Unsupervised Depth Estimation, 3D Face Rotation and Replacement

Joel Ruben Antony Moniz¹, Christopher Beckham²,³, Simon Rajotte²,³, Sina Honari², Christopher Pal²,³
¹Carnegie Mellon University, ²MILA-University of Montreal, ³Ecole Polytechnique of Montreal
¹jrmoniz@andrew.cmu.edu, ²honaris@iro.umontreal.ca, ³firstname.lastname@polymtl.ca

Abstract

We present an unsupervised approach for learning to estimate three dimensional (3D) facial structure from a single image while also predicting 3D viewpoint transformations that match a desired pose and facial geometry. We achieve this by inferring the depth of facial keypoints of an input image in an unsupervised manner, without using any form of ground-truth depth information. We show how it is possible to use these depths as intermediate computations within a new backpropable loss to predict the parameters of a 3D affine transformation matrix that maps inferred 3D keypoints of an input face to the corresponding 2D keypoints on a desired target facial geometry or pose. Our resulting approach can therefore be used to infer plausible 3D transformations from one face pose to another, allowing faces to be frontalized, transformed into 3D models or even warped to another pose and facial geometry. Lastly, we identify certain shortcomings with our formulation, and explore adversarial image translation techniques as a post-processing step to re-synthesize complete head shots for faces re-targeted to different poses or identities.

1 Introduction

Faces are a particularly important type of visual object as underscored by the fact that the human brain contains a specialized region for recognizing faces known as the fusiform face area. Correspondingly, facial pose normalization, which is sometimes referred to as frontalization [8], and pose-invariant face verification are both important and well studied problems in computer vision [22, 17]. Pose normalization can be an important pre-processing step leading to improved accuracy for face classifiers or verifiers [17, 26, 19, 11], or when used as input to a wide variety of other facial analysis tasks. These tasks become much more difficult when a given face is not facing the camera.

In this paper we present a novel unsupervised learning technique for face rotation and warping from a 2D source image – whose facial appearance will be used in the rotation – to a target face – to which the facial pose and geometry inferred from the source image is mapped. This is achieved through estimating the source face depth and the 3D affine parameters that warps the source to the target face using neural networks. These neural networks use a novel loss formulation for the structured prediction of keypoint depths. Once the 3D affine transformation matrix is estimated, it can be used to warp the source image onto the target face geometry using a textured triangular mesh. The use of a 3D affine transform means that we can capture both a 3D rotation of the face to a new viewpoint as well as a global non-Euclidean warping of the geometry to match a target face. We call these neural networks Depth Estimation-Pose Transformation Hybrid Networks, or DepthNets in short.

Our first contribution is to propose a neural architecture that predicts both the depth of source keypoints as well as the parameters of a 3D geometric affine transformation which constitute the explicit outputs of the DepthNet model. The predicted depth and affine transformation could be then used to map a source face to a target face for object orientation, distortion and viewpoint changes.

Preprint. Work in progress.
Our second contribution consists of making the observation that given 3D source and 2D target keypoints, closed form least squares solutions exist for estimating geometric affine transformation models between these sets of keypoint correspondences, and we can therefore develop a model that captures the dependency between depth and the affine transformation parameters. More specifically, we express the affine transformation as a function of the pseudoinverse transformation of 2D keypoints in a source image – augmented by inferred depths – and the target keypoints. Thus, the second and major contribution in this work is therefore the fact that with our formulation we are able to capture the relationship between estimated affine transformation and the inferred depth as a deterministic relationship, without introducing extra parameters for the geometric affine transformation. In this formulation, DepthNet only predicts depth values explicitly and the affine parameters are indirectly estimated through pseudoinverse transformation of source and target keypoints.

Our proposed DepthNet can map the central region of the source face to the target geometry. This leads to background mismatch when warping one face to another. Finally, our third contribution is to use an adversarial unpaired image-to-image transformation approach to repair the appearance of 3D models inferred from DepthNet. Together these contributions allow 3D models of faces that construct realistic images in the target pose. Our proposed method can be used for pose normalization or face swap with no manually specified 3D face model. To the best of our knowledge, this is the first such neural network based model that estimates a 3D affine transformation model for face rotation which neither requires ground-truth 3D images nor any ground truth 3D face information such as depth.

2 Our Approach

As we have outlined above, our approach uses neural networks for inferring depth and geometric transformation – referred to as DepthNets; and, an adversarial image-to-image transformation network which improves the quality of the appearance of a 3D model inferred from a DepthNet.

DepthNets

We propose three DepthNet formulations, described in Sections 2.1, 2.2, and 2.3. For each of the above three models we explore two architectural scenarios: (A) a Siamese-like architecture that uses the source and target images themselves as well as keypoints extracted from these images, and (B) a fully-connected neural network variant which uses only facial keypoints in the source and target images. See Figure 1 (left) for details.

