Viewing Real-World Faces in 3D

Tal Hassner
The Open University of Israel, Israel
hassner@openu.ac.il

Abstract

We present a data-driven method for estimating the 3D shapes of faces viewed in single, unconstrained photos (aka "in-the-wild"). Our method was designed with an emphasis on robustness and efficiency – with the explicit goal of deployment in real-world applications which reconstruct and display faces in 3D. Our key observation is that for many practical applications, warping the shape of a reference face to match the appearance of a query, is enough to produce realistic impressions of the query's 3D shape. Doing so, however, requires matching visual features between the (possibly very different) query and reference images, while ensuring that a plausible face shape is produced. To this end, we describe an optimization process which seeks to maximize the similarity of appearances and depths, jointly, to those of a reference model. We describe our system for monocular face shape reconstruction and present both qualitative and quantitative experiments, comparing our method against alternative systems, and demonstrating its capabilities. Finally, as a testament to its suitability for real-world applications, we offer an open, online implementation of our system, providing unique means of instant 3D viewing of faces appearing in web photos.

1. Introduction

The problem of estimating the 3D shapes of faces from photos is deceptively simple. We all have similar looking and similar shaped faces. Yet when considering unconstrained, "in-the-wild" photos, our appearances can vary a great deal. Many of these variations are due to the many sources of appearance variability in general. Often, these stem from the camera systems, and include pose variations, changing lighting, noise and more. When considering photos of faces, these are all exacerbated by variabilities of the faces themselves: expression changes, different ages, occlusions (often facial hair and glasses), makeup and others (e.g., Fig. 1). To reliably estimate the shape of a face, all of these issues must be overcome.

Over the years this problem has attracted considerable attention, and several highly effective approaches have emerged. These were shown to be capable of estimating accurate and detailed shapes from single photos, even to the point of capturing the shapes of expressions [29, 31], wrinkles, and other fine facial features [14].

Figure 1. Faces can vary greatly in appearances, thereby challenging Machine Vision systems for 3D reconstruction whenever such variations are allowed. Despite this, the underlying facial shapes are similar. This fact can be exploited in order to allow the generation of novel, realistic, 3D views of even challenging face photos, such as those shown here.

To achieve this accuracy, some employ collections of carefully aligned, 3D face models, used for spanning the space of face shapes [11, 29]. Others require manual segmentation and localization of query faces [3, 7, 14]. Yet others, including the commercial systems of [22] and [28], are designed for controlled viewing conditions.

In this paper we are motivated by the observation that although highly accurate reconstruction is always *desirable* it is not always *necessary*; for many practical purposes, tolerance to challenging imaging conditions, fast processing, minimal user interaction, and simplicity of training and data acquisition, are as important, if not more. One such example application is 3D viewing of faces appearing in casual web photos (e.g., in Picasa and Facebook albums), where minimal interaction and robustness to "in-the-wild" viewing

conditions is paramount. Another example is face recognition in unconstrained photos. Here, high-end face recognition systems rely on the 3D reconstruction process mostly to create an appealing alignment and cancel out-of-plane pose differences, and do not necessarily require exact 3D measurements [23].

We therefore propose a novel method for estimating the 3D shapes of faces in unconstrained photos. At the heart of our method is a robust optimization process which employs a single reference face, represented by its appearance (photo) and shape (depth map). This process attempts to transfer reference depth values to match a query photo, while preserving a global face shape. This, by optimizing for the joint similarities of appearances and depths of the two faces. These similarities are considered *locally*, at each pixel, with the optimization ensuring that global consistency is achieved once the algorithm converges. We show that this method converges quickly, and produces realistic face shape estimates.

In summary, our contributions are as follows.

- A novel optimization technique and 3D reconstruction system designed to enable new, 3D-view generation of challenging face photos.
- Qualitative and quantitative tests demonstrating our method's performance compared to a range of alternative systems and techniques.
- Finally, as testament to our method's suitability for reconstruction of unconstrained face photos, we offer an open, on-line implementation allowing 3D reconstruction of faces in casual, real-world photos.

2. Related work

Reconstructing the 3D shape of an object viewed in a single photo has long since been a central research theme in Computer Vision. Motivated by the importance and unique properties of faces as a class, many have turned to designing monocular, face-shape estimation methods. Broadly speaking these methods can be classified as either "shape from X" approaches or learning based techniques.

