**CSE 260 - Computer Graphics Project Final Report**

# 3D Face Reconstruction using iPhone's TrueDepth Sensor

Prepared By

Kesav Ravichandran (keravich)
Utkarsh Gupta (utgupta)
Yunquian Cheng (ychen827)

# Goals

The project has the following goals:

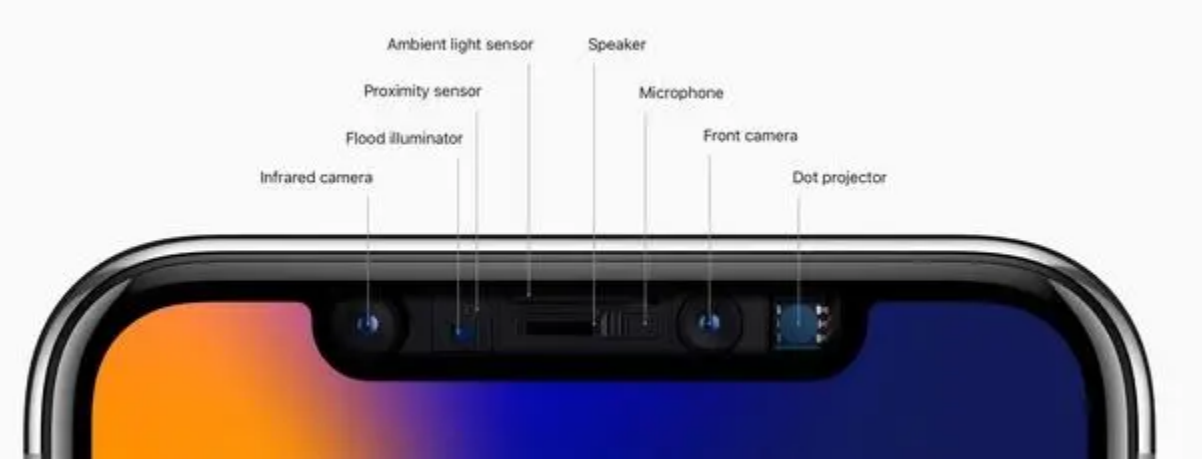
1. Develop an iOS application that captures a 3D point cloud of the user's face using the iPhone’s front-facing True Depth camera.
2. Fit the FLAME model to the data to create a ‘complete’ face model with only one view.
3. An experiment was conducted to determine the minimum amount of light required for Machine Learning models like DECA and MICA to produce results that are comparable to the depth capture results of an iPhone.
4. Compare results with baseline models that use RGB images from the NoW challenge (MICA and DECA) mainly in low lighting/pitch-dark conditions.

# Raw Depth Capture

## Working of TrueDepth Camera

A dot projector projects structured Infrared light patterns onto the object infront of it.

An infrared lens receives reflected dot projection pattern, calculates the deformation of the pattern to get the depth of each pixel on the object.

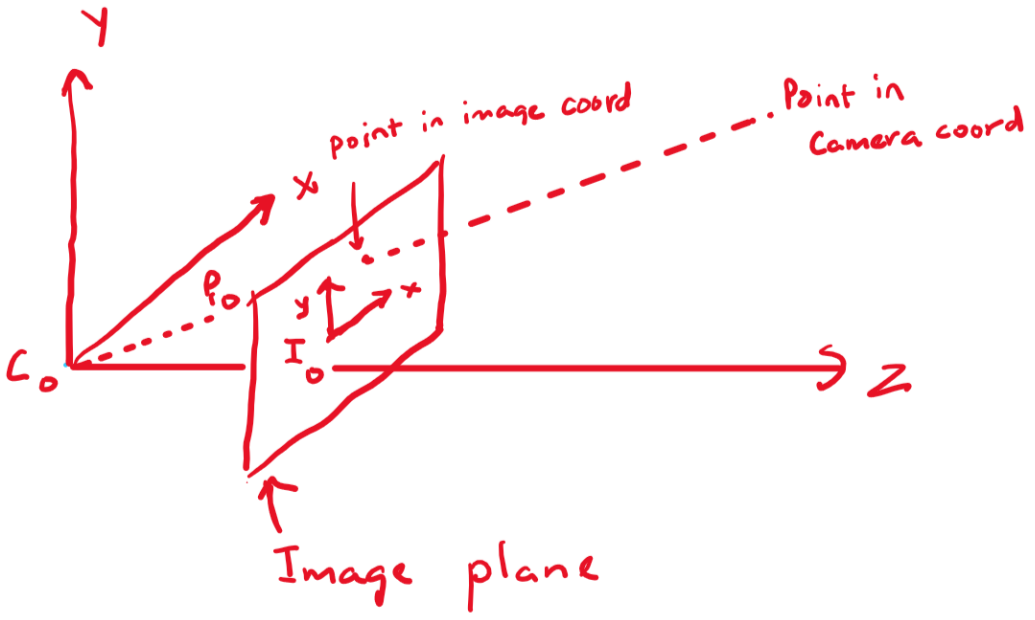


# Coordinate Systems

**Pixel Coordinate System**
Top left of the image pixel is taken as origin. The x and y axis are parallel to the height and width of the image. This is what is shown on our screens.

**Image Coordinate System**
Center of the camera sensor is taken as the origin on the image plane. x and y axis are parallel to the sides of the image plane.

**Camera Coordinate System**
Taking the optical center of the camera as the origin. X and y axis are similar to image coordinate and the z axis is the optical axis of the camera.




$C_o$ - Camera Origin.

$I_o$ - Image / Camera sensor Origin.

$P_o$ - Pixel Origin (0,0) Top left.

## Conversion between coordinate systems

For our case, we need to convert points from Pixel space (x, y) to Camera space (X, Y, Z). In other words, we convert our pixel coordinates from 2D to 3D using camera calibration matrix and depth information.

**Camera Calibration Matrix (From AVFoundation API)**

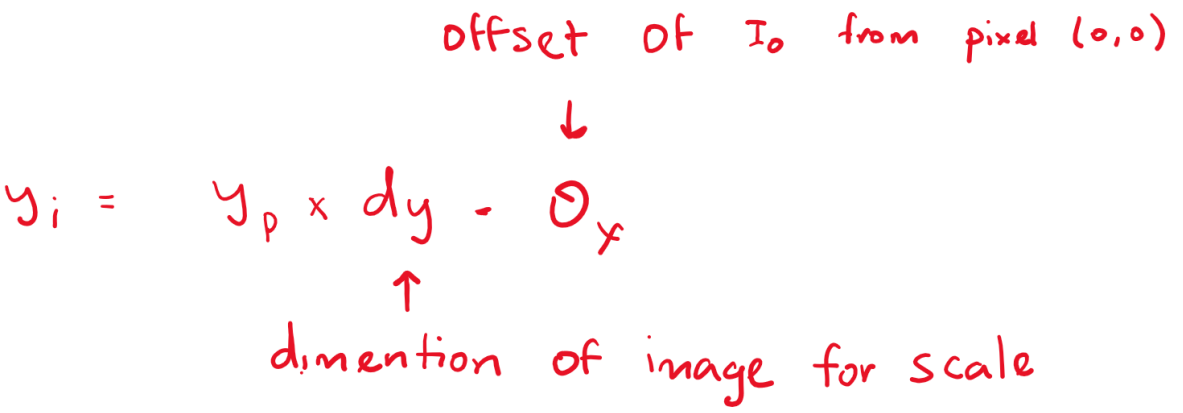
$$\begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

$o_x$, $o_y$ → offset of center in image coordinate ($I_o$) from center in pixel coordinate

$f_x$, $f_y$ → Pixel focal length of camera.

**Pixel Coordinates ($x_p$, $y_p$) -> Image Coordinates ($x_i$, $y_i$)**

We are scaling the x and y coordinates in pixel coordinate system and subtracting the offset (center of image coordinate in pixel coordinate) to get the x and y coordinates in image coordinate system..