![Figure 1: (Left) DepthNet architecture. The blue region is only used in case (A) and the red part is used in both cases (A) and (B), described in Section 2. The orange output (the 8 affine transformation parameters) is predicted only by model variations described in Sections 2.1 and 2.2, and not the model described in section 2.3. All three models predict the N depth values of the source keypoints. C, P, and FC correspond to valid conv, pool and fully-connected layers. The two paths of Siamese network share parameters and the black dots indicate concatenating keypoint values to FC units. (Right) Visualizing face rotation by re-projecting a frontal face (far left) to a range of other poses defined by the faces in the row above (in each pair of rows). In this experiment, we only use keypoints from the top-row in the DepthNet model (Model 7 in Table 1).](https://example.com/dnn.png)

The depths learned for keypoints by these approaches are not necessarily true depths, but are likely to strongly correlate with the actual depth of each keypoint. This is because even though the method succeeds (as we shall see below) in aligning poses, the inferred depth and the affine transform may each be scaled by factors so as to cancel each other out (i.e., by factors which are multiplicative inverses of each other). Real world viewpoint geometry also involves perspective projection.

It is interesting to note that if DepthNets are used to register a set of images of objects to the same common viewpoint, the same image and geometry can be used as the target. This is the case for
the frontalization of faces, for example. While the DepthNet framework is sufficiently general to be applied to any object type where 2D keypoint detections have been made, our experiments here focus on faces. We describe the three variants of DepthNets below.

2.1 Predicting Depth and Viewpoint Separately

In this variant of DepthNets, the model predicts both depths and viewpoint geometry, but as separate explicit outputs of a neural network. The input is comprised of only the geometry and pose of the source and target faces (encoded in the form of a 2D keypoint template), in case (B), or both keypoints and images of the source and target faces, in case (A). The key phases of this stage are described by the sequence of steps given below:

1. keypoint extraction: Raw \((x, y)\) pixel coordinates corresponding to the keypoints of each image are extracted using a Recombinator Network (RCN) architecture, and then concatenated before being passed into the keypoint processing step.

2. (Optional) Image Feature Extraction: DepthNets can be conditioned on only keypoints, case (B), or on keypoints and the original images, case (A). We can therefore optionally subject the source and target images to alternating conv-maxpool layers. If this component of the architecture is used, the last spatial feature maps in the Siamese architecture are concatenated before being given to a set of densely connected hidden layers.

3. keypoint processing: In this step keypoints are passed through a set of hidden layers. If the Image Feature Extraction stage is used, the keypoints are concatenated to image features, the output of which is in turn fed to densely connected layers. The output layer of this phase will be of size \(N\), where \(N\) is the number of keypoints. The first \(N\) points represent the Depth proxy, and the last 8 points form a \(4 \times 2\) matrix representing the learned parameters of the affine transform. See Figure 1.

4. Geometric Affine Transformation Normalizer: This phase applies the predicted affine transform on each (depth augmented) source keypoint to obtain its target location. Let \((x_s^i, y_s^i)\) represent the \(i\)th source keypoint, \((x_s^m, y_s^m)\) the corresponding source normalized keypoint, estimated by applying the affine transformation matrix, \((x_t^i, y_t^i)\) the \(i\)th target keypoint (as ground truth (GT)), and \(I_s\) and \(I_t\) represent the source and target images respectively. Depending on which underlying architectural variant we use two cases arise: one that utilizes only the keypoints (B), and the other utilizing both the keypoints and the images (A). Since the keypoints are generated using RCNs, they are technically functions of the input images: \([x_s, y_s] = R(I_s)\), and \([x_t, y_t] = R(I_t)\). Depending on the (A) or (B) variant, the \(i\)th keypoint’s predicted depth proxy \(z_s^i\) is inferred as a function of the input keypoints, or both input keypoints and input images. The keypoints in turn being derived from the images, we get \(z_s^i = z_p^i(I_s, I_t)\) in both cases. Similarly, the 3D-2D affine transform \(F\) is a function of the input points or of both the input points and images, such that \(F = F(I_s, I_t)\), where the 8 predicted parameters are: \(m = \{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8\}\). These constitute the 3D-2D affine transform which is common to all keypoints. In other words, each of the \(i\) points is transformed using \(x_s^i = F(I_s, I_t) \cdot x_s^i\), or:

\[
\begin{bmatrix}
    x_s^i \\
    y_s^i
\end{bmatrix} = \begin{bmatrix} m_1 & m_2 & m_3 & t_x \\
    m_4 & m_5 & m_6 & t_y \end{bmatrix} \begin{bmatrix}
    x_s^i \\
    y_s^i \\
    1
\end{bmatrix}
\]

Note, however, in the case of (B), when it uses a 2D template in lieu of the target image keypoints, the target is effectively hard-coded, and no longer forms a variable input. The predicted depth and affine then become functions purely of the source input image: \(z_p^i = z_p^i(I_s)\) and \(F = F(I_s)\).

The loss function of a DepthNet is obtained by transforming the source face to match the target face using the simple squared error of the corresponding target object’s keypoint vector \(x_t = [x_t, y_t]^T\), as GT values, and the source object’s normalized keypoint vector \([x_n, y_n]^T\), all normalized to the range \([0, 1]\) by dividing with the total image size. The loss for one example where we predict depth and affine viewpoint geometry can therefore be expressed as:

\[
L = \sum_{i=1}^{K} \left\| x_t^i - F(I_s, I_t) [x_s^i, y_s^i z_p^i(I_s, I_t)]^T \right\|^2
\]

5. Image Warper: This phase consists of using the depth proxy and affine transform matrix generated to actually warp the face from its source pose to be matched to the target object geometry. The final projection to 2D is achieved by simply dropping the transformed \(z\) (which corresponds
to an orthographic projection model). In the case of DepthNets, this orthographic projection is effectively embedded in the Geometric Affine Transformation Normalizer step, since the affine correspondence to the $z$ coordinate is not predicted, essentially dropping it. As we operate on keypoints, the actual warping of pixels can be performed with a high quality OpenGL pipeline that performs the warp separately from the rest of the architecture. Source Image, keypoints augmented with depth, and the affine matrix are passed to OpenGL pipeline to warp the source image towards the target pose. This warping is not needed during training, which has the advantage of processing depth and 3D geometry regardless of image resolution at train and test time.

