Regressing a 3D Face Shape from a Single Image

Sergey Tulyakov and Nicu Sebe
University of Trento, Italy
sergey.tulyakov@unitn.it, sebe@disi.unitn.it

Abstract

In this work we present a method to estimate a 3D face shape from a single image. Our method is based on a cascade regression framework that directly estimates face landmarks locations in 3D. We include the knowledge that a face is a 3D object into the learning pipeline and show how this information decreases localization errors while keeping the computational time low. We predict the actual positions of the landmarks even if they are occluded due to face rotation. To support the ability of our method to reliably reconstruct 3D shapes, we introduce a simple method for head pose estimation using a single image that reaches higher accuracy than the state of the art. Comparison of 3D face landmarks localization with the available state of the art further supports the feasibility of a single-step face shape estimation. The code, trained models and our 3D annotations will be made available to the research community.

1. Introduction

Over the last several years 2D face alignment has reached maturity making it possible to detect landmarks in the wild at very high frame rates [12, 20, 13, 5, 21]. The mainstream direction is based on learning a sequence of regressors in a cascade fashion starting from the mean shape, and consequently refining the shape prediction at the later stages of the cascade [8, 2]. Previous works for face alignment used Active Appearance Models (AAM) [6, 18, 9] and Constrained Local Models (CLM) [7, 25, 22]. Cascaded regressor methods are advantageous over the AAM and CLM based approaches in several respects: (i) running a sequence of regressors is faster than solving an optimization problem for every image, (ii) the offline training stage allows cascaded approaches to take advantage of the large available sets of training images, (iii) shape-invariant feature sampling makes these methods robust to rotations.

As a natural extension of 2D face alignment methods, 3D face analysis from a 2D image also experiences significant breakthrough [12, 5] reaching comparable results to depth-based methods [27, 26, 16, 24]. These methods first detect 2D landmarks and as a second step fit a previously learned high resolution 3-dimensional face model to estimate a face shape. However, in many applications this high precision face shape estimation is not always required, while frame rates and low hardware requirements often become more critical [11, 30].

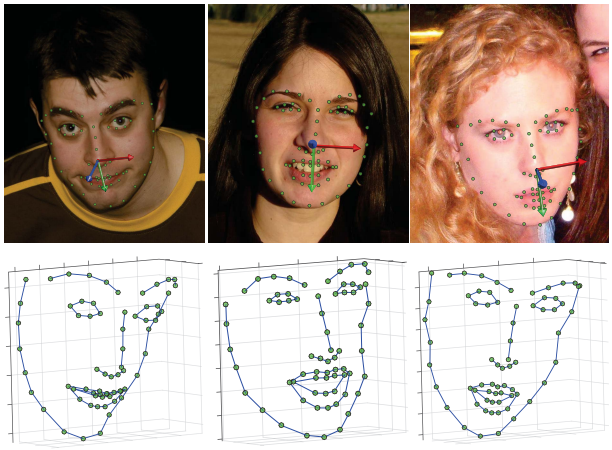


Figure 1: Selected examples of images from the HELEN database [15] processed by our method. Top: the projections of the landmarks to the image plane. Arrows represent the face bases used to find the head pose. Bottom: estimated faces shapes in the world coordinate system.

In this work, we present a novel cascaded regressor-based method to estimate a 3D shape of a face from a single 2D image. Motivated by the recent success of sequential approaches for 2D face alignment, we extend the framework to naturally detect 3D landmarks positions from a single image at state-of-the-art accuracy and processing speed. In contrast to existing two-step systems that first detect landmarks and only then recover a 3D shape, our method provides a reliable estimate of a face shape in one step. To support the applicability of our method to real-world problems we report the results on a large set of face images. Moreover, since our method outputs a 3D shape, we show how this shape can be used to accurately estimate the head pose of a face improving state-of-the-art accuracy. Selected examples of face landmarks estimated by our method are

given in Figure 1.

The contribution of our work is threefold:

1. **Single-step 3D face shape estimation**. Previous two-step face shape recovery methods first detected 2D face landmarks. Shape estimation was then done by fitting a 3D model to the estimated landmarks. We use 3D information in the learning pipeline and propose 3D shape invariant feature indexing. To the best of our knowledge, we are the first to estimate 3D face landmarks in a single step fashion.

2. **Localization of actual facial landmarks.** Prior works focused on either removing the occluded landmarks from the pipeline or using the nearest visible points instead. Our method estimates the actual 3D positions of the landmarks even if some points of the shape are not visible due to self-occlusions of the face. This helps us preserve the face shape during training and testing and accurately estimate head pose orientation.

3. **Data and the code.** We release our 2D and 3D annotations of the BU-4DFE [31] database as well as the code for our system to the research community.

The paper is structured as follows: in Section 2 we review relevant works. Section 3 describes the proposed method, starting with the description of the framework of cascade regressors (Section 3.1). In Section 3.2 we describe the shapes learned by our method. Section 3.3 proposes three ways of indexing features in 3D. The head pose estimation method is given in Section 3.4. We report our experimental results in Section 4 and conclude in Section 5.

2. Related work

Three-dimensional face shape estimation from a single RGB image is not a new topic in computer vision. Many works are done in the context of pose-invariant face recognition [19, 1, 23, 30]. The classical work of Blanz and Vetter [1] uses manual initialization as a first step, followed by a 3D model fitting. They achieve very accurate face models at a cost of low processing speed.

