hand parts. (c) Image overlaid with per-pixel probability map for each hand part by one-vs-all prediction. (d) Probability map of hand part obtained by one-vs-all prediction. (e) Image overlaid with per-pixel probability map for each hand part by our method. (f) Probability map of hand part obtained by our method.

5.3.1. Support from Labelling in the Neighborhood

We first performed a pilot study to train four separate detectors of patch size 16×16 for each part of the hand in one-vs-all fashion, which treated the other

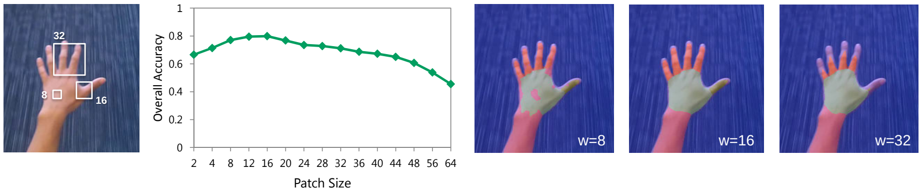


Figure 20: Performance of different patch sizes used for training.

parts as background pixels using previous hand detection framework. Next our structured forest was trained under the same setting to compare with the above detectors qualitatively. The detection results for each hand part are shown in Figure 19. From the probability map, we can see that for one-vs-all hand part detectors, confusion may occur between *thumb* and *finger* region, as well as *palm* and *forearm* region. On the contrary, if we switch to the posterior distribution for each label under multi-class framework, the confusion is suppressed greatly for these regions. This suggests that the class labels in the neighborhood of a hand pixel provide contextual support for the recognition of each part.

5.3.2. Patch Size

Next we examined the effect of different sizes of patches used for training. Similar to the experiment settings of hand detection in EDSH dataset, we downsampled the video to 320 × 180 and tested a wide range of patch sizes from 2 × 2 pixels to 64 × 64 pixels. Overall per-class accuracy was used to evaluate the multi-class prediction for hand region. It was computed as the mean of diagonal elements of the confusion matrix between predicted labels and ground truth labels. The metric had the same weights for each hand part in spite of their different sizes. We evaluated our method on the whole image in the test set and collected the metric in ground truth hand region. The overall accuracy of all classes *vs.* size of patch is shown in Figure 20. When the patch size was small, there was great confusion between palm center and forearm region. This is probably because the colour and texture information are quite similar for these patch samples and no information from the shape can help to differentiate the *palm* pixels from *forearm* pixels. As the patch growed larger, the prediction for fingers were often suppressed by *background* pixels probably due to insufficient support from its nearby regions. Therefore, like in Shotton *et al.*'s observation in [22], we also confirm that larger neighborhood helps defining the identity of current pixels. However, it may suffer from insufficient training data if the size becomes too large (see the drop in Figure 20).

We further show F-score for each class in Figure 21. The metrics of *finger* and *thumb* drop greatly when the patch size becomes larger while there is a slight increase for *palm* and *forearm* region. The size of 16×16 is a good trade-off between small parts such as hand fingers and large parts such as palm and forearms.

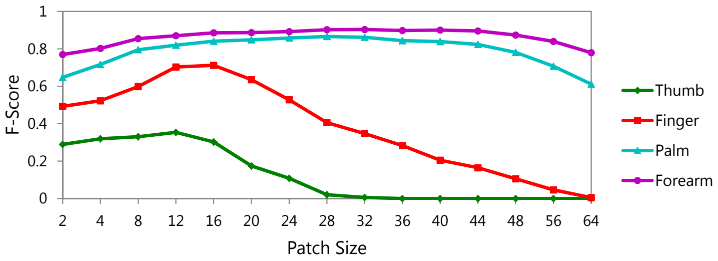


Figure 21: F-score for different hand part classes.

6. Conclusions

We have presented a new approach to exploit the inherent contextual information from structured hand labelling for pixel-level hand detection and hand part labelling. By using a random forest framework, our method can predict the hand mask and hand part labels in an efficient and robust manner. Through experiments, we demonstrate that our method can outperform state-of-the-art pixel-level detection methods in ego-centric videos, and further be able to parse hand parts in details. This may provide us better information for gesture analysis and hand-object interaction.

[1] M. Kölsch, M. Turk, Robust Hand Detection., in: Proc. IEEE Conference on Automatic Face and Gesture Recognition, 2004.

[2] X. Zhu, J. Yang, A. Waibel, Segmenting hands of arbitrary color, in: Proc. IEEE Conference on Automatic Face and Gesture Recognition, 2000.

[3] Y. Wu, T. Huang, View-independent recognition of hand postures, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2000.

[4] A. Mittal, A. Zisserman, P. Torr, Hand detection using multiple proposals, in: Proc. British Machine Vision Conference, 2011.

[5] C. Li, K. Kitani, Pixel-level hand detection in ego-centric videos, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2013.

[6] Y. Wu, Q. Liu, T. Huang, An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization, in: Proc. 4th Asian Conference on Computer Vision, 2000.

