Fast and Accurate Head Pose Estimation via Random Projection Forests

Donghoon Lee[1], Ming-Hsuan Yang[2], and Songhwai Oh[1]
[1]Electrical and Computer Engineering, Seoul National University, Korea
[2]Electrical Engineering and Computer Science, University of California at Merced
donghoon.lee@cpslab.snu.ac.kr, mhyang@ucmerced.edu, songhwai@snu.ac.kr

Abstract

In this paper, we consider the problem of estimating the gaze direction of a person from a low-resolution image. Under this condition, reliably extracting facial features is very difficult. We propose a novel head pose estimation algorithm based on compressive sensing. Head image patches are mapped to a large feature space using the proposed extensive, yet efficient filter bank. The filter bank is designed to generate sparse responses of color and gradient information, which can be compressed using random projection, and classified by a random forest. Extensive experiments on challenging datasets show that the proposed algorithm performs favorably against the state-of-the-art methods on head pose estimation in low-resolution images degraded by noise, occlusion, and blurring.

1. Introduction

The gaze of a person is important for a number of applications such as surveillance, human-computer interaction, and psychophysical studies, to name a few. In a surveillance system, the gaze of a person can be used to study interaction between people and characterize objects of interest [29]. Gaze information has been used in human-computer interaction for controlling smart devices and helping collaboration between humans and robots [26].

In this paper, we consider the problem of estimating the gaze of a moving person in a crowded scene, where head images are assumed to be obtained from a tracking or detection algorithm. As the image resolution is usually low, e.g., 50×50 pixels, it is especially challenging to estimate a person's gaze using their head image. A wide range of variations in skin color, hair styles, and head shapes exacerbate the problem [27]. The problem is further complicated since useful facial features cannot be reliably extracted from low-resolution images. An efficient gaze estimation algorithm is of great interest for practical applications, e.g., surveillance and human-computer interaction. However, it is difficult, if not impossible, to infer gaze estimation from a low-resolution image. Therefore, this problem, in practice, is posed as a head pose estimation task due to the high correlation of these visual cues. In this paper, we estimate the head pose in the discrete and continuous domains with classification and regression schemes.

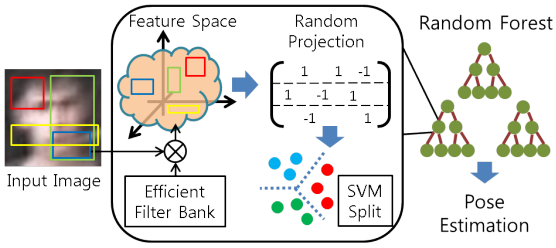


Figure 1. The proposed random projection forest algorithm. The responses of the designed filter bank are sparse and contains the color and gradient information of an image. Each node of a random forest compresses the responses by random projection. An SVM is trained using the compressed responses to split the data. The head pose is estimated by merging the distribution of leaf nodes.

In this work, we address the aforementioned challenging problems by exploiting expressive representation of compressive features and effective classification and regression of the proposed random projection forest as shown in Figure 1. To obtain compressive features, we first design an efficient filter bank that generates sparse high dimensional responses. The filter bank contains multi-channel, multi-scale, and multi-orientation box filters that capture color and gradient properties from an image. Then, the high-dimensional responses are compressed using random projection, preserving essential information of an image.

The random projection forest algorithm is based on the compressive features and a random forest [4]. The compressive features alone are not discriminative descriptors due to the generative framework of compressive sensing. When a random forest is constructed with compressive features where each node chooses the best random projection matrix based on the impurity measure (e.g., information gain), the whole classifier is likely to be more discriminative. In addition, when the random projection matrix satisfies the restricted isometry property (RIP) condition [8], the infor-

mation used to split the data is preserved at each node. Furthermore, the sparse form of a random projection matrix induces small correlations between trees by decreasing the probability of the same measurement being made twice. Therefore, the random projection forest is likely to have low generalization errors by strengthening the discriminability of each tree while weakening the correlation between trees. On the other hand, a random forest has been successfully used in numerous problems, e.g., classification, regression and clustering.

Extensive experiments on five challenging benchmark datasets are carried out to evaluate the proposed algorithm against the state-of-the-art methods for head pose estimation. The proposed algorithm performs well with a classification accuracy of 98% on the HIIT dataset and regression accuracy of 1.1° on the CMU Multi-PIE dataset where each frame is processed within a few milliseconds. The proposed approach performs well against other algorithms on low resolution images, (where each head image size is smaller than 50×50 pixels) and degraded images with noise, occlusion, and blurring. We also demonstrate that the proposed algorithm with a hierarchical structure using a random forest is more accurate and robust than alternative approaches.

2. Related Work

We present an overview of head pose estimation approaches in seven categories: appearance template, detector array, nonlinear regression, manifold embedding, flexible model, geometric, and tracking methods [22].

Appearance template methods divide training head images into a finite number of poses and generate prototypes for estimation with an SVM classifier [23]. Since two images of the same person in different poses are known to be more similar than images of different people in the same pose, such methods do not perform well [22].

Detector array methods [32] train multiple detectors for different pose estimation. However, it is difficult to resolve the situation in which two or more detectors identify the same head image as different poses. In addition, such classifiers can be easily biased due to unbalanced positive and negative training samples.