2.2 Estimating Viewpoint Geometry as a Second Step

In this model variant, training is similar to Section 2.1 and the model outputs depth and 3D affine transformation parameters. However, at test time, rather than using the predicted 3D affine transformation for pairs of faces, we use only the predicted depths and estimate the affine geometry parameters as a second estimation step. More precisely, given 3D points for a scene and the corresponding 2D points for a target geometry it is possible to formulate the estimation of a 3D affine transformation as a linear least squares estimation problem. An overdetermined system of the form $Am = x_t$ for this problem can be constructed as shown in (2).

This corresponds to an affine camera model followed by an orthographic projection to 2D keypoints. This setup also leads to the following closed form solution for the affine transformation parameters:

$$m = [A^T A]^{-1} A^T x_t,$$

(3)

where this pseudoinverse based transformation is parameterized by the reference points and their predicted depths.

2.3 Joint Viewpoint and Depth Prediction

Our key observation is that one can alternatively use the closed form analytical solution, measured in Eq. (3), for the least squares estimation problem as the underlying affine transformation matrix within the loss function. This leads to a special form of structured prediction problem for geometrically consistent depths and affine transformation matrix. For each image we have $L = \sum_{i=1}^{K} \sum_{i=1}^{N} \left\| x_i^j - \operatorname{reshape}([A^T A]^{-1} A^T x_t) x_i^j \right\|_2^2$

where the matrix $A$ is parameterized as a function of $x_o$ as shown in Eq. (2). In this variant, the model explicitly outputs only depth values during train and test time. The affine transformation matrix in the equation above is replaced by Eq. (3), which measures the affine transformation as a pure function of source and target keypoints plus the inferred depth. The big difference of this formulation compared to Sections 2.1 and 2.2 is that geometric affine transformation parameters are no longer predicted by DepthNet during training and at both train and test time – it solves the least square loss through the pseudoinverse based transformation. Since $z_i^j = x_o^j \{(x_i^j, y_i^j, x_t^j, y_t^j)\}_{j=1...N}$ is predicted within the analytical formulation of the solution to the least squares minimization problem, we can backpropagate through the solution of a minimization problem that depends on the predicted depths.

Adversarial Image-to-Image Transformation

DepthNet transforms the central region of the source face to the target pose. Inevitably, the face background will be missing, which might make the proposed method unsuitable for many application where the full face is required. To address this issue, we utilize CycleGAN [25], an adversarial image-to-image translation technique. This serves to repair the background of faces that have undergone frontalization or face swap through the DepthNet pipeline. Importantly, the adversarial
nature of CycleGAN allows one to perform image transformation between two domains without the requirement of paired data. In our work, we perform experiments translating between various domains of interest but one example is translating between the domain of images in the dataset (i.e. the ground truth) and the domain of images where the DepthNet output is pasted onto the face region (in the case of face-swap). By doing so we clean the face background in an unsupervised manner.

3 Related Work

3.1 3D Transformation and Faces

While there is a large body of literature on 3D facial analysis, many standard techniques are not applicable to our setting here. As an example, morphable models [1] cover a wide variety of approaches which are capable of high quality 3D reconstructions, but such methods usually require 3D face scans or reconstructions from multi-view stereo to be assembled so as to learn complex parametric distributions over face shapes. One of the closest approaches to our own is that of [7] on viewing real world faces in 3D. Like our work this approach does not require aligned 3D face scans, highly engineered models or manual interventions. In this work they make the observation that if 2D keypoints can be obtained from a single input image of a face and these keypoints are matched to an arbitrary 3D target geometry, then standard camera calibration techniques can be used to estimate plausible intrinsics and extrinsics of the camera. This allows the estimated camera matrix, 3D rotation matrix and 3D translation vector to be used to transform the target 3D model to the pose of the query image from which an approximate depth can be obtained. Another work has explored the use of a single unmodified 3D surface as an approximation to the shape of all input faces [8]. In contrast, our approach only requires 2D keypoints from the source and target faces as input. It then estimates the depth of the source face keypoints, thereby inferring an image specific 3D model of the face.

DeepFace [17] uses face frontalization to improve the performance of a face verification system. It uses a 3D mask composed of facial keypoints, detect the corresponding locations of these keypoints in the image, and map the 3D model onto the plane of the face. They then use this model to do a piecewise-affine transform on the face to frontalize it. In contrast, we do not require a 3D target face model. We only require an arbitrary 2D target keypoints - no target appearance or textured 3D model is needed. Also, in our model only parameters of one affine camera model are learned for the entire facial keypoints, rather than a separate, per-pixel set of weights. Our approach is therefore less tied to pixel values of faces in particular and is applicable to any sort of object where 2D keypoint locations can be placed into semantic correspondence and geometric warping under a 3D affine model.