Shape from X methods make assumptions on the properties of the scene itself, here the faces appearing in the photos. Some exploit the symmetry of faces [7]. More often, however, methods rely on the reflectance properties of facial skin, along with assumptions on scene lighting, and extract shapes of faces from shading [1, 14]. Because these methods are sensitive to background clutter, they require careful segmentation of the foreground face from any background in the photo in order to perform accurately. In addition, they are sensitive to occlusions, further limiting their applicability in the scenarios considered here.

Machine learning based methods can further be classified as those which explicitly model a distribution of allowable face shapes and those which employ non-parametric sampling (data-driven) techniques. Notable examples of the former are the well known Morphable Models [3, 4, 16, 29, 31] and the related methods of [11, 24, 30]. These all require carefully aligned 3D face scans to be assembled and then used to learn the space of faces.

Related to our work here is the on-line, commercial Vizago system [28], based on a recent extension of the Morphable Models [16]. Unlike our own on-line system (Sec. 4.3) it requires careful manual localization of facial points and selection of the query gender. Some comparisons of our reconstructions with those obtained using Vizago are presented in Fig. 2. The commercial FaceGen software [22], likewise requires manual localization, though slower to converge.

Figure 2. **Visual comparison with Vizago.** Top: Query photos; Middle: Shapes estimated by the on-line, commercial system of [28], which implements the recent Morphable Models variant described in [16]; Bottom: Our own results.

Our own method belongs to the latter family of techniques. These attempt to match parts of the query photo to parts of reference faces, in order to obtain local estimates for the query's depth. A final shape is assembled by combining the reference estimates. To our knowledge, the first such "by-example" technique for single-view reconstruction was presented in [9] (see also [10]). Face Reconstruction In-the-Wild was recently proposed by [15], combining an example based approach with a shape from shading face model. These previous methods, however, typically require long run-times [9] or hundreds of query photos [15] and are therefore unsuitable for on-line processing.

In contrast, our system, though easily extendable to handle multiple references, requires only a single reference to produce realistic looking estimates. It requires no manual interaction. It is robust to illumination, expression changes as well as occlusions and viewpoint differences. Finally, it

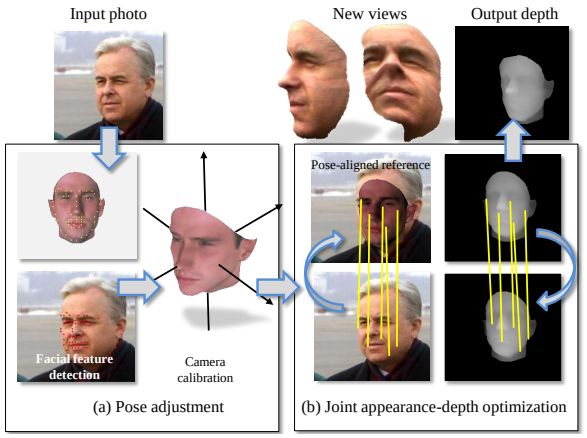

Figure 3. **Overview of our depth estimation process.** (a) During pose adjustment we obtain a camera matrix for the query photo, given detected facial features (Sec. 3.1). (b) Using this matrix, the reference, 3D model is re-rendered and our joint appearance-depth optimization process is applied using the aligned, rendered reference and its depth (Sec. 3.2). Note the reference photo superimposed on the query. See Sec. 3.2 for further details.

is efficient, estimating depths in seconds.

3. Depth estimation

For a single query photo I_Q, we seek an estimate for the shape of the face appearing in it. Here, we represent the shape as a depth-map which assigns every pixel coordinate $p = (x, y)$ in the query photo a distance to the surface of the face along the ray emanating from the optical center through p. Fig. 3 provides an overview of our system. It involves two main steps: (a) Pose adjustment and reference generation; (b) Iterative optimization. In the following sections we describe these in detail.

3.1. Pose adjustment

Unconstrained photos often present faces in varying, possibly extreme, pose differences. Previous methods have dealt with this either by warping the query image to a "normalized" pose (e.g., [15]), or by integrating pose estimation directly into the depth estimation process [3, 9]. Here, our system relies on an initial pose adjustment step, which produces a reference photo I_R and matching depth D_R in approximately the same pose as the query.

Specifically, we assume a single, reference, textured, 3D-model (i.e., triangulated mesh) of a face. In all our experiments we used the same 3D model, arbitrarily selected from the USF Human-ID database [25]. We begin by rendering this reference model in a fixed, frontal pose. 68 facial landmarks $p'_{k,k\in[1,68]} = (x'_k, y'_k)$ are detected in this image using the method of [33], selected for its accuracy in real-world face photos. Our rendering software provides us with the 3D coordinates $P' = (X', Y', Z')$ of the surface point projected onto each rendered pixel p', thereby associating each detected point p'_k with a 3D point P'_k.

