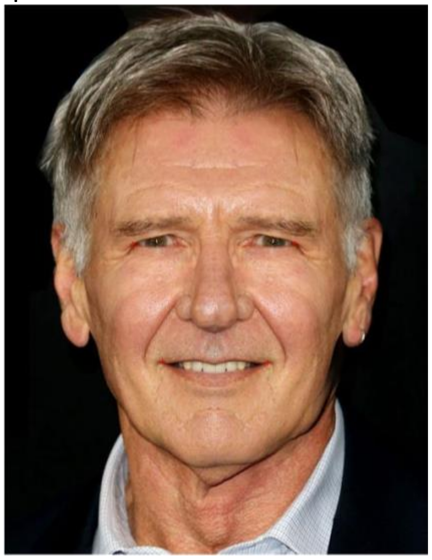
Assignment – 4
CAP 6419 – 3D Computer Vision

Abhianshu Singla

In this assignment, our goal is to create 3D models of the faces using only one picture for a given face to reconstruct it. Human face is mirror symmetric which means that for every point (or feature), there is a corresponding mirror symmetric point. Here, we are manually choosing left eye corner and its corresponding symmetric point- right eye corner and left lip corner and its symmetric feature right lip corner to determine the 3D modeling of the faces. Below are the steps to achieve this task.

1. Feature Extraction

We can choose either traditional feature extraction technique such as SIFT (Scale Invariant Feature Detection), SURF (Speed-Up Robust Features), BRISK (Binary Robust Invariant Scalable Keypoint), MSER (Maximally Stable Extremal Regions), Harris Corner Detection Algorithm and many more or we can also use deep learning technique which gives the most promising results. However, as everything comes with a price, these methods are highly time-consuming. We just need two pairs of symmetric points to get the vanishing points along the x and y-direction. For this assignment, I selected 4 points manually using ginput function in order – left eye outer corner, right eye outer corner, left lip outer corner and right lip outer corner. Also, for 3D modeling, I manually selected the points and stored it in face-name.mat.



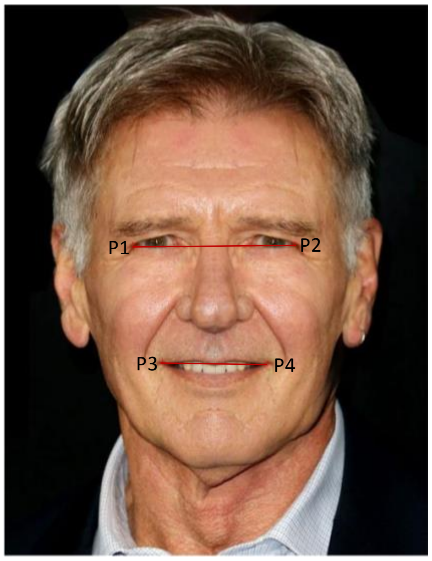
2. Compute x-vanishing point and y-vanishing point to finally compute Camera Calibration Matrix, Rotation Matrix, Rectifying Homography and Camera Projection Matrices

I used the extcaoforooshcalib.m file to obtain the two projection matrices – one for the original image and the other for the mirrored image. It also gives us the homology matrix and the rectifying homography.

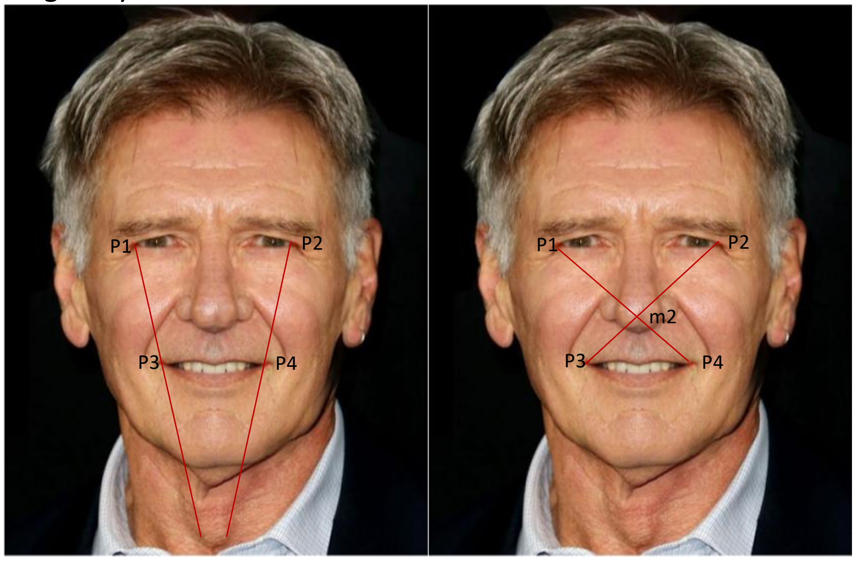
The image origin is assumed to be at the image center, so map all the four clicked points according to origin.

```
h=round(size(im,1)/2);
w=round(size(im,2)/2);
X=X-w; %x-coordinate of clicked points
Y=Y-h; %y-coordinate of clicked points
```

Now, find the x-vanishing point and y-vanishing point using theorem – Point at the intersection of two lines is obtained by the cross-product of two lines and also the line that goes through two points is obtained by the cross-product of the two points. It is because points and line are dual to each other in 2-D plane.