Head pose estimation can be posed as learning a regression function from the space of image features to 2D or 3D parameters. In [9], a mapping from the space of depth features to the corresponding head pose is learned using random regression forests. However, regression methods are less effective for low-resolution images as two images of the same person with different poses may be mapped closer than images of different people with the same pose [27].

Manifold embedding algorithms assume that head images form a low-dimensional manifold, on which similarity is measured for pose estimation. Recently, Tosato et al. [30] demonstrated state-of-the-art head pose estimation results using manifold embeddings and a weighted array of descriptors computed from overlapping patches, where each is described by a covariance matrix of image features. However, this method is computationally expensive.

Other approaches, such as flexible models, geometric methods, and tracking based algorithms are not closely related to this work (see references in [22]). Flexible models and geometric methods are suitable for analyzing larger images, in which facial structures or features can be extracted reliably. Tracking based algorithms determine head poses from consecutive observations, and the proposed method can be easily combined with such approaches.

Recently, Ho and Chellappa [15] presented a head pose estimation algorithm based on randomly projected dense SIFT descriptors and support vector regression. However, this method operates on larger sized images (as SIFT features need to be reliably extracted) with a random projection method proposed in [1] which is denser than the random projection approach presented in this work. Furthermore, it is computationally expensive to extract dense SIFT features.

3. Proposed Algorithm

In compressive sensing, the original signal x is compressed as follows:

$$y = Ax, \tag{1}$$

where A is an $m \times n$ matrix with $m \ll n$ and y is a compressed signal. In order to preserve the essential information of x using compressive sensing, x must be a sparse signal and the matrix A has to satisfy the RIP condition [8]. It is well known that an image can be represented by sparse coefficients in the wavelet domain [5] and the low dimensional vector y contains essential information of an image.

We note that we do not use discriminative features (e.g., HOG or SIFT) as low resolution images are considered in this work. In addition, it is computationally expensive to extract such features. Instead, we use a high dimensional feature space that encompasses all possible combinations from the proposed filter bank, which is designed to generate sparse responses efficiently. The filter responses are compressed via random projections and utilized for pose estimation using a random forest.

3.1. Efficient Filter Bank

We propose a multi-channel, multi-scale, and multi-orientation box filter bank that captures the color and gradient information from an image. The channels consist of the color $C = \{I_i | i \in \mathcal{C}\}$, gradient magnitude I_{mag}, and gradient orientation I_{ori}, where $\mathcal{C}$ consists of gray, RGB, HSV, and YCbCr color spaces. The filter bank contains two types of box filters, F^C and F^G, which are designed to collect color and gradient responses, respectively, as illustrated in Figure 2.

For the color filter response, F^C is parameterized by its width w, height h, and γ, which indicates the color channel. For a $\underline{w} \times \underline{h}$ input image, the value at (x, y) of a $w \times h$ filter

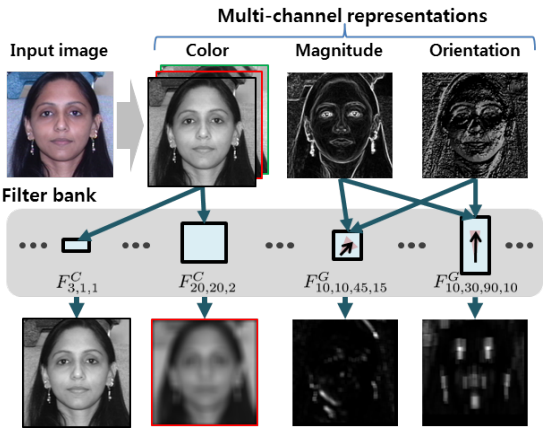


Figure 2. Four responses of the proposed filter bank. The filter bank is applied to each channel of the input image. The first two blue boxes are the filters from the color channel and the others extract gradient information. The filter bank contains all possible sizes inside the input image. Different θ and ϕ are represented by arrows and shaded regions in boxes, respectively.

is defined as

$$F^C_{w,h,\gamma}(x,y) = \frac{1}{wh} \times \begin{cases} 1, & \text{if } 1 \le x \le w, 1 \le y \le h, I_\gamma \in C \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where w and h represent all possible widths and heights of a box, i.e., $1 \le w \le \underline{w}$ and $1 \le h \le \underline{h}$. The convolved image is sparse in the wavelet domain. By concatenating all vectorized filter responses, we obtain a high dimensional descriptor of an image. However, computing all possible filter responses is computationally expensive since the number of possible boxes for a channel of 50×50 pixels already exceeds 10^7. A compressed representation of the filter responses is described in the next section.

For the gradient filter response, F^G considers the orientation of a gradient θ and an angle ϕ that quantizes the orientation. The value at (x, y) of a $w \times h$ filter is defined as

$$F^G_{w,h,\theta,\phi}(x,y) = \frac{1}{\mathrm{Z}} \times \begin{cases} 1, & \text{if } 1 \le x \le w, 1 \le y \le h, \\ & \theta - \phi \le I_{ori}(x', y') < \theta + \phi, \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where (x', y') is the location of (x, y) in the image and Z is a normalization constant which is equal to the number of nonzero elements in the filter response. The parameters vary in all possible ranges, i.e., $0 < \theta \le 2\pi$ and $0 < \phi \le \pi$, and w and h are the same as in the case of F^C. This filter is aimed to collect the averaged magnitudes of gradients within a certain orientation range. It resembles the Gabor filter bank but is more effective. The Gabor filter bank suffers from the curse of dimensionality due to dense filter responses. Typically, this issue is resolved by downsampling the magnitude responses of the Gabor filter bank using a grid or a feature selection scheme [28]. Nevertheless, the concatenated vector of the downsampled magnitude responses is still high dimensional. Usually, their dimension is further reduced by a subspace projection technique, such as principal component analysis (PCA) or linear discriminant analysis (LDA), with some loss of information. In the next section, we describe an efficient way to reduce the dimensionality of the filter responses while preserving the essential information.