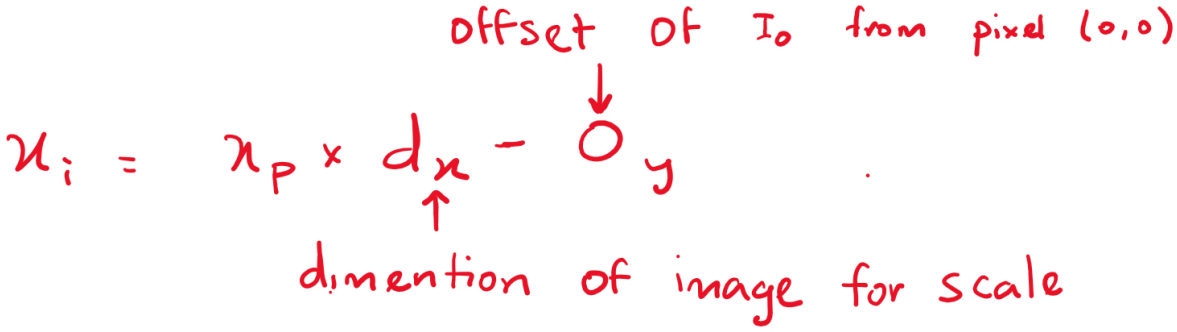
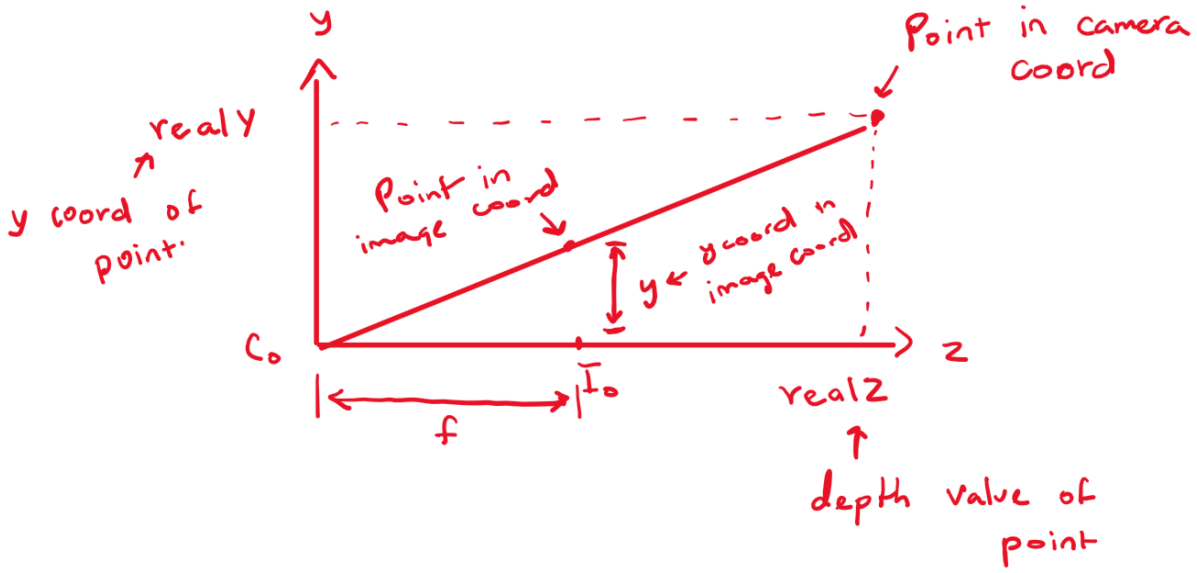



**Image Coordinates ($x_i$, $y_i$) -> Camera Coordinates (realX, realY, realZ)**




We use the triangle symmetry property to get the X and Y in camera coordinate system with respect to the obtained depth.

$$\frac{y_i}{f} = \frac{realY}{realZ} \quad \therefore \quad realY = \frac{y_i}{f} \times realZ$$

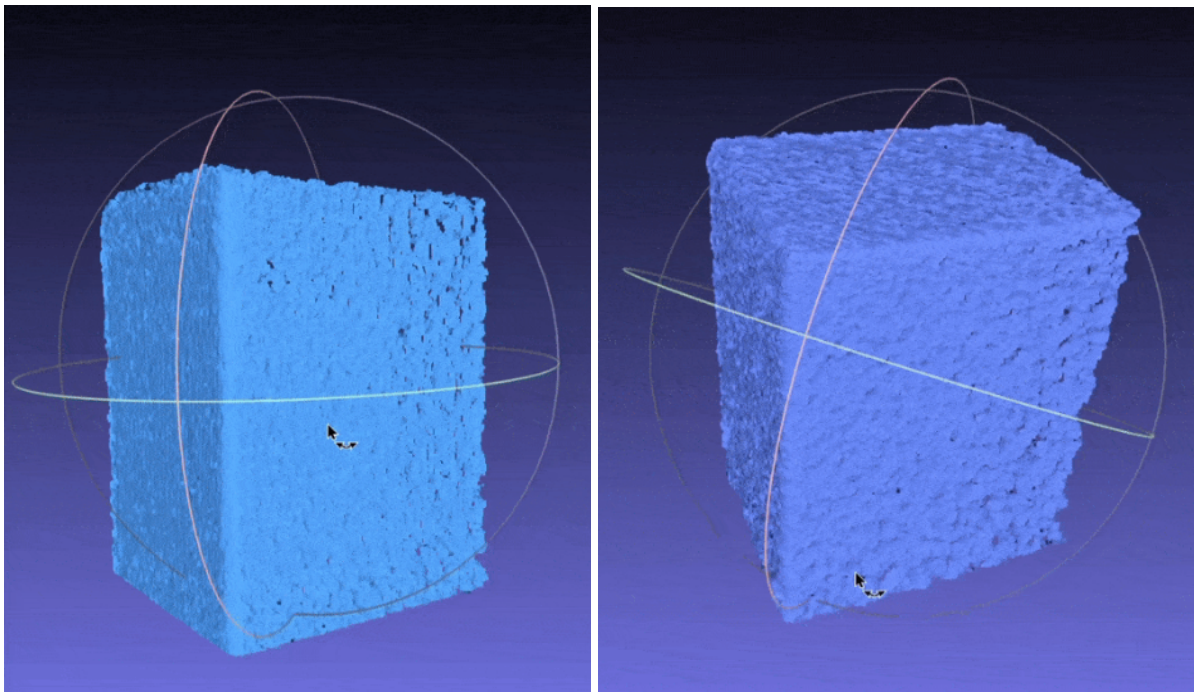
Similarly,

$$\frac{x_i}{f} = \frac{realX}{realZ} \quad \therefore \quad realX = \frac{x_i}{f} \times realZ$$

# Point Cloud Captures

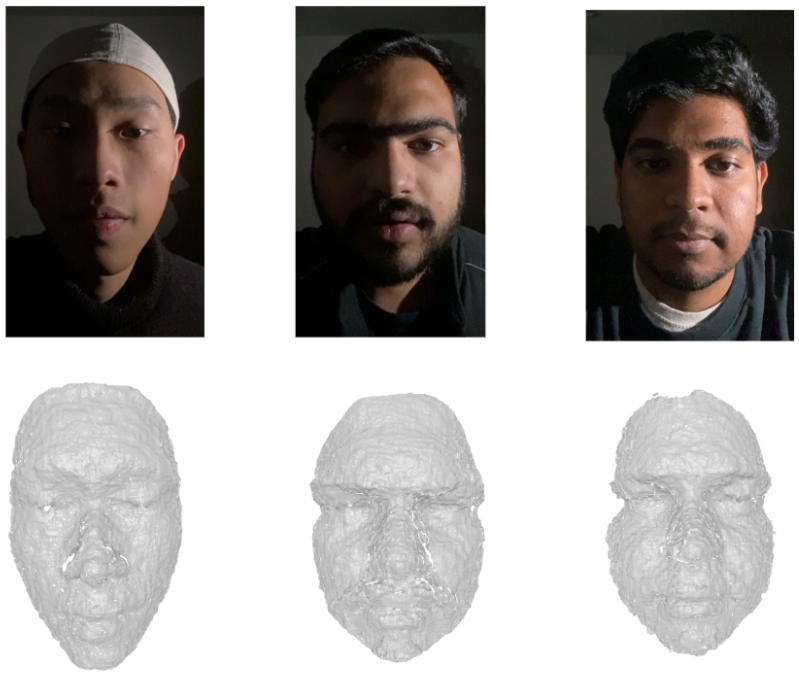
## Box Trial Capture

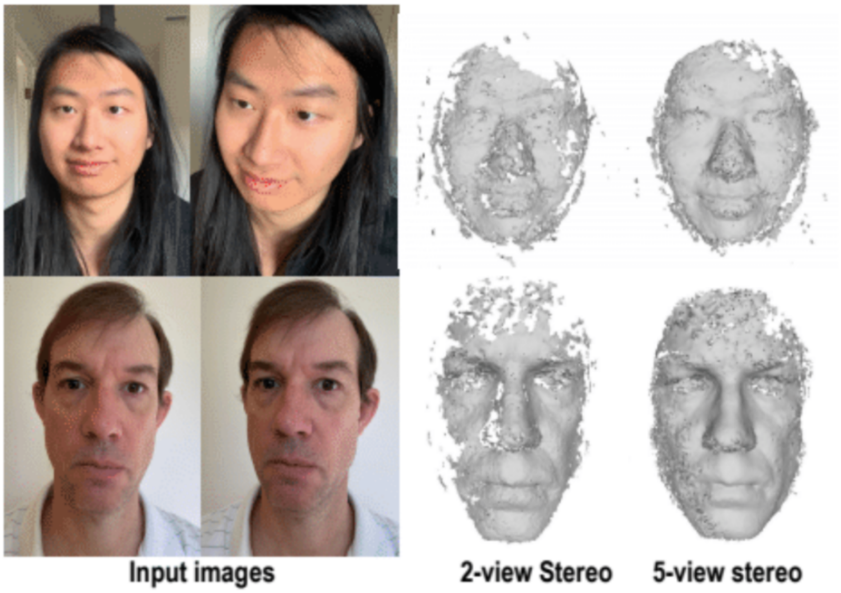
**(To check for 90 degree edge and 3 edge vertex angles)**



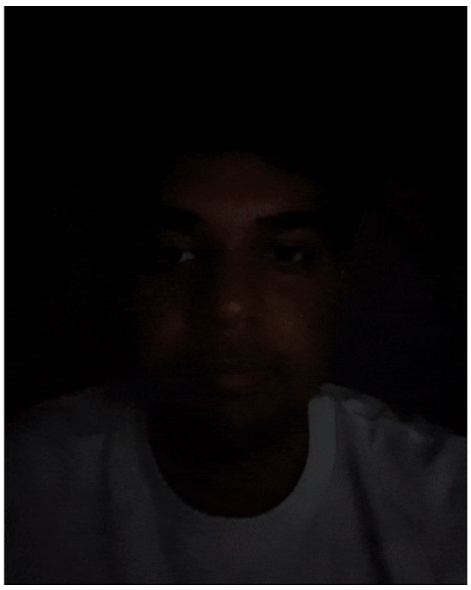
# Comparison of TrueDepth generated mesh to Stereo generated mesh