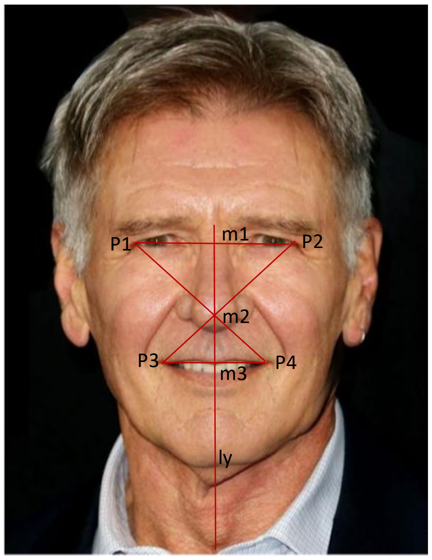
- To compute the x-vanishing point, do $(P1 \times P2) \times (P3 \times P4)$
- To compute the y-vanishing point which is along the y-axis, we first compute the vanishing points along the y-axis.



Point m is obtained by $(P1 \times P3) \times (P2 \times P4)$ and Point m2 is obtained by $(P1 \times P4) \times (P2 \times P3)$ as shown in the above two figures.
Then, the line along the y-axis is computed: $ly = m \times m2$

Also, compute $m1 = (P1 \times P2) \times ly$ and $m3 = (P3 \times P4) \times ly$

After obtaining these three points m1, m2 and m3, we can either use the golden ratio assumption about faces or equality of ratios from similar triangles in the world, which translates to equality of the cross-ratios in the image. I tried both of these and the results are quite indistinguishable to the human eye. All the results shown below are computed using Golden Ratio: $cr = \frac{1+sqrt(5)}{2}$



Now, compute the y-vanishing point by solving:

$eq1 = \frac{\frac{m1(2)-m2(2)}{m2(2)-m3(2)}}{\frac{m1(2)-vyy}{m3(2)-vyy}} - cr$ in vyy

$eq2 = ly' * [vyx; vyy; 1]$ in vyx (l*p = 0 if point p lies on line l)

$vy = [vyx; vyy; 1]$

Now, the points vx and vy are the vanishing points along the x and y-direction and these are orthogonal points. The calibration matrix is computed using Choleski Function which states that "A positive-definite symmetric matrix A may be uniquely decomposed as A = KK^T where K is an upper-triangular real matrix with positive diagonal entries."

Also, the first and second column of the camera calibration matrix is computed using the x- and y-vanishing points and then solving it using singular value decomposition.

Rectifying Homography is computed using four-point coordinates and the golden cross-ratio.

Finally, the two projection matrices are computed based on the assumption that m1 is a world-point coordinate.

3. Linear Triangulation

Compute the homogeneous coordinates for each feature point in the face-name.mat file by adding ones to the last column. Then, compute rectified feature points1 and rectified feature points2 in the mirrored image using the rectified homography.

```
xy_ = [xy; ones(1,n_columns)];
pnts1 = hnormalise(Hr*xy_);

z = [-1 0 0; 0 1 0; 0 0 1];
pnts2= hnormalise(Hr*z*hnormalise(H*pnts1));
```

Now, apply the linear triangular method to all the points1 and points2 using Projection Matrix P1 and P2 respectively. It is analogous to Direct Linear Transform which we used in the last assignment. Initially, the homogeneous factor is eliminated using cross-product to get three equations for each image point. Two equations are linearly independent, so we can ignore the third equation. `x*P(3,:)-P(1,:)=0 and y*P(3,:)-P1(2,:)=0.`

```
function [A] = triangulation(P1, P2,x1,x2)
A = [    x1(1) * P1(3,:) - P1(1,:);
         x1(2) * P1(3,:) - P1(2,:);
         x2(1) * P2(3,:) - P2(1,:);
         x2(2) * P2(3,:) - P2(2,:)];
end
```

After linear triangulation method, solve matrix A using SVD and take the values from the last column of V matrix and append it to p3d points.