3.2. Compact Representation of Filter Responses

The filters F^C and F^G generate sparse responses as shown in Figure 2. By concatenating all these responses into a vector, we represent an image in a high dimensional, sparse feature space. We apply compressive sensing to reduce the dimensionality of these filter responses using (1). Specifically, we adopt an $m \times n$ sparse random projection matrix [19] as follows:

$$a_{ij} = \sqrt{s} \times \begin{cases} 1, & \text{with probability } \frac{1}{2s}, \\ 0, & \text{with probability } 1 - \frac{1}{s}, \\ -1, & \text{with probability } \frac{1}{2s}, \end{cases} \tag{4}$$

where $s \in \mathrm{o}(n)$ and $A = [a_{ij}]$. By setting $s = n/\log(n) \in \mathrm{o}(n)$, the expected number of nonzero elements per row of the matrix A is $\log(n)$. Therefore, the actual number of calculated filter responses is exponentially decreased. This enables us to bypass computing all filter responses while preserving the essential information. The random matrix A needs to be computed only once off-line and is fixed while testing a new image. As a result, an element of the compressed vector is a weighted linear combination of random box filter responses. Note that when only F^C is used, the process is similar to the generalized Haar-like features [31].

By representing images with compressed vectors, we are able to carry out classification or regression tasks. However, features projected by a single random projection may not be discriminative or robust, as shown in Section 4. This is because a random projection matrix is designed with a random basis that does not take the training data into account. We address this issue with the proposed random projection forest algorithm that hierarchically selects random projection matrices which maximize the impurity measure.

3.3. Random Projection Forest

A random forest is an ensemble of decision trees where each node of a tree has its own split function. The split function divides the input data into two or more partitions which are delivered to each child node. During the training stage, a training set is fed to each tree and the leaves store the distribution of the samples. Each non-leaf node stores a split function which optimizes the impurity measure. In the test stage, an example traverses each random tree and reaches a leaf node. The probability distribution over the classes of poses is computed by taking the average of the distribution for all reached leaf nodes in the random forest.

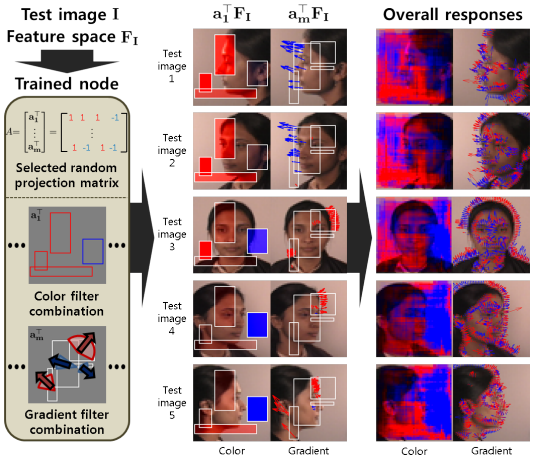


Figure 3. An example of a split function in a node. This node is trained to split images ranging from -90° to 90° of the yaw angle into two subsets: $[-90^\circ, 0^\circ)$ and $[0^\circ, 90^\circ]$. The middle column shows responses selected by row vectors of A and the right column shows the overall responses selected by A. The responses of color and gradient filters are represented with painted rectangles and arrows, respectively. Each red and blue response corresponds to +1 and -1 of the random projection matrix. Darker boxes and longer arrows represent stronger responses. The gradient responses that are larger than a threshold are shown for better visualization.

A random projection forest integrates a random forest with random projection. We perform random projection at each node and split the training data based on the compressed vector. Each node selects a random projection matrix given the input data that maximizes the impurity measure to make the tree more discriminative. It does not only generate a better basis to describe the training data but also makes the tree more robust to the bias in a single random projection matrix. Note that a random projection is a generalized representation of the split functions that have been widely used: each node splits the data using a small portion of input variables, or using a linear combination of them, which are special cases of the linear mapping in (1).

Figure 3 shows examples of selected filters by a node of the trained random projection forest. In this case, the node is trained to split head poses into left and right. Example combinations of selected filters are shown in the left column. The red and blue boxes or arrows represent $+\sqrt{s}$ and $-\sqrt{s}$ terms in the random projection matrix, respectively. For each head pose, selected filters generate different responses around the face, hair, neck, and background region as shown in the middle column. As a result, the combination of color filters for the first image yields positive value (responses of red boxes are strong) while the last image yields negative value (response of blue box is relatively strong). Gradient filters also generate distinguishable responses for each head pose. The responses from the first image are negative (blue arrows) while the last image yields positive responses (red arrows) from different box filters. The right column shows all selected filters in the node. The color filters show symmetrical responses for the left and right head pose images. The gradient filters generate responses along the head shape and they are informative since the response is different at the certain location of each head pose image.