Point cloud data is generated from single frame

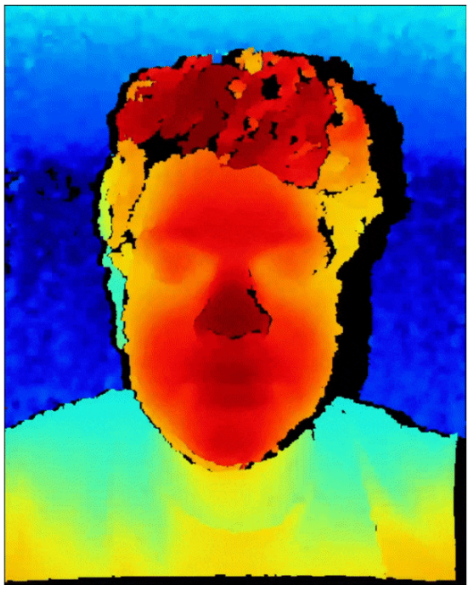




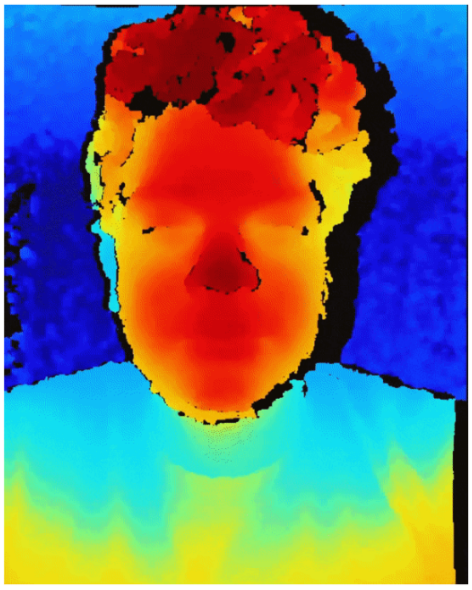
# Point Cloud Capture in Dark and Room Lighting



->

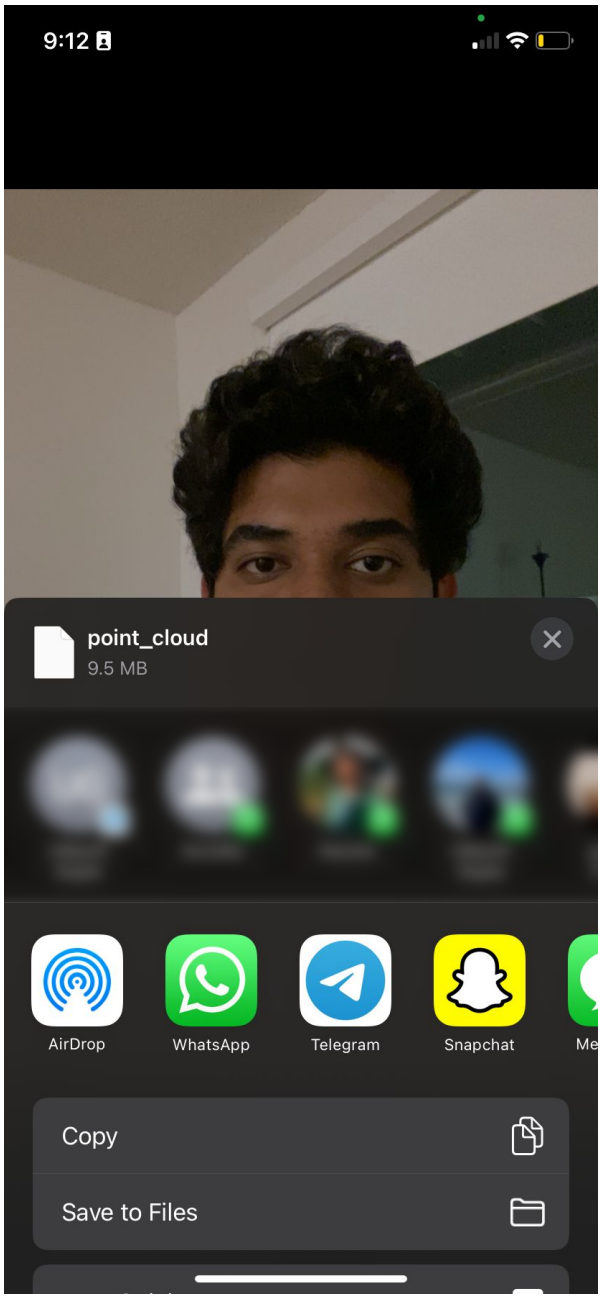


->



# App Screenshots


Preview and Capture Scene



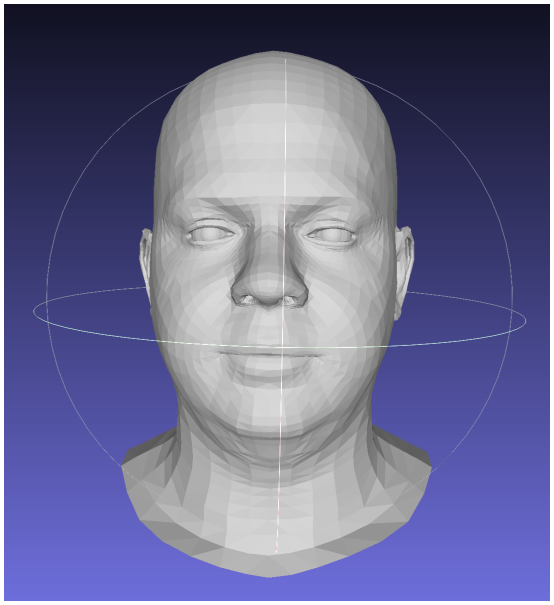

Share Menu for Point Cloud

# FLAME

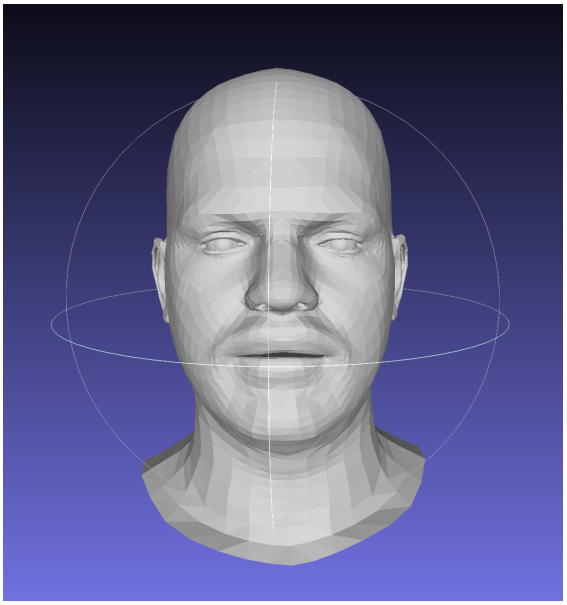
We set up the FLAME model by following the instructions present in its GitHub repository. We learned the following ideas while setting up the flame:

1. We were able to set up the PyTorch as well as the ChumPy version of the FLAME.
2. We learned about the scaling parameters that are used in the fit_scan.py file.