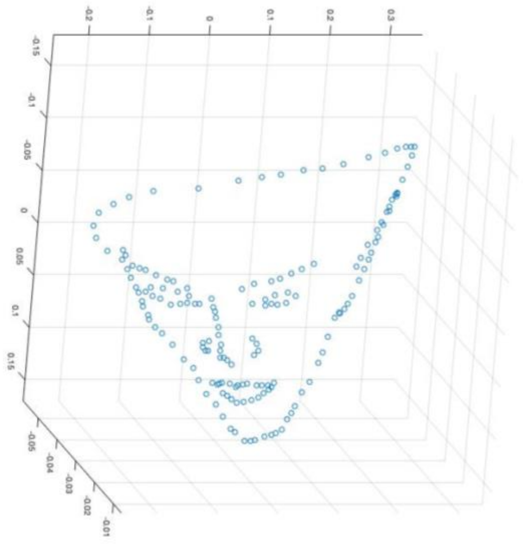
```
[U,D,V] = svd(A);
 x_w = V(:,end);
 x_w = x_w./x_w(4);
 X_wn = [x_w(1); x_w(2); x_w(3)];
 p3d = [p3d X_wn]
```

Finally, plot these p3d points using scatter function.

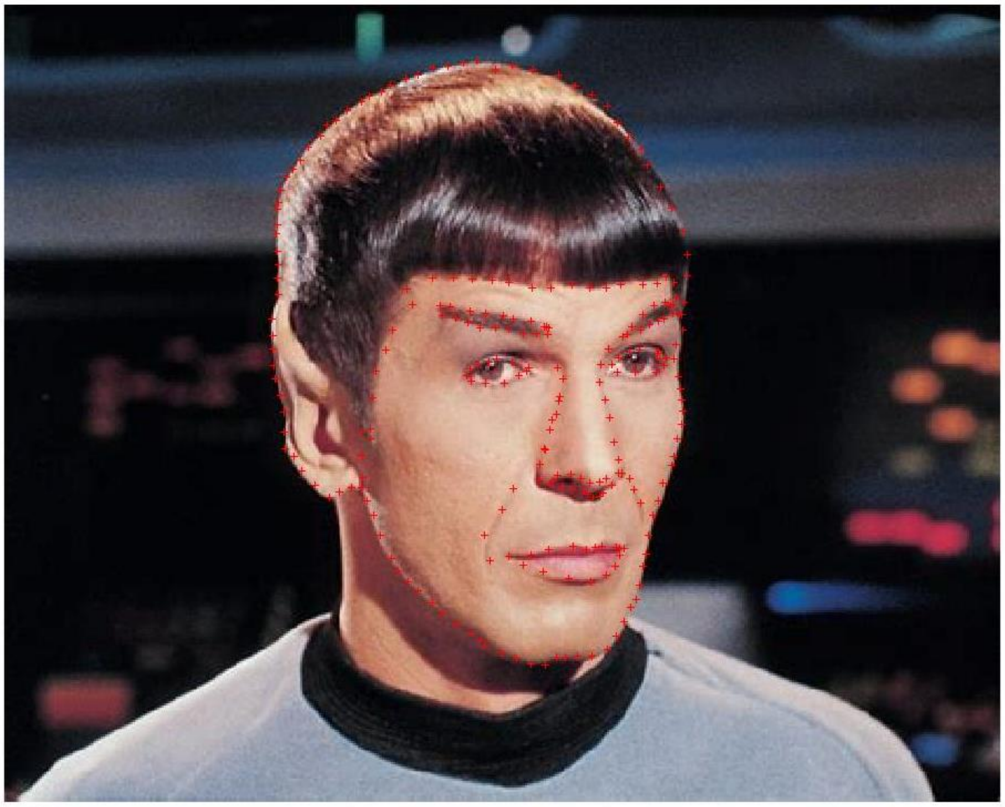
```
figure;
scatter3(p3d(1,:), p3d(2,:), p3d(3,:));
axis vis3d;
```

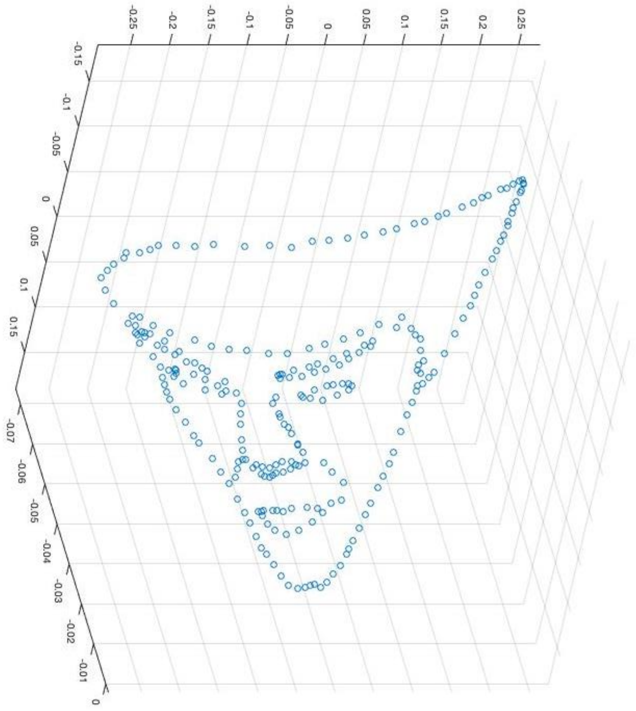
Results:

