Facial Reconstruction and Alignment

Ajitesh Gupta

Face Alignment Across Large Pose : A 3D Solution

Motivation

- Fitting a face model to an image
- Extracting semantic information from pixels
  - Eye corner
  - Nose tip
  - Chin center
- Essential precursor to many face analysis tasks
  - Face recognition
  - Expression recognition

Two main approaches

- Analysis by synthesis - Minimizing reconstruction error of regenerated image
- Regression - Matching key points to ground truth landmark via feature based regression
- Good algorithms developed for cases where the face is mostly frontal view (Yaw < 45 degrees)
- But what happens when we start to look at profile views of faces i.e. large pose variation ?

Issues - Modelling

- Landmark based models (those depending on feature extraction) assume all landmarks are visible.
- Medium rotation cases can be handled by adjusting the face silhouette.
- But in large pose variations, landmarks are bound to get occluded by the face itself, thus model no longer works.

Issues - Fitting

- What about different landmark model for each face configuration ?
- May work but computationally very expensive to look for each configuration
- Need to have a unified solution

Issues - Data labelling

- Manual labelling of occluded faces is very tedious.
- Cannot have automated initializations.
- “Guessing” occluded landmarks is hard.
- Current datasets are either medium variation or only have visible landmark information.

Modelling faces in using 3D information

- **3DMM** - A compressed representation of a 3D face model formed using PCA.
- Parameterized by mean face shape, their neutral face component and expression component.
- Weak perspective projection onto image plane is taken.
- Components to model the face-
  - P = [R, t, f, Aid, Aexp]

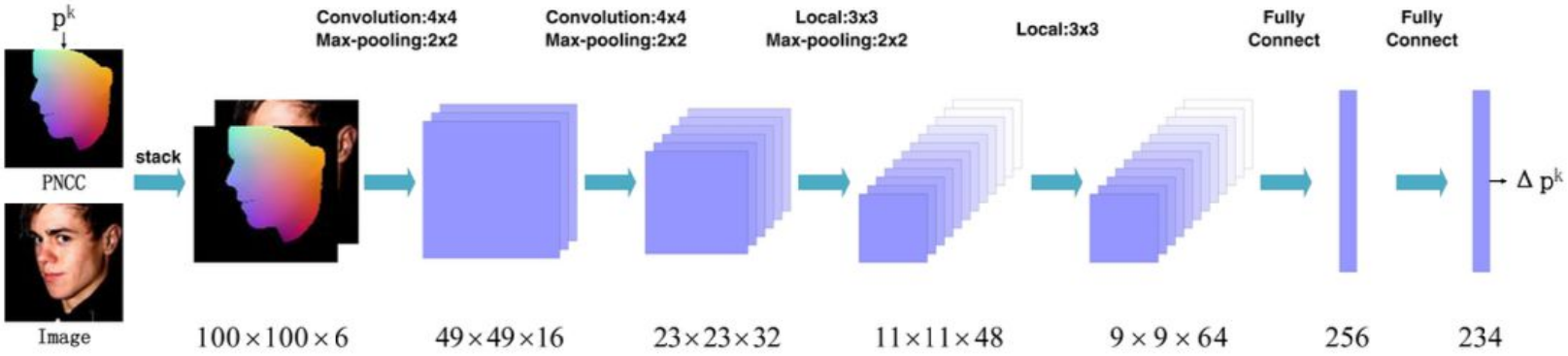
$$\mathbf{S} = \overline{\mathbf{S}} + \mathbf{A}_{id}\boldsymbol{\alpha}_{id} + \mathbf{A}_{exp}\boldsymbol{\alpha}_{exp},$$

$$V(\mathbf{p}) = f * \mathbf{Pr} * \mathbf{R} * (\overline{\mathbf{S}} + \mathbf{A}_{id}\boldsymbol{\alpha}_{id} + \mathbf{A}_{exp}\boldsymbol{\alpha}_{exp}) + \mathbf{t}_{2d},$$

Network Structure

- 9 layer convolutional neural network
- 234 dimensional output - 6 for pose, 199 for shape and 29 for expression
- Uses the image along with PNCC features to predict parameter updates
- Iterative refinement is done as old parameters are used to predict new ones

$$\Delta \mathbf{p}^k = Net^k(\mathbf{I}, \mathrm{PNCC}(\mathbf{p}^k)).$$



Projected Normalized Coordinate System

- 3D mean face is normalized to 0-1 with 3D coordinate of each point being called Normalized Coordinate Code.
- Project the 3D face with parameter p using Z buffer.
- Encode the depth at each point using RGB values.
- 3 Majors properties - Feedback, Convergence, Convolvability

$$\mathrm{NCC}_d = \frac{\overline{\mathbf{S}}_d - \min(\overline{\mathbf{S}}_d)}{\max(\overline{\mathbf{S}}_d) - \min(\overline{\mathbf{S}}_d)} \quad (d = x, y, z),$$