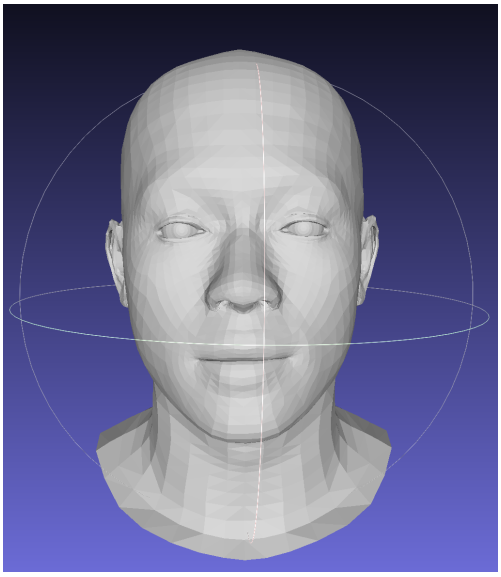
## Results



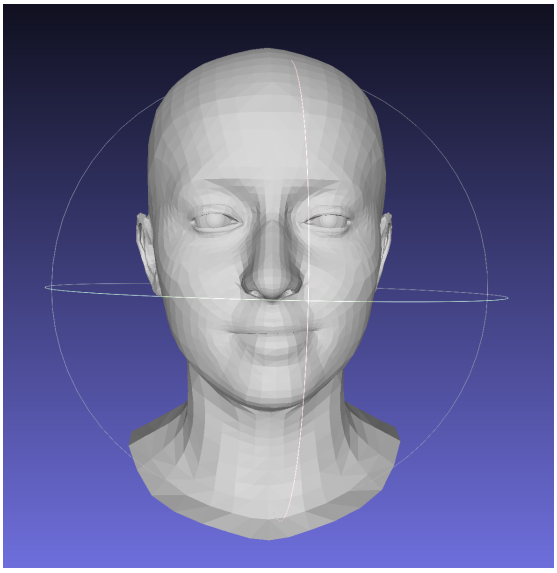
**Kesav Ravichandran**



**Utkarsh Gupta**



**Yunquian Cheng**



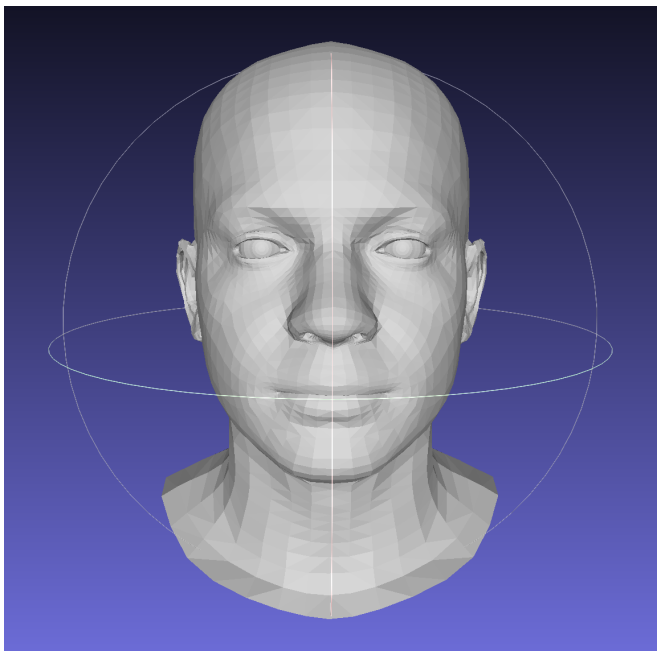
**Harshini Venkataraman**

# MICA

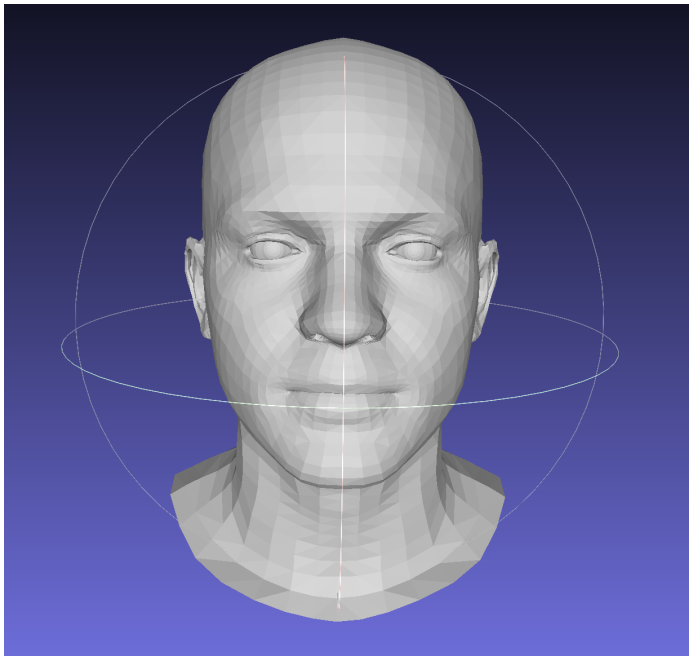
We set up the MICA model by following the instructions present in its GitHub repository. We learned the following ideas while setting up the flame:

1. Setting up the virtual environment for MICA was the easiest part. We only had to use the environment.yml file to download the required packages and set the environment.
2. MICA directly outputs the seven key points that are required by the NoW evaluation pipeline for face alignment. Therefore, we did not have to make any changes to the code of MICA.

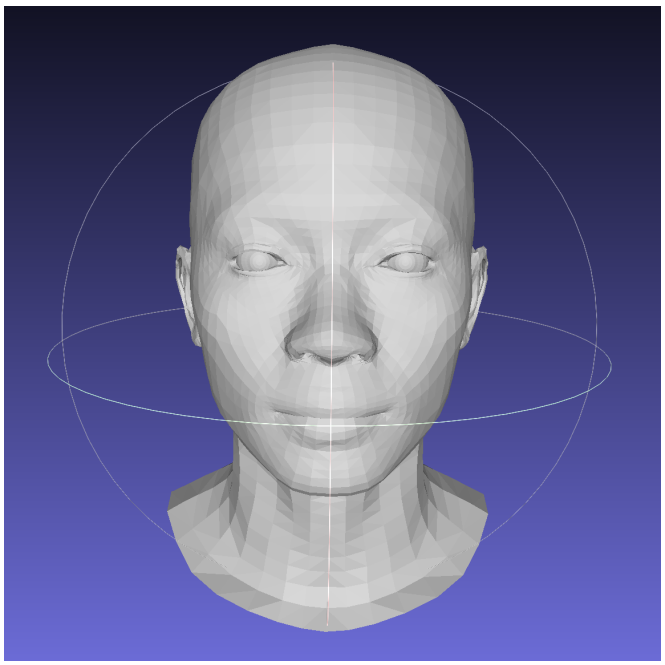
## Results



**Kesav Ravichandran**



**Utkarsh Gupta**

**Yunquian Cheng**

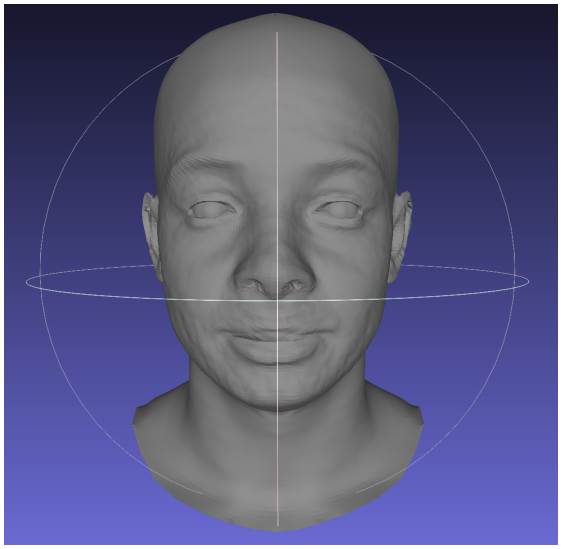
# DECA

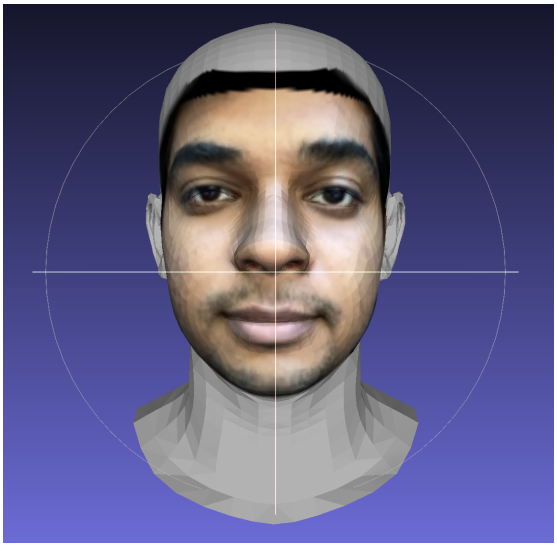
We set up the DECA model by following the instructions present in its GitHub repository. We learned the following ideas while setting up the flame:

1. Setting up the virtual environment for DECA was not easy as setting up MICA. It took us some time to figure out that we need to upgrade the C++ compiler version that DECA was using to compile their C++ code.
2. DECA does not output the seven key points in the FLAME topology that are required to evaluate the model using the NoW metric. Hence, we have to make changes in their inference code and we started saving seven key points in the flame topology to make it work in evaluation.
3. While saving the points in the FLAME topology, we made a mistake in saving the seven points in the correct order. And therefore, the ordering of points in the npy file is critical.