3.2 Depth Estimation

While several models [12; 2; 14] estimate depth with direct supervision, there has been recent models [24; 5; 3] that estimate depth in an unsupervised training procedure. These models rely on pixel reconstruction by using frames that are captured from similar scenes, e.g. nearby frames of a video [24] or left-right frames captures by stereo cameras [5; 3]. These models estimate depth on one frame and then by using the disparity map, measure how pixel values of nearby frames compare to each other. To do this, they also require camera intrinsic parameters, e.g. focal length or distance between cameras. Unlike these models, our approach does not require pixel level reconstruction. This allows mapping faces from different people with completely different skin colors, without knowing camera parameters or how they are positioned with respect to each other.

Tung et. al [20] estimate 3D human pose in videos, where they use synthetic data to pre-train internal parameters of the model and fine-tune them by keypoint, segmentation and motion loss. Adversarial Inverse Graphics Networks (AIGN) [21] estimates 3D human pose from 2D keypoint heatmaps in a semi-supervised manner with a similar formulation to that of CycleGAN. They apply an adversarial loss on the 3D pose to make them look realistic. These models leverage the depth values either through synthetic data [20], or by adversarial usage of ground truth depth values [21]. Unlike these models, our model does not rely on any depth signal, either directly or indirectly. MOFA [18] builds a 3D face mesh using a single image, where the 3D face parameters such as 3D shape and skin reflectance are estimated by an encoder and then using a differentiable model they are rendered back to the image by the decoder. This model requires manual initialization to map the input image to the 3D mesh, since otherwise it is doing an unconstrained optimization by adapting both the face pose and the skin reflectance. Our model, does not require any manual initialization or tuning.
The goal of aforementioned models is either to predict depth in images/videos [24; 5; 3] or to measure 3D human pose [20; 21; 18]. Our model, however, measures depth as an intermediate step without any supervision, in order to find the transformation parameters that maps a source to a target face. Therefore, estimating depth is not the final goal of our model as we aim to solve a different problem compared to prior work, which is finding parameters that map one face to another.

### 3.3 Generative Adversarial Networks

There has undoubtedly been an abundant literature exploring generative adversarial networks. Recently, adversarial models in [11; 19] have explored face rotation using purely neural networks. TP-GAN [11] performs face frontalization through introducing several losses to preserve identity and symmetry of the frontalized faces. The model requires pairs of images of the same identity during train. DR-GAN [19] rotates faces to any target pose by using a discriminator that also does identity classification, to maintains identity of faces, in addition to pose prediction to verify accurate face rotation. While these models do pure face rotation of an input 2D face, our model can warp the input face to any other target face, allowing warping the input face to any other identity, with a different geometry and pose. Moreover, our model also estimates the 3D geometric affine transformation parameters explicitly, allowing these parameters to be used later, e.g., for face texture swap.

### 4 Experiments

#### 4.1 DepthNet Evaluation on Paired Faces

For the experiments in this section, we use a subset of the VGG dataset [15], with training and validating on all possible pairs of images belonging to the same identity for 2401 identities. This yields 322,227 train and 43,940 validation pairs. Check experimental setup details in Supplementary.

| Model | Color | MSE | MSE_norm |
|-------|-------|-----|----------|
| 1) A simple 2D affine registration | grey | 1.562 | 9.547 |
| 2) A 3D affine registration model using an average 3D face template | purple | 0.724 | 4.486 |
| 3) A DepthNet that separately estimates depth and geometry | brown | 0.566 | 3.292 |
| 4) The model above, but with a Siamese CNN image model | violet | 0.539 | 3.115 |
| 5) Secondary least squares estimation for visual geometry using the depths from 3 | red | 0.400 | 2.184 |
| 6) Secondary least squares estimation for visual geometry using the depths from 4 | green | 0.399 | 2.175 |
| 7) Backpropagation through the pseudo-inverse based solution for visual geometry | orange | 0.357 | 4.932 |
| 8) The model above, but with a Siamese CNN image model | blue | 0.349 | 4.891 |

Table 1: (left) Comparing the Mean Squared Error (MSE) and MSE normalized by inter-ocular distance (MSE_norm) of Different Models. (right) Histogram of Mean Squared Errors. The second column in the Table (on left) corresponds to the color of the model in the figure (on right).

We explore the three variants of DepthNets described in Sections 2.1, 2.2, and 2.3 each with two architectural cases (A) and (B), depending on whether image features are used in addition to keypoints or not. We also compare with a number of baselines. We measure the mean square error (MSE) between the estimated keypoints on the target face and ground truth target keypoints. Results for the following models are shown in Table 1:

1) A baseline model registrations using a simple 2D affine transformation.

2) We generate a 3D average face template from the 3DFAW dataset [13; 23; 6] by aligning the 3D keypoints of all faces in the dataset to a front-facing face using Procrustes superimposition. We report error by mapping the template face to each source face via Procrustes superimposition (to get a 3D face f) and then use an affine transformation from the 3D face f to the target face.

3, 4) We use our proposed approach to predict both depth and geometry (described in Sections 2.1, 2.2, and 2.3) each with two architectural cases (A) and (B), depending on whether image features are used in addition to keypoints or not. We also perform a secondary optimization to determine the optimal geometry at test time (described in Section 2.2). Note that during training, these two cases are similar to models 3 and 4 in Table 1.

7, 8) For these models we express the complete visual geometry estimation solution using the pseudo-inverse formulation, allowing backpropagation through the depth (described in Section 2.3).