In [30] a low resolution model is fit to 2D landmarks for determining feature sampling points. Although fitting the final model is far from being perfect, the authors report improvement in face recognition results, while getting reasonable processing speed.

The problem of estimating a face shape is tackled from a different perspective in [11]. The author shows a SIFTFlow-based [17] method to warp a depth-RGB image pair of a reference person to a single RGB image of a query person. Consequently, the method is rather slow and can estimate depth only for visible parts of the face. Again, the first step is performed by the 2D face alignment system presented in [32].

Several commercial systems have appeared. Vizago[1] and FaceGen[2] both require careful manual initialization, and therefore are two-step systems.

Cao et al. [3] tackles the problem of automated avatar animation. They propose to jointly estimate a parametric 3D face model together with 2D landmarks from a video of a human performer. The method uses the landmarks estimated for the previous frame to simultaneously regress the 3D and 2D shapes for the current frame. However, when applied to a single image, the previously estimated landmarks are not available, and the method requires 2D landmarks estimated using [4] as an initialization, which makes it a two-step method according to our classification.

A very recent work of Jeni et al. [12] while still belonging to the two-step group, reaches significant frame rate increase. The authors use a cascade regressor framework similar to the one presented in [29] to estimate a dense grid of 2D landmarks and fit a 3D part-based model to this grid as a second step.

The major difference of our work with respect to the described two-step systems [19, 1, 23, 30, 11, 3, 12] is that our method is single-step and requires only a single image. A key advantage of our single-step method is that it is much faster, since it does not require computationally expensive 3D model fitting, while providing highly accurate shape estimates. We detect 3D landmarks from a single image by using a framework of cascade regressors. We evaluate our method on a large corpus of faces annotated in 3D and show that the estimated shapes can be used for head pose estimation with higher accuracy than methods previously presented in the literature.

3. Method

Although our work extends [13], we introduce several major novelties as compared to the original paper: (i) our shape estimates are 3D, (ii) we propose and compare three methods for 3D shape invariant feature indexing, and (iii) we show a 3D shape-based head pose estimation method that improves state-of-the-art accuracy while being computationally simple.

3.1. A Framework of Cascade Regressors

A general cascade regression approach produces an estimate $\hat{\boldsymbol{S}}$ of a shape $\boldsymbol{S}$ for an image of a face I by producing several shape increments $\Delta\boldsymbol{S}_t$ $(t = 1, ..., N)$ at every level t of the cascade in the following fashion:

$$\Delta\boldsymbol{S}_t = r_t(H_t(I, \hat{\boldsymbol{S}}_{t-1})), \quad (1)$$

$$\hat{\boldsymbol{S}}_t = \hat{\boldsymbol{S}}_{t-1} + \Delta\boldsymbol{S}_t, \quad (2)$$

[1] http://www.vizago.ch

[2] http://www.facegen.com/

where H_t is a feature extraction function, r_t is a regressor function learned at the t^{th}-level of the cascade and N is the total number of levels in the cascade. A shape vector $\boldsymbol{S} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]^T$ represents a set of facial landmarks. We denote $\hat{\boldsymbol{S}} = \boldsymbol{r}(I, \bar{\boldsymbol{S}})$ as the final shape estimate made by the cascade of regressors $\mathbf{r}(\cdot, \cdot)$ for an image I and the initial average shape $\bar{\boldsymbol{S}}$.

In previous works, every point $\boldsymbol{x}_i$ of the face shape vector was represented either by x, y-coordinates in the image, or was augmented by an additional label m_i that represents a flag indicating whether a point is visible or occluded: $\boldsymbol{x}_i = [x_i, y_i, m_i]^T$. Hereinafter we drop the index i and write $\boldsymbol{x}$ to denote a point of a shape to simplify the notation. Instead of adding an extra flag for every point we augment the usual x, y-coordinates of a point in the plane with the z-coordinate of the landmarks in the 3D space. Having a third dimension in the training set at every step of the cascade we learn the 3D shape increment $\Delta \boldsymbol{S}_t \in \mathbb{R}^{3\times n}$.

The feature extraction function $H_t(I, \hat{\boldsymbol{S}}_{t-1})$ in eq. 1 depends not only on image I but also on the previous shape estimate $\hat{\boldsymbol{S}}_{t-1}$, this allows the cascade to extract shape independent features. We propose to extend a face shape with the third dimension so that $\Delta\, \boldsymbol{S}_t,\ \hat{\boldsymbol{S}}_t,\ \boldsymbol{S} \in \mathbb{R}^{3\times n}$. Several models can be used as a regressor; we train a number of regression trees at each level of the cascade, since they have shown remarkable results in the literature [20, 13].

3.2. From World Coordinates to 3D Landmarks

To learn a face shape predictor one has to decide upon the landmarking scheme and perform annotation of the available training data. In our case, such annotation is hardly possible even for a human annotator due to the difficulty to estimate a z-coordinate by observing just a single 2D image. However, we propose the solution based on performing the 2D annotation as usual, and then augmenting the annotation of the z-coordinate estimated in a different way. To do so we use the available 2D+3D database BU-4DFE [31]. Manual annotation is performed on a frontal set of images provided in the database. Since 2D-3D correspondences are known, we map 2D coordinates of the point in a frontal RGB image to the corresponding 3D point on the mesh.