## Results

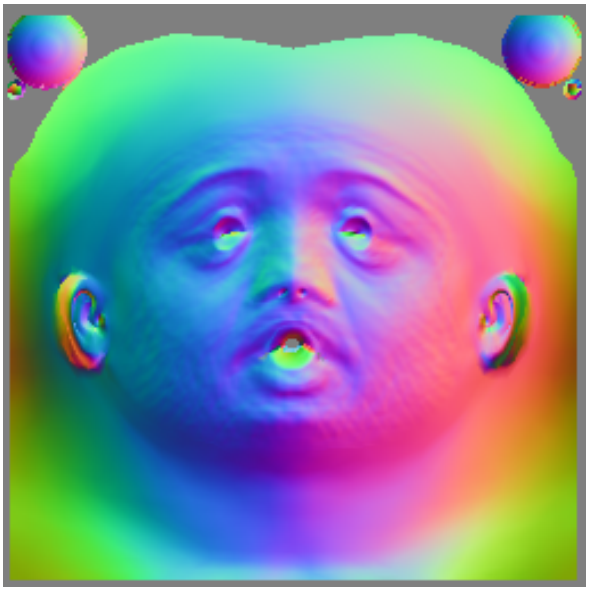
**Kesav Ravichandran**





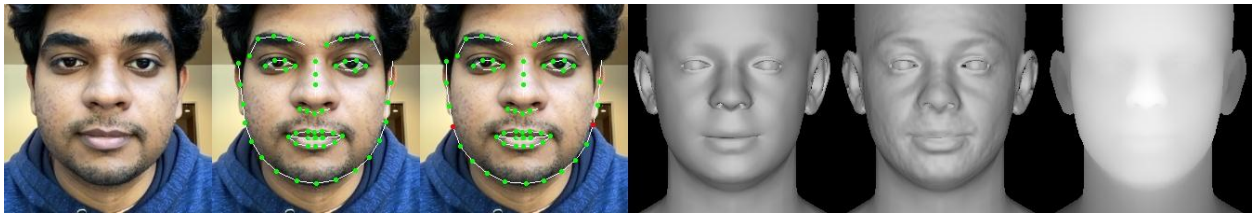
Mesh

Texture

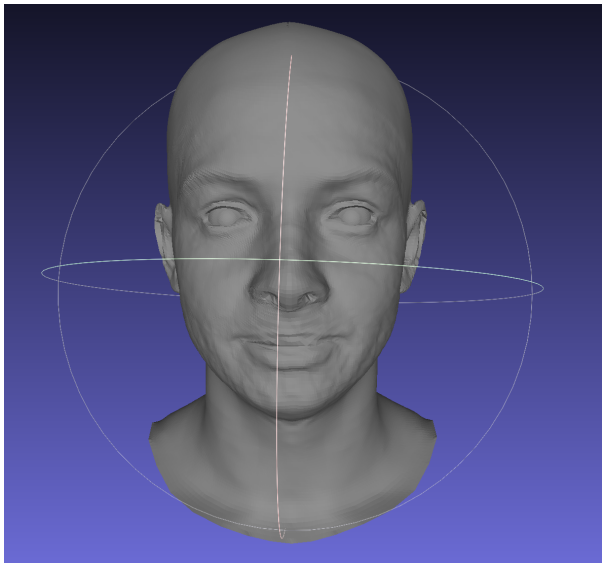


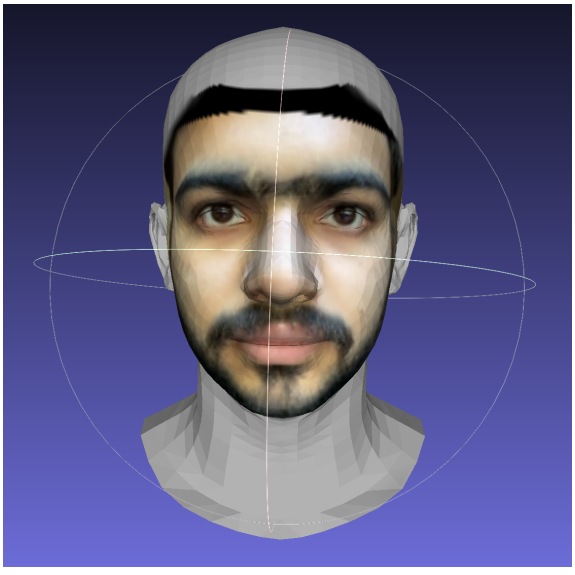
Depth Map

Normal Map



Utkarsh Gupta



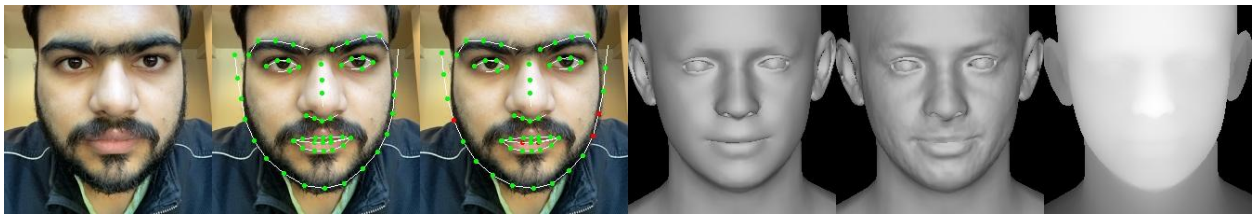


**Mesh**

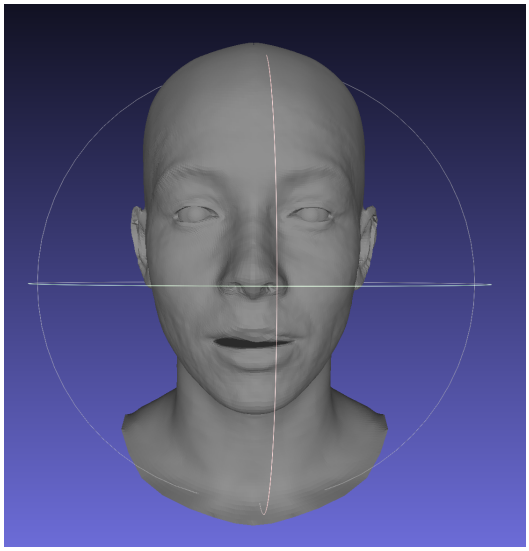
**Texture**

**Depth Map**

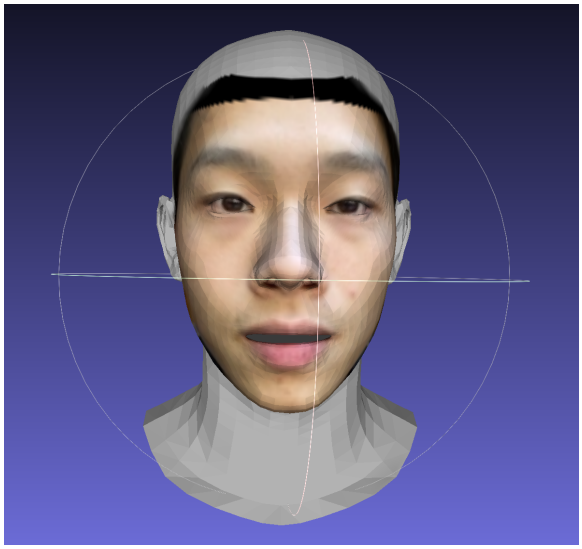
**Normal Map**



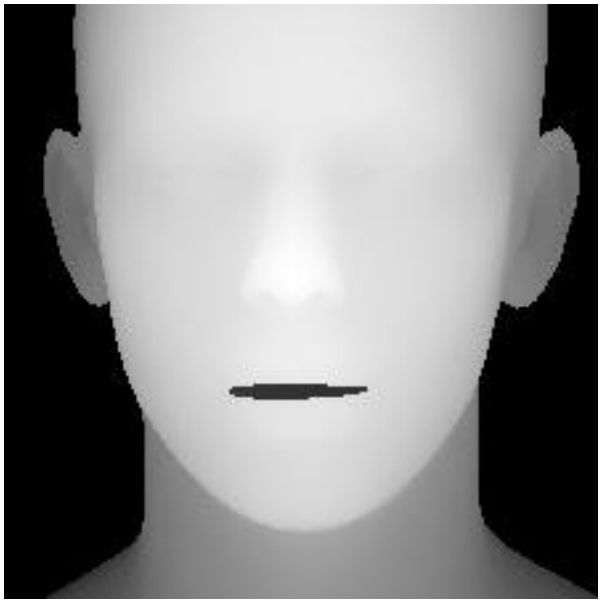
## Yunquian Cheng



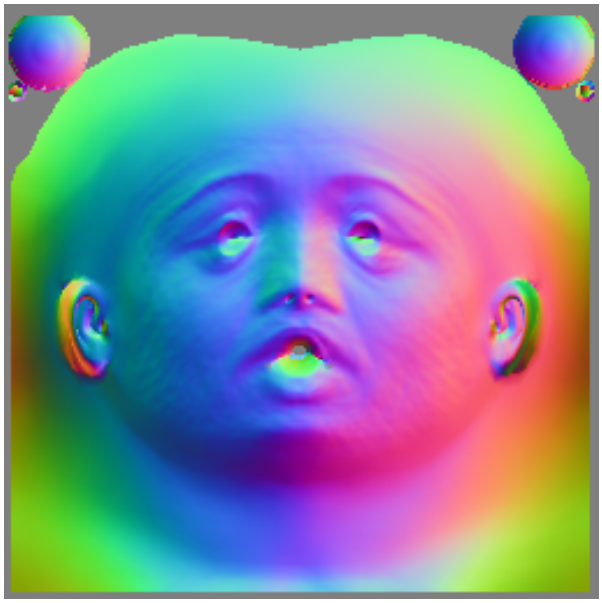
Mesh



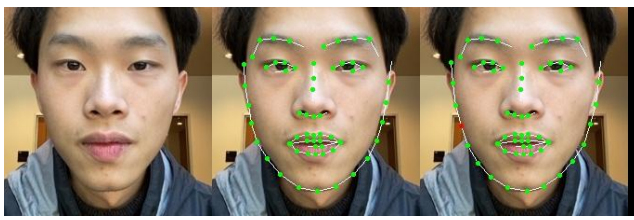
Texture

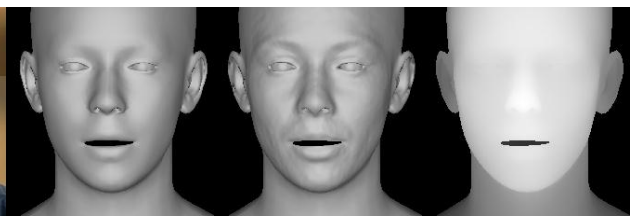


Depth Map



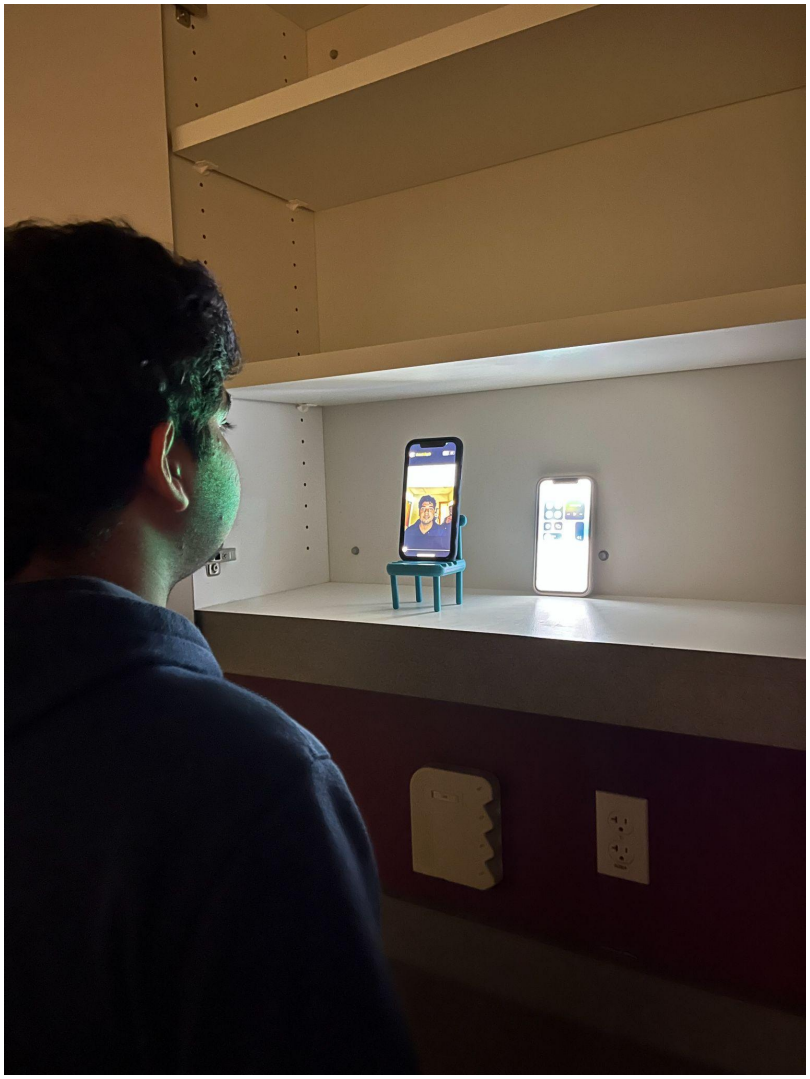
Normal Map





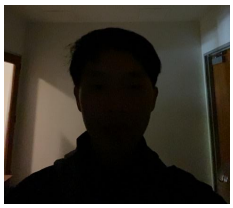
# Low Lighting Experiment

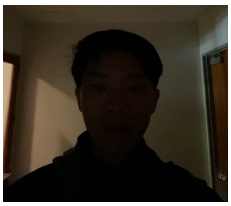
The objective of this low lighting experiment was to find the minimum amount of lighting where machine learning model can work. Therefore, we captured the images by gradually increasing the amount of brightness on the face by another iPhone that is kept in front of the camera. See the figure below to understand the experiment setup.



Low lighting experiment setup