$$\mathrm{PNCC} = \textit{Z-Buffer}(V_{3d}(\mathbf{p}), \mathrm{NCC})$$

$$V_{3d}(\mathbf{p}) = f * \mathbf{R} * \mathbf{S} + [\mathbf{t}_{2d}, 0]^T$$

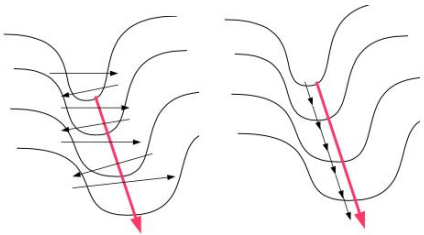
Cost Function

- Parameter distance cost
  - Simple L2 distance between expected ground truth update and calculated update.
  - Ignores the fact that different parameters might cause different magnitude of effect of errors.

$$E_{pdc} = \|\Delta \mathbf{p} - (\mathbf{p}^g - \mathbf{p}^0)\|^2.$$

- Vertex distance cost
  - Minimize distance between fitted and ground truth faces.
  - Pathological curvature as different parameters cause widely varying directions of gradient function.

$$E_{vdc} = \|V(\mathbf{p}^0 + \Delta \mathbf{p}) - V(\mathbf{p}^g)\|^2,$$



Cost Function

- Weighted Parameter Distance Cost
- Weighs each parameter by how much error it causes due to mis-prediction
- For simplicity W is considered constant while computing derivative.
- CNN starts with optimizing important parameters and then moves to smaller ones.

$$E_{wpdc} = (\Delta\mathbf{p} - (\mathbf{p}^g - \mathbf{p}^0))^T \mathbf{W} (\Delta\mathbf{p} - (\mathbf{p}^g - \mathbf{p}^0))$$

$$where \quad \mathbf{W} = diag(w_1, w_2, ..., w_n)$$

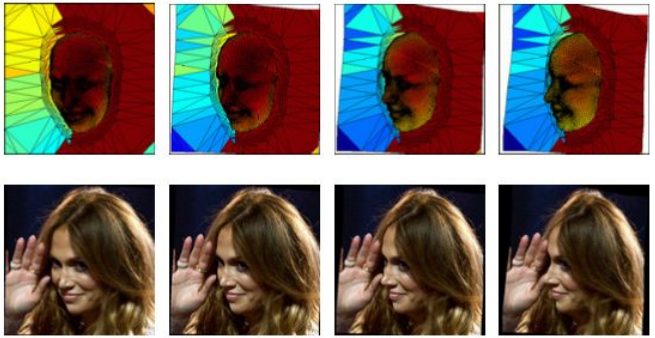
$$w_i = \|V(\mathbf{p}^d(i)) - V(\mathbf{p}^g)\| / \sum w_i$$

$$\mathbf{p}^d(i)_i = (\mathbf{p}^0 + \Delta\mathbf{p})_i$$

$$\mathbf{p}^d(i)_j = \mathbf{p}^g_j, \quad j \in \{1, \ldots, i-1, i+1, \ldots, n\},$$

Generating profile views of faces

- Good frontalisation techniques exist, so invert them to create profile faces!
- 3D Image Meshing
  - Use 3DMM with existing techniques to estimate depth with main focus of accuracy on face region
  - Solid landmark features visible so results are almost always good
  - Some outliers are manually adjusted
- 3D Image Rotation
  - Rotate depth map in 3D for large pose variations
  - Project face on only visible part to avoid artifacts.
- Good source of data augmentation



Implementation Details - Initialization Regeneration

- Network at deeper cascade might receive almost zero error due to overfitting
- Fitting error depends highly on ground truth face posture.

$$\mathrm{FP} = \mathbf{Pr} * \mathbf{R}^g * (\overline{\mathbf{S}} + \mathbf{A}_{id}\boldsymbol{\alpha}_{id}^g + \mathbf{A}_{exp}\boldsymbol{\alpha}_{exp}^g)_{landmark},$$

- For each training sample have a subset of validation samples with similar FP.
- At iteration k use the validation samples to regenerate a new initial parameter.

$$\mathbf{p}^k = \mathbf{p}^g - (\mathbf{p}_{v_i}^g - \mathbf{p}_{v_i}^k),$$

- Kind of regularization ?