To generate various head poses for training and testing purposes we render meshes under tilt and yaw rotations uniformly distributed in the range of $[-50, 50]$ degrees. Since the rendering parameters are known, we can get the locations of the points by using the pinhole camera model:

$$\lambda \boldsymbol{x}^c = \boldsymbol{A}\boldsymbol{R}\boldsymbol{x}^w + \boldsymbol{t}, \tag{3}$$

where $\boldsymbol{x}^w = [x^w, y^w, z^w]^T$ is the point in the world coordinate system, $\boldsymbol{x}^c = [x^c, y^c, 1]^T$ is the point in the camera coordinates, λ is the homogeneous scaling factor, $\boldsymbol{A}$ is the matrix of intrinsic parameters or the camera matrix, $\boldsymbol{R}$ and $\boldsymbol{t}$ are the rotation matrix and the translation vector correspondingly. We note here that the z-coordinate is still available after the transformation. We augment the point in the camera coordinates with this z-coordinate to form $\tilde{\boldsymbol{x}}^c = [x^c, y^c, \lambda]$. In this way every training example is formed by $\{I(\text{yaw}, \text{tilt}), \tilde{\boldsymbol{S}}^c\}$, where $\tilde{\boldsymbol{S}}^c = [\tilde{\boldsymbol{x}}_1^c, \tilde{\boldsymbol{x}}_2^c, ..., \tilde{\boldsymbol{x}}_n^c]^T$. Although $\tilde{\boldsymbol{S}}^c$ is a 3D shape, its points are distorted by the camera matrix and therefore, proportions no longer correspond to the normal face proportions. This needs to be compensated. To this end we define a point $\boldsymbol{x}^{wR} = \boldsymbol{R}\boldsymbol{x}^w$ and a corresponding shape $\boldsymbol{S}^{wR}$ which is rotated according to the extrinsic rotation matrix, while being represented in the world coordinates. During testing a cascade of regressors produces a shape estimate $\hat{\boldsymbol{S}}^c = \boldsymbol{r}(I, \bar{\boldsymbol{S}}^c)$, where the shape $\hat{\boldsymbol{S}}^c$ is given by augmented points: $\hat{\boldsymbol{S}}^c = [\hat{\boldsymbol{x}}_1^c, \hat{\boldsymbol{x}}_2^c, ..., \hat{\boldsymbol{x}}_n^c]^T$. Then, if the camera matrix $\boldsymbol{A}$ is known, we can rewrite eq. 3 to get $\hat{\boldsymbol{x}}^{wR}$:

$$\hat{\boldsymbol{x}}^{wR} = \boldsymbol{A}^{-1}(\lambda \hat{\boldsymbol{x}}^c - \boldsymbol{t}). \tag{4}$$

However, at testing time the matrix $\boldsymbol{A}$ is unknown, and therefore needs to be estimated. To get the estimate $\hat{\boldsymbol{A}}$ we perform camera calibration using $\bar{\boldsymbol{S}} \in \mathbb{R}^{3\times n}$ as the coordinates in the world coordinate system and only x, y-values of the points in $\hat{\boldsymbol{S}}^c$ as the coordinates in the image plane. Finally, we substitute $\hat{\boldsymbol{A}}$ into eq. 4 to get $\hat{\boldsymbol{x}}^{wR}$.

We train and test our model on the actual landmarks positions even if they are invisible because of face rotations. Figure 2 shows an example of this. In other works, the closest visible pixels to the invisible landmarks are usually used instead. For example, the boundary of the face is often considered as a jawline when the actual jawline is not visible. However, this operation changes the natural proportions of the estimated shape, which is acceptable for two-step systems, where the shape is regularized during the second step.

Our experiments show that it is possible to estimate the actual 3D positions of the invisible points. Moreover, since

Figure 2: An example of the actual landmark positions. Left image shows an annotated mesh with several landmarks occluded. Central image shows the landmarks on the frontal face. Right image shows the projections of the actual landmarks onto the image plane.

a recovered shape is unchanged and is represented in the world coordinates we can accurately determine the head pose (see Sections 4.1 and 4.2).

3.3. 3D Invariant Features

At every level of the cascade, we build tree-based regressors to produce a shape increment. The decision function of a tree uses simple intensity difference features extracted at the points $\boldsymbol{u}$ and $\boldsymbol{v}$ indexed with respect to a mean shape. The points $\boldsymbol{u}$ and $\boldsymbol{v}$ are randomly generated during training. The goal of feature indexing is to have a way to compute $\boldsymbol{u}$ and $\boldsymbol{v}$ for every face geometrically close to their true locations, taking into account scaling, rotation and translation.

Indexing starts by defining an offset from $\boldsymbol{u}$ to the nearest point $\boldsymbol{x}_{k_{\boldsymbol{u}}}$ in the mean shape (we follow the notation in [13]):

$$\delta \boldsymbol{x}_{\boldsymbol{u}} = \boldsymbol{u} - \bar{\boldsymbol{x}}_{k_{\boldsymbol{u}}}, \tag{5}$$

where $\delta \boldsymbol{x}_{\boldsymbol{u}}$ is selected during training. To determine $\boldsymbol{u}'$, a point geometrically corresponding to the point $\boldsymbol{u}$, we first find the scaling and rotation transformations between the mean shape $\bar{\boldsymbol{S}}$ and the current shape estimate $\hat{\boldsymbol{S}}_t$:

$$\{s, \boldsymbol{R}, \boldsymbol{t}\} = \underset{s,\boldsymbol{R},\boldsymbol{t}}{\operatorname{argmin}} \sum_{i=1}^{n} \|\bar{\boldsymbol{x}}_i - (s\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t})\|^2, \tag{6}$$

where $s, \boldsymbol{R}, \boldsymbol{t}$ represent scaling, rotation and translation correspondingly. Then $\boldsymbol{u}'$ is determined in the following way:

$$\boldsymbol{u}' = \boldsymbol{x}_{k_{\boldsymbol{u}}} + \frac{1}{s}\boldsymbol{R}^T \delta \boldsymbol{x}_{\boldsymbol{u}}. \tag{7}$$

If one considers the case when $\boldsymbol{S} \in \mathbb{R}^{2\times n}$, then the rotation matrix $\boldsymbol{R} \in \mathbb{R}^{2\times 2}$, which accounts for in-plane rotations, such as roll angle.

To address head rotation from a 3D perspective, for the current cascade level t we define a face basis $\boldsymbol{F}_t$. The basis is spanned by the normal $\vec{\boldsymbol{n}}_t$, the vector connecting the centers of the eyes $\vec{\boldsymbol{e}}_{1,t}$ and $\vec{\boldsymbol{e}}_{2,t} = \vec{\boldsymbol{n}}_t \times \vec{\boldsymbol{e}}_{1,t}$, where $\times$ is a cross product operation. The vector $\vec{\boldsymbol{n}}_t$ is determined as the eigenvector with the smallest eigenvalue of the following covariance matrix:

$$\boldsymbol{C}_t = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_{i,t-1} - \bar{\boldsymbol{x}}_{t-1})(\boldsymbol{x}_{i,t-1} - \bar{\boldsymbol{x}}_{t-1})^T, \tag{8}$$

where $\boldsymbol{x}_{i,t-1} \in \hat{\boldsymbol{S}}_{t-1}$, $\boldsymbol{C}_t \in \mathbb{R}^{3\times 3}$. Since the direction of the normal vector $\vec{\boldsymbol{n}}_t$ can vary from iteration to iteration, depending on the face rotation, to obtain the normal consistently oriented with the observer direction $\vec{\boldsymbol{n}}_o$, we need to satisfy the following equation:

$$\vec{\boldsymbol{n}}_t \cdot \vec{\boldsymbol{n}}_o > 0, \tag{9}$$

where $\cdot$ is a dot product operation. We assume that $\vec{\boldsymbol{n}}_o$ is perpendicular to the image plane and directed to the observer. Having the basis $\boldsymbol{F}_t$ and the estimated scaling s we rewrite eq. 7 in the following way:

$$\tilde{\boldsymbol{u}}' = \boldsymbol{x}_{k_{\boldsymbol{u}}} + \frac{1}{s}\boldsymbol{F}_t^T \delta \tilde{\boldsymbol{x}}_{\boldsymbol{u}}, \tag{10}$$

where $\delta \tilde{\boldsymbol{x}}_{\boldsymbol{u}} = [\delta \boldsymbol{x}_{\boldsymbol{u}}|0]$, the operation of vector concatenation $[\cdot|\cdot]$ adds the third dimension, so that $\delta \tilde{\boldsymbol{x}}_{\boldsymbol{u}}, \tilde{\boldsymbol{u}}' \in \mathbb{R}^3$. After the transformation, this third dimension is truncated. In this way we find the coordinates of the offset vector in the face basis $\boldsymbol{F}_t$. Now we define three ways of indexing features:

- **Baseline** indexing is based on directly using eq. 7. In this case the only difference with the original method [13] is that the learned shape is 3-dimensional.
- **3D transform** indexing. The difference with the baseline method is that minimization in eq. 6 is performed in a 3D space, resulting in rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$.
- **Basis transform** indexing determines pixel sampling points by first estimating a basis $\boldsymbol{F}_t$ and then computing $\tilde{\boldsymbol{u}}'$ with eq. 10.

The same analysis can be applied to get $\boldsymbol{v}'$. We report the comparison results of these methods in the experimental section.

3.4. Head Pose Analysis

All the shapes throughout our analysis are 3D. This is the advantage of our single-step approach that allows us to use a simple yet reliable method to estimate the head pose of a face. In the previous section we defined a face basis $\boldsymbol{F}_t$ that is associated with the direction of the face. Clearly, the directions of the basis vectors of $\boldsymbol{F}_t$ can reveal the head pose of the analyzed face. We exploit this fact to determine the head direction.

The final shape estimate $\hat{\boldsymbol{S}}^c = \boldsymbol{r}(I, \bar{\boldsymbol{S}}^c)$ is represented in the camera coordinates. Although it is three-dimensional, its proportions no longer correspond to the actual face proportions, and therefore the estimated basis will not accurately correspond to the face direction. To address this we apply the analysis detailed in Section 3.2. By using eq. 4 and estimating the camera matrix $\boldsymbol{A}$ we transform every point of $\hat{\boldsymbol{S}}^c$ to the world coordinate system and obtain $\hat{\boldsymbol{S}}^{wR}$, for which the face proportions are preserved. We then determine the angles that the normal to $\hat{\boldsymbol{S}}^{wR}$ forms with xz and yz planes to get the tilt and yaw angles correspondingly. This simple method outperforms the state-of-the-art systems as shown in Section 4.2. Examples of bases estimated using $\hat{\boldsymbol{S}}^c$ and $\hat{\boldsymbol{S}}^{wR}$ are given in Figure 3.