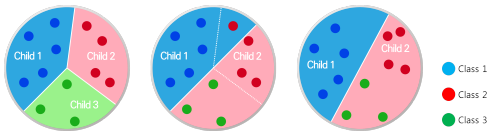
Another important aspect of this work is that random projections also help lower the generalization error of the random forest. It has been shown that the generalization error bound of a random forest is $\overline{\rho}(1-s^2)/s^2$, where $\overline{\rho}$ is the average correlation between trees and s is the strength of the trees [4]. To minimize the maximum generalization error, the correlation between trees needs to be minimized and the strength of trees has to be maximized. However, when we strengthen a tree, we also increase the correlation between different trees. In order to strengthen a tree, we need a large number or combination of input variables to learn better split functions. Simultaneously, the correlation between trees increases, as trees now have overlapped redundant information. Therefore, the number of considered variables or the number of linear combinations have been empirically chosen. In contrast, we use the random projection matrix which satisfies the RIP condition as the linear mapping, and analyze the proposed algorithm below.

Correlation Between Trees. As each node compresses the input data and splits them, it is important to obtain diverse representations of the compressed data to decrease the correlation between trees without losing important details from the input data. The random projection approach yields diverse representations of compressed data, which we can credit to the use of a random basis. Since the linear projection matrix described in (4) has 3^{mn} different representations and n is the number of filter responses, which is larger than 10^7, the probability that a different node has the same basis is negligible.

Strength of Trees. Trees with random projections that satisfy the RIP condition are always stronger, as the relative distance between samples, i.e., the information for the split, is preserved in each node. In other words, once the RIP condition is met, it is not necessary to measure an arbitrarily large number of input variables to strengthen the tree. Thus, for high dimensional sparse signals, the trade-off between the strength and the correlation of trees can be resolved by random projections. As demonstrated in experiments, this property can increase the discriminative capability of a random forest.

3.4. Random Projection Forest for Pose Estimation

Based on randomly projected compressive features at a tree node, a split function is trained using an SVM. By using an SVM, we can split the data with the maximum margin while efficiently evaluating test samples using the trained hyperplane. Three different types of SVMs (Figure 4) are considered for classification and regression. For



(a) (b) (c)

Figure 4. Three split function candidates considered in this paper. In this example, twelve training samples with three different classes reach the node. To incorporate multi-class SVMs into a forest, three options are considered. (a) Training multi-class SVM and dividing data by all hyperplanes. (b) Training multi-class SVM and dividing data by one of the hyperplanes. (c) Training binary SVM and dividing data by the hyperplane.

the discrete head pose estimation, where a small number of discrete head poses are considered, we use the multi-class SVM in Figure 4(a). On the other hand, for the continuous head pose estimation, a binary SVM is chosen, as multi-class SVMs are often unable to stably divide many classes at once. The benefit of the split is measured by the information gain.

During the training phase, a tree is grown until one of the following conditions is met: the tree reaches a pre-defined maximum depth, the number of samples in the node that comes from the same class exceeds 99%, or the number of samples in the node gets too small. In this paper, we limit tree depth to 10 levels, and each node is required to have a minimum of 10 samples.

The same forest is used in the test phase. A test sample reaches to a single leaf node for each tree. A posterior of the forest is calculated by averaging the stored posterior at reached leaf nodes. For classification, the head pose is classified as the class with the maximum probability. For regression, the head pose is estimated by averaging the posterior of the forest.

4. Experiments

We use the HIIT [30], QMUL [23], and QMUL with background datasets [30] for head pose classification experiments. In addition, we use the CMU-MultiPIE [12] and FacePix [20] datasets for head pose regression evaluations.

HIIT Dataset. The HIIT dataset contains 24,000 images with 6 head poses in a static background with no occlusions. The dataset is challenging because it consists of images from different datasets (e.g., QMUL [23] and CMU Multi-PIE [12]) with large variations in appearance.

QMUL Dataset. The QMUL dataset contains 15,660 images with 4 head poses at different illuminations with occlusions. The QMUL dataset with 3,099 additional background images is referred as QMULB in this paper. Both datasets are challenging as the images are acquired in airport terminals with heavy occlusions and large crowds.

CMU-MultiPIE Dataset. The images of the the CMU-MultiPIE database are acquired from 337 subjects with different poses from $-90°$ to $90°$ with $15°$ intervals and 13 yaw directions. For the experiments, we use all images of 6 expressions under the frontal light sources. We note these high-resolution images have been used in the past for other head pose estimation methods. The head region of each image must be cropped and aligned using manually annotating facial features. In this work, we consider more realistic scenarios. We crop 360×360 center pixels of the head images and downsample it to 50×50 pixels. Due to the varying height and facial shape of each person, the cropped images are not aligned, which is more suitable for real world applications. We use images from a randomly selected 50% of the subjects for training and the others for tests. This dataset is challenging as the images are acquired from a large number of different subjects with different expressions.

Table 1. Classification accuracy using the corrected dataset: Frobenius and CBH denote two algorithms from [30].