Given a query image, it is processed by first running the Viola-Jones detector [27]. The detected face is then expanded to 2.2 times its size, cropped and rescaled to 250×250 pixels. We refer to this as the query photo I_Q. We again use [33] to detect the same 68 landmarks in I_Q, giving us points $p_k = (x_k, y_k)$. Using these, we form correspondences (p_k, P'_k), from 2D pixels in the query photo to 3D points on the model (Fig. 3(a)). We then solve for both intrinsic and extrinsic camera parameters, using a standard calibration method [8]. The obtained camera matrix $A_{3\times3}$, 3D rotation matrix $R_{3\times3}$ and 3D translation vector $t_{3\times1}$, relate 3D reference points with the query photo's viewpoint by $p_i \sim A[R\ t]P'_i$.

These camera parameters allow us to re-render the reference model, along with its depth, in a query-adjusted pose. Although we found the detection step to be accurate, it is currently the run-time bottleneck. In order to expedite this step, detection is performed on a small, 150×150 pixels, query photo. The reference image and depth subsequently used to estimate depths are rendered at 250×250 pixels.

3.2. Depth optimization

Following the pose adjustment step, the reference photo and depth, I_R and D_R, are in approximately the same pose as the query. Small pose estimation errors, along with differences in the shape of the reference head compared to that of the query, still remain, as demonstrated in Fig. 4(b). We next bridge these differences and fit a depth estimate tailored to the query's appearance. To this end, we begin by considering what makes for a good face depth estimate.

What is a suitable depth estimate? Intuitively, we expect similar looking facial features to have similar shapes; any variations in the *appearance* of one face should imply a like variation in its *shape*. In other words, although the sought depth D_Q should be different from the reference, these differences should be minimized, and should follow variations in the appearance of the query compared to the reference.

Formally, we define the local appearance (depth) at pixel p as $f(I_Q, p)$ and $f(I_R, p')$ (similarly, $f(D_Q, p)$ and $f(D_R, p')$). The function f represents a feature transform, applied to pixel p (p'). Here, we use the SIFT descriptor [19], extracted at a constant scale and orientation (i.e., Dense-SIFT [26]) and applied to both images and their depth maps. Our goal can now be stated as follows: We seek to warp the reference depth D_R to match the query image; that is, to obtain for each query pixel p a vector $w(p) = [u(p), v(p)]^T$, mapping it to a pixel p' in the reference, thereby assigning it with the reference depth. Following the reasoning above, a good warp should satisfy the

Figure 4. **Reconstructions with different references.** (a) Four pose-adjusted references used in separate reconstructions of the same query. (b) Following pose adjustments, noticeable differences remain between the references' depths and the query's appearance. These are particularly evident around the nose (on the right). (c) Our depth optimization (Sec. 3.2) reshapes the reference depths, matching them to features of the query. Note the minor effect of using different references, evident when comparing our results on the last row. (d) The query photo.

following requirements:

1. **Small displacement.** Depth assigned to query pixel p should come from a nearby reference pixel. That is, $|u(p)| + |v(p)|$ is minimized for all pixels.

2. **Smoothness.** Adjacent query pixels, p_1 and p_2 should be assigned depth values from adjacent reference pixels p'_1 and p'_2. That is, $|u(p_1) - u(p_2)|$ and $|v(p_1) - v(p_2)|$ are minimized for all pixels.

3. **Joint depth-appearance similarity.** We require that for any pixel p, its appearance *as well as depth*, will resemble the appearance-depth of its matched reference pixel, minimizing $||f(I_Q, p) - f(I_R, u(p))||_1 + ||f(D_Q, p) - f(D_R, u(p))||_1$ for all pixels.

Requirements 1 and 2 have been imposed by previous methods for optical-flow (see, e.g., [5]). Moreover, [17, 18] have extended these by additionally requiring the similarity of SIFT descriptors, partially meeting our third requirement. Unlike [17, 18], however, we require that the warped *depths* be also similar to those of the reference. This requires the matching procedure to take into account the (yet unknown) query depth D_I, and not only its appearance.