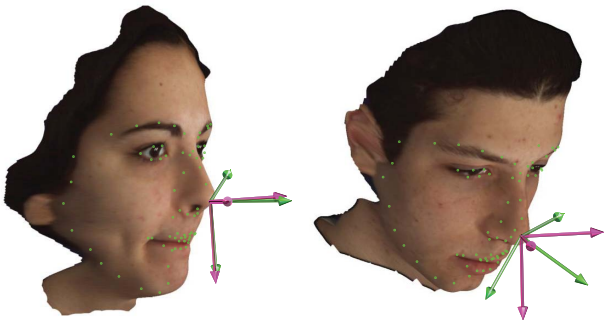


Figure 3: Examples of face bases estimated using $\hat{\boldsymbol{S}}^{wR}$ (green) and $\hat{\boldsymbol{S}}^{c}$ (pink). Note that the bases estimated using shapes in the world coordinate system (green) are more consistent with the head rotation. The detected points are plotted in green. The background was removed for visualization purposes after detection.

3.5. Learning

Our learning framework is similar to one presented in Kazemi et al. [13]. We train N levels of the cascade, where each level contains K regression trees. A node split is performed with the following split function:

$$h(I, \hat{S}_t, \boldsymbol{\theta}) = \begin{cases} 1 & I(\boldsymbol{u}') - I(\boldsymbol{v}') > \tau \\ 0 & \text{otherwise,} \end{cases} \quad (11)$$

where $\boldsymbol{\theta} = (\boldsymbol{u}', \boldsymbol{v}', \tau)$, $\boldsymbol{u}'$ are $\boldsymbol{v}'$ are obtained by using eq. 7 or 10 depending on the indexing strategy. The split parameters in $\boldsymbol{\theta}$ are randomly generated at each split node and a tree is trained with a gradient boosting algorithm that minimizes the sum of squared error.

4. Experiments

Since most of the works for face alignment estimate only 2D landmarks from an RGB image and invisible landmarks are either skipped from the estimation or their nearest visible neighbors are predicted, direct comparison is not possible. Another difficulty is that the datasets for face alignment used by the community have only 2D annotations. To overcome these difficulties we generate a large set of training and testing images and perform manual annotation of this set. For comparison purposes we train the method presented in [13] on x, y-coordinates of our 3D annotations, keeping their default parameters unchanged. We use the open-source implementation of [13] made available by [14].

Database. We build our training and testing set by using the BU-4DFE [31] database. This database contains 2D and 3D videos for six posed prototypical facial expressions (anger, disgust, fear, happy, sad, surprise) for 101 ethnically diverse subjects (58 female and 43 male). The database contains more than 60K 2D-3D pairs. Since BU-4DFE does not contain facial landmarks annotations, we performed manual annotation. We followed the widely accepted MultiPIE [10] 68-landmarks scheme. The 60K samples of the database were uniformly sampled to obtain 3000 face images with the corresponding 3D meshes. Manual annotation was performed on these 2D images, and the annotations were augmented with the third coordinate by finding the reference points on the mesh. As a result, we have 3000 images of faces annotated with the 3D landmarks positions. To generate images of faces with various head poses we rendered the meshes under uniformly distributed face rotations taken from the range $[-50, 50]$ degrees for yaw and tilt angles. In total we have 120K images. To add variability to this generated set we used images from the SUN database [28] as backgrounds, removing images annotated as containing a person. The selected BU-4DFE recordings ids as well as the 3D annotations will be made available to the research community.

Running time analysis. At every stage of the cascade $t = 1, ..., N$ we need to propagate the trees $O(KF)$ and compute a face basis $O(n^2p + p^3)$, where K is the number of weak regressors, F is the number of trees, n - the number of landmarks and p - the dimensionality of each landmark. Therefore, for a single image the running time complexity of our algorithm is constant $O(N(KF + n^2p + p^3))$. For a case $\{N = 10, K = 500, F = 5, n = 68, p = 3\}$ the method takes on average 9 ms to process an image on an Intel Core i7-4702HQ processor

4.1. 3D Landmarks Localization

To test the accuracy of our method we randomly split the rendered images into folds and perform 6-fold cross-validation. We report the averaged results for all the folds. We use the commonly accepted metric that measures the distance from a landmark to its ground truth position normalized by dividing it by the interocular distance for each image. Table 1 shows the results. We perform a separate comparison for 2D and 3D. For 2D only the first two coordinates (x, y) were used.

Table 1 shows that learning a 3D shape improves the accuracy even if we are only interested in 2D points in the image plane. Basis transform indexing shows a slightly better performance for 3D case than the other methods (not statistically significant). The intuition for this effect is that a face is inherently a 3D object, and therefore three-dimensional indexing is able to more reliably estimate the corresponding sampling points.

The values in Table 1 are close to those reported in the literature for 2D face alignment. This fact proves the difficulty of our testing set compared to the commonly used benchmarks such as [15].