	Original dataset		Corrected dataset	
	Frobenius	CBH	Frobenius	CBH
HIIT	95.3%	96.5%	95.3%	95.7%
QMUL	93.2%	94.3%	94.3%	94.9%
QMULB	90.6%	91.2%	92%	92.2%

FacePix Dataset. This dataset contains 30 subjects and 181 images for each person (one image per yaw degree from $-90°$ to $90°$). There are total of 5,430 aligned head images with static backgrounds. We perform the leaving one subject out evaluation on this dataset. The dataset is challenging due to fine intervals in the yaw orientation.

We identify misclassified images in the HIIT and QMUL datasets and manually correct these labels (the corrected ground truth data will be released). Table 1 shows classification accuracy using the original and corrected datasets.

4.1. Analysis of the Proposed Algorithm

We examine the properties of the proposed method using different channels, random projection matrices, and random projection forest settings.

Effect of Different Channels. The color channels we consider are based on gray, RGB, HSV, and YCbCr. We have tested all combinations and report the results of four combinations in Table 2. Among all color channels, the gray color space plays the most important role. However, better results can be attained when filters from all color channels are utilized (which can be computed efficiently). Note that the best accuracy is obtained by combining the color channel and the gradient channel. To demonstrate the discriminability of the proposed filter bank, we use HOG features to represent head images. The classification accuracy with the HOG representation is 92% on the HIIT dataset at significantly higher computational cost.

Effects of Random Projection Matrices. The random projection matrix in (4) projects each sample from $\mathbb{R}^n$ to $\mathbb{R}^m$

Table 2. Effects of different channels. Color: Gray + RGB + HSV + YCbCr. All: Color + Gradient.

Channel	Gray	Color	Gradient	All
HIIT	94.1%	96.8%	94.6%	97.6%
QMUL	91.4%	92.1%	91.6%	94.3%
QMULB	84.7%	89.8%	82.9%	92.2%

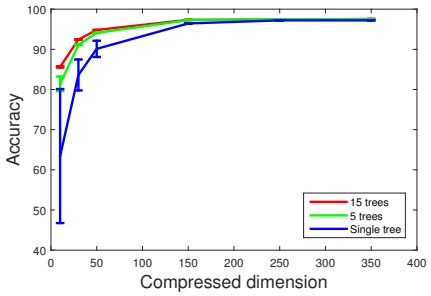


Figure 5. Estimation accuracy with respect to the dimension of the compressed vector for each node (HIIT dataset).

with a small number of nonzero elements governed by the parameter s. When the dimensionality of the projected domain is too small or the matrix is too sparse, the RIP condition does not hold. We show how the proposed algorithm performs with different values of m and s for the random projection matrix.

The head pose estimation accuracy at different values of m is shown in Figure 5. It shows that the proposed algorithm performs well in the 250-dimensional feature space. For comparisons, we apply PCA to reduce the dimensionality of features. The estimation accuracy is only about 50% on the HIIT dataset when PCA reduces the dimensionality of 50×50 pixels of head images to a 250 dimension.

We carry out experiments with denser random projection matrices by varying s. As s increases, the random projection matrix gets denser, which measures a larger number of rectangular filter responses. Figure 3(a) in the supplementary materials shows classification accuracy at different values of s when m is 250. We note that the proposed algorithm performs well using feature vectors with only four nonzero elements.

Effects of Random Projection Forest Parameters. We consider four parameters of a forest: the number of trees, the number of guesses to find the one that maximizes the information gain at each node, the compression rate at each node, and different ways to split the data at each node.

Figure 3(b) in the supplementary materials shows the effect of the number of trees and guesses where the average accuracy and error bars are computed from ten independent runs. High accuracy (and low variance in accuracy) is obtained with 15 trees and 10 guesses. This is a result of each node discovering a better split function when the number of trees and guesses are sufficiently large. This is another indication that the proposed forest outperforms a tree.

Table 3. Classification accuracy on the HIIT, QMUL, and QMULB datasets at different image sizes. [30]-a and [30]-b are methods proposed by [30] based on the Frobenius distance and the CBH distance, respectively. The results of [23] and [29] are from their paper.

Dataset	Size	[23]	[29]	[30]-a	[30]-b	Proposed
HIIT	15×15	-	-	82.4%	84.6%	**97.6**%
	25×25	-	-	89.6%	90.4%	**97.6**%
	50×50	-	-	95.3%	95.7%	**97.6**%
QMUL	15×15	-	-	59.5%	59.8%	**94.1**%
	25×25	-	-	82.6%	83.2%	**94.3**%
	50×50	82.3%	93.5%	94.3%	**94.9**%	94.3%
QMULB	15×15	-	-	54.5%	57%	**91.9**%
	25×25	-	-	76.5%	76.9%	**92.1**%
	50×50	64.2%	89%	92%	92.2%	**92.2** %

Figure 5 shows the effect of the number of trees and compression rate. The average accuracy and the variance are obtained from ten independent runs. When the dimension of the compressed vector is increased, the accuracy is improved but saturated after 250 dimensions. The results show that the forest achieves higher accuracy and lower variance in accuracy compared to the case of a single tree.

For a classifier at each node, we use both a linear SVM and a radial basis function (RBF) SVM. When using 50×50 pixel images, the RBF SVM outperforms the linear SVM by about 5% for all datasets. The parameters of an SVM are estimated by 5-fold cross validation to avoid overfittting.