With the given features:

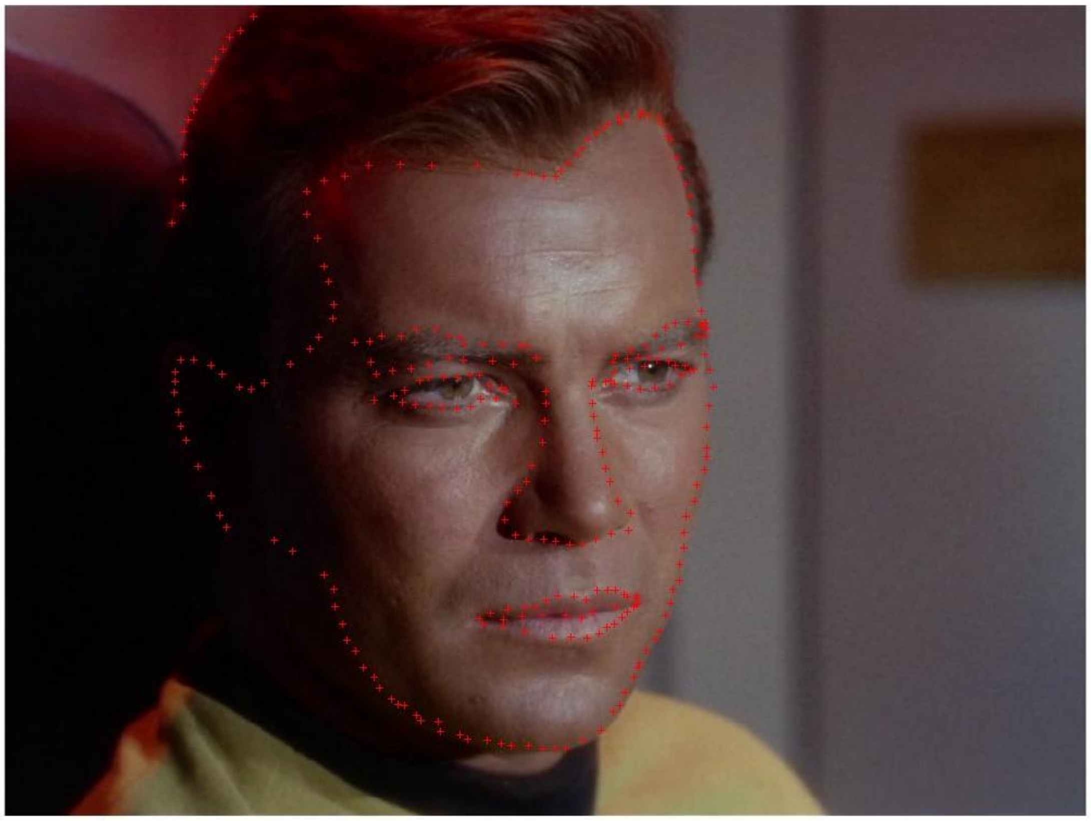


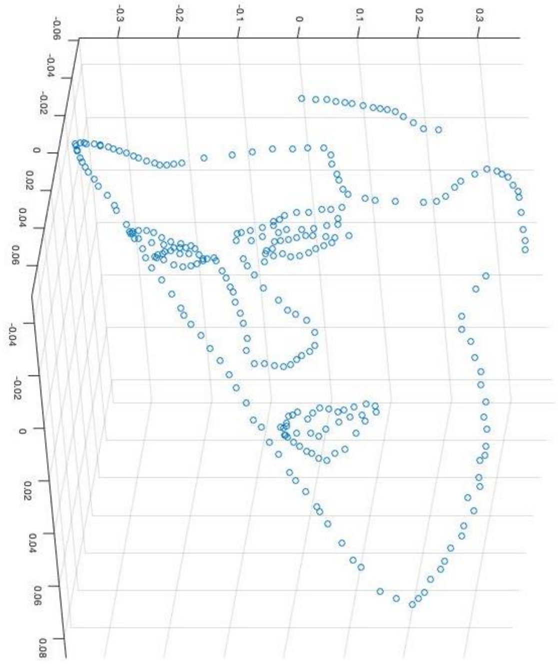
With slghtly different features



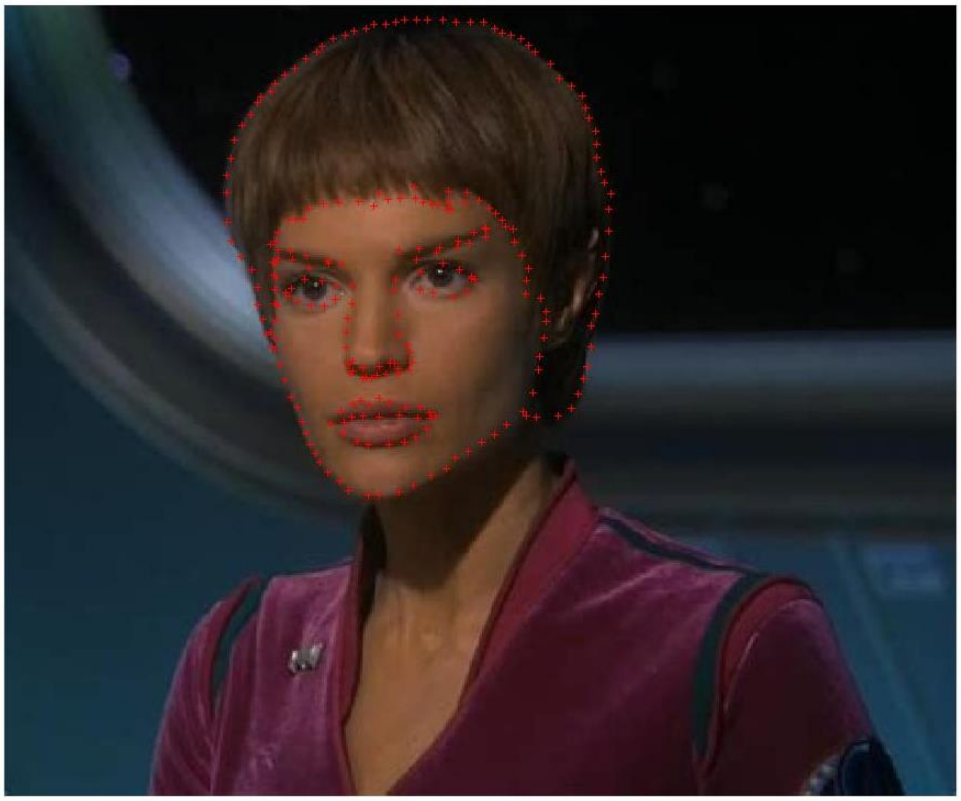