Implementation Details - Landmark Refinement

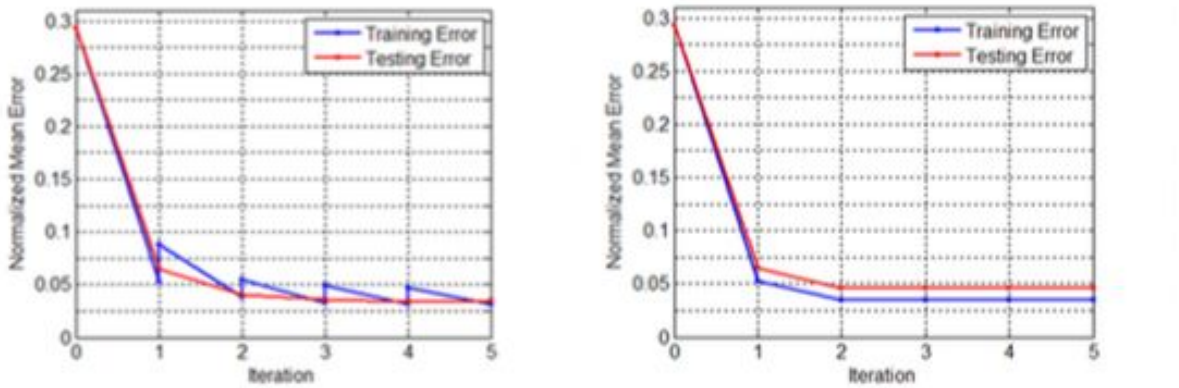
- By default we fit all vertexes.
- But if we are only interested in landmarks, errors can be reduced further
- Ex: For 2D face alignment
  - Extract HOG features at landmark location
  - Refine the location using linear regressor

Datasets Used

- 300W-LP : Combination of multiple datasets aligned with 68 landmarks. 122,450 samples after profiling and flipping. **Used to test medium pose face alignment.**
- AFLW : 21,080 in-the-wild faces with large pose variations. Up to 21 visible landmarks annotated in each image. **Used to test large pose face alignment.**
- AFLW2000-3D : Used the first 2000 AFLW images to reconstruct 3D faces and corresponding 68 landmarks. **Used to test 3D face alignment.**

Performance Analysis

Effect of Initialization regeneration

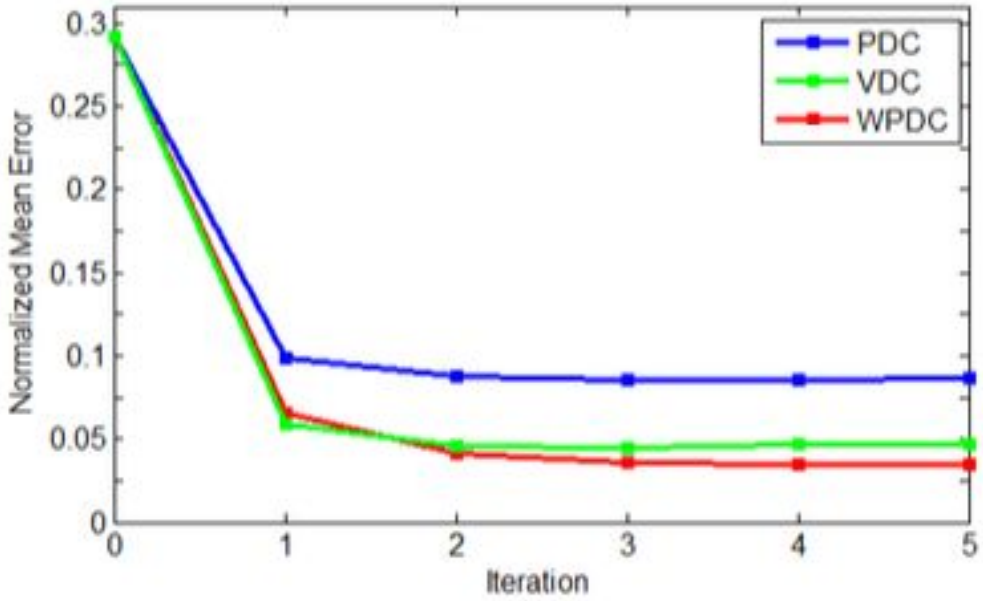


(a) (b)

Figure 7. The training and testing errors with (a) and without (b) initialization regeneration.

Performance Analysis

Effect of Cost Functions



Comparison Experiments - Large Pose

- 300W+300W-LP used for training, AFLW with 21 landmarks used for testing.
- Bounding boxes in AFLW used for initialization.
- During training, for 2D methods projected 3D landmarks are used as ground truth.
- For 3DDFA they directly regress the 3DMM parameters.
- During testing 3 subsets according to yaw angle -
  - [0.30] - 11,596 samples
  - [30-60] - 5457 samples
  - [60-90] - 4027 samples
- Normalized mean error measure - Average of visible landmark error normalized by the bounding box size.
- Standard deviation is also a good measure of pose robustness.