The above experimental results show that the proposed random projection forest algorithm is insensitive to parameter changes and sensible values can easily be determined for accurate head pose estimation.

4.2. Evaluation of Head Pose Estimation Methods

Head Pose Classification. We evaluate the proposed algorithm against state-of-the-art head pose classification methods [23, 29, 30] in regard to image scale variation, noise, occlusion, blurring, and computational time. Table 3 summarizes the performance of head pose estimation methods on three datasets with different image sizes. Figure 6 shows the overall accuracy with respect to image sizes. Some confusion matrices for the HIIT and QMULB datasets are shown in Figure 7 and Figure 8, respectively. Overall, the proposed algorithm performs robustly with respect to size variation against the other methods. The proposed algorithm achieves almost the same estimation accuracy, for example, 97.6% on the HIIT dataset until the image size is reduced to 10×10 pixels. We note that it has been shown that estimating gaze direction from low-resolution images of 10×10 pixels can be achieved when provided feature vectors of head motion in videos [25]. However, this requires video inputs, and the method has not been quantitatively evaluated in terms of precision and recall on benchmark datasets. We note that accuracy from the method in [30] decreases rapidly when the image size is reduced below 50×50 pix-

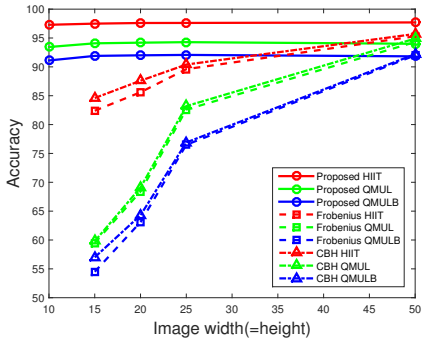


Figure 6. Classification accuracy at different image size. Frobenius: the Frobenius norm based method from [30]. CBH: the CBH norm based method from [30].

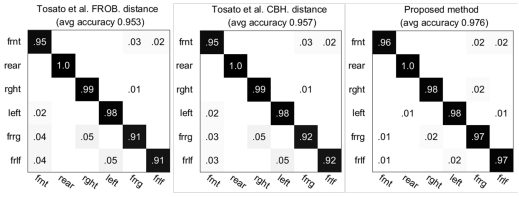


Figure 7. Confusion matrices of head pose estimation results on the images of 50×50 pixels from the HIIT dataset (frnt: front, rght: right, frrg: front right, frlf: front left).

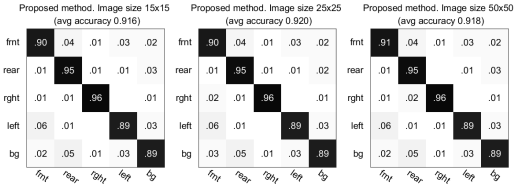


Figure 8. Confusion matrices of head pose estimation results on QMULB dataset at different image sizes. (bg: background)

els, and does not operate when the image size is smaller than 15×15 pixels (using the provided code).

The confusion matrices of the QMULB dataset show that the proposed algorithm is capable of estimating head poses while filtering out 90% of background images. Based on this observation, we estimate head poses (including background) on other datasets such as the Towncentre dataset [3] and the PETS2009 dataset [10] while the proposed algorithm is trained using the QMULB dataset. These datsets are challenging to estimate head poses because camera angles and lightning conditions are different from the QMULB dataset. We train the sliding-window based state-of-the-art detector [7] for the head detection. Finally, head poses are estimated as shown in Figure 9. The results show that false positives are effectively removed and head poses are fairly well estimated by the proposed algorithm.

We report the computational time required for head estimation on an image of 50×50 pixels. For the methods proposed in [30], the Frobenius norm based method and the CBH norm based method take 550 ms and 1,689 ms per image, respectively. In contrast, the proposed method takes

Figure 9. An example of head pose estimation result on the Towncentre dataset. Red arrows indicate the estimated direction. Its confidence score is written near the box. Dashed-line rectangles are the detections that are estimated as the background (i.e., false positive) by the proposed algorithm (see supplementary materials for more results).

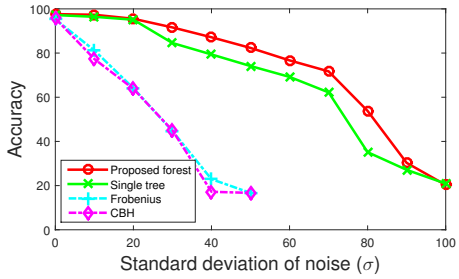


Figure 10. Head pose estimation accuracy at different noise levels. Since there are six classes, a random classifier can achieve an accuracy of 16.7% on average. Hence, we do not plot the results using methods from [30] when σ is larger than 50 (HIIT dataset).

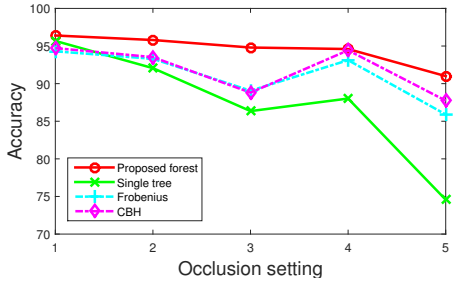


Figure 11. Head pose estimation accuracy at different occlusion settings (HIIT dataset).

only 10 ms per image, using 15 trees with parallel processing. All processing is done on a computer with a 3.3 GHz CPU. Overall, the proposed algorithm is about 170 times faster than the CBH norm based method [30].