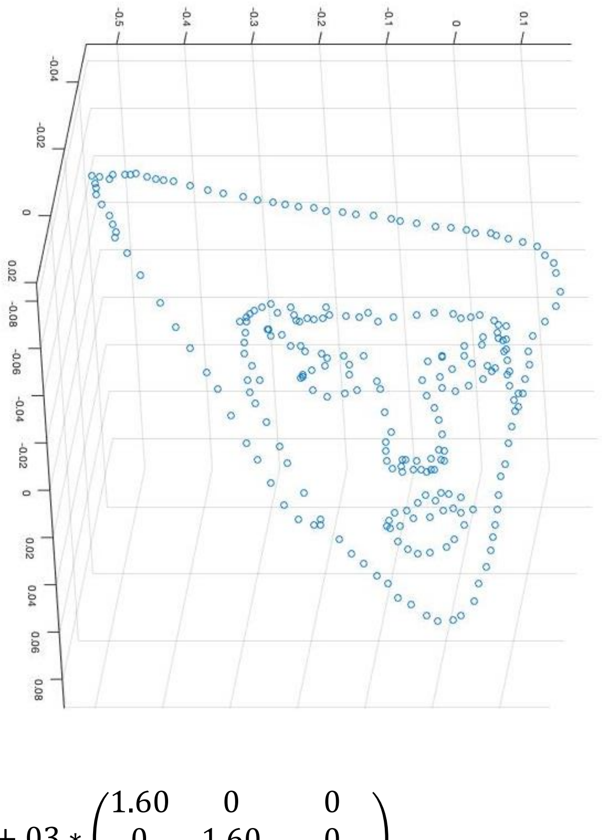
Calibration Matrix: $1.0\mathrm{e}+03 * \begin{pmatrix} 1.14 & 0 & 0 \\ 0 & 1.14 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$