Results - Large Pose

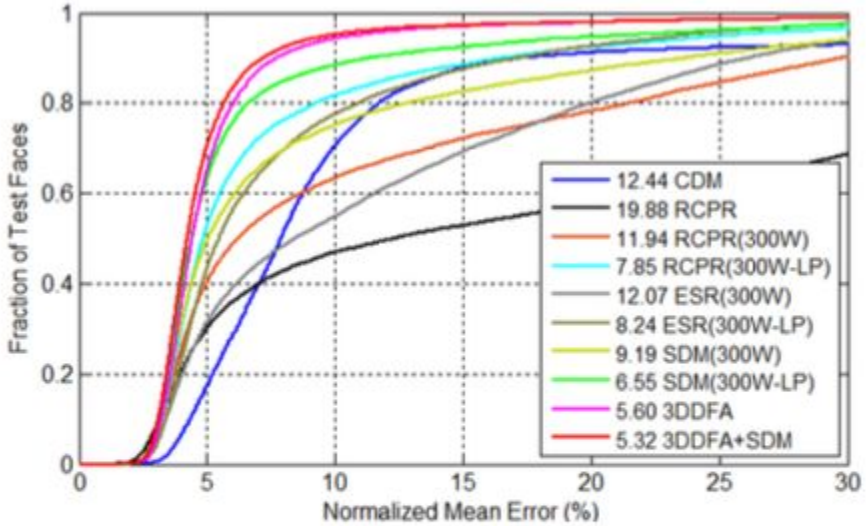


Figure 9. Comparisons of cumulative errors distribution (CED) curves on AFLW. To balance the pose distribution, we plot the CED curves with a subset of 12,081 samples whose absolute yaw angles within $[0°, 30°]$, $[30°, 60°]$ and $[60°, 90°]$ are 1/3 each.

Experiment - 3D Alignment

- Tested on AFLW2000-3D
- Degraded to an all landmark evaluation - considering all 68 landmarks instead of just 21 visible ones.
- Again bounding boxes provided for initialization.
- During testing 3 subsets according to yaw angle -
  - [0.30] - 1306 samples
  - [30-60] - 462 samples
  - [60-90] - 232 samples

Results - 3D Pose

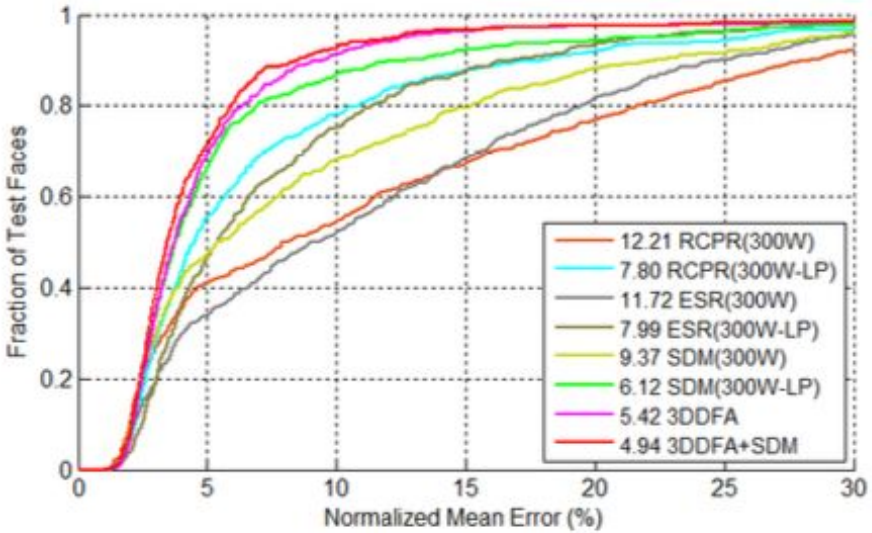


Figure 10. Comparisons of cumulative errors distribution (CED) curves on AFLW2000. To balance the pose distribution, we plot the CED curves with a subset of 696 samples whose absolute yaw angles within $[0°, 30°]$, $[30°, 60°]$ and $[60°, 90°]$ are 1/3 each.

Results - Large and 3D

	AFLW Dataset (21 pts)					AFLW2000-3D Dataset (68 pts)				
Method	[0, 30]	[30, 60]	[60, 90]	Mean	Std	[0, 30]	[30, 60]	[60, 90]	Mean	Std
CDM [49]	8.15	13.02	16.17	12.44	4.04	-	-	-	-	-
RCPR [7]	6.16	18.67	34.82	19.88	14.36	-	-	-	-	-
RCPR(300W)	5.40	9.80	20.61	11.94	7.83	4.16	9.88	22.58	12.21	9.43
RCPR(300W-LP)	5.43	6.58	11.53	7.85	3.24	4.26	5.96	13.18	7.80	4.74
ESR(300W)	5.58	10.62	20.02	12.07	7.33	4.38	10.47	20.31	11.72	8.04
ESR(300W-LP)	5.66	7.12	11.94	8.24	3.29	4.60	6.70	12.67	7.99	4.19
SDM(300W)	**4.67**	6.78	16.13	9.19	6.10	**3.56**	7.08	17.48	9.37	7.23
SDM(300W-LP)	**4.75**	5.55	9.34	6.55	2.45	3.67	4.94	9.76	6.12	3.21
3DDFA	5.00	**5.06**	**6.74**	**5.60**	**0.99**	3.78	**4.54**	**7.93**	**5.42**	**2.21**
3DDFA+SDM	**4.75**	**4.83**	**6.38**	**5.32**	**0.92**	**3.43**	**4.24**	**7.17**	**4.94**	**1.97**