As observed in Table 1, a simple 2D affine transform (model 1) without estimating depth and a template 3D face (model 2) get high errors on mapping to the target faces. DepthNet models get lower errors and the pseudo-inverse formulation (models 7 and 8) further reduces the error by 10%. The CNN models slightly reduce errors compared to their equivalent models that rely only on keypoints.
4.2 Depth Estimation and Comparison to other Models

In this section we train DepthNet on unpaired faces belonging to different identities and compare with other models that estimate depth. We use the 3DFAW dataset [13, 23, 6] that contains 66 3D keypoints to facilitate comparing with ground truth (GT) depth. It provides 13,671 train and 4,500 valid images. We extract from the valid set, 75 frontal, left and right looking faces yielding a total of 225 test images, which provides a total of 50,400 source and target pairs. We train the pseudoinverse DepthNet model that relies on only keypoints (model 7 in Table 1). We also train a variant of DepthNet that applies an adversarial loss on the depth values (DepthNet+GAN). This model uses a conditional discriminator that is conditioned on 2D keypoints and discriminates GT from estimated depth values. The model is trained with both keypoint and adversarial losses.

| Source   | DepthNet | DepthNet + GAN |
|----------|----------|----------------|
| Left     | 9.55     | 15.06          |
| Front    | 8.67     | 7.29           |
| Right    | 15.08    | 15.14          |
| Avg      | 17.65    | 58.23          |

Table 2: Comparing DepthCorr for different DepthNet models when mapping variant source to target poses. The Avg column measures the average over the three preceding columns.

We measure the correlation matrix between GT and estimated depths, where the element $k$ in the diagonal indicates the correlation between estimated and ground truth depth values for keypoint $k$, yielding a value between -1 and 1. We report the sum of absolute values of the diagonal of this matrix, indicated by DepthCorr. We compare DepthNet models on DepthCorr in Table 2. For this experiment we take every possible pair of source to target faces, where source and target are one of {left, front, right} looking faces. This yields a total of 5,550 pairs when the source and the target are from the same subset, and 5,625 pairs otherwise. This experiment indicates the cases in which the estimated depth values are more realistic. As shown in Table 2, the more distinct the source and target poses, the better are the results of the base DepthNet model. This is expected, because the model needs a more accurate depth estimation to correctly warp the face to a different pose. DepthCorr improves for the DepthNet+GAN, indicating a direct loss on depth can enhance the depth estimation.

| Model          | Need Depth | Manual Init. | MSE | Depth Correlation Matrix Trace (DepthCorr) |
|----------------|------------|--------------|-----|------------------------------------------|
| GT Depth       | Yes        | -            | 0.0415 ± 0.0215 | 56.797 |
| AIGN [21]      | Yes        | No           | 0.00627 ± 0.0207 | 17.615 |
| MOFA [18]      | No         | Yes          | 0.01741 ± 0.0074 | 57.145 |
| DepthNet (Ours)| No         | No           | 0.00102 ± 0.0007 | 17.653 |
| DepthNet + GAN (Ours)| Yes | No | 0.000080 ± 0.00005 | 57.797 |

Table 3: Comparing MSE and DepthCorr for different models. A lower MSE indicates the model maps better to the target faces. A higher DepthCorr indicates more correlation between estimated and GT depths.

We compare our two DepthNet models with three baselines: 1) AIGN [21], 2) MOFA [18] and 3) GT Depth (no model trained). AIGN estimates 3D keypoints conditioned on 2D heatmaps of the keypoints. MOFA estimates a 3D mesh using only an image. We implemented the AIGN model and asked the authors of MOFA to run their model on our test-set. They provided MOFA’s results for 134 images in the test set. In Table 3 we compare these three models with our DepthNet models on DepthCorr. We also compare them on MSE, which is measured between GT and estimated target keypoints. Since the three baselines estimate depth on a single image due to their different model formulation, we first measure $m$ using closed form solution in Eq. 3 and then apply $m$ to the estimated source keypoints to get the target keypoint estimations. We contrast the estimated values with the GT target keypoints. As shown in Table 3, GT depth has the highest DepthCorr (the maximum possible value). The depths estimated by AIGN and DepthNet+GAN have stronger correlation to GT depth compared to the baseline DepthNet and MOFA. On MSE the DepthNet models get smaller error when mapping to target faces, indicating they are better suited for this task.

In Figure 2 we plot heatmaps of the estimated depth of different models (on Y axis) and the GT depth (on X axis) aggregated over all 66 keypoints on all test data. As can be seen, the depth estimated by DepthNet+GAN and AIGN models form a 45 degree rotated ellipses showing a stronger linear correspondence with respect to the GT depth compared to the the baseline DepthNet and MOFA.

In Figure 3 we show some estimated depth samples for different models (see more samples in Figure S1). AIGN and DepthNet+GAN generate more realistic results. MOFA generates very similar face
templates for different faces. The baseline DepthNet model generates flatter faces, since the estimated depth and affine params can be scaled by a factor to cancel each other out.

By comparing different models in Table 3, MOFA requires proper initialization to map face meshes to the image. AIGN requires depth labels to train the model. Our baseline DepthNet model neither require any depth labels nor any manual tuning. We would also like to emphasize that MOFA and AIGN are designed to estimate a 3D model, while DepthNet is designed to estimate the parameters that facilitate warping a face pose to another without having depth values, so these models are designed to solve different problems.