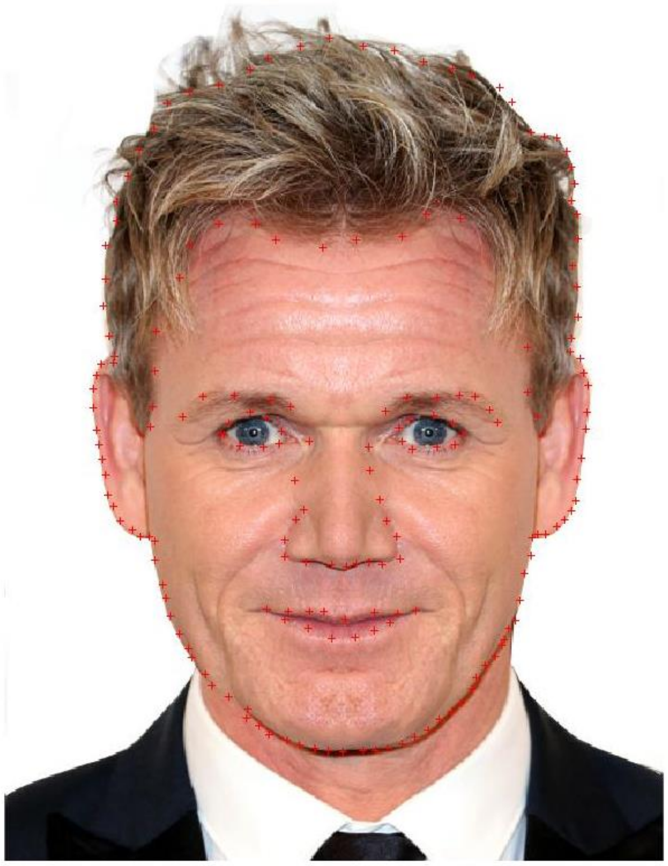


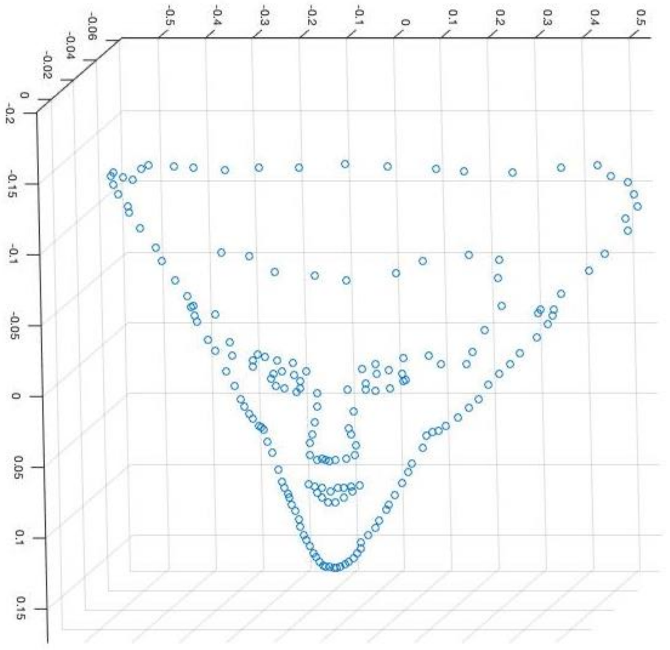
Calibration Matrix: $1.0e+03 * \begin{pmatrix} 4.46 & 0 & 0 \\ 0 & 4.46 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$