Experiment - Medium pose

- LFPW, HELEN and AFW for training - 50,521 images
- Test set
  - Common - LFPW + HELEN
  - Challenging - IBUG
  - Full - Common + Challenging

Method	Common	Challenging	Full
TSPM [56]	8.22	18.33	10.20
ESR [10]	5.28	17.00	7.58
RCPR [7]	6.18	17.26	8.35
SDM [45]	5.57	15.40	7.50
LBF [32]	**4.95**	11.98	6.32
CFSS [54]	**4.73**	**9.98**	**5.76**
3DDFA	6.15	10.59	7.01
3DDFA+SDM	5.53	**9.56**	**6.31**

Main Points

- New accurate method to generate profile faces from existing face images.
- CNN with rather simple architecture able to solve image alignment in large pose images with state of the art accuracies.
- Modelling image alignment problem in 3D leads to more robust solution
- A mixed algorithm in terms of alignment and reconstruction
- New feature (PNCC) which can be very well used in a feedback network.

Learning detailed face reconstruction from single image

Motivation

- To generate detailed geometric structure of face given a single image
- Key to various applications -
  - Motion capture
  - Reenactment

Current Approaches

3DMM Methods

- Represent faces using PCA basis formed using real faces, thereby only likely solutions possible
- But loss of detail due to PCA itself
- Not good with one image

Template Based Methods

- Deform a template to match the input using shape from shading, depth similarity, appearance similarity etc.
- But they are limited in their global shape by the template used for initialization

Current Approaches

Data driven method

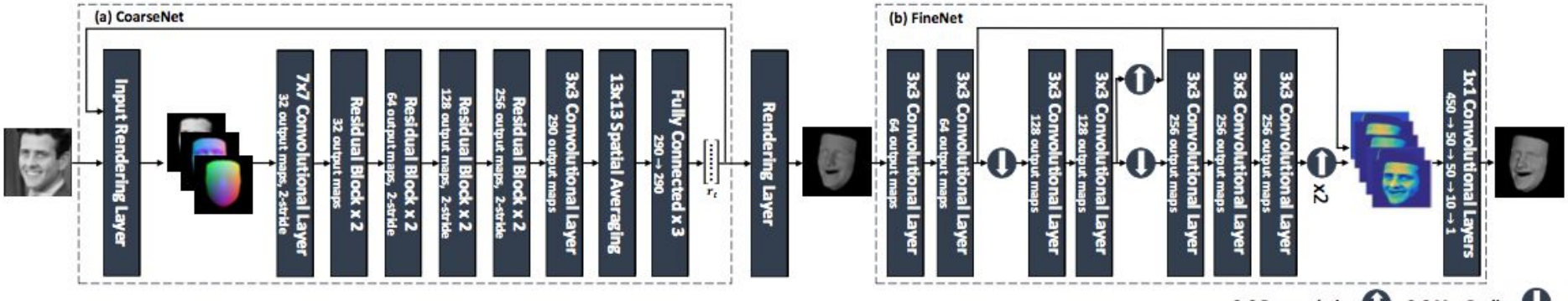
- Using some form of regression to connect between input image and reconstruction representation
  - Ex. Regression on sparse landmarks, regression on features
- Depends highly on landmarks and many methods still require pre/post processing to get good detail.

This Approach

A **3DMM** is used to define the input for **template** based refinement step where both parts are learned using a **data-driven** model.

Network Architecture

- It has a coarse to fine approach with a CNN for each connected via a novel rendering layer to convert
- CoarseNet forms a 3DMM model, rendering layer converts it to depth map and FineNet converts finetunes the depth map.
- CoarseNet based on ResNet architecture
- FineNet based on VGG-Face architecture



Step 1 - Coarse Geometry Reconstruction

Modeling the Solution Space

- As previously used, here too a 3DMM based model
- 290 parameters in total including alignment, identity and expression.

$$S = \mu_S + A_{\text{id}}\alpha_{\text{id}} + A_{\text{exp}}\alpha_{\text{exp}}.$$

CoarseNet Iterative training network

Feedback representation

- As previous paper, they also use original image and PNCC as input.
- They also add a Normal map - A depth map rendered as an RGB image.
  - It preserves local features of the image unlike PNCC
- Iterative refine 4 times for each training sample