Joint depth-appearance optimization. To obtain a depth estimate which meets the above criteria, we employ a "coordinate descent" optimization [2]. Specifically, we define the following cost function:

$$\begin{aligned} C^t(w) \quad = \quad & \sum_p \min\left(||f(I_Q, p) - f(I_R, u(p))||_1, k\right) + \\ & \sum_p \nu\left(|u(p)| + |v(p)|\right) + \\ & \sum_{(p_1,p_2 \in N)} \left[\, \min\left(\alpha|u(p_1) - u(p_2)|, d\right) + \right. \\ & \qquad \left. \min\left(\alpha|v(p_1) - v(p_2)|, d\right) \,\right] + \\ & \sum_p \min\left(||f(D_Q^{t-1}, p) - f(D_R, u(p))||_1, k\right) \end{aligned} \tag{1}$$

Where k and d are constant threshold parameters and N defines neighboring pixel pairs (i.e., p_1 and p_2 are close). The second and third terms of the cost function, Eq. 1, reflect our requirements for small displacements and smoothness. The first is similar to the one defined by [17, 18] and enforces appearance similarities. Here, however, we add the fourth term, which expresses our requirement that the estimated depths be similar to the reference depths. In order to do that, we define our cost at time t, by comparing the reference depths to those estimated at time $t-1$. To minimize this cost, at time t, we establish correspondences using the modified optical flow formulation [6], implemented by the SIFT-Flow system [17]. We use it here by representing the features extracted from the photos and the depths as vector images, and adding the depth channels, vector images representing the features extracted from D_Q^{t-1} and D_R, to the input provided to SIFT-Flow. Finally, we set the initial depth estimate at time $t = 0$ as the reference depth.

Dealing with background. Background clutter has been addressed by previous methods either by requiring that the face be manually segmented (e.g., [9, 14]), imposing the segmentation of a reference face onto the query [15], or else assuming that the shape to be reconstructed encompasses the entire photo, as in outdoor scene reconstruction [13].

Here, the uniform background of a rendered image I_R can result in correspondences to arbitrary reference pixels, wherever it does not contain background information and the query does. This has the adverse effect of letting the *size* of the background influence the quality of the estimated depth. We address this by superimposing the pose-adjusted reference (Sec. 3.1) onto the query image, and use that as the reference image (See Fig. 3(b)). Thus, both query and reference share the same background, encouraging zero flow in this region. This leaves only face pixels with the freedom of being assigned flow vectors according to the differences in appearance. This has the additional advantage of allowing the optimization to automatically infer the segmentation of the foreground face from its

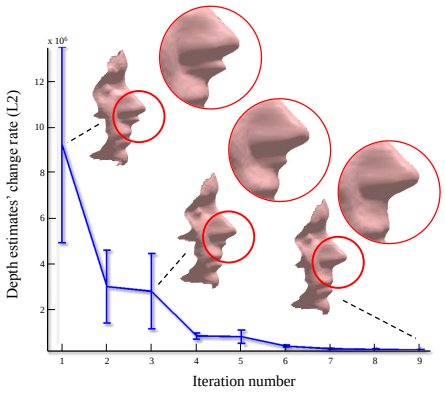

Figure 5. **Convergence of the iterative optimization.** The L2 distance between depth maps estimated in successive iterations, along with the Standard Errors, averaged over all 76 queries. Alongside the convergence, we provide an example reconstruction, demonstrating how features inconsistent with the appearance-shape of the reference face, gradually dissolve. Note that the first iteration is equivalent to depth estimation using SIFT-Flow alone ([17], see also Sec. 4.1).

background (see also Sec. 4.2).

Complexity and run-time. Informally, in each iteration, our optimization progresses by choosing an assignment of depth values to all pixel, thereby improving (minimizing) the value of the target function. It can therefore be considered an instance of coordinate descent optimization, well known to converge to a local minimum of the cost function [20]. We verify the convergence property of our method by estimating depths for the rendered views of 76 of the 77 models from [25], using the first model as a reference. Fig. 5 presents the diminishing change in depth estimates from one iteration to the next, by plotting the $L2$ distances between estimates from successive iterations, averaged over all queries. Evidently, depth estimates converge in as little as four iterations. This whole optimization is thus equivalent to running SIFT-Flow, four times, on 256 dimensional feature vectors. The total runtime for this optimization was approximately 20sec., measured on a 2.4GHz Intel Core i5 Processor notebook with 4Gb RAM.