Method	2D	3D
Kazemi et al. [13]	0.0522	-
Baseline indexing	**0.0515**	0.0610
3D Transform	**0.0515**	0.0607
Basis Transform	0.0518	**0.0592**

Table 1: Landmark localization errors. The numbers represent the average distance from an estimated landmark to its ground truth location normalized by the interocular distance.

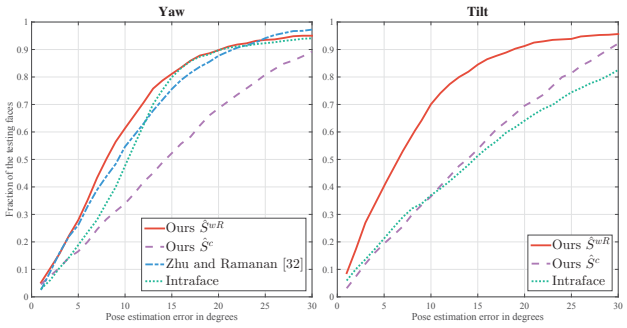


Figure 4: Cumulative error distribution rates for head pose estimation for yaw and tilt angles. We do not report results of [32] for tilt since the method provides only yaw estimates. To initialize our method we used the face detector in [14]. The benchmark systems use their internal face detectors to find faces.

4.2. Head pose estimation

We compare our method with the available state-of-the-art methods of Zhu and Ramanan [32] and Intraface[3] by Xiong and De la Torre [29]. For comparison, we use a subset of 1300 images of faces from our rendered set. The head pose is uniformly distributed within $[-50, 50]$ degrees. The images were taken from the testing folds of the trained models. The advantage of such generated dataset is that it uniformly covers all head poses in a range, and, more importantly, requires no manual annotation, because head poses are known exactly. If the competing systems were not able to detect the face in an image, we removed the image from the testing set. In total 1123 images were left. We report the results of our model that uses basis transform as the indexing method.

Table 2 shows the fraction of correctly classified images within the $\pm 15^\circ$ error tolerance, which is the commonly accepted metric in head pose analysis literature (also used in [32]). The table shows that our method based on analyzing the normal vector to $\hat{\boldsymbol{S}}^{wR}$ scores the best. The method based on $\hat{\boldsymbol{S}}^{c}$ still shows reasonable performance for tilt, but

[3]http://www.humansensing.cs.cmu.edu/intraface/

	Yaw	Tilt
Ours $\hat{\boldsymbol{S}}^{c}$	0.52	0.54
Zhu and Ramanan [32]	0.76	-
Intraface	0.80	0.51
Ours $\hat{\boldsymbol{S}}^{wR}$	**0.81**	**0.85**

Table 2: Head pose estimation results. The numbers show the fraction of faces correctly labeled within $\pm 15^\circ$ error tolerance.

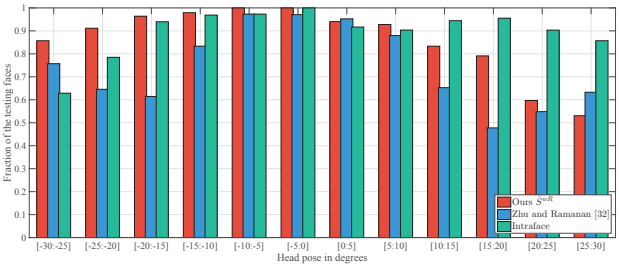


Figure 5: The distribution of the fraction of correctly recognized images within $\pm 15^\circ$ error tolerance over the yaw angle.

these results prove that 3D information contained in $\hat{\boldsymbol{S}}^{c}$ is not sufficient for head pose estimation, while the analysis in Section 3.2 is a tool to restore the shape of the face. In Figure 4 we plot the dependency of the fraction of the correctly labeled testing faces on the error tolerance value. In addition we report the fraction of correctly classified images as a function of the yaw angle for error smaller than 15° of our best method versus [32] and Intraface [29].

We note that our method has several additional advantages over [32]: (i) our method predicts continuous output for both angles (yaw and tilt), while the method in [32] recognizes only yaw angle and only for a fixed number of allowed head poses in a range of $[-90, 90]$ with a step of 15° and (ii) the computational performance of our method is superior: the whole pipeline on average took 9 ms per image as compared to 40-100 seconds of [32]. Qualitative results for head pose estimation are given in Figure 6.

4.3. Qualitative results

Qualitative results are given in Figure 7. The subjects in Figure 7a are taken from the testing set of the trained model. Figure 7b shows the results of our method applied to the HELEN database [15]. The bases on the both figures are estimated using $\hat{\boldsymbol{S}}^{wR}$. Some dots in the odd rows may look misleading, since the actual positions of the landmarks are plotted regardless of the visibility of the points. Note that the points in the world coordinate system reveal the face orientation that is often difficult to understand from a single

Figure 6: Selected examples of estimated head poses from the testing set. Yaw angles are reported. The green points show the estimated landmarks by our best model.

2D image, when a face looks frontal, while having a slight inclination. In some cases the y-direction (the green arrow) of the face basis can be estimated incorrectly. However, this makes no influence on the face normal vector (the blue arrow) that is used in head pose estimation.