CoarseNet Iterative training network

Acquiring Data

- No large dataset of 3D faces available
- Reconstruction from 2D would limit quality
- Synthesise artificial dataset using random geometry and pose and render them with random lighting, texture etc
- Train by starting with samples with random noise added to them using bernoulli sampling

$$r_t = \beta \cdot r_{gt} + (1 - \beta) \cdot r_{rnd}, \qquad 0 \leq \beta \leq 1,$$

CoarseNet Iterative training network

Loss Function

- Geometric Mean Squared Error for geometry parameters

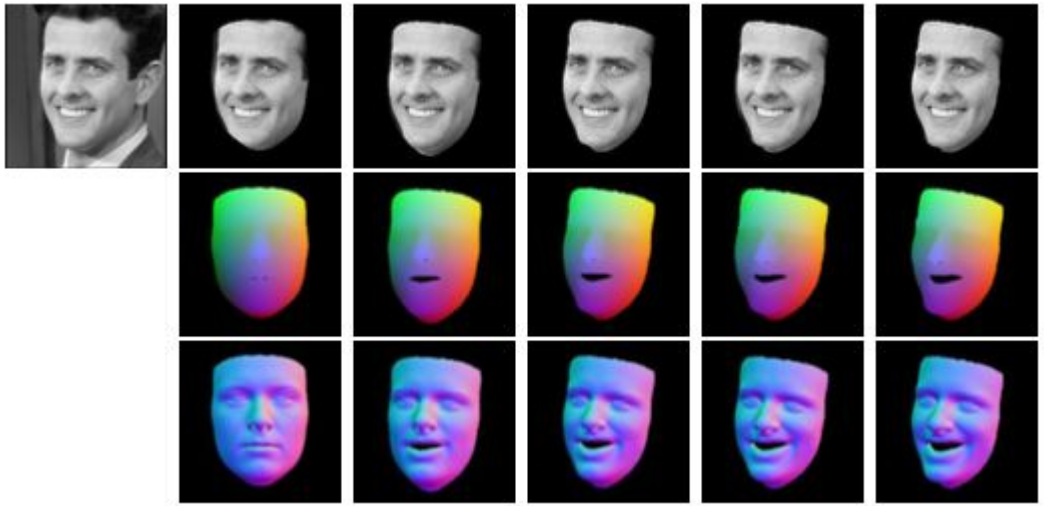
$$L(\hat{\alpha}, \alpha) = \left\| \left[A_{\text{id}} | A_{\text{exp}} \right] \hat{\alpha} - \left[A_{\text{id}} | A_{\text{exp}} \right] \alpha \right\|_2^2,$$

- Mean squared Error for pose parameters (R,t,f)

CoarseNet Iterative training network

Training

- 200x200 face image input.
- Initial parameters set to 0, corresponding to mean face.
- Input image always masked with visible vertices
- Mask gets gradually refined and so does geometry

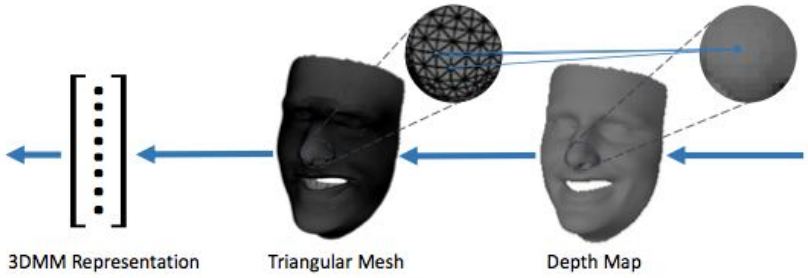


Coarse to fine

- 3DMM models cannot model the face accurately due to loss to data caused by PCA.
- It is constrained to its basis vectors.
- For detailed outputs containing wrinkles and dimples, we need to transfer the problem to unconstrained image plane.
- Thus we represent it as a depth map
- For this we use a novel rendering layer.

Rendering Layer

- First converts the geometry parameters into a 3D Mesh
- Mesh is then rendered using z-buffer renderer
- Since mesh is made up of triangles, need to interpolate z values
  - Done using Barycentric coordinates
- During backprop gradients are passed onto original triangular vertices using their weights attached to the rendered pixel.



$$\tilde{z} = \lambda_0 z_0 + \lambda_1 z_1 + \lambda_2 z_2,$$

$$\frac{dE}{dz_i} = \frac{dE}{d\tilde{z}} \frac{d\tilde{z}}{dz_i} = \frac{dE}{d\tilde{z}} \lambda_i,$$

FineNet Network

- Based on a hypercolumn architecture (concatenate activations from each layer for final prediction)
- Pooling leads to a difference in size of the input and output, hence they have to upsample.
- Instead of bilinear upsampling they use 2 strided 2x2 upconvolution to maintain quality.
- Used VGG-Face network architecture since its already trained on faces
- Fully convolutional, hence allows any image size as input