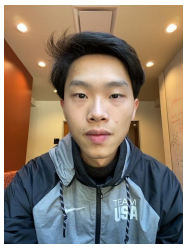
We control the light intensity on the face by changing the brightness of the phone that is kept behind the phone that is capturing the image. We captured the images in four different brightness levels of the phone. The four levels are 25% (A), 50% (B), 75% (C), and 100% (D) brightness levels. The captured images are shows as follows:


(A: 25%)

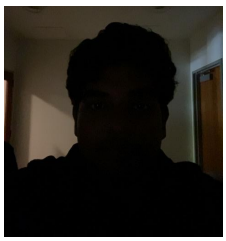

(B: 50%)


(C: 75%)


(D: 100%)


(E: natural)


(F: 25%)


(G: 50%)


(H: 75%)


(I: 100%)


(J: natural)


(K:25%)


(L:50%)

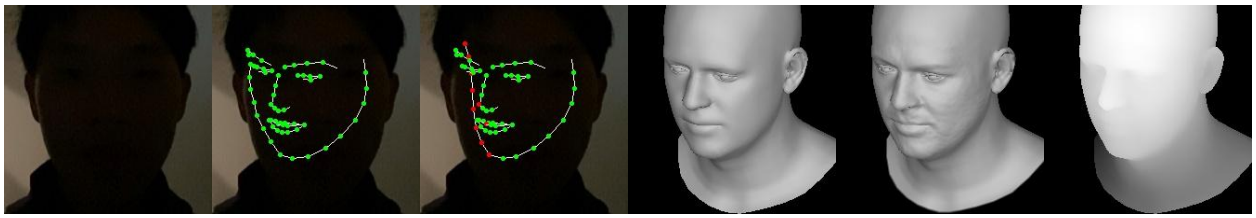

(M: 75%)


(L:100%)

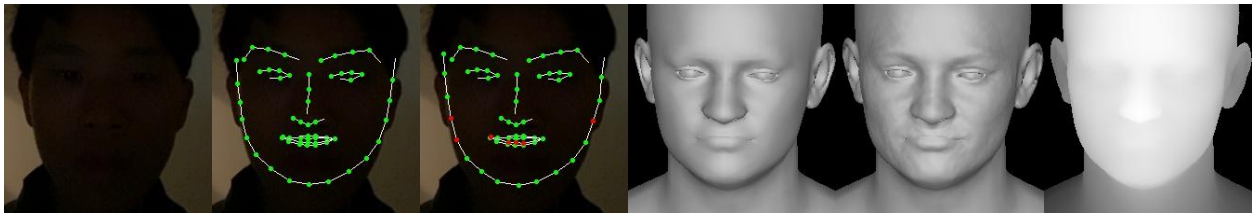

(O: natural)

In an experiment, two machine learning algorithms, MICA and DECA, were tested for their feasibility in low lighting conditions. We were able to observe some reconstruction at 75% brightness level, but the quality of the face reconstructions was still inferior to those obtained using a natural lighting setup.

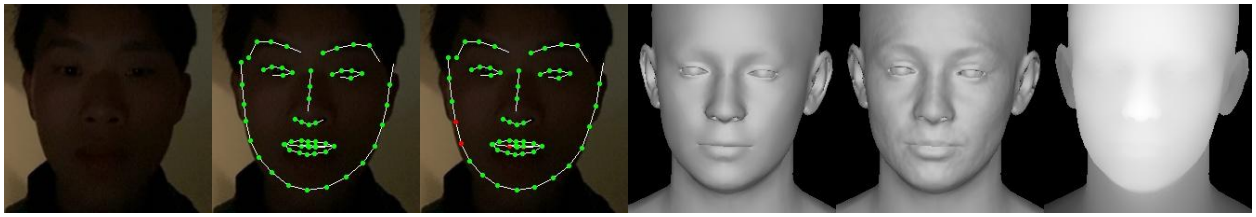
## DECA results:



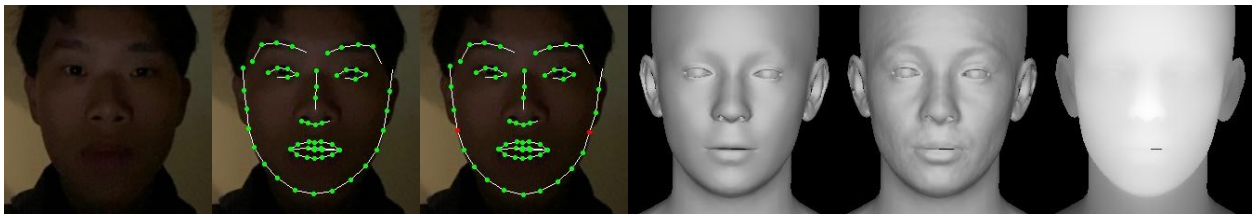
(A)



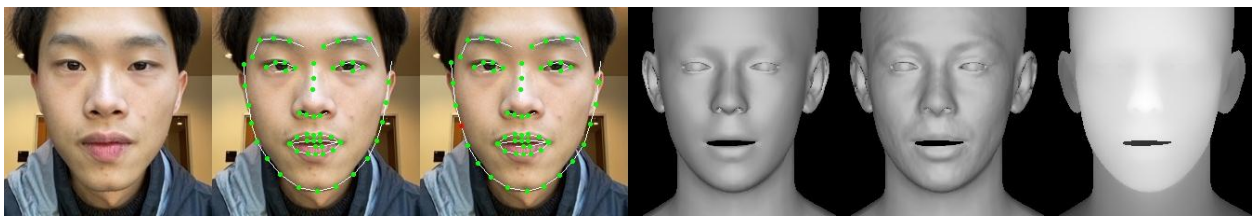
(B)



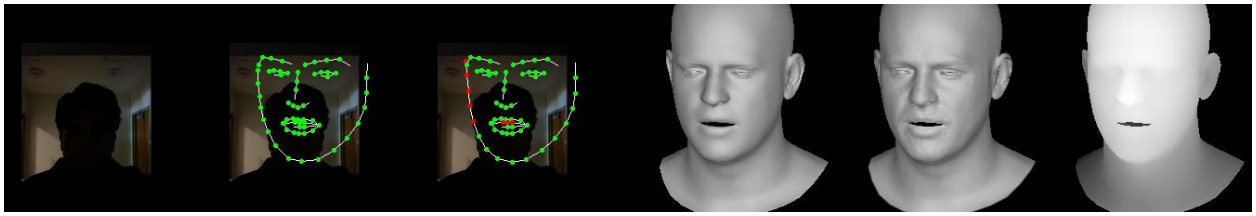
(C)