5. Conclusions

We have presented a novel, accurate and fast single-step method for estimating a 3D face shape from a single 2D image. We have shown that when treating a face as a 3D object, the overall recognition error decreases even when considering only 2D landmarks. Including additional knowledge about the face yields substantial improvement, and allows a simple head pose estimation method to show results superior to available systems. Since 2D face alignment has reached impressive results in speed and accuracy, we believe, that 3D shape regression is a promising area that should be explored further.

References

[1] V. Blanz and T. Vetter. Face recognition based on fitting a 3D morphable model. *PAMI*, 25(9):1063–1074, 2003. 2

[2] X. P. Burgos-Artizzu, P. Perona, and P. Dollar. Robust face landmark estimation under occlusion. In *ICCV*, pages 1513–1520, 2013. 1

[3] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. In *SIGGRAPH*, volume 33, 2014. 2

[4] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3D Shape Regression for Real-time Facial Animation. In *SIGGRAPH*, volume 32, 2013. 2

[5] X. Cao. Face alignment by Explicit Shape Regression. In *CVPR*, pages 2887–2894, 2012. 1

[6] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *PAMI*, pages 681 – 685, 2001. 1

[7] D. Cristinacce and T. Cootes. Automatic feature localisation with constrained local models. *PR*, 41(10):3054–3067, 2008. 1

[8] P. Doll, W. Pietro, and P. Perona. Cascaded Pose Regression. In *CVPR*, pages 1078 – 1085, 2010. 1

[9] R. Gross, R. Gross, I. Matthews, I. Matthews, S. Baker, and S. Baker. Generic vs. Person Specific Active Appearance Models. *IVC*, 23(11):1080–1093, 2005. 1

[10] R. Gross, I. Matthews, J. Cohn, and T. Kanade. Multi-PIE. In *FG*, pages 1 – 8, 2008. 5

[11] T. Hassner. Viewing Real-World Faces in 3D. In *ICCV*, pages 3607–3614, 2013. 1, 2

[12] A. Jeni, J. F. Cohn, and T. Kanade. Dense 3D Face Alignment from 2D Videos in Real-Time. In *FG*, 2015. 1, 2

[13] V. Kazemi and S. Josephine. One Millisecond Face Alignment with an Ensemble of Regression Trees. In *CVPR*, pages 1867–1874, 2014. 1, 2, 3, 4, 5, 6

[14] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009. 5, 6

[15] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang. Interactive facial feature localization. In *ECCV*, pages 679–692, 2012. 1, 5, 6, 8

[16] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. In *SIGGRAPH*, volume 32, page 1, 2013. 1

[17] C. Liu, J. Yuen, S. Member, and A. Torralba. SIFT flow: dense correspondence across difference scenes. *PAMI*, 33(5):978 – 994, 2011. 2

[18] I. Matthews and S. Baker. Active Appearance Models Revisited. *IJCV*, 60(2):135–164, 2004. 1

[19] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3D Face Model for Pose and Illumination Invariant Face Recognition. In *International Conference on Advanced Video and Signal Based Surveillance*, pages 296 – 301, 2009. 2

[20] S. Ren, X. Cao, Y. Wei, and J. Sun. Face Alignment at 3000 FPS via Regressing Local Binary Features. In *CVPR*, pages 1685 – 1692, 2014. 1, 3

[21] E. Sangineto. Pose and expression independent facial landmark localization using dense-surf and the hausdorff distance. *PAMI*, 35(3):624–638, 2013. 1

[22] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *IJCV*, 91(2):200–215, 2011. 1

[23] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *CVPR*, pages 1701 – 1708, 2014. 2