An interesting observation is that GT depth gets a higher MSE compared to DepthNet. This can be due to not having a perspective projection between source and target faces. However, since DepthNet is trained to map to the target faces, it learns the affine parameters in a way to minimize this loss.

4.3 Face Rotation, Replacement and Adversarial Repair

In this section, we show how the DepthNet model can be used for different applications. In Figure 1 we visualize the face rotation by re-projecting a frontal face, from Multi-PIE [6], (far left) to a range of other poses defined by the faces in the row above. These views are rendered from a 3D model in OpenGL. Note the model can align well to the pose of the target faces.

In another experiment, we do face frontalization with synthesized background. Here we use CycleGAN to add background detail to a face that has been frontalized with DepthNet. Referring to Figure 4, we perform this by conditioning the CycleGAN on the DepthNet image (column 3) and the background of column 2 (masking interior face region determined by the convex hull spanned by the keypoints). The second domain contains ground truth frontal faces.

Finally, we do face swaps, where we warp the face of one identity onto the geometry and background of another identity using DepthNet. To do so, we paste the rotated face by DepthNet onto the background of the target image and train a CycleGAN to map from the domain of `swapped in faces`
to the ground truth faces in our dataset, effectively learning to clean up face swaps so that the face region matches the hair and background. Some examples of this procedure are shown in Figure 5.

Figure 5: Left to right: source face; target face; warp to target; repaired result.

5 Conclusion

We have proposed a novel neural network approach to 3D face model creation which enables relative pose normalization or frontalization without using any ground truth depth data. We achieve our best quantitative keypoint registration results using our novel formulation for predicting depth and 3D visual geometry simultaneously, learned through backpropagating through the analytic solution for the visual geometry estimation problem expressed as a function of predicted depths. We have illustrated the quality and utility of the depths and 3D transformations obtained using our method by transforming source faces to a wide variety of target poses and geometries. Our technique can be used for face rotation and replacement and when combined with adversarial appearance repair it can blend warped faces to also synthesize the background. The proposed model, however, has some deficiency when face is not neutral due to learning a shared affine parameters for all keypoints. Moreover, for extreme non-frontal faces, while DepthNet can extract the transformation params (since it only relies on keypoints), OpenGL cannot extract texture due to occlusion. We show an example of how to address this in the supplementary material.

6 Acknowledgment

We would like to thank Samsung for partially funding this project. We are also thankful to Compute Canada and Calcul Quebec for providing computational resources, and to Poonam Goyal for helpful discussions.

References

[1] Blanz, Volker and Vetter, Thomas. A morphable model for the synthesis of 3d faces. In Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp. 187–194. ACM Press/Addison-Wesley Publishing Co., 1999.

[2] Eigen, David, Puhrsch, Christian, and Fergus, Rob. Depth map prediction from a single image using a multi-scale deep network. In Advances in Neural Information Processing Systems, pp. 2366–2374, 2014.

[3] Garg, Ravi, BG, Vijay Kumar, Carneiro, Gustavo, and Reid, Ian. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In European Conference on Computer Vision, pp. 740–756. Springer, 2016.

[4] Glorot, Xavier and Bengio, Yoshua. Understanding the difficulty of training deep feedforward neural networks. In International conference on artificial intelligence and statistics, pp. 249–256, 2010.

[5] Godard, Clément, Mac Aodha, Oisin, and Brostow, Gabriel J. Unsupervised monocular depth estimation with left-right consistency. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[6] Gross, Ralph, Matthews, Iain, Cohn, Jeffrey, Kanade, Takeo, and Baker, Simon. Multi-pie. Image and Vision Computing, 28(5):807–813, 2010.

[7] Hassner, Tal. Viewing real-world faces in 3d. In Proceedings of the IEEE International Conference on Computer Vision, pp. 3607–3614, 2013.
[8] Hassner, Tal, Harel, Shai, Paz, Eran, and Enbar, Roee. Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4295–4304, 2015.

[9] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

[10] Honari, Sina, Yosinski, Jason, Vincent, Pascal, and Pal, Christopher. Recombinator networks: Learning coarse-to-fine feature aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5743–5752, 2016.

[11] Huang, Rui, Zhang, Shu, Li, Tianyu, and He, Ran. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[12] Jackson, Aaron S, Bulat, Adrian, Argyriou, Vasileios, and Tzimiropoulos, Georgios. Large pose 3d face reconstruction from a single image via direct volumetric cnn regression. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 1031–1039. IEEE, 2017.

[13] Jeni, László A, Cohn, Jeffrey F, and Kanade, Takeo. Dense 3d face alignment from 2d videos in real-time. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pp. 1–8. IEEE, 2015.

[14] Liu, Fayao, Shen, Chunhua, and Lin, Guosheng. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5162–5170, 2015.

[15] Parkhi, Omkar M, Vedaldi, Andrea, and Zisserman, Andrew. Deep face recognition. *Proceedings of the British Machine Vision*, 1(3):6, 2015.

[16] Sagonas, Christos, Tzimiropoulos, Georgios, Zafeiriou, Stefanos, and Pantic, Maja. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 397–403, 2013.

[17] Taigman, Yaniv, Yang, Ming, Ranzato, Marc’Aurelio, and Wolf, Lior. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014.

[18] Tewari, Ayush, Zollhöfer, Michael, Kim, Hyeongwoo, Garrido, Pablo, Bernard, Florian, Perez, Patrick, and Theobalt, Christian. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, 2017.