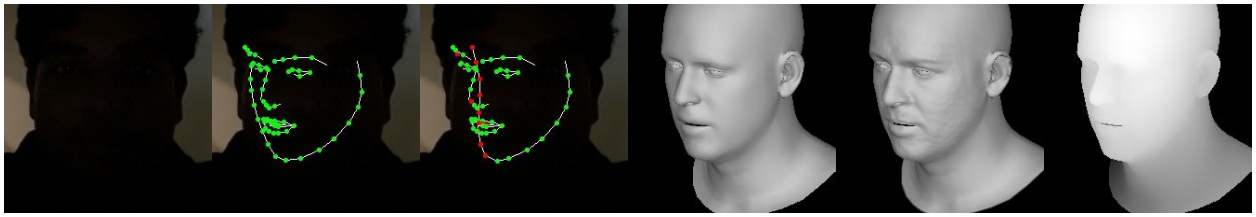
(D)



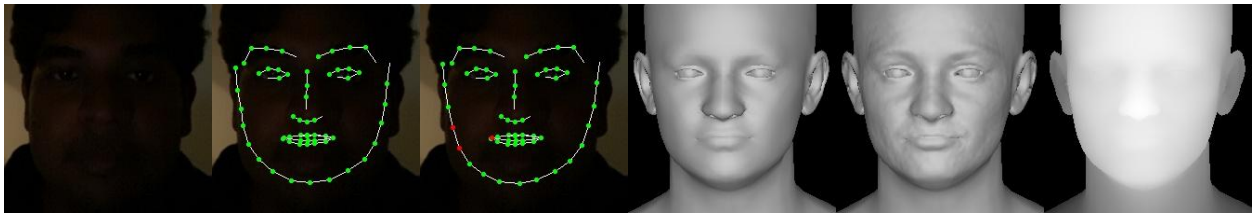
(E)



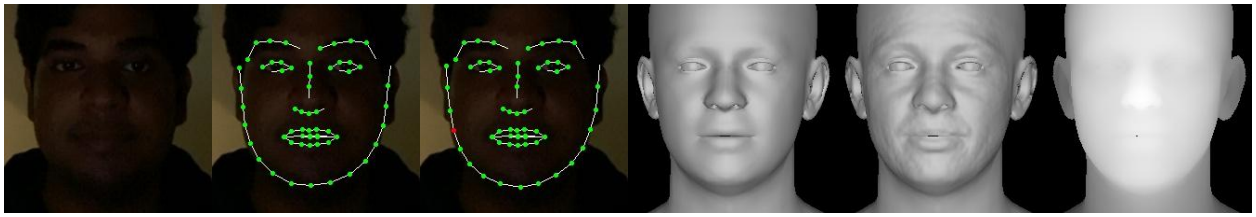
(F)



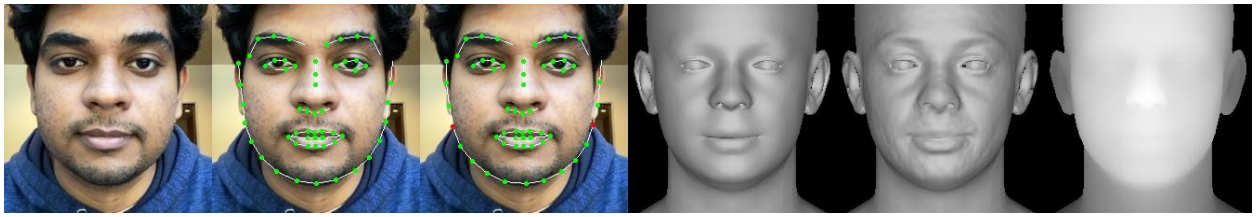
(G)



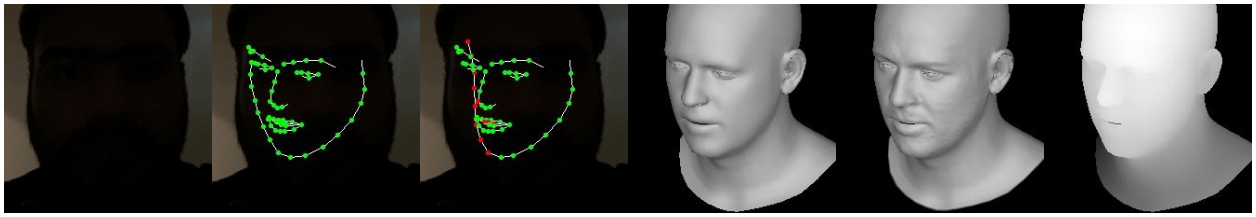
(H)



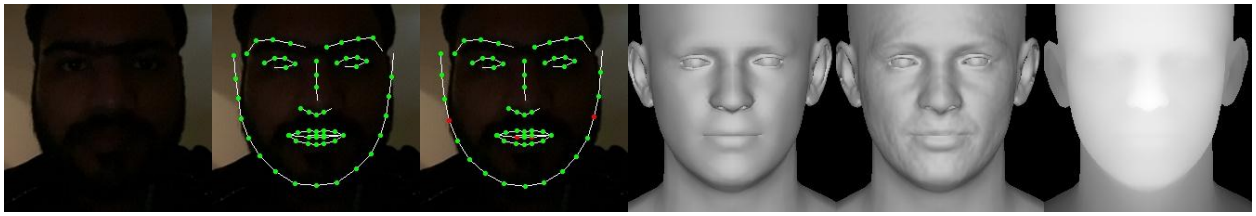
(I)



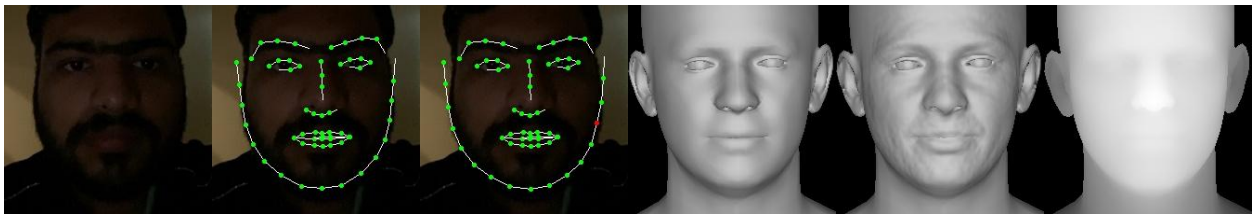
(J)



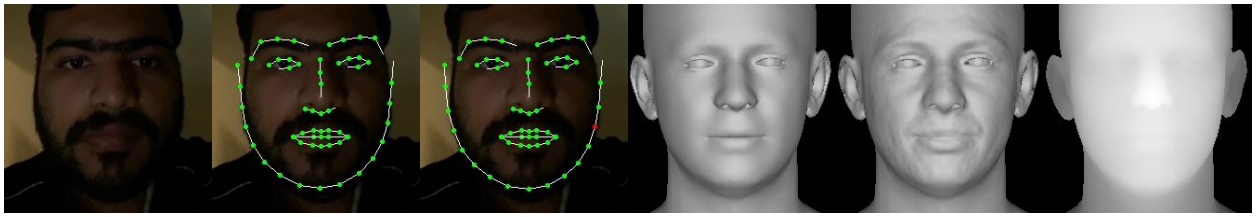
(K)



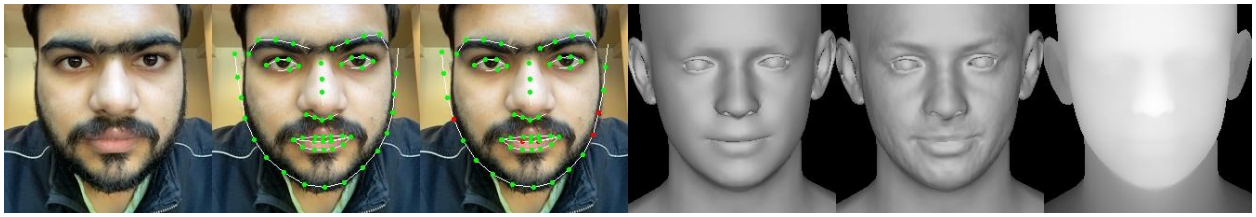
(L)



(M)



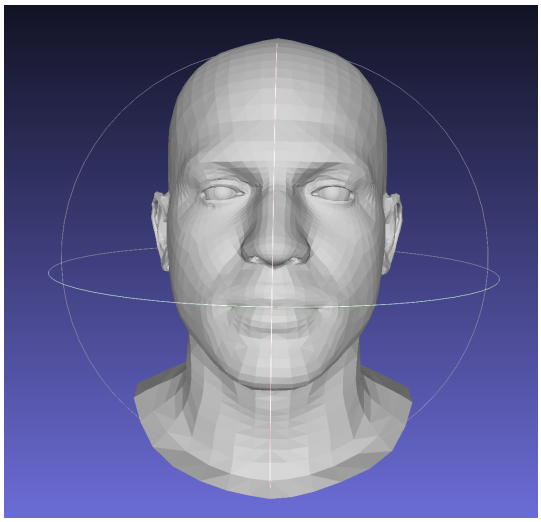
(N)

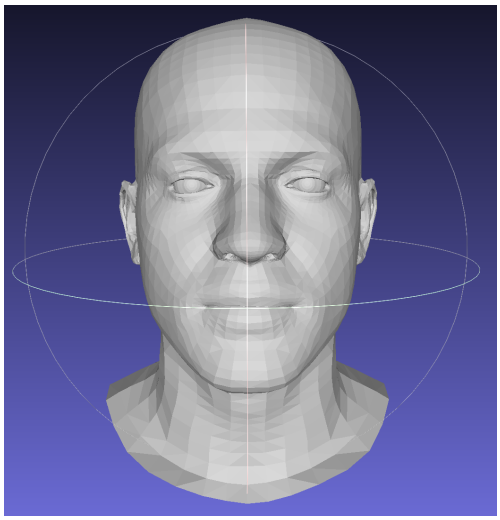


(O)