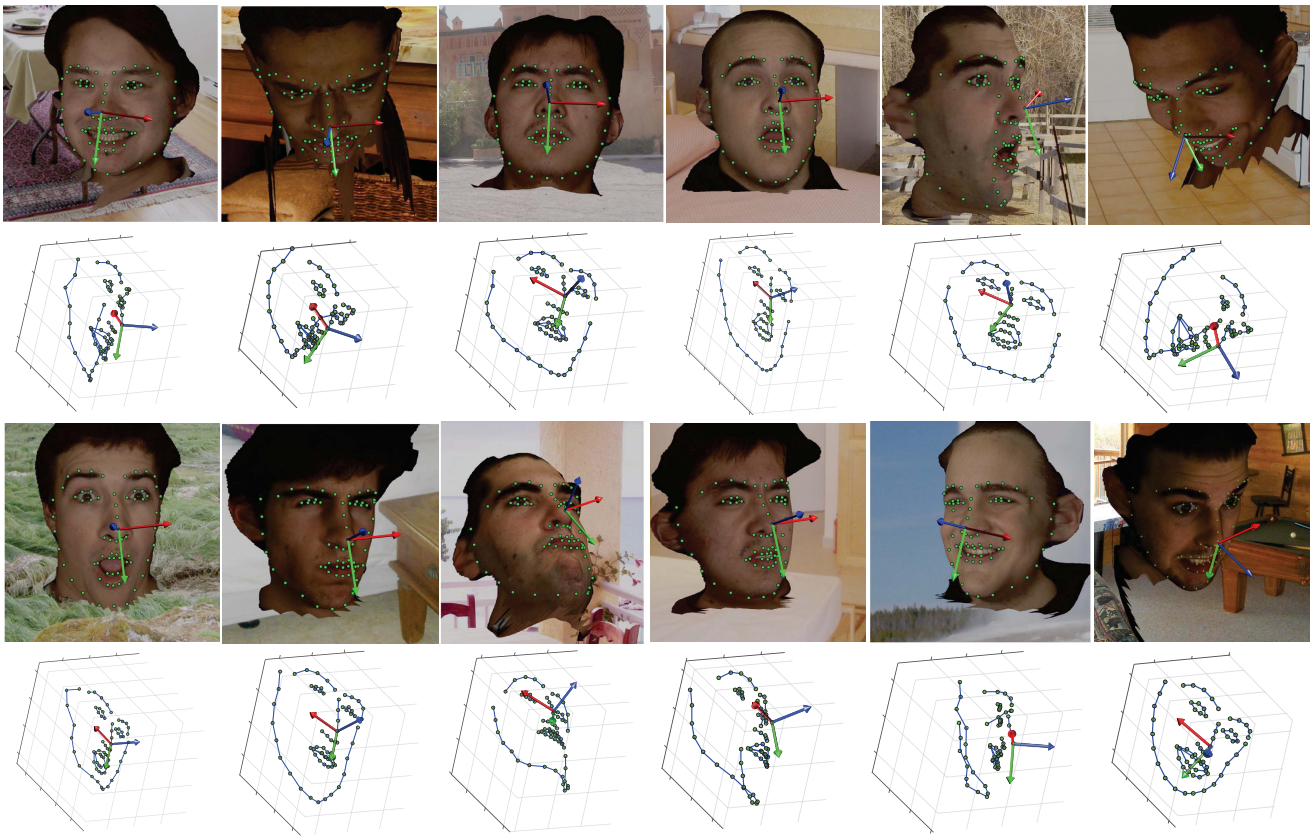
[24] S. Tulyakov, R. L. Vieriu, S. Semeniuta, and N. Sebe. Robust Real-Time Extreme Head Pose Estimation. In *ICPR*, 2014. 1

[25] Y. Wang, S. Lucey, and J. Cohn. Enforcing convexity for improved alignment with constrained local models. In *CVPR*, June 2008. 1

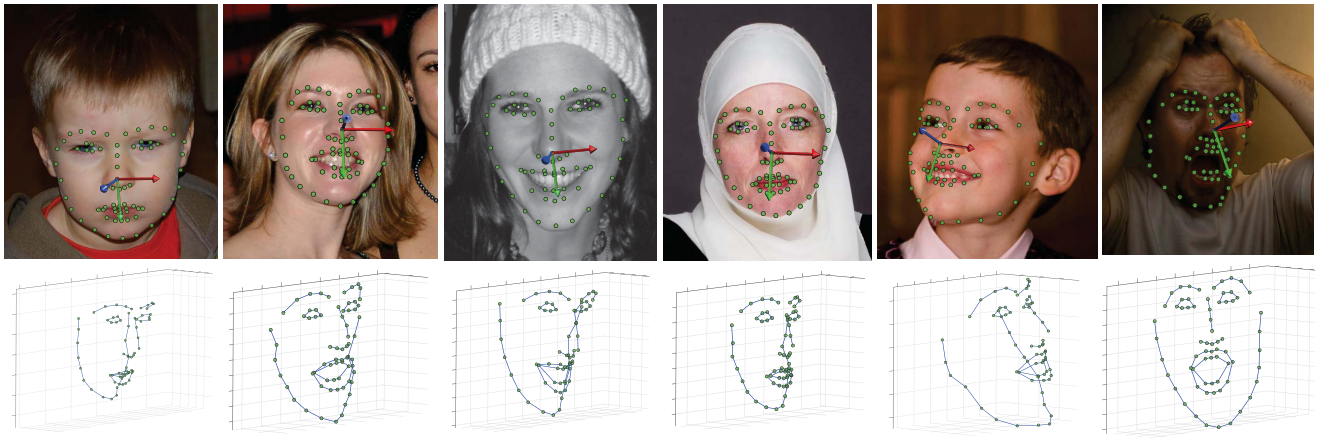
[26] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. In *SIGGRAPH*, 2011. 1

[27] T. Weise, H. Li, L. Van Gool, and M. Pauly. Face/off: Live facial puppetry. In *SIGGRAPH*, pages 7–16, 2009. 1

[28] J. Xiao, J. Hays, K. a. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pages 3485–3492, 2010. 5



(a) Estimated 3D landmarks positions from the generated test set.



(b) Qualitative results of our method applied to the HELEN database [15]

Figure 7: Selected examples of estimated face shapes from the generated test set (a) and from the HELEN database (b). Odd rows show the estimated points in the camera coordinates projected onto the image plane. Even rows show the shapes in the world coordinate system. Arrows represent bases computed using $\hat{\boldsymbol{S}}^{wR}$.

[29] X. Xiong and F. De La Torre. Supervised descent method and its applications to face alignment. In *CVPR*, pages 532–539, 2013. 2, 6

[30] D. Yi, Z. Lei, and S. Z. Li. Towards Pose Robust Face Recognition. In *CVPR*, pages 3539–3545, 2013. 1, 2

[31] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale. A high-resolution 3d dynamic facial expression database. In *FG*, 2008. 2, 3, 5

[32] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark estimation in the wild. In *CVPR*, pages 2879 – 2886, 2012. 2, 6