We analyze the performance of each method when facing noisy test images. We add Gaussian noise with kernel width σ to each test image of 50×50 pixels and ensure that the intensity of the corrupted pixel value is between 0 and 255. Figure 10 shows that the proposed method performs better than other methods against large image noise.

We evaluate head pose estimation methods and analyze how they perform when the input images are occluded. The occluded images are generated in five settings as depicted

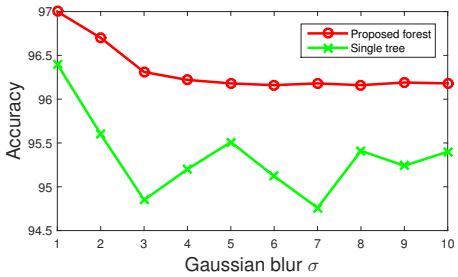


Figure 12. Head pose estimation accuracy with blurry images (with 5×5 Gaussian filter of different width on the HIIT dataset).

in Figure 2 of the supplementary materials: randomly generated (1) one 10×10 rectangle, (2) two 10×10 rectangles, (3) three 10×10 rectangles, (4) one 15×15 rectangle, and (5) two 15×15 rectangles. The intensity value of each pixel in the occluded region ranges between 0 and 255. Figure 11 shows that our method performs well against [30] when faced with occluded images. This can be attributed to the fact that, unlike the methods based on holistic representations, the proposed algorithm obtains information from multiple local patches, thus allowing good performance on images that are partially occluded.

We evaluate whether the proposed algorithm performs well on blurry low-resolution images as shown in Figure 12, where the images are degraded with a 5×5 Gaussian kernel with different width, σ. Overall, the proposed algorithm performs well with different settings. In contrast, the accuracy of the manifold method [30] decreases significantly. With a small Gaussian kernel width of one pixel, the accuracy decreases by 65%. When the kernel width is larger than one pixel, the manifold method works as a random classifier, i.e., the accuracy is 16.7% for six classes.

Head Pose Regression. We compare the proposed algorithm with the state-of-the-art head pose regression methods. Table 4 and 5 summarize the performance of head pose regressors on the CMU Multi-PIE dataset and the FacePix dataset, respectively. We use the same parameters as those in the classification task. The mean absolute error (MAE) between the estimated head pose and ground truth head pose in degree is computed for each method. Overall, the proposed algorithm performs favorably against the other methods for head pose regression.

We note that the CMU Multi-PIE and FacePix datasets are not developed specifically for pose estimation, and existing methods in the literature use different numbers of subjects and images for experiments. For comparison, we report the number of subjects and images used for the evaluation on head pose regression in Table 4. The proposed algorithm is evaluated on more subjects and images than any other approaches. Overall, the proposed algorithm performs favorably against other methods.

For the FacePix dataset, the methods [11, 21] use the same evaluation scheme; the leave-one-out cross validation on the original dataset, that we report in this work. In [17], the yaw interval of the dataset is set to 2 degrees (instead of 1 degree) and 5 subjects are used for training, leaving 25 subjects for tests. The method [2] is trained with a 3D dataset and evaluated on the yaw degrees ranging from $-45°$ to $45°$. The approach [6] uses the yaw ranging from $-45°$ to $45°$ with $15°$ interval for experiments where 5 subjects are used for training and 25 subjects for testing. In contrast, the proposed algorithm performs favorably against the other methods, based on an evaluation of the entire dataset (i.e., 5,430 images with yaw degree from $-90°$ to $90°$ and leave-one-out cross validation). As the source code of the previously mentioned methods are not available to the public, we are unable to carry out experiments using noisy or occluded images.

Table 4. Regression accuracy on the Multi-PIE dataset. # S: number of subjects used in the experiment. # I: number of images used in the experiment. MAE: Mean absolute error in degrees.

	[13]	[16]	[24]	[18]	[14]	Proposed
# S	144	336	30	337	337	337
# I	2,700	5,648	540	8,762	32,682	32,682
MAE	5.31°	4.33°	4.12°	2.99°	1.25°	**1.12°**

Table 5. Regression accuracy on the FacePix dataset. MAE: Mean absolute error in degrees.

	[17]	[2]	[11]	[21]	[6]	Proposed
MAE	6.1°	3.96°	2.75°	2.74°	2.71°	**2.38°**

5. Conclusions

In this paper, we propose a fast and accurate head pose estimation algorithm by exploiting compressive features and a random projection forest. Compressive features are obtained by compressing the responses of a large filter bank that captures the color and gradient information of an image. The proposed random projection forest algorithm effectively splits the compressive features for effective head pose estimation. In addition, the proposed algorithm achieves high accuracy with a fraction of the running time. Extensive experiments on challenging benchmark datasets show that the proposed algorithm performs favorably against the state-of-the-art methods on low-resolution images degraded by noise, occlusion, and blurring.

Acknowledgements. The work of D. Lee and S. Oh is supported in part by a grant to Bio-Mimetic Robot Research Center funded by Defense Acquisition Program Administration and Agency for Defense Development (UD130070ID) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2013R1A1A2065551). The work of M.-H. Yang is supported in part by NSF CAREER Grant (No.1149783) and NSF IIS Grant (No.1152576).