Unsupervised training

- Large dataset of detailed facial geometries with corresponding 2D images unavailable.
- Synthetic data cannot be generated using morphable models as that would miss finer details.
- Use shape from shading methods to create a loss function suitable for the problem.
- Model the loss function as the difference between input intensity image and regenerated albedo, using depth values and lighting

$$E_{sh} = \left\| \rho \left\langle \vec{l}, \vec{Y}(\hat{z}) \right\rangle - I \right\|_2^2 .$$

Unsupervised Loss - Albedo estimation

- Limit the space of possible albedos using 3DMM because this problem is constrained to faces.
- Average face texture plus differences. $\rho \approx T = \mu_T + A_T \alpha_T.$
- Assume just average face texture and recover lighting using

$$\vec{l}^* = \underset{\vec{l}}{\operatorname{argmin}} \left\| \hat{\rho} \left\langle \vec{l}, \vec{Y}(z_0) \right\rangle - I \right\|_2^2.$$

- Now that lighting is given recover proper albedo as

$$\alpha_T^* = \underset{\alpha_T}{\operatorname{argmin}} \left\| (\mu_T + A_T \alpha_T) \left\langle \vec{l}^*, \vec{Y}(z_0) \right\rangle - I \right\|_2^2.$$

Unsupervised Loss

- Use albedo and lighting to calculate E_sh
- Regularization -
  - Fidelity - Constraining the solution to not wander away far from original solution
  - Smoothness - Normal smoothness term to prevent overfitting

$$\begin{aligned} E_f &= \|\hat{z} - z_0\|_2^2, \\ E_{sm} &= \|\Delta\hat{z}\|_1, \end{aligned}$$

- Final total loss

$$L(\hat{z}, z_0, I) = \lambda_{sh} E_{sh}(\hat{z}, I) + \lambda_f E_f(\hat{z}, z_0) + \lambda_{sm} E_{sm}(\hat{z}).$$

Unsupervised Loss

- No need for annotated dataset
- Not limited by performance of algorithms used for data annotation
- Albedos and lighting terms are only needed while training, no need while testing.

End to end training

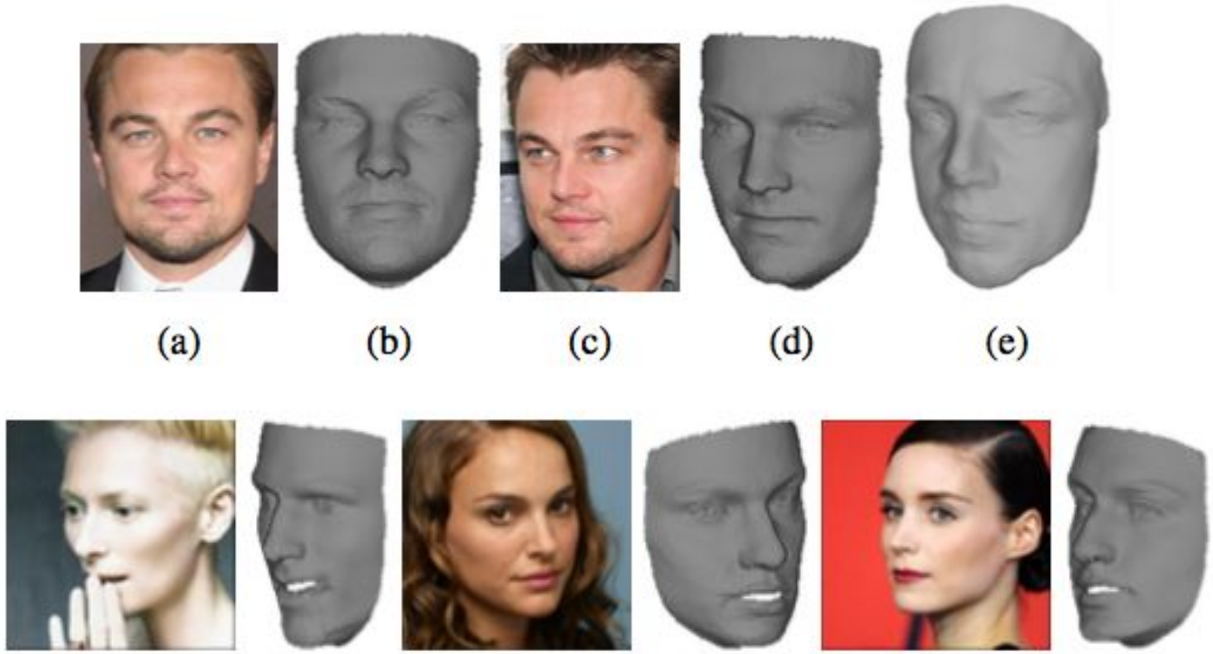
- Run input through CoarseNet for 4 iterations
- Rendering layer converts the 3DMM output to depth map
- FineNet calculates the dense depth map error and propagates it back
- CoarseNet gets fine tuned but in order to prevent it from departing from original solution completely, it is also fed a fidelity loss
- That fidelity loss is nothing but MSE between current solution and original solution.