[19] Tran, Luan, Yin, Xi, and Liu, Xiaoming. Disentangled representation learning gan for pose-invariant face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[20] Tung, Hsiao-Yu, Tung, Hsiao-Wei, Yumer, Ersin, and Fragkiadaki, Katerina. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pp. 5242–5252, 2017.

[21] Tung, Hsiao-Yu Fish, Harley, Adam W, Seto, William, and Fragkiadaki, Katerina. Adversarial inverse graphics networks: Learning 2d-to-3d lifting and image-to-image translation from unpaired supervision. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, 2017.

[22] Zhang, Xiaozheng and Gao, Yongsheng. Face recognition across pose: A review. *Pattern Recognition*, 42(11):2876–2896, 2009.

[23] Zhang, Xing, Yin, Lijun, Cohn, Jeffrey F, Canavan, Shaun, Reale, Michael, Horowitz, Andy, Liu, Feng, and Girard, Jeffrey M. Bp4d-spontaneous: a high-resolution spontaneous 3d dynamic facial expression database. *Image and Vision Computing*, 32(10):692–706, 2014.
[24] Zhou, Tinghui, Brown, Matthew, Snavely, Noah, and Lowe, David G. Unsupervised learning of depth and ego-motion from video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[25] Zhu, Jun-Yan, Park, Taesung, Isola, Phillip, and Efros, Alexei A. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593, 2017.

[26] Zhu, Xiangyu, Lei, Zhen, Yan, Junjie, Yi, Dong, and Li, Stan Z. High-fidelity pose and expression normalization for face recognition in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 787–796, 2015.
S.1 Experimental Setup

Here we describe the details of the experiments carried out in Section 4.1. Our DepthNet architectures require keypoints of both source and target images in a pair to be extracted. For this, the image is first passed through the VGG-Face [15] face detector. The face crops are then scaled down to the size $80 \times 80$ and converted to greyscale, following which they are passed through the RCN to obtain $N = 68$ keypoints on each image. The RCN is trained exactly as described in [10], using the 300W dataset [16].

The keypoint only variant of our model involves concatenating all detected keypoints and passing them through a two-layer deep fully connected network, with 256 hidden and $o$ output units respectively, where $o$ depends on whether we are predicting only the depth, when $o = N$, or the depth and an affine transform as well, when $o = N + 8$.

As discussed above, it is possible to augment these models with a Siamese CNN module (case A). The model variants that also use the image, pass both the source and the target images through three conv-maxpool layers with shared weights of size $(32, 4, 2), (48, 3, 2), (64, 2, 2)$, respectively for the representation $(\text{num\_filters}, \text{filter\_size}, \text{pool\_size})$. The output is concatenated before passing them into a 4-layered fully connected network with respective output sizes of 2048, 512, 256 and $o$. The keypoints are concatenated to the 512 units before being passed to the last two layers. See Figure 1 for illustration of the model. We explore these Siamese CNN augmented variants in Model 4, 6 and 8 in Table 1.

We set the initial learning rate of 0.001 and use a Nesterov momentum optimizer (with a momentum of 0.9) in all our experiments. With the exception of the last layer, we initialize all weights using a Glorot initialization scheme [4], with the weights sampled from a uniform distribution. We use a ReLU gain [9], set all biases to 0, and apply a ReLU non-linearity after every layer. In the final output layer, we do not apply any non-linearity and initialize the weights to 0. The biases of units that represent depths are initialized to a random Normal distribution with $\mu = 0$ and $\sigma = 0.5$, while those that form the predicted affine transform are initialized with the equivalent of a “flattened” identity transform. All models have been trained for 500 epochs.

We point out that except for a comparison between learning rates in the set $\{0.01, 0.001, 0.0001\}$ over a few (less than 10) epochs to find a learning rate that the model seems to train well with, we have not performed a hyperparameter search, and anticipate that the performance of the model can be made even better by searching the hyperparameter space on a per model basis and by using deeper (or modified) architectures.

S.2 Depth Visualization

We show further estimated depth values by different models in Figure S1. AIGN and DepthNet+GAN generate the closest depth values to the GT depth. The baseline DepthNet model produces relatively flatter faces, since the estimated depth and the affine params can be scaled by a factor to cancel out each other. In DepthNet, the estimated depth indicates the position of each keypoint relative to other keypoints rather than with respect to a source and importantly it is done without any supervision. MOFA, which is also unsupervised, generates very similar face templates for different cases.

S.3 Additional Camera Sweep Visualizations

In this section, we present additional visualizations along the lines of those shown in Figure 1(right) in Section 2 of the main paper. Frontal faces selected from the Multi-PIE dataset [6] are re-projected to match several other poses corresponding to a person with a different identity. The DepthNets predicts reliable proxy depths, which when coupled with the analytically obtained affine transform (obtained from the least-squares pseudo-inverse-based solution described in Section 2.3 of the main
Figure S1: Depth visualization for different models (color coded by depth). From left to right: RGB image, Ground Truth, DepthNet, DepthNet+GAN, AIGN and MOFA estimated depth values.

paper) yields faces close to the desired target face geometry when passed through the OpenGL pipeline.

Figure S2: Projecting a frontal face (far left) to a range of other poses define by the faces in the row above. We show camera sweeps for frontal source faces in Figures S2 and S3 and non-frontal source faces in Figure S4. In Figure S5, we use the same target identity as the source face, showing how much the generated face differs from the ground truth target. For all samples in Figures S2, S3, S4, and S5 we use DepthNet model that relies on only key-points (model 7 in Table 1).