References

[1] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687, 2003. 2

[2] A. Asthana, T. K. Marks, M. J. Jones, K. H. Tieu, and R. MV. Fully automatic pose-invariant face recognition via 3D pose normalization. In *Proc. of the IEEE International Conference on Computer Vision*, 2011. 8

[3] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *Proc. of the IEEE Computer Vision and Pattern Recognition*, pages 3457–3464. IEEE, 2011. 7

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 1, 4

[5] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *Signal Processing Magazine*, 25(2):21–30, 2008. 2

[6] A. Dahmane, S. Larabi, I. M. Bilasco, and C. Djeraba. Head pose estimation based on face symmetry analysis. *Signal, Image and Video Processing*, pages 1–10, 2014. 8

[7] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014. 7

[8] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006. 1, 2

[9] G. Fanelli, J. Gall, and L. V. Gool. Real time head pose estimation with random regression forests. In *Proc. of the IEEE Computer Vision and Pattern Recognition*, 2011. 2

[10] J. Ferryman and A. Shahrokni. PETS2009: Dataset and challenge. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 1–6. IEEE, 2009. 7

[11] J. Foytik and V. K. Asari. A two-layer framework for piecewise linear manifold-based head pose estimation. *International journal of computer vision*, 101(2):270–287, 2013. 8

[12] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010. 5

[13] M. A. Haj, J. Gonzalez, and L. S. Davis. On partial least squares in head pose estimation: How to simultaneously deal with misalignment. In *Proc. of the IEEE Computer Vision and Pattern Recognition*, 2012. 8

[14] B. Han, S. Lee, and H. S. Yang. Head pose estimation using image abstraction and local directional quaternary patterns for multiclass classification. *Pattern Recognition Letters*, 45:145–153, 2014. 8

[15] H. T. Ho and R. Chellappa. Automatic head pose estimation using randomly projected dense SIFT descriptors. In *Proc. of the IEEE International Conference on Image Processing*, 2012. 2

[16] D. Huang, M. Storer, F. D. la Torre, and H. Bischof. Supervised local subspace learning for continuous head pose estimation. In *Proc. of the IEEE Computer Vision and Pattern Recognition*, 2011. 8

[17] H. Ji, R. Liu, F. Su, Z. Su, and Y. Tian. Robust head pose estimation via convex regularized sparse regression. In *Proc. of the IEEE International Conference on Image Processing*, 2011. 8

[18] F. Jiang, H. K. Ekenel, and B. E. Shi. Efficient and robust integration of face detection and head pose estimation. In *Proc. of the IEEE International Conference on Pattern Recognition*, 2012. 8

[19] P. Li, T. J. Hastie, and K. W. Church. Very sparse random projections. In *Proc. of the ACM International Conference on Knowledge Discovery and Data mining*, 2006. 3

[20] D. Little, S. Krishna, J. Black, and S. Panchanathan. A methodology for evaluating robustness of face recognition algorithms with respect to variations in pose angle and illumination angle. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2005. 5

[21] B. Ma, R. Huang, and L. Qin. Vod: A novel image representation for head yaw estimation. *Neurocomputing*, 148:455–466, 2015. 8

[22] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):607–626, 2009. 2

[23] J. Orozco, S. Gong, and T. Xiang. Head pose classification in crowded scenes. In *Proc. of the British Machine Vision Conference*, 2009. 2, 5, 6

[24] X. Peng, J. Huang, Q. Hu, S. Zhang, and D. Metaxas. Head pose estimation by instance parameterization. In *Proc. of the IEEE International Conference on Pattern Recognition*, 2014. 8

[25] N. Robertson and I. Reid. Estimating gaze direction from low-resolution faces in video. In *Proc. of the European Conferecne on Computer Vision*, 2006. 6

[26] K. Sakita, K. Ogawara, S. Murakami, K. Kawamura, and K. Ikeuchi. Flexible cooperation between human and robot by interpreting human intention from gaze information. In *Proc. of the IEEE Intelligent Robots and Systems*, 2004. 1

[27] T. Siriteerakul. Advance in head pose estimation from low resolution images: A review. *International Journal of Computer Science Issues*, 9(2), 2012. 1, 2

[28] V. Struc, R. Gajsek, and N. Pavesic. Principal gabor filters for face recognition. In *Proc. of the IEEE International Conference on Biometrics: Theory, Applications, and Systems*, 2009. 3

[29] D. Tosato, M. Farenzena, M. Spera, V. Murino, and M. Cristani. Multi-class classification on Riemannian manifolds for video surveillance. In *Proc. of the European Conference on Computer Vision*, 2010. 1, 6

[30] D. Tosato, M. Spera, M. Cristani, and V. Murino. Characterizing humans on Riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1972–1984, 2013. 2, 5, 6, 7, 8

[31] K. Zhang, L. Zhang, and M.-H. Yang. Fast compressive tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(10):2002–2015, 2014. 3

[32] Z. Zhang, Y. Hu, M. Liu, and T. Huang. Head pose estimation in seminar room using multi view face detectors. In *International Evaluation Workshop on Classification of Events, Activities and Relationships*, 2007. 2