

Scientific Discoveries and Accomplishments

CyLab Researcher Marios Savvides

3D Pose Reconstruction from Single 2D Image: Depth Estimation by a Linear Deformable Model

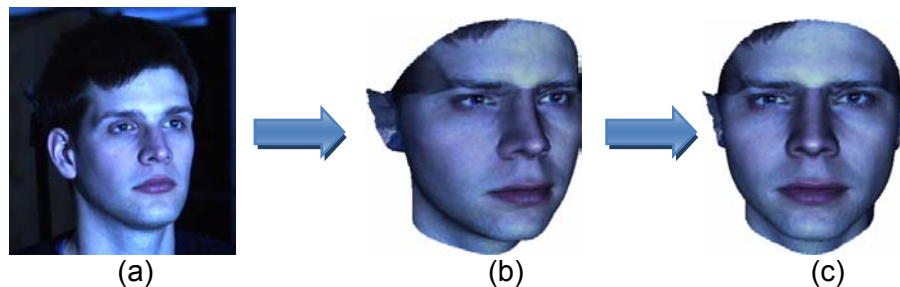


Figure 1: 3D reconstruction(b) from single 2D image (a), followed by pose correction (c)

Problem definition

In current surveillance for automatic facial identification, one of the biggest challenges is dealing with 'un-cooperative' subjects, but also even with many scenarios, pose is the biggest challenge making many facial recognition technology fall to their knees in poor performance. This project is aimed at working on a pre-processing approach for pose correction. Particularly when trying to enroll an image that is non-frontal (i.e. a person of interest from a video or a newspaper article or clip), one must be able to enroll the image in a database. This involves doing pose correction, and the only way to perform that robustly is to use a 3D model. However one does not exist since we only have a single 2D image. Thus in this approach we propose to learn 3D face by building a 3D model which we can morph to provide the same 2D projection. We can then rotate the extrapolated 3D model to frontal pose (for pose correction) to obtain the normalized 2d image as shown in Fig 1 above.

Methodology

By applying interpolation using barycentric coordinates, we get X and Y coordinates of the points inside a triangular mesh of a face. From the results of interpolation, we propose a method to infer Z coordinates (depth information) of the points and X, Y and Z coordinates of the points. We propose two linear deformable models with two different PCA subspaces learned from the inside points in the triangular mesh and all the points. Let z_{all} be a $3N_{all} \times 1$ vector defined by $z_{all} = (x_1, y_1, z_1, \dots, x_{N_{all}}, y_{N_{all}}, z_{N_{all}})^T$, where $N_{all} = 75,917$. In the same way, the 3D points inside the triangular mesh can be represented by $z_{in} = (x_1, y_1, z_1, \dots, x_{N_{in}}, y_{N_{in}}, z_{N_{in}})^T$ where $N_{in} = 36,112$. Then, two deformable models for z_{all} and z_{in} are learned from a 3D training set, where a training face consists of N_{all} 3D vertices taken from a frontal view;



$$z_{all} = W_{all}c_{all} + \mu_{all},$$

and

$$z_{in} = W_{in}c_{in} + \mu_{in},$$

where $W_{all} \in \mathbb{R}^{N_{all} \times m}$ and $W_{in} \in \mathbb{R}^{N_{in} \times m}$ are eigenvector matrices for N_{all} points and N_{in} points in a 3D training face respectively. The two eigenvector matrices take the same number of eigenvectors in descending order according to their eigenvalues.

Note that N_{in} points are part a subset of N_{all} so the coefficients c_{all} and c_{in} is ideally the same; the common coefficients c can be defined as $c = c_{all} = c_{in}$. Given a 2D facial image unseen during training, c_{in} of the image can be easily calculated with the N_{in} interpolated points z_{in} by $c_{in} = W_{in}^T z_{in}$. Consequently, we can also calculate all N_{all} 3D points z_{all} by $z_{all} = W_{all}c_{in} + \mu_{all}$. Finally, we estimate the shape of a given 2D image by inferring all the 3D vertices taken from the frontal view. (Figure on next page.)