Note that for non-frontal source faces, the quality of images is reduced specially for frontal target faces. This is due to lack of adequate texture on the occluded side of the face to be transferred to the target pose by OpenGL pipeline, rather than inaccuracies in the affine transformation parameters. In order to reduce the side-affects, we use either of the source face or its flipped version, that is closer to the target face pose, and then warp the face to the target keypoints.
S.4 CycleGAN

Suppose we have some images belonging to one of two sets $x \in X$ and $y \in Y$, where $x$ denotes a a DepthNet-frontalized face and $y$ a ground truth face which is frontal. We wish to learn two functions $F : X \to Y$ and $G : Y \to X$ which are able to map an image from one set to the corresponding image in the other. Correspondingly, we have two discriminators $D_X$ and $D_Y$ which try to detect whether the image in that particular set is real or generated. As mentioned in the main text, while we are only interested in the function $F : X \to Y$ (since this essentially tries to clean up the face and make it plausibly look like it came from the ground truth) the formulation of CycleGAN requires that we learn mappings in both directions during training. We optimize the following objectives for the two generators $F$ and $G$:

$$\min_{G,F} \mathbb{E}_{x,y} \left[ \ell(D_X(G(y), 1) + \ell(D_Y(F(x), 1) + \lambda||y - F(G(y))||_1 + \lambda||x - G(F(x))||_1 \right] \quad (4)$$

And the following for the two discriminators $D_X$ and $D_Y$:

$$\min_{D_X,D_Y} \mathbb{E}_{x,y} \left[ \ell(D_X(x), 1) + \ell(D_X(G(y)), 0) + \ell(D_Y(y), 1) + \ell(D_Y(F(x)), 0) \right], \quad (5)$$

where 0/1 denote fake/real, $\ell()$ is the squared error loss and $\lambda$ is a coefficient for the cycle-consistency (reconstruction) loss.
In the case where we condition on an extra image, the underlying math is exactly the same, only that this time $x$ is a channel-wise concatenation of the DepthNet-frontalized face and some extra conditioning image, such as the background of the original (pre-frontalized) image. Once the network has been trained, we can disregard all other functions and use $F$ to clean up faces which are low quality due to artifacts from warping.

In terms of architectural details the generators and discriminators used were those described in the appendix of the CycleGAN paper. In short, the generator consists of three conv-BN-relu blocks which downsample the input, followed by nine ResNet blocks (which can be interpreted as iteratively performing transformations over the downsampled representation), followed by deconv-BN-relu blocks to upsample the representation back into the original input size. For training, we use the same hyperparameters as most CycleGAN implementations which is using the Adam optimizer with learning rate $\alpha = 2 \times 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.999$. However, instead of using a batch size of 1 we use the largest possible batch size, which was 16 for a 12GB GPU.

Note that in order to produce better translations, the dataset we used for all CycleGAN experiments contain both the VGG and the CelebA datasets, which has significantly more images.

**S.4.1 Background Synthesis**

We present extra visualizations for the CycleGAN which performs background synthesis on CelebA, corresponding to Figure 4 in the main paper. These are shown in Figure S6.

![Figure S6: Background synthesis with CycleGAN. Left to right: source face; keypoints overlaid; DepthNet (DN); DN + background → frontal](image)

**S.4.2 Face Replacement and Adversarial Repair**

In Figure S7 we provide extra face swap samples on CelebA, corresponding to Figure 5 in the main paper, where a source face is warped to a target face pose using DepthNet, pasted onto the target image and then passed to a CycleGAN to adapt the face skin of the warped source face to the background and hairstyle of the target face.

**S.4.3 Extreme Pose Face Clean-up**

If the source image has an extreme pose, the texture will be missing on the occluded side of the face and the OpenGL pipeline cannot rotate the face without artifacts. Note that this shortcoming is due to lack of texture on the occluded side of the face rather than a deficiency of the transformation parameters measured by DepthNet.

We performed an experiment using CycleGAN to fix such artifacts. For this experiment we take source images from CelebA and first frontalize it by using DepthNet. Since the frontalized faces have artifacts due to stretch of texture on the occluded side of the face by the OpenGL pipeline, we train a CycleGAN that takes DepthNet frontalized faces plus the background of the original non-frontal image (as two images) in one domain and the ground truth frontal faces in the other domain. The CycleGAN learns to clean-up these artifacts. Finally, we take the GAN-repaired frontalized faces and project it to different target poses using DepthNet. In Figure S8, we visualize camera sweep for source faces in the wild that have extreme poses. The cycleGAN reasonably cleans the face artifacts.
Figure S7: Face swap experiment with CycleGAN. Left to right: source face; target face; warp to target with DepthNet; repaired result with CycleGAN. The source face is taken and warped onto the target face. The background and hairstyle is then adapted to the target face.

and then DepthNet projects it to different poses. This is just one approach to address the extreme pose occlusion artifacts. We see alternative methods for addressing this issue as promising directions for future research.

Figure S8: Re-projecting a non-frontal face (far left) from CelebA to a range of other poses define by the faces in the row above. Top row (in each pair) depicts the target faces from Multi-PIE [6]. The bottom row shows from left to right: source face, source face frontalized by DepthNet, adversarial-repaired face, the repaired source face projected to the target poses (4th to 10th columns).