Comparison with Depth Transfer. It is instructional to compare our approach to the recent, related "Depth Transfer" of [13] which also produces depth estimates by warping reference depths, using SIFT-Flow to obtain initial matches. Like us, they observed that doing so may be inaccurate, as it disregards the reference depths and how they may constrain the estimated output. Their proposed means of addressing this, however, is different from our own. Specifically, following an initial correspondence estimation using SIFT-Flow [17, 18], they optimize the output depth *separately* from the appearance in a subsequent process which is

	Method	$\log_{10}$	RMSE	REL
(i)	Baseline	0.0772	24.7806	1.8839
(ii)	Face-Molding [14]	0.0753	24.6197	2.2197
(iii)	SIFT-Flow [17]	0.0746	24.2617	1.8293
(iv)	Depth Transfer [13]	0.0766	24.2897	1.8165
(v)	Our method, 4 Itr.	0.0743	24.0949	1.7877
(vi)	Our method, 5 Itr.	**0.0742**	24.0907	1.7902
(vii)	Our method, 10 Itr.	**0.0742**	**24.0681**	**1.7839**

Table 1. Empirical results on the USF Human-ID database [25]. Lower values are better. Bold values are the best scoring. Please see text for further details.

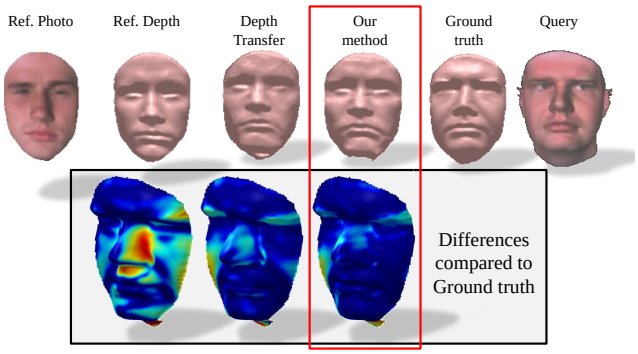

Figure 6. **Visual comparison of different reconstruction methods.** Shapes estimated for an example query in our empirical tests (Table 1). From left to right: The reference photo and depth-map; Depth-Transfer [13]; our own method; the ground truth depth; finally, the query photo. We provide also the $\log_{10}$ absolute difference between the estimated and ground truth depth at each pixel, superimposed on the ground truth depth. Here, **darker (bluer) colors are better** – implying smaller differences.

applied to the depth values alone. Although Depth Transfer was shown to far out-perform the state-of-the-art in single-image, depth estimation of outdoor scene photos, we have noticed that when applied to faces this process may over-smooth the estimated depth (see Sec. 4.1).

4. Experiments

Our system employs the SIFT-Flow and SIFT descriptor code from [17], using their default values for all parameters, unchanged. Standard OpenCV routines were used for camera calibration. The reference model was rendered, along with its depth-map and 3D coordinates of the surface projected onto each pixel, using our own rendering software, developed in OpenGL for this purpose. Finally, we use the facial feature detector of [33] for the pose estimation (Sec. 3.1), using their original MATLAB code.

4.1. Empirical tests

We compare the accuracy of our method to several existing state-of-the-art methods. As there is no standard, publicly available benchmark for face depth reconstruction, we design our own benchmark tests. To this end, we use the 77 models of the USF Human-ID database [25], fixing the first model, ID "03500_1", as reference and all others as queries. We use the evaluation protocol defined by [21] and compare each estimated depth-map, D_Q, to its known ground truth depth, D_Q^*, by computing for each method the following error measures: The $\log_{10}$ Error, $(\log_{10})$, $|\log_{10}(D_Q) - \log_{10}(D_Q^*)|$, the Root Mean Square Error (RMSE), defined by $\sqrt{\sum_{i=1}^{76}\left(D_{Q_i} - D_{Q_i}^*\right)^2/N}$, and the Relative Error (REL), defined by $\frac{|D_Q - D_Q^*|}{D_Q^*}$. The $\log_{10}$ and REL errors were averaged over the 76 queries.

Table 1 lists error rates for the following methods: (i) Baseline performance computed by setting $D_Q = D_R$; that is, taking the reference depth itself as the estimated depth-map; (ii) the shape-from-shading, "Face-Molding" approach of [14]. We use our own implementation of their method, as no publicly available implementation exists; (iii) SIFT-Flow [17] used to compute dense correspondences between appearances alone, returning the reference depth, warped using the obtained flow; finally, (iv) Depth Transfer [13], using a single reference photo and depth-map. All these were compared to our own error rates, (v-vii), reported for four, five, and ten optimization iterations.

We note that the methods we tested were selected due to being recent, efficient, and not requiring manual assistance (as in [22, 28]) or reference data beyond a single reference photo and depth-map pair (e.g., [15]). The only exception is Face-Molding [14] which requires a mask to be positioned around the central part of the face. We empirically selected the shape and size of this mask to optimize performance, and report the best results we have obtained. No other attempt was made to optimize parameters for any method.