[7] M. Jones, J. Rehg, Statistical color models with application to skin detection, International Journal of Computer Vision 46 (1) (2002) 81 – 96.

[8] L. Sigal, S. Sclaroff, V. Athitsos, Skin Color-Based Video Segmentation under Time-Varying Illumination, IEEE Trans. on Pattern Analysis and Machine Intelligence 26 (2004) 862 – 877.

[9] M. Kolsch, M. Turk, Hand tracking with flocks of features, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2005.

[10] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color-Based Probabilistic Tracking, in: Proc. 7th European Conference on Computer Vision, 2002.

[11] E. Ong, R. Bowden, A boosted classifier tree for hand shape detection, in: Proc. IEEE Conference on Automatic Face and Gesture Recognition, 2004.

[12] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 2001.

[13] N. Dalal, B. Triggs, D. Europe, Histograms of Oriented Gradients for Human Detection, in: Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 2005.

[14] A. Thayananthan, P. H. S. Torr, S. Member, B. Stenger, R. Cipolla, Model-based hand tracking using a hierarchical Bayesian filter, IEEE Trans. on Pattern Analysis and Machine Intelligence 28 (9) (2006) 1372–84.

[15] I. Oikonomidis, N. Kyriazis, A. A. Argyros, Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints, in: Proc. British Machine Vision Conference, 2011.

[16] P. Buehler, M. Everingham, D. P. Huttenlocher, A. Zisserman, Upper Body Detection and Tracking in Extended Signing Sequences, International Journal of Computer Vision 95 (2) (2011) 180–197.

[17] Y. Sheikh, O. Javed, T. Kanade, Background Subtraction for Freely Moving Cameras, in: Proc. IEEE 12th International Conference on Computer Vision, 2009.

[18] A. Fathi, X. Ren, J. M. Rehg, Learning to recognize objects in egocentric activities, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[19] H. Trinh, Q. Fan, P. Gabbur, S. Pankanti, Hand tracking by binary quadratic programming and its application to retail activity recognition, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2012.

[20] C. Li, K. Kitani, Model recommendation with virtual probes for egocentric hand detection, in: Proc. 14th IEEE International Conference on Computer Vision, 2013.

[21] L. Baraldi, F. Paci, G. Serra, Gesture Recognition in Ego-Centric Videos using Dense Trajectories and Hand Segmentation, Computer Vision and Pattern Recognition (2014) 688–693.

[22] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake, Real-Time Human Pose Recognition in Parts from a Single Depth Image, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[23] C. Qian, X. Sun, Y. Wei, X. Tang, J. Sun, Realtime and Robust Hand Tracking from Depth, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[24] C. Xu, L. Cheng, Efficient Hand Pose Estimation from a Single Depth Image, in: Proc. 14th IEEE International Conference on Computer Vision, 2013.

[25] D. Tang, H. Chang, A. Tejani, T. Kim, Latent Regression Forest: Structured Estimation of 3D Articulated Hand Posture, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[26] C. Keskin, F. Kraç, Y. Kara, L. Akarun, Real Time Hand Pose Estimation using Depth Sensors, in: IEEE International Conference on Computer Vision Workshops, 2013.

[27] J. Shotton, M. Johnson, R. Cipolla, Semantic texton forests for image categorization and segmentation, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2008.

[28] P. Kontschieder, S. R. Bulo, M. Pelillo, H. Bischof, Structured Labels in Random Forests for Semantic Labelling and Object Detection, IEEE Trans. on Pattern Analysis and Machine Intelligence 36 (10) (2014) 2104–2116.

[29] L. Breiman, J. Friedman, C. J. Stone, R. A. Olshen, Classification and regression trees, Chapman and Hall/CRC, 1984.

[30] A. Criminisi, Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning, Foundations and Trends in Computer Graphics and Vision 7 (2-3) (2011) 81–227.

[31] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models., IEEE Trans. on Pattern Analysis and Machine Intelligence 32 (9) (2010) 1627–45.

[32] P. Dollár, Z. Tu, P. Perona, S. Belongie, Integral Channel Features., in: Proc. British Machine Vision Conference, 2009.

[33] P. Dollár, C. Wojek, B. Schiele, P. Perona, Pedestrian Detection: An Evaluation of the State of the Art., IEEE Trans. on Pattern Analysis and Machine Intelligence 34 (4) (2012) 743 – 761.

[34] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, M. Pantic, 300 Faces in-the-Wild Challenge: The First Facial Landmark Localization Challenge, in: IEEE International Conference on Computer Vision Workshops, 2013.

[35] S. Bagon, O. Brostovski, M. Galun, M. Irani, Detecting and sketching the common, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2010.

[36] K. Crammer, On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines 2 (2001) 265–292.

[37] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, ACM Transactions on Graphics 23 (3) (2004) 309 – 314.

[38] G. Serra, M. Camurri, L. Baraldi, Hand segmentation for gesture recognition in EGO-vision, Proceedings of the 3rd ACM International Workshop on Interactive Multimedia on Mobile & Portable Devices (2013) 31–36.

[39] J. Tompson, M. Stein, Y. Lecun, K. Perlin, O. Database, Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks.