# Sieving Regression Forest Votes for Facial Feature Detection in the Wild

Heng Yang and Ioannis Patras
Queen Mary University of London
{heng.yang, i.patras}@eecs.qmul.ac.uk

## Abstract

*In this paper we propose a method for the localization of multiple facial features on challenging face images. In the regression forests (RF) framework, observations (patches) that are extracted at several image locations cast votes for the localization of several facial features. In order to filter out votes that are not relevant, we pass them through two types of sieves, that are organised in a cascade, and which enforce geometric constraints. The first sieve filters out votes that are not consistent with a hypothesis for the location of the face center. Several sieves of the second type, one associated with each individual facial point, filter out distant votes. We propose a method that adjusts on-the-fly the proximity threshold of each second type sieve by applying a classifier which, based on middle-level features extracted from voting maps for the facial feature in question, makes a sequence of decisions on whether the threshold should be reduced or not. We validate our proposed method on two challenging datasets with images collected from the Internet in which we obtain state of the art results without resorting to explicit facial shape models. We also show the benefits of our method for proximity threshold adjustment especially on 'difficult' face images.*

## 1. Introduction

Detecting semantic facial features on face images is often the first step for many tasks in Facial Analysis including face and facial expression recognition [16, 21]. Recent works attempt to make a transition from facial images recorded in laboratory settings or in controlled conditions to face images "in the wild" [4, 10, 12, 30, 6]. However, they still have difficulties with low quality face images, head pose variation and partial occlusions, especially when real-time detection is needed.

In this paper, we address the problem using random forests, given their good performance in various challenging computer vision tasks like video synopsis[29], action recognition [13], human pose estimation [23], object recognition [26] and facial feature detection [10]. In this framework

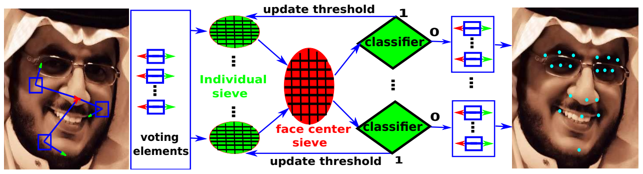


Figure 1: Our approach estimates the facial feature points in 2D images using votes from random regression forest. Before accumulating the votes to a Hough map, our method refines them by a cascade of sieves.

(see Figure 1), we follow a part-based approach in which first observations are extracted at several local regions in the face, and then each observation is propagated in the tree and votes for the localization of facial features as well as for a number of hidden variables such as the center of the face and the head pose angles. Our approach introduces a bank of sieves, each one associated with one of the variables in question (i.e. either a facial feature, or the center of the face). Essentially, each sieve operates as a filter that rejects votes that are not relevant/useful for the estimation of the variable associated with the sieve in question.

The sieve that is associated with the hidden variable (i.e. the face center) rejects votes that are not consistent with hypotheses that are generated by a search for local maxima in the voting space for the variable in question. This introduces global consistency, imposes geometric constraints in the form of implicit facial shape models, and deals with irrelevant votes due to, for example, occlusions. This differentiates our method from classic regression forests that treat votes in a completely independent way, i.e. there is no mechanism that encourages consistent predictions. The sieves associated with the individual facial features are responsible of rejecting votes with low accuracy. Each sieve adjusts a threshold that controls the minimum allowed proximity (equivalently the maximum allowed offset) between the facial feature in question and the location at which the observations are extracted. With a large threshold, that is at high proximity, we select votes with small offsets to the facial feature in question. Those are expected to have high localization accuracy, unless they are contaminated due to conditions like noise, shadows and occlusions. With a small threshold, we select votes with large offsets and in this way introduce facial shape constraints and robustness to occlusions. Such a threshold is widely used in regression forests applications like [13, 23, 10, 28]. In contrast to using a fixed threshold that is learned during training, in our work we learn a classifier who controls a procedure in which the proximity threshold is gradually reduced. In this procedure, the decision on whether the threshold should be decreased or not is taken by a classifier that is built on middle-level features that are extracted from the current voting map for the location of the feature in question.

Finally, the detection is carried out on Hough voting maps formed by the cast votes after they are filtered by the cascade of sieves. Our contributions are validated on two challenging face image datasets, namely, the Labeled Faces in the Wild and the Annotated Facial Landmarks in the Wild. We show that with the proposed approach, and without explicitly introducing shape models, we obtain state-of-the-art performance in both datasets. We also show how that the benefits of using sieves and determining an image-dependent and facial-feature-dependent threshold are higher for the 'difficult' images in both datasets.

## 2. Related work

Two different sources of information are usually exploited for facial feature detection: face appearance and spatial shape. Instead of modelling them together like the classic AAMs [8], several methods have focused on using shape models to regularize the local detections like the Constrained Local Models (CLMs) [22], Branch & Bound optimization [2] and tree structured shape models [30]. Instead of using parametric shape models, non-parametric representations of shape constraint include [4] and a series of cascaded pose regression approaches [12, 11, 6, 5]. Local detectors can be grouped into two categories: classification based methods like SVM in [4]; regression-based methods like boosted regression [24] and Regression Forests (RF) [10, 28]. [7] and [27] proposed to impose shape models on RF. In this work, within the RF framework, we introduce non-parametric shape constraints.

In terms of rejecting irrelevant observations for regression, our work is related to [19]. Our approach is also related to methods that analyse RF votes. In particular, [3] modelled the joint distribution over all the votes and the hypotheses in a probabilistic way, rather than simply accumulating the votes. [17] studied the geometric compatibilities of the votes in a pairwise fashion within a game-theoretic setting. [20] learned latent variables for votes weighing. These methods were developed for person/object detection and focused on *intra-class* geometrical agreement, while the consistency in our problem is *inter-class*, since we consider the localization of all facial points.

## 3. Sieving Random Forest votes

Our method can be built on top of any part-based voting approaches. In this section, we first describe two baseline regression forests and then present our proposal to use a cascade of sieves to refine votes.

### 3.1. Regression forests

The forests that we use here, use image patches as observations. Once a regression forest (or Hough forest) is trained, at testing stage, observations arrive at tree leaves and cast votes for the hidden or target variables [9]. We build on two recent conditional regression forest frameworks: **CRF-D** [10] and **CRF-S** [23].

#### 3.1.1 CRF-D [10]

CRF-D introduces conditional regression forests that model the appearance and location of facial feature points conditioned on the head pose yaw angle. Specifically, the head pose is quantized in five discrete labels and the training set is also partitioned into five subsets.Then one forest is trained on each subset. An additional forest is trained for the task of head pose estimation. During testing, the head pose forest is utilized first, and then the estimated head pose information determines how many trees will be selected from each subset.

Following the usual practice, the information gain $IG$ criterion is used to select the split function. It is defined as $IG(\phi) = \mathcal{H}(\mathcal{P}) - \sum_{S\in\{L,R\}} \frac{|\mathcal{P}_S(\phi)|}{|\mathcal{P}|}\mathcal{H}(\mathcal{P}_S(\phi))$. $\phi$ is a split function candidate which, when applied to a set of image patches $\mathcal{P}$, splits it into two sub sets: $\mathcal{P}_L$ and $\mathcal{P}_R$. Let us denote with $\mathcal{H}(\mathcal{P})$ the class uncertainty and define it as $\mathcal{H}(\mathcal{P}) = -\sum_{i=1}^{N} \frac{\sum_j p(c_i|\mathcal{P}_j)}{|\mathcal{P}|} \log\left(\frac{\sum_j p(c_i|\mathcal{P}_j)}{|\mathcal{P}|}\right)$, where $p(c_i|\mathcal{P}_j)$ denotes the probability that the patch $\mathcal{P}_j$ is informative about the location of the facial feature point $i$, that is defined as:

$$p(c_i|\mathcal{P}_j) \propto f(|d_j^i|) = \exp\left(-\frac{|d_j^i|}{\alpha}\right). \tag{1}$$

$f(\cdot)$ is a function that transforms the distance $d_j^i$ into a proximity measure. This proximity metric is used throughout this paper. The constant $\alpha$ controls the steepness of this function and is set to $\alpha = \frac{1}{8}$. The distance $d_j^i$ is defined as fractions of the face size.

At leaf nodes, the relative offsets to each facial feature point are summarized by a weighted offset vector. This vector is used to cast votes for observations that arrive at the leaf in question during testing. Similarly, at each leaf node we summarize the relative offsets to the face center. Formally, at a leaf note, the vote associated with the $i$th facial feature point is given by:

$$v_i = (\Delta_i, \omega_i, \Delta_i^0, \omega_i^0), \tag{2}$$

where $\Delta_i = \bar{d}_i, \omega_i = \frac{1}{trace(\Sigma_i)}$ with $\bar{d}_i$ and $\Sigma_i$ are, respectively, the mean and the covariance matrix of the offsets of the $i$th facial feature point. The first two items are the same as in [10]. $\Delta_i^0$ is the mean value calculated on the offsets of the face center and $\omega_i^0$ is the corresponding weight. Figure 1 shows some voting elements, each with a red arrow to the face center and a green arrow to one point target.

### 3.1.2 CRF-S [23]

The CRF-S [23] was used for human pose estimation. [28] adapted their **Partial Model** for facial feature localization. This model shares tree structures for all states of the global variables. At the leaf node, one model is learned for each state $k$ of the global variable. Then, a relative vote associated with $i$th facial feature point in state $k$ is given as:

$$v_{ik} = (\Delta_{ik}, \omega_{ik}, \Delta_{ik}^0, \omega_{ik}^0) \tag{3}$$

where $\Delta_{ik}$ is the mean-shift mode of the largest cluster and $\omega_{ik}$ is the relative size of the cluster. $\Delta_{ik}^0$ and $\omega_{ik}^0$ that are related to the offsets to the center of the face are learned in the same way.

During testing, as in [23] and [28], the state of the global variable, i.e. the $k$ in Eq. (3) is estimated first and then the voting model with the estimated state will be used. The model becomes the same as in Eq. (2). Thus, we drop the index $k$ in the following discussion.

### 3.1.3 Inference from votes

During testing, the patches extracted from a test image are fed to the forest and when they arrive at a leaf note, they cast votes for the localization of the facial features. Let us denote by $V_i = \{v_i\}$ the set of votes for the facial point $i$. Each voting element $v_i$ is associated with a location $z_i$ in the image from where the corresponding patch was extracted. Then, the voting element $v_i$ casts a vote at $y_i = z_i + \Delta_i$. Following the dominant paradigm, e.g., [14, 6, 10], we construct a Hough map for each facial feature point by accumulating the votes. The density for the facial feature point $i$ at pixel location $y_i'$ in the Hough map is therefore approximated as:

$$p(y_i') \propto \sum_{v_i \in V_i} \omega_i \exp\left(-\|\frac{y_i' - y_i}{h_i}\|_2^2\right) \cdot \delta(f(\Delta_i) > \lambda_i) \tag{4}$$

where $h_i$ is a learned per-point bandwidth. $f(\Delta)$ is the proximity metric defined in Eq. (1). $\delta(\cdot)$ is the Dirac delta function that only allows votes for which the proximity test, using the proximity threshold $\lambda_i$, is satisfied. This vote selection process is regarded as a sieve, which in this case is associated with an individual facial point. Then a mode finding algorithm like mean-shift can be applied on the Hough map for detection.

However, spurious maxima can occur in the voting maps. This can happen for many reasons, including occlusions due to head pose and visual obstructions such as hair, glasses, hands, etc. In this work, instead of explicitly learning shape models to constrain the local detection, as in [7, 27], we propose to use a cascade of sieves that impose geometric constraints and filter the votes in order to remove false hypotheses.

## 3.2. Face center sieve

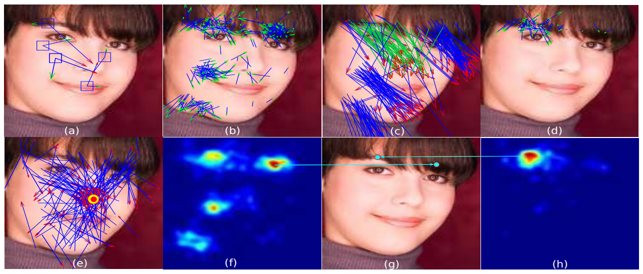


Figure 2: Face center sieve. (a) A vote consists of two offset vectors, one to the target point (green arrow) and the other to face center (red arrow). (b) Original set of votes for the left brow center. (c) The absolute face center votes, those in green are regarded as consistent to the face center. (d) The remaining voting elements filtered by the face center sieve. (e) All voting elements are used to localize the face center (red dot). (f) and (h) are the Hough maps generated from votes of (b) and (d) respectively. (g) shows the corresponding detection results.

A face center sieve is used to reject votes that are not consistent with the localization of the facial center. Recall that each voting element also contains a weighted relative vector to the face center, i.e. $(\Delta_i^0, \omega_i^0)$. Then, the absolute vote to the face center is $y_i^0 = z_i + \Delta_i^0$. As shown in Figure 2e when calculating the face center, voting elements from all the facial points are accumulated into $V_0 = \{(\Delta_i^0, \omega_i^0)\}_{i=1}^N$. We aggregate the votes as in Eq. (4), i.e., the density estimation for the face center at $y_0'$ is approximated by:

$$p_0(y_0') \propto \sum_{(\Delta_i^0, \omega_i^0) \in V_0} \omega_i^0 \exp\left(-\|\frac{y_0' - y_i^0}{h_0}\|_2^2\right) \tag{5}$$

where $h_0$ is a fixed bandwidth. A mean-shift algorithm is employed to find the mode in the voting map that is the estimate of the face center $\hat{y}_0$.

Since the face center is calculated using votes from the whole image, its localization is very robust to partial occlusions even if the center itself is heavily occluded. Consequently, we choose to use it as a stable point that is used by our sieve to reject voting elements that cast votes far from it. This is shown in Figure 2c. Votes that are cast far away from

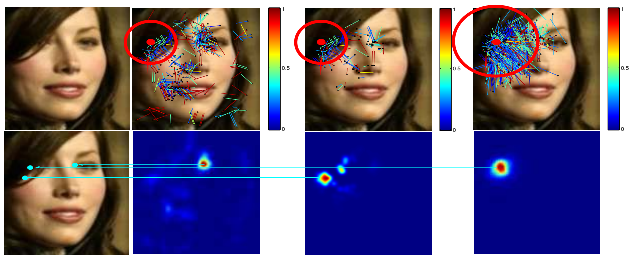


Figure 3: Threshold updating. From left to right, the first row shows the original face image, all votes for the point ($\lambda = 0.35$), votes passed center sieve and the votes with updated threshold ($\lambda = 0.22$) passed center sieve. The color indicates the vote weight and the dark terminal is the voting destination. The second row shows the detection results, Hough map for original voting, after filtering and re-voting.

the estimated center $\hat{y}_0$ are not consistent with the global hypothesis and therefore unlikely to contribute to the correct localization of facial features. Such votes are rejected by the sieve. Formally, the sieve re-weighs a voting element $v_i$, according to the proximity between the location of the vote for the face center (i.e. $y_i^0$) and the estimated face center $\hat{y}_0$, that is:

$$\omega_i := \omega_i \cdot \delta(f(|y_i^0 - \hat{y}_0|) > \lambda_0) \tag{6}$$

where $f(\cdot)$ is the proximity function and $\delta(\cdot)$ is the delta function. As shown in Figure 2d, after filtering by the sieve, voting elements that violate the face center consistency and vote for other face center hypotheses, are assigned a zero weight, i.e. they are removed from the votes set. The ones that do not violate the face center consistency are kept. In what follows we will denote set of the remaining votes by $V^F$. The Hough images that are formed when all the voting elements are used, the Hough images that are formed after the re-weighing and the mean-shift modes found on them are shown in Figure 2.

## 3.3. Individual sieve threshold updating

Recall that when collecting the votes for an individual feature point, a threshold is applied. This works as a sieve that prohibits long distant voting. This threshold is typically optimized during training and is constant during testing. This works well in most cases but fails when a feature point is heavily occluded. As shown in Figure 3, in that case, only few valid voting elements remain after the sieve of the face center is applied and those voting elements may contain wrong information. In such cases, the proximity threshold should be reduced, and patches from further away regions allowed to cast their votes. Such votes, essentially introduce stronger facial shape constrains.

In order to determine an image-dependent proximity threshold $\lambda_i$ for the sieve $i$, we propose a iterative scheme

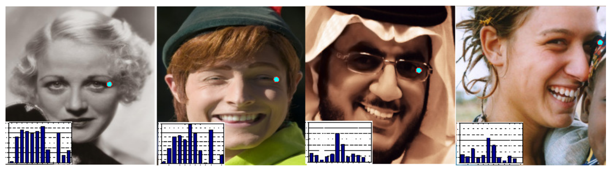


Figure 4: Comparison of vote orientation histograms $h_o$. For the same facial point, $h_o$ of occluded ones (the right two) differs significantly from non-occluded ones (the left two). By contrast, $h_o$ of non-occluded points are similar despite the fact that the face images are quite different.

in which we start with the optimal (for the whole validation set) value for the proximity threshold $\lambda_i$ and at each iteration decide whether to decrease it or not. The decision on whether to adjust (i.e. decrease) the threshold is taken by a classifier who is applied on features that are extracted from the voting map calculated using the current threshold value. If a decision to adjust the threshold is taken, then a new voting map is calculated considering all the voting elements that pass through the sieve, and the classifier is again applied on features that are extracted from the new voting map. We train the individual classifiers in a supervised manner. For each facial point we collect a set of training instances $(x_i, \tau)$ consisting of a feature vector $x_i$ that is extracted from the voting map, as explained below, and an adjustment operation label $\tau \in \{1, 0\}$. $\tau = 0$ means that the point can be localized with high accuracy using the current threshold value from which $x_i$ is extracted. $\tau = 1$ means an adjustment (i.e. decrease) of the threshold leads to a better estimate. We note that we only consider enlarging the voting length, i.e. decreasing the value of $\lambda$. In our formulation, we learn a function $f : \mathcal{X}_v \rightarrow \{1, 0\}$ that minimizes the number of wrong decisions, where each decision is binary: to decrease the threshold or not. In contrast to other methods that explicitly try to detect occlusions using image features, we propose to use middle-level features that are extracted directly from the voting maps that are formed using the current threshold before and after the face center sieve is applied, i.e., $V_i$ and $V_i^F$ respectively. A description of the middle-level features $x_i = (x_i^1, x_i^2, x_i^3)^T$ is included below:

- $x_i^1 = \frac{\sum_{v \in V_i^F} v}{\sum_{v \in V_i} v}$, the ratio of the sum of vote weights after and before the center face sieve is applied.

- $x_i^2 = \frac{\sum_{v_i \in V_i^F} f(|y_i - \bar{y}_i|)}{|V_i^F|}$, the concentration of the votes after face center sieve. $\bar{y}_i$ is the mean value of all the $y_i$, that is calculated from $v_i$. $f(\cdot)$ is the proximity function and $|V_i^F|$ is the number of votes in the set.

- $x_i^3 = h_o$, the orientation histogram of the votes after filtering. It consists of 12 equally divided bins, i.e., 30° per bin, as shown in Figure 4.

In order to train the decision classifier, we collect the training samples as follows. On a validation dataset, first, we get the detection results by applying the detector with the pre-optimized $\lambda_i^0$ and extract the features $x_i$. Then, for each facial feature, we decrease the value of proximity threshold $\lambda_i$ (i.e. allow distant votes) gradually. For each facial feature point, we find the optimal value of the threshold $\hat{\lambda}_i$, i.e. the one that has led to the most accurate detection. If $\hat{\lambda}_i < \lambda_i^0$, we set $\tau = 1$ and vice versa. We then add in the training set the appropriate training pairs, i.e. $(x_i, \tau = 0)$ or $(x_i, \tau = 1)$. Then, an RBF-kernel based SVM classifier that predicts $\tau$ is trained for each point separately.

### 3.4. Algorithm summary

The voting map of a feature point is re-estimated if the threshold of its sieve has been updated (Figure 1). Then, the new votes are filtered by the face center sieve. This is done iteratively until there is no proximity threshold update or the maximum iterations are reached. In order to have real-time performance we set the maximum iteration number to 3 and the update step is set to 20% of the current threshold. Almost identical results are obtained in our experiment with a smaller step and a larger max number of iterations. With a small step size the input to the classifier does not change much from iteration to iteration; it takes more iterations until either the classifier output changes or the max iterations are reached. We summarize our method in Algorithm 1.

**Algorithm 1** Sieving votes from regression forests

```
1:  initialize all thresholds $\Lambda = \{\lambda_i\}$ with pre-optimized
    values and set maximum iteration number to $MaxIter$
2:  while $\Lambda \neq \emptyset$ and $MaxIter > 0$ do
3:      for all $\lambda_i \in \Lambda$ do
4:          collect voting elements $V_i$ based on $\lambda_i$
5:          apply face center sieve and obtain $V_i^F$
6:          calculate the middle level feature $x_i$
7:          $\tau \leftarrow \mathrm{svm}_i(x_i)$      ▷ svm classifier for prediction
8:          if $\tau == 1$ then
9:              $\lambda_i := \lambda_i - 0.2\lambda_i$
10:         else
11:             remove $\lambda_i$ from $\Lambda$
12:         end if
13:     end for
14:     $MaxIter--$
15: end while
```

## 4. Experiments

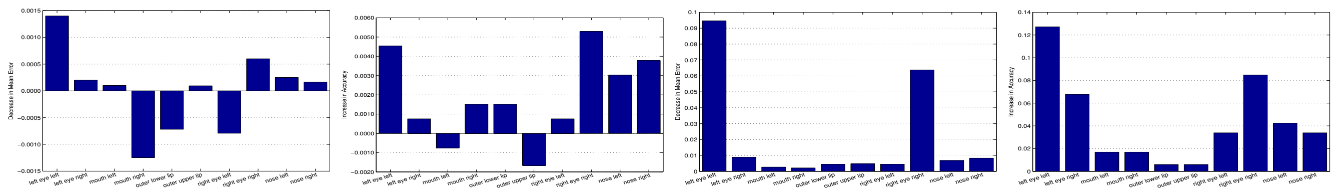
We demonstrate the efficacy of our proposed contributions on two public datasets that contain face images collected from the Internet. The first, contains the 13,233 images of the **Labelled Face in the Wild (LFW)** dataset [15] that are annotated [10] with the location of 10 facial points and the face bounding boxes. The second, is the **Annotated Face Landmarks in the Wild (AFLW)** [18] that contains real-world face images from Flickr. These images exhibit a very large variability in pose, lighting, expression as well as general imaging conditions. Many images exhibit partial occlusions that are caused by head pose, objects (e.g., glasses, scarf, mask), body parts (hair, hands) and shadows. In total, 25993 faces are annotated with up to 21 landmarks per face. We selected a subset in which all 19 frontal landmarks (i.e. excluding the two ear lobes) were annotated (about 6200 images and some are with occlusion). From this subset, we randomly select 1000 images for testing and 600 images for validation (300 of which were selected manually to ensure they contain occlusions). The rest were used for training the CRF-S model. Throughout our experiments we report the localization error as a fraction of the inter-ocular distance [10, 6, 24].

### 4.1. Implementation details

**CRF-D** We used the code and the trained model of the CRF-D that are publicly available [10]. At each leaf node, the trained model provides offset vectors to 10 facial points and it also provides a mean patch offset vector to the center of the bounding box, i.e. $\Delta^0$ in our work. We assign a unit weight to face center vote, i.e. we set $\omega^0 = 1$. Despite the fact that the definition of the face center is slightly different, this allows us to evaluate our contributions using the CRF-D as a baseline.

**CRF-S** We train the CRF-S on the AFLW dataset. We discretize the head yaw angle into 3 labels and at each leaf node conditional models are learned as discussed in Section 3.1.2. We use the same settings such as image features, maximum tree depth (20), number of tests at the internal nodes (2000) and forest size (10 trees in total), that are used by CRF-D. The per-point bandwidth $h_i$ are set as by CRF-D and the thresholds of individual sieves are optimized on the validation set using a grid search (from 0 to 0.8 with a step value 0.05), as in [10].

**Sieve parameters** The parameters that are related to the face center sieve are the bandwidth for the face center $h_0$ in Eq. (5) and the threshold $\lambda_0$ in Eq. (6). All bandwidths, that is $h_0$ and $h_i$, are set to the same value that is learned in the validation set. A key parameter is the face center proximity threshold $\lambda_0$. Large values of $\lambda_0$ impose larger consistency in the votes for the face center, while smaller values relax the constraint and allow more votes to pass through the sieve. We optimized this value through grid search in a validation set ($\lambda_0 = 0.4$). The SVM models used for thresholds updating are trained on the validation set of AFLW using a 10-fold cross validation. Since there are far fewer occlusions on the three nose points we do not update the thresholds for them.



(a) Mean error decrease on LFW_TestI (b) Accuracy increase on LFW_TestI (c) Mean error decrease on LFW_TestII (d) Accuracy increase on LFW_TestII

Figure 5: Results on **LFW**, compared to [10]. Note that the range of the $Y$ axis in right two is different from the left two.

## 4.2. Experiments on LFW

We first evaluate the contribution of our face center sieve by comparing with CRF-D [10]. The latter has achieved state-of-the-art results on the LFW. We randomly select 1000 images from the dataset for testing and put them into two sets according to the average localization error of the CRF-D detector. In this way we create an 'easy' partition, namely the **LFW_TestI**, where the average point localization error of the CRF-D is less than 0.1, and a 'difficult' partition, namely the **LFW_TestII**, consists of the rest of the images. We repeated this 4 times and on average 118 out of 1000 face images ended up into LFW_TestII. This small number is due to the fact that the face images in the LFW dataset are relatively easier. Only a few of them contain occlusions caused by head pose, hair and sunglasses. The absolute difference of mean error and accuracy (using the definition of [10]) on the LFW_TestI and LFW_TestII are shown in Figure 5. On LFW_TestI, there are some points our method performs slightly worse, but the difference is negligible. To give the reader an idea, the maximum difference in the average point error is around 0.05 pixels. The maximum difference in the accuracy is also very small, namely around 0.5%. This is expected since our method is designed to maintain the performance of the baseline RF on "easy" images. On the contrary, the improvement on LFW_TestII is noticeable. The reduction in the mean error for the *left eye left* point is around 4 pixels and that of the *right eye right point* is around 3 pixels. The differences on other points are not so noticeable. There are three points (*left eye left*, *left eye right* and *right eye right*) with more than 6% increase in detection accuracy.

## 4.3. Experiments on AFLW

We split the test set containing 1000 images into two sets, **AFLW_TestI** and **AFLW_TestII**, as we did the split in the LFW dataset. That is, we apply the plain CRF-S detector on the whole test dataset and put into the AFLW_TestI the face images with average localization error less than 0.1 (663 face images) and into the AFLW_TestII the rest (337 face images). We report results on them separately.

**Face center localization.** Since our proposed scheme is based on the face center detection, we evaluate the stability by measuring the nose tip localization error. As is shown in Figure 7a, though the localization is not highly accurate, the performance is very stable: only 2 out of the 1000 test images have localization error larger than 0.4 and more than 98% of them have localization error less than 0.3.

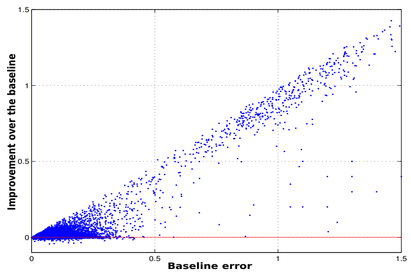
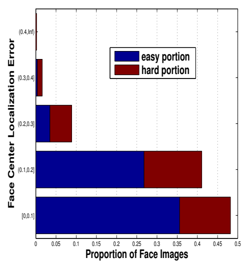


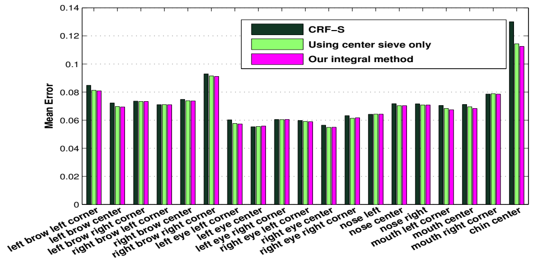
Figure 6: Improvement by our method against the error of CRF-S (all the 19 $\times$ 1000 points are shown).

**Performance of the sieves.** In Figure 7 we report results that summarize our evaluation of the efficacy of our sieves. In this comparison, we use the face bounding boxes from the dataset. In detail, on AFLW_TestI, the relative average error improvement is small, 1.3% (0.0706 vs. 0.0715). The sieves show improvements only on a few difficult points, e.g. the *left eye brow left corner* (4.6%, 0.0809 vs. 0.0848) and the *chin corner* (12%, 0.115 vs. 0.13). The procedure of updating individual sieve thresholds does not have much effect. On the contrary, the relative improvement on AFLW_TestII is large. The average relative improvement in mean error by using the face center sieve is 37% (0.1039 vs. 0.1648). By automatically updating the thresholds for individual sieve, there is a further 3% improvement in average and on difficult points, the relative improvement using threshold updating is more noticeable, e.g. 18% for both *left eye brow left corner* and *right eye brow right corner*. Figure 6 shows that the improvement of our method over the baseline CRF-S has clearly a positive correlation with localization error of CRF-S. This shows the efficiency of our method on difficult images. The reported results are averages over 4 runs. The improvements are statistically significant for 4/19 points on the easy images, for 18/19 on difficult images (all, except the nose tip) and for 16/19 points overall (all, except the 3 nose points).

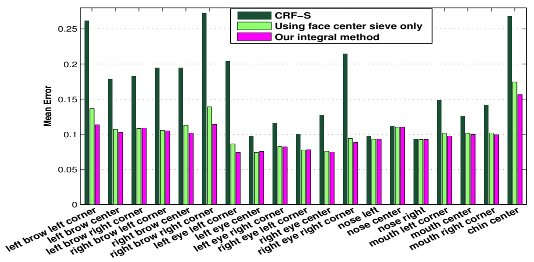
**Comparison with the state-of-the-art.** We compare the



(a)

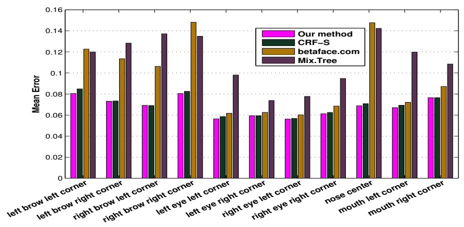


(b) Mean error on AFLW_TestI.

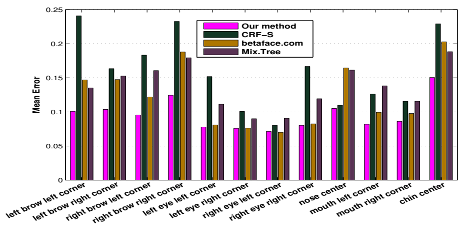


(c) Mean error on AFLW_TestII.

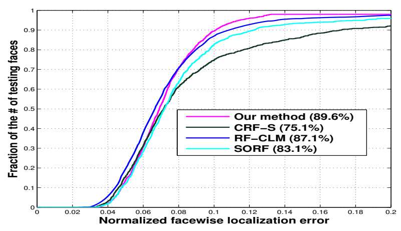
Figure 7: Performance on **AFLW**. (a) Distribution of face center localization error. (b) and (c), mean error results.



(a) Mean error comparison on AFLW_TestI.



(b) Mean error comparison on AFLW_TestII.



(c) Cumulative error distribution

Figure 8: Results of our method on **AFLW**, compared to [30, 27, 7] and betaface.com. The numbers in legend of (c) are the percentage of test faces that have average error below 0.1.

overall performance of our proposed method with methods from the academic community as well as commercial systems, namely (1) the mixture-of-trees (Mix.Tree) [30], (2) the structured-output regression forests (SO-RF) in [27], (3) the regression forests based CLM (RF-CLM) [7] and (4) betaface.com's face detection module [1]. Since betaface.com and Mix.Tree models combine face detection and landmarks detection, for fair comparison we build our algorithm on top of a Viola-Jones face detector from the Matlab computer vision toolbox. We manually discard missed or incorrect detections (e.g. sometimes Mix.Tree detected a half face) from all the methods when calculating the error. There are 74 missed face detections for betaface.com, 113 for Mix.Tree and 89 for matlab Viola-Jones detector. Mix.Tree failed to detect small faces because they were trained on high resolution face images. The intersection test set then contains 776 images (555 in AFLW_TestI and 221 in AFLW_TestII). We compare results of 12 common points to betaface.com and Mix.Tree as shown in Figure 8a and Figure 8b. On the AFLW_TestI we see that both CRF-S and our method perform better than Mix.Tree and betaface.com. On AFLW_TestII, CRF-S performs significantly worse while the other two methods, and our method have more stable performance. Our method has the best performance on both testing sets. In Figure 8c we compare the average localization error of all the 18 internal points on a face (the chin center is excluded) of our method with SO-RF and RF-CLM. We train SO-RF model on AFLW using the code in [27] using the same experimental setting as that of CRF-S and compare the reported result of RF-CLM. The markup of RF-CLM is slightly different since their results are for 17 points and two are annotated by the authors. Our method performs on par with RF-CLM and better than SORF and CRF-S though both RF-CLM and SORF are based on shape model fitting. An example image is shown in Figure 9 where our method performs better than not only the local detection method, like CRF-S, but also the ones using shape models such as [30, 27]. In addition, we have found that in terms of computational complexity and in terms of how well it deals with low quality images, our method performs considerably better than the Mix.Tree model. However, as shown in Figure 10, unlike the Mix.Tree, our method fails on side view faces since we have not used such images in training.

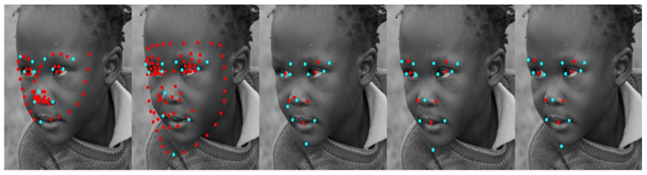


Figure 9: Left to right: Results for Mix.Tree, betaface.com, CRF-S, SO-RF and our method on an image from AFLW. The blue dots are the 12 common points listed in Figure 8a.

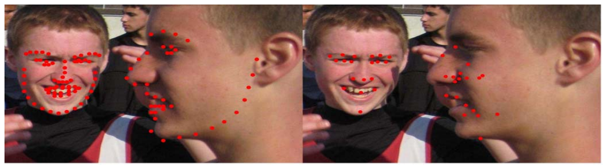


Figure 10: An example image from AFW [30] with results from Mix.Tree (Left) and our method (Right)

## 5. Conclusion and Future Work

In this paper we have proposed a cascade of sieves that are able to refine votes from regression forests for facial feature detection. By doing so, the Hough map of each facial point is only built on reliable votes. Our proposed framework achieves the state-of-the-art results on two public challenging datasets with face images in the wild, without resorting to explicit shape models.

Our results raise some interesting questions. Other than the face center consistency, can we develop more efficient ways to measure the quality of the votes before accumulating them into the Hough map? Can we extract more useful middle-level features from the votes for high-level vision tasks such as to measure the face similarity and to recognize the facial expression? We plan to investigate these questions in our future work.

## Acknowledgement

This work is partially supported by EU funded IP project REVERIE (FP-287723). Heng Yang thanks the CSC/QMUL joint PhD student scholarship and Yichi Zhang for inspiring and heated conversations.

## References

[1] http://www.betaface.com/.

[2] B. Amberg and T. Vetter. Optimal landmark detection using shape models and branch and bound. In *ICCV*, 2011.

[3] O. Barinova, V. Lempitsky, and P. Kholi. On detection of multiple object instances using hough transforms. *TPAMI*, 2012.

[4] P. Belhumeur, D. Jacobs, D. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *CVPR*, 2011.

[5] X. P. Burgos-Artizzu, P. Perona, and P. Dollár. Robust face landmark estimation under occlusion. In *ICCV*, 2013.

[6] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In *CVPR*, 2012.

[7] T. Cootes, M. C. Ionita, and S. P. Robust and accurate shape model fitting using random forest regression voting. In *ECCV*, 2012.

[8] T. Cootes, G. Wheeler, K. Walker, and C. Taylor. View-based active appearance models. *Image and Vision Computing*, 20(9-10), 2002.

[9] 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):81–227, 2011.

[10] M. Dantone, J. Gall, G. Fanelli, and L. Van Gool. Real-time facial feature detection using conditional regression forests. In *CVPR*, 2012.

[11] P. Dollár, P. Welinder, and P. Perona. Cascaded pose regression. In *CVPR*, 2010.

[12] B. Efraty, C. Huang, S. Shah, and I. Kakadiaris. Facial landmark detection in uncontrolled conditions. In *IJCB*, 2011.

[13] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *TPAMI*, 2011.

[14] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *ICCV*, 2011.

[15] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, Univ. of Massachusetts, Amherst, October 2007.

[16] S. Koelstra, M. Pantic, and I. Patras. A dynamic texture-based approach to recognition of facial actions and their temporal models. *TPAMI*, 32(11):1940–1954, 2010.

[17] P. Kontschieder, S. R. Bulò, M. Donoser, M. Pelillo, and H. Bischof. Evolutionary hough games for coherent object detection. *Computer Vision and Image Understanding*, 2012.

[18] M. Kostinger, P. Wohlhart, P. Roth, and H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *ICCV Workshops*, 2011.

[19] I. Patras and E. Hancock. Coupled prediction classification for robust visual tracking. *TPAMI*, 32(9):1553–1567, 2010.

[20] N. Razavi, J. Gall, P. Kohli, and L. Van Gool. Latent hough transform for object detection. In *ECCV*, 2012.

[21] O. Rudovic, M. Pantic, and I. Patras. Coupled gaussian processes for pose-invariant facial expression recognition. *TPAMI*, 2012.

[22] J. Saragih, S. Lucey, and J. Cohn. Face alignment through subspace constrained mean-shifts. In *ICCV*, 2009.

[23] M. Sun, P. Kohli, and J. Shotton. Conditional regression forests for human pose estimation. In *CVPR*, 2012.

[24] M. Valstar, B. Martinez, X. Binefa, and M. Pantic. Facial point detection using boosted regression and graph models. In *CVPR*, 2010.

[25] T. Wang, X. He, and N. Barnes. Learning structured hough voting for joint object detection and occlusion reasoning. In *CVPR*, 2013.

[26] H. Yang and I. Patras. Face parts localization using structured-output regression forests. In *ACCV*, 2012.

[27] H. Yang and I. Patras. Privileged information-based conditional regression forests for facial feature detection. In *FG*, 2013.

[28] X. Zhu, C. C. Loy, and S. Gong. Video synopsis by heterogeneous multi-source correlation. In *ICCV*, 2013.

[29] X. Zhu and D. Ramanan. Face detection, pose estimation and landmark localization in the wild. In *CVPR*, 2012.