Experiments

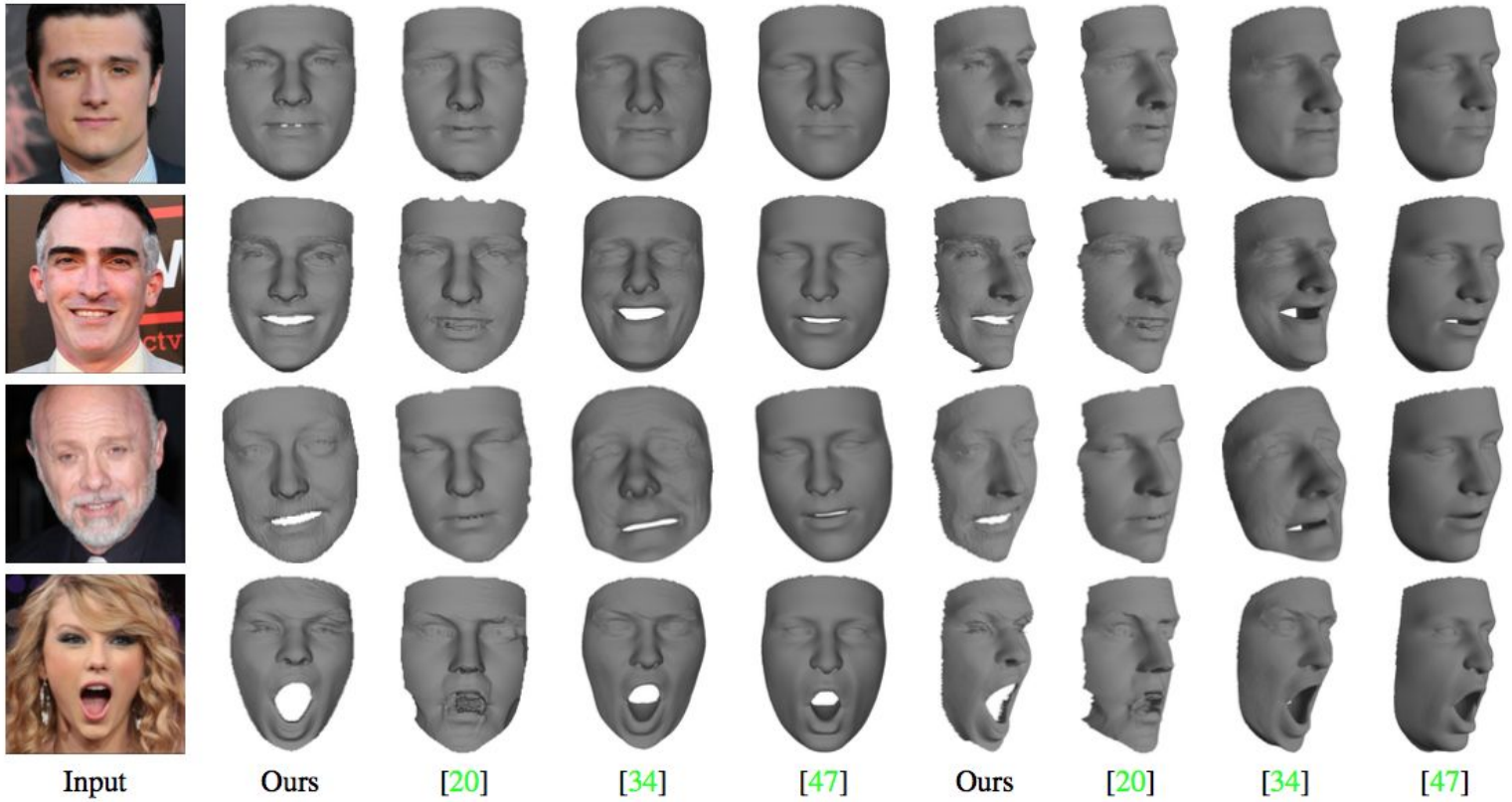
Facial Recognition Grand Challenge Dataset V2

Method	Ave. Depth Err. [mm]	90% Depth Err. [mm]
Ours	**3.22**	**6.69**
[20]	3.33	7.02
[34]	4.11	8.70
[47]	3.46	7.36

Experiments



Experiments



Main Points

- New unsupervised method to refine depth map using shape from shading ideas, so no need of labelled data
- Single image can be used to reconstruct with high accuracy what other algorithms require multiple images for
- A novel rendering layer allowing backpropagation from a depth map to a 3DMM model
- End to end trainable solution for geometric reconstruction, so no need of any kind of pre/post processing of input/output