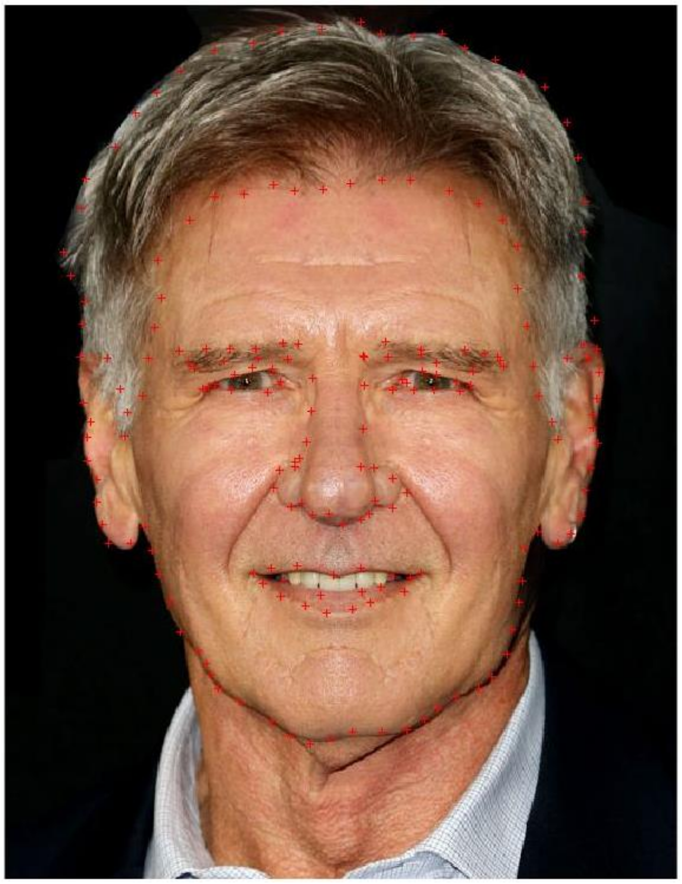


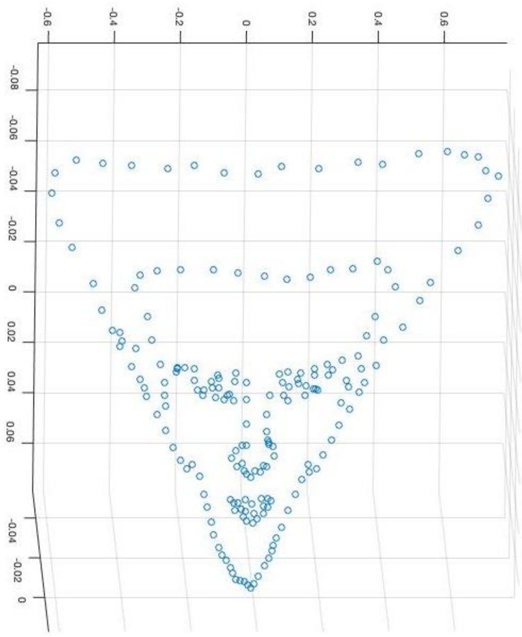
Calibration Matrix: $1.0\mathrm{e}+03 * \begin{pmatrix} 1.60 & 0 & 0 \\ 0 & 1.60 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$



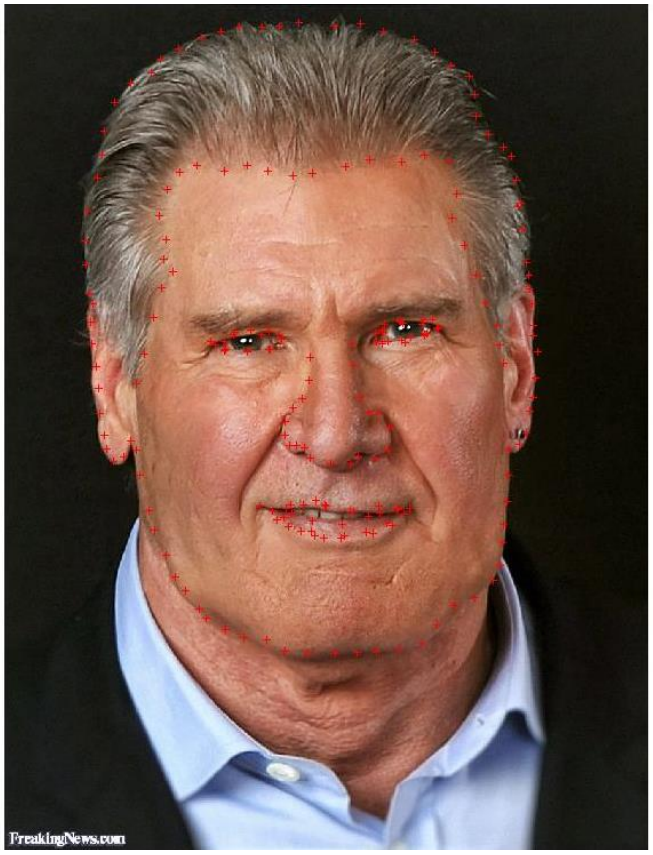


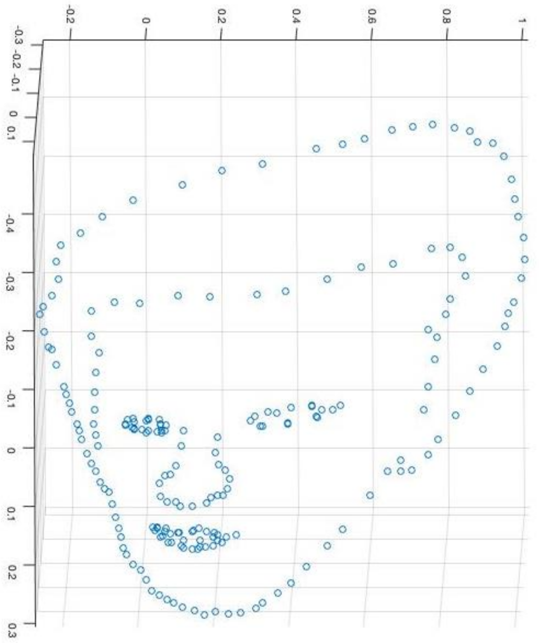
Calibration Matrix: $1.0e+03 * \begin{pmatrix} 1.89 & 0 & 0 \\ 0 & 1.89 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$