Discussion. Several things are evident from Table 1. First, row (i) suggests that although the numerical differences between the methods are small, these differences are significant; the reference depth is typically an unsuitable depth estimate, as demonstrated in Fig. 4(b), yet its error scores are only slightly higher than other methods. Second, despite being the state-of-the-art for outdoor scene reconstruction, Depth Transfer [13] performs worst than the simpler SIFT-Flow, as measured by both the $\log_{10}$ error and RMSE. This, despite using SIFT-Flow for initialization. We believe this is due to over smoothing of the depth, which may be less suitable for face photos than outdoor scenes. All three of our methods outperform the others, with four-iterations better than SIFT-Flow based on appearance alone, and with performance improving with additional iterations.

Yaw error	$\log_{10}$	RMSE	REL
0°	0.0743	24.0949	1.7877
10°	0.0828	25.7653	2.2485
20°	0.1110	31.9553	2.9834
30°	0.1336	36.8970	3.4663
40°	0.1559	41.4488	3.8008

Table 2. The robustness of our depth optimization (Sec. 3.2) to pose adjustment errors. Please see text for further details.

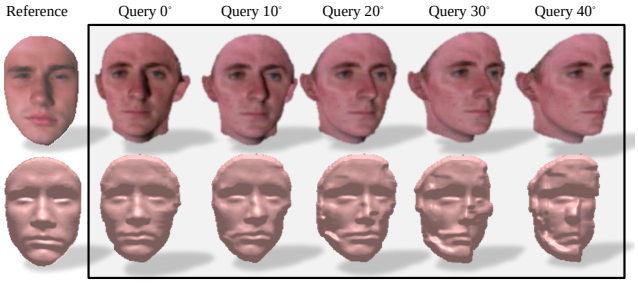

Figure 7. **Depth optimization (Sec. 3.2) performed without adjusting the reference pose (Sec. 3.1).** Illustrating the results in Table 2. Leftmost column is the reference. Next columns, left to right, are the query, with increasing yaw angle difference to the reference. On top are the photos; bottom shows estimated depths. Without pose adjustments, quality can noticeably degrade, breaking down at around 20° yaw, pose error.

Fig. 6 visualizes some of these results, by presenting depth estimates for several of the methods that were tested. To better illustrate the misalignment errors, Fig. 6 also color codes the per-pixel $\log_{10}$ absolute distances between the estimated depths and the ground truth. Clearly, our method has fewer noticeable misalignments compared to the others. Also evident is that all methods are influenced by the selection of the reference model, as noted also by [14]. As evident from Fig 4(c), however, differences in depth estimates due to the use of different references, have minor effects when visualizing the face in 3D.

Robustness to pose adjustment errors. Pose adjustment errors can occur whenever the facial feature detector fails to accurately localize the key-points used to estimate the query's pose. Table 2 provides reconstruction errors, again computed using the USF database [25] with model, ID "03500_1" as reference, and averaging over all other models as queries. Here, we progressively increase the yaw angle between the query photo and the reference, from 0°to 40°. Four optimization iterations were performed, and so the first row is equal to row (v) in Table 1.