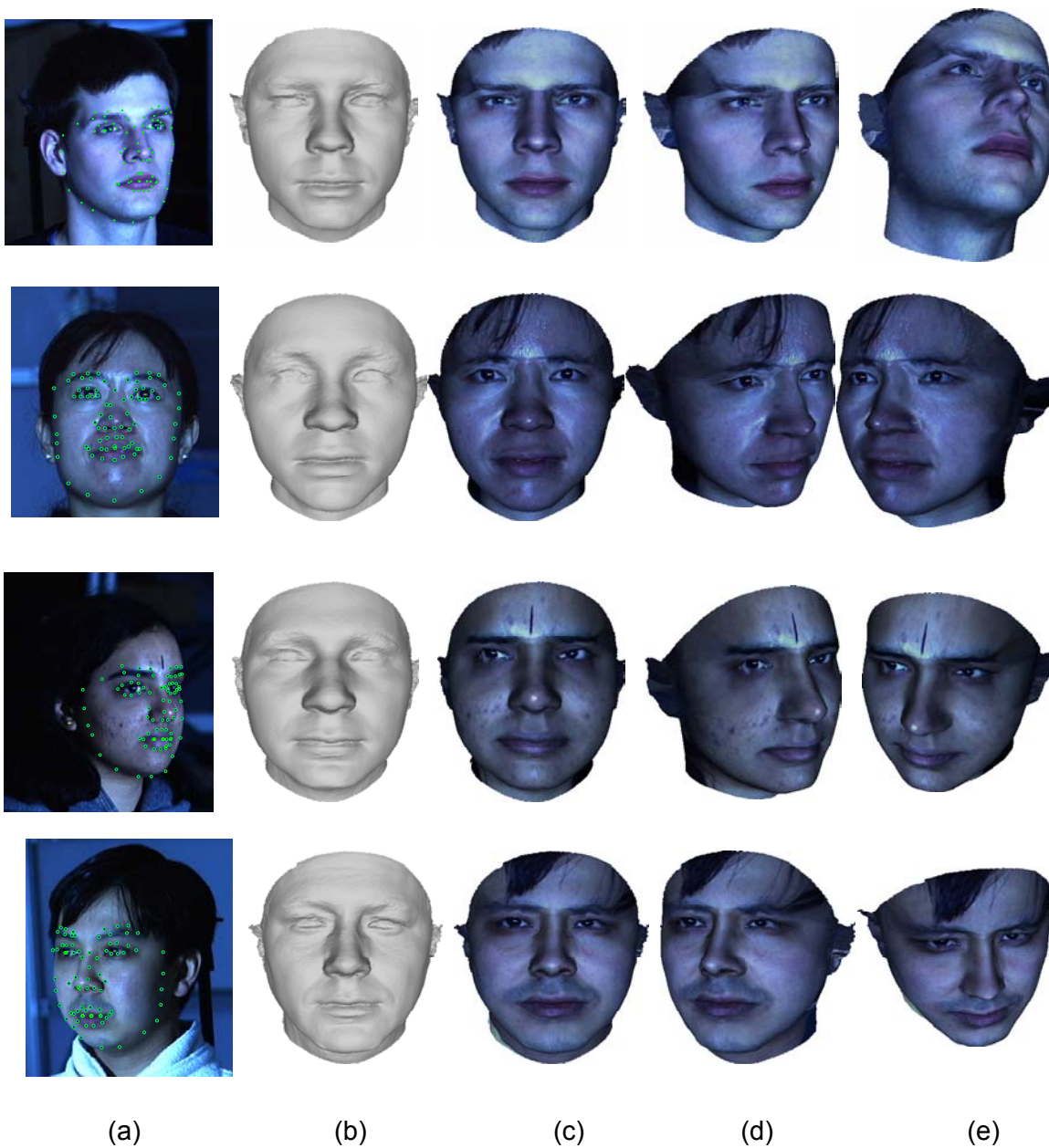


Figure 2: Example 3d reconstruction (d) from single image in (a). Shape information shown in (b), pose normalized image in (c), synthesized alternate pose in (e)

After reconstructing the shape of a 3D face, its texture (or color) is also mapped from the given 2D facial image. Texture mapping is easy for the N_{in} points since we calculate the



correspondence between a 3D vertex in the reconstructed 3D face and a 2D point in the given 2D image. Otherwise, to map the texture to the other $N_{all} - N_{in}$ vertices in the 3D reconstruction, we apply the interpolation with barycentric coordinates again, but in the opposite direction: from a 3D vertex in the 3D reconstruction to a 2D point in the 2D image. We take a triangle which is the closest to the 3D vertex and calculate the corresponding 2D point by interpolation although the vertex is not inside the triangle. One of the merits of the interpolation with barycentric coordinates is that it also can interpolate the points out of the triangular mesh; a point p is inside the triangle formed by a , b and c if and only if $0 < \alpha < 1$, $0 < \beta < 1$, and $0 < \gamma < 1$.