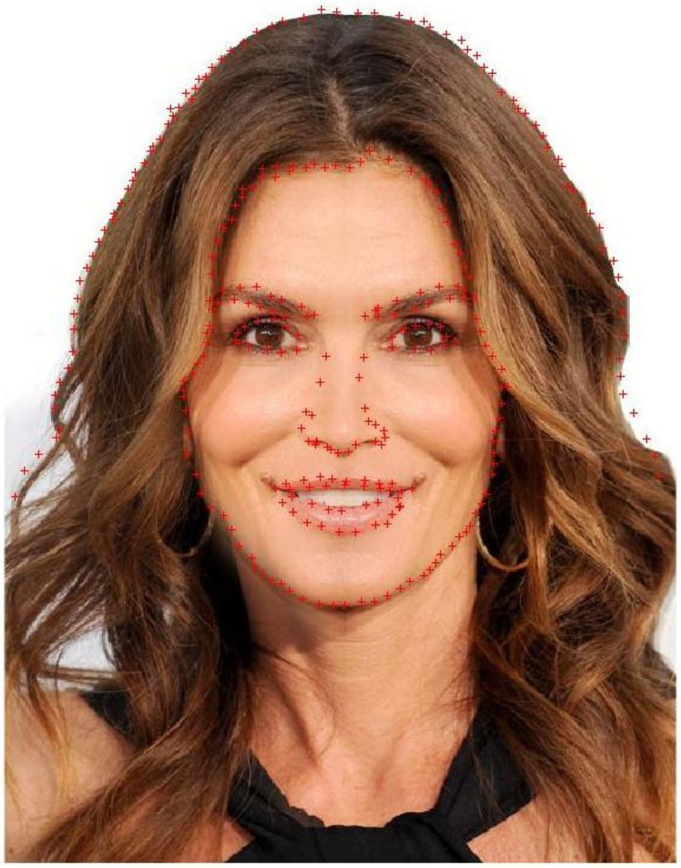


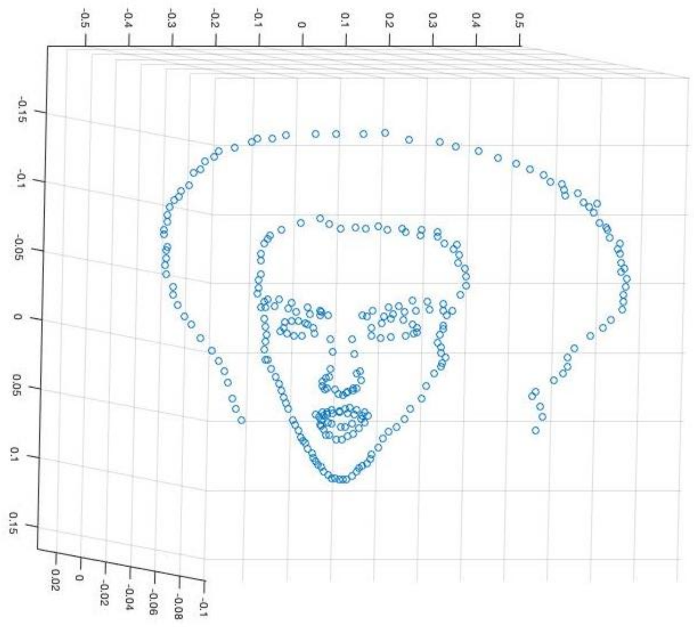
Calibration Matrix: $1.0e + 03 * \begin{pmatrix} 3.23 & 0 & 0 \\ 0 & 3.23 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$





Calibration Matrix: $1.0\mathrm{e}+03*\begin{pmatrix}1.27 & 0 & 0\\ 0 & 1.27 & 0\\ 0 & 0 & 0.001\end{pmatrix}$





Calibration Matrix: $1.0e + 03 * \begin{pmatrix} 1.27 & 0 & 0 \\ 0 & 1.27 & 0 \\ 0 & 0 & 0.001 \end{pmatrix}$

Discussion:

- The rightful selection of the feature points for 3D modeling of faces is a crucial step. I used the manual method because with SURF or SIFT feature extraction technique, there are some features which are not that relevant, however, one can utilize his/her human knowledge in manually selecting the appropriate corresponding features. But, a drawback of human error (not correct at pixel level) is present in the manual selection process.
- To compute the y-vanishing point, we can use either golden ratio assumption about faces or equality of ratios from similar triangles in the world, which translates to equality of the cross-ratios in the image. I tried both of these and the results are quite indistinguishable to the human eye.
- In the result images, the 3D modeling of the faces accurately depicts the depth of the facial features such as nose, eyes and lips. Also, there is huge distance seen between the forehead and back-head because of the depth effect.

References:

1. Xiaochun Cao and Hassan Foroosh, Camera calibration using symmetric objects, IEEE Transactions on Image Processing, vol. 15, issue 11, pages 3614-3619, 2006.
2. Richard Hartley and Andrew Zisserman, Multiple View Geometry in Computer Vision