The results in Table 2 show that reconstruction errors

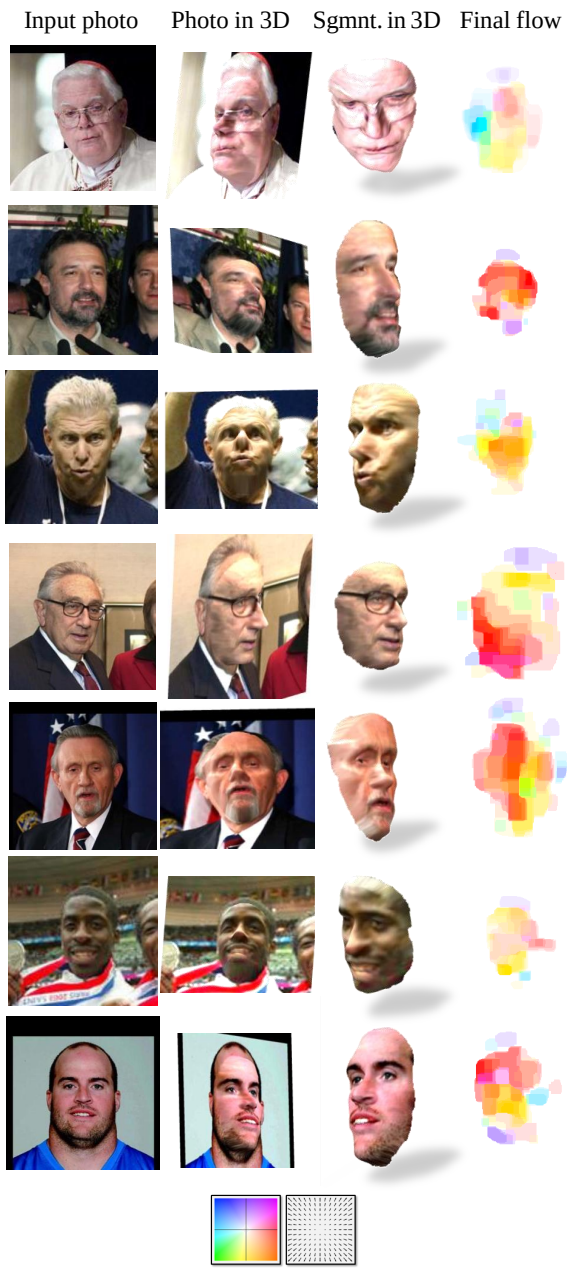

Figure 8. **Example faces in 3D.** From left to right: example photos from the LFW collection [12]; 3D views of the photos with estimated 3D faces; 3D views of automatically segmented faces; final flows from the reference face. Flow values are color coded; flow color legend provided at the bottom of the figure.

gradually increase with larger pose errors. Fig. 7 visualizes these results by presenting the depths obtained for the same query at increasing angle differences. Interestingly, the depth obtained at 10°appears rotated correctly, despite the misaligned reference. The same is true for 20°, though with such a large pose difference, some artifacts appear. These artifacts grow worst with larger alignment errors.

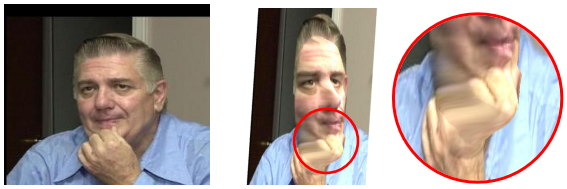

Figure 9. **Faulty estimation.** An example of a typical failure, here resulting from an occluding hand.

4.2. Qualitative results

Fig. 1, 2, 4, and 8 present many examples of faces reconstructed and rendered from various views in 3D. These photos were mostly taken from the original (unaligned) LFW collection [12], or from the web, making a point of choosing those which would be considered challenging due to facial hair, expressions, varying poses, and more. In all these cases, the reconstructed geometry allows exploring the faces in 3D, with few noticeable artifacts. Moreover, in the segmented images (e.g., Fig. 8, second column from the right), faces were segmented *automatically*, by considering only pixels in the largest connected component of non-zero depth values. Segmenting faces in photos is itself a challenging problem, and it would be worth while to explore how well this approach performs in comparison with existing art [32]. Here, we found our depth estimates to provide reliable means for face segmentation, and did not pursue these alternatives. Finally, in Fig. 9 we provide an example of a typical faulty reconstruction, resulting from the hand occluding part of the face.

4.3. Our on-line face reconstruction system

In order to demonstrate our system's capabilities, we have designed an on-line system for web-based 3D viewing of real-wold face photos, publicly accessible using modern web-browsers. We implemented our method as a server-side process, providing a simple interface for client-side applications. One such client application is a Google Chrome extension. It enables users to select photos appearing in web-pages by a simple mouse click. A selected photo is then automatically sent to the server service, the shape of the face is estimated on the server, and then displayed back on the Chrome browser. Users typically wait about a minute before a surface is returned and can be explored in 3D, much of this time required for data transfer. To access the system, please visit our project webpage, available from `www.openu.ac.il/home/hassner/projects/ViewFaces3D`.

5. Conclusions

Today, as web-browsers are integrated with 3D engines, and with the advent of 3D printers, there is an increased demand for accessible ways of creating 3D models. In response to this, we present an efficient and robust system

for 3D estimation of faces, designed to tolerate extreme variabilities in facial appearances. Our system is designed around an optimization which uses appearance and depth jointly to regularize the output depth. We tested our method extensively, comparing it to existing alternatives, including commercial products, in order to evaluate its capabilities. Finally, we demonstrate how this process may be employed within a real-world system by offering a public system for on-line face shape estimation.

References

[1] J. T. Barron and J. Malik. Shape, albedo, and illumination from a single image of an unknown object. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 334–341, 2012.

[2] D. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[3] V. Blanz, K. Scherbaum, T. Vetter, and H. Seidel. Exchanging faces in images. *Comput. Graphics Forum*, 23(3):669–676, 2004.

[4] V. Blanz and T. Vetter. Morphable model for the synthesis of 3D faces. In *Proc. ACM SIGGRAPH Conf. Comput. Graphics*, pages 187–194, 1999.

[5] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *European Conf. Comput. Vision*, pages 25–36, 2004.

[6] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *Int. J. Comput. Vision*, 61(3):211–231, 2005.

[7] R. Dovgard and R. Basri. Statistical symmetric shape from shading for 3D structure recovery of faces. *European Conf. Comput. Vision*, pages 99–113, 2004.

[8] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[9] T. Hassner and R. Basri. Example based 3D reconstruction from single 2D images. In *Proc. Conf. Comput. Vision Pattern Recognition*, 2006.

[10] T. Hassner and R. Basri. Single view depth estimation from examples. *CoRR*, abs/1304.3915, 2013.

[11] Y. Hu, D. Jiang, S. Yan, L. Zhangg, and H. zhang. Automatic 3D reconstruction for face recognition. In *Int. Conf. on Automatic Face and Gesture Recognition*, pages 843–848. IEEE, 2004.

[12] 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. University of Massachusetts, Amherst, TR 07-49, 2007.

[13] K. Karsch, C. Liu, and S. B. Kang. Depth extraction from video using non-parametric sampling. In *European Conf. Comput. Vision*, 2012.

[14] I. Kemelmacher-Shlizerman and R. Basri. 3D face reconstruction from a single image using a single reference face shape. *Trans. Pattern Anal. Mach. Intell.*, 33(2), 2011.

[15] I. Kemelmacher-Shlizerman and S. Seitz. Face reconstruction in the wild. In *Proc. Int. Conf. Comput. Vision*, pages 1746–1753. IEEE, 2011.

[16] R. Knothe. *A Global-to-local model for the representation of human faces*. PhD thesis, University of Basel, 2009.

[17] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, 2011. Available: `people.csail.mit.edu/celiu/SIFTflow/`.

[18] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. Freeman. Sift flow: dense correspondence across different scenes. In *European Conf. Comput. Vision*, pages 28–42, 2008. Available: `people.csail.mit.edu/celiu/ECCV2008/`.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[20] Z. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *J. of Optimization Theory and Applications*, 72(1):7–35, 1992.

[21] A. Saxena, M. Sun, and A. Ng. Make3D: Learning 3D scene structure from a single still image. *Trans. Pattern Anal. Mach. Intell.*, 31(5):824–840, 2009. Available: `http://make3d.cs.cornell.edu/`.

[22] Singular Inversions Inc. Facegen modeller manual. `www.facegen.com`, 2009.

[23] Y. Taigman and L. Wolf. Leveraging billions of faces to overcome performance barriers in unconstrained face recognition. *arXiv preprint arXiv:1108.1122*, 2011.

[24] H. Tang, Y. Hu, Y. Fu, M. Hasegawa-Johnson, and T. S. Huang. Real-time conversion from a single 2d face image to a 3D text-driven emotive audio-visual avatar. In *Int. Conf. on Multimedia and Expo*, pages 1205–1208. IEEE, 2008.

[25] USF. DARPA Human-ID 3D Face Database:. Courtesy of Prof. Sudeep Sarkar, University of South Florida, Tampa, FL.

[26] A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proc. int. conf. on Multimedia*, pages 1469–1472, 2010. Available: `www.vlfeat.org/`.

[27] P. Viola and M. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, 2004.

[28] Vizago Research GmbH. 3D face reconstruction. `www.vizago.ch`.

[29] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Trans. on Graphics*, 24(3):426–433, 2005.

[30] C. Wang, S. Yan, H. Li, H. Zhang, and M. Li. Automatic, effective, and efficient 3D face reconstruction from arbitrary view image. In *Pacific Rim Conf. on Multimedia*, pages 553–560. Springer-Verlag, 2004.

[31] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas. Expression flow for 3D-aware face component transfer. *ACM Trans. on Graphics*, 30(4):60, 2011.

[32] M. Zhou, L. Liang, J. Sun, and Y. Wang. AAM based face tracking with temporal matching and face segmentation. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 701–708, 2010.

[33] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Proc. Conf. Comput. Vision Pattern Recognition*, pages 2879–2886, 2012. Available: `www.ics.uci.edu/~xzhu/face/`.