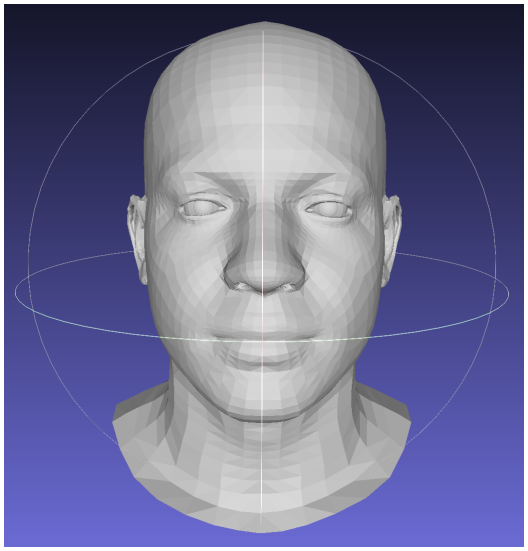
In an experiment involving three subjects, it was observed that DECA, failed to produce accurate reconstructions at 25% brightness level for all subjects. At 50% brightness level, it worked for two out of the three subjects. However, it was necessary to increase the brightness level to 75% to obtain some reconstructions using DECA. Despite this, the quality of the face reconstructions was still inferior to those obtained using a natural lighting setup.

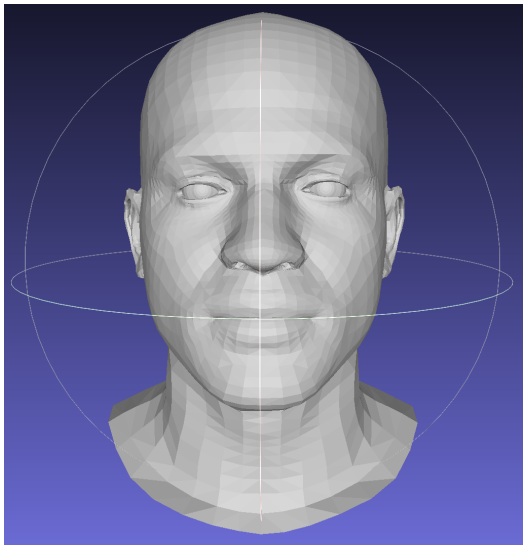
# MICA results:



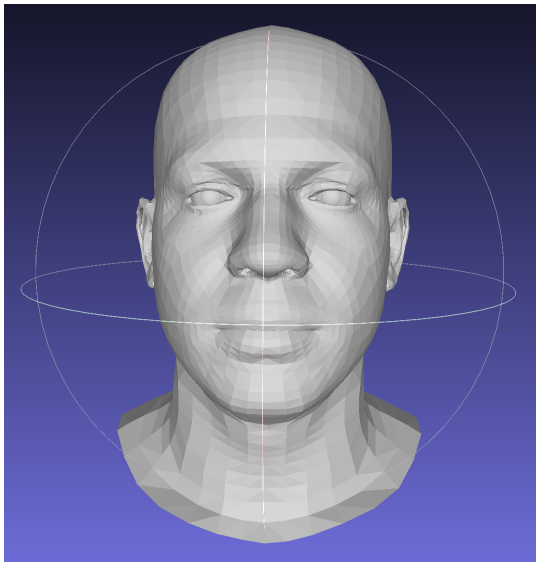


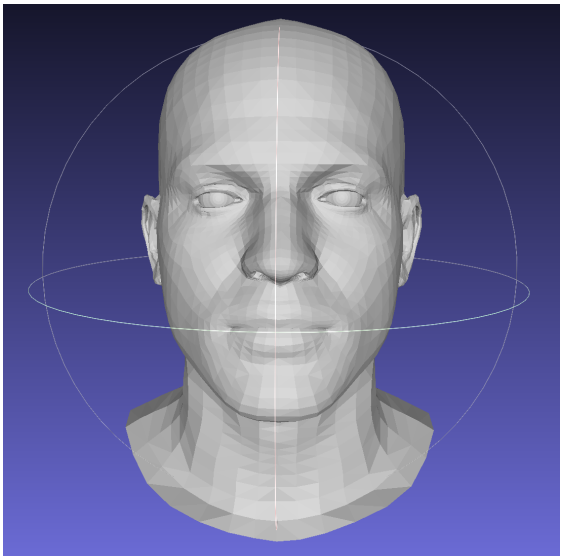
**A: 25% brightness**



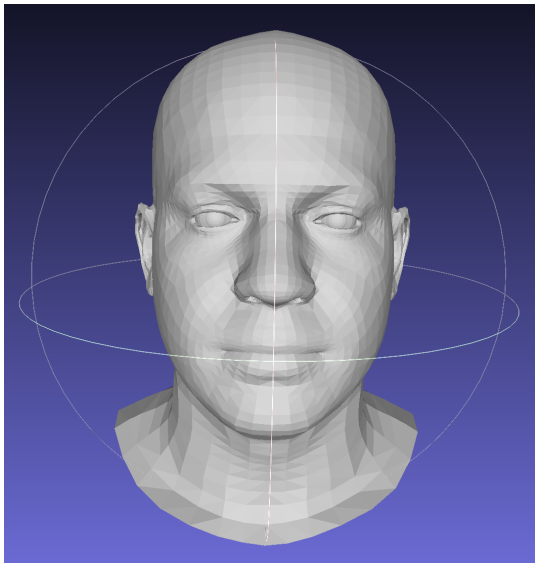


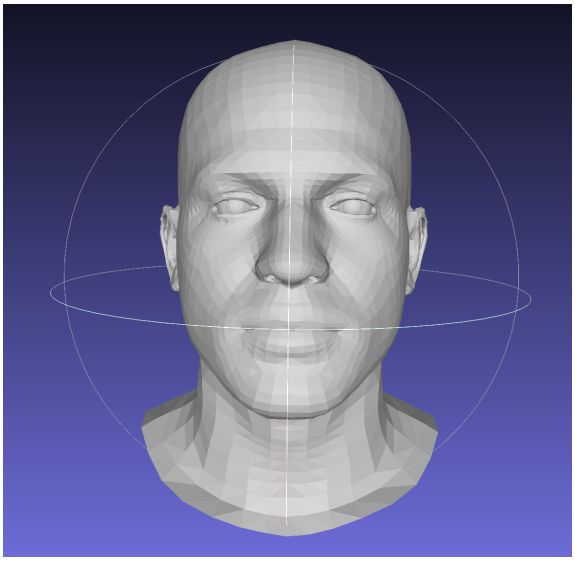
**B: 50% brightness**



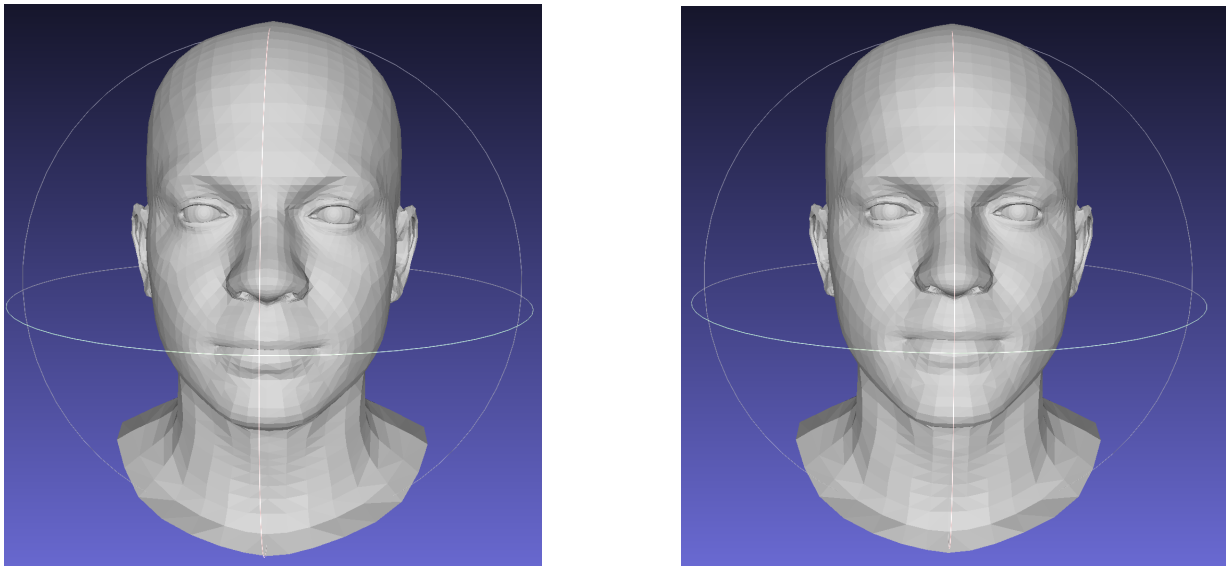


**C: 75% brightness**





**D: 100% brightness**



**E: natural lighting**

MICA is a technology that generates 3D face models, but in an experiment where the brightness level was set at 25% and 50%, the 3D face reconstructions for two different subjects appeared similar. This led to the hypothesis that MICA may be using a default face model to generate these reconstructions when the lighting in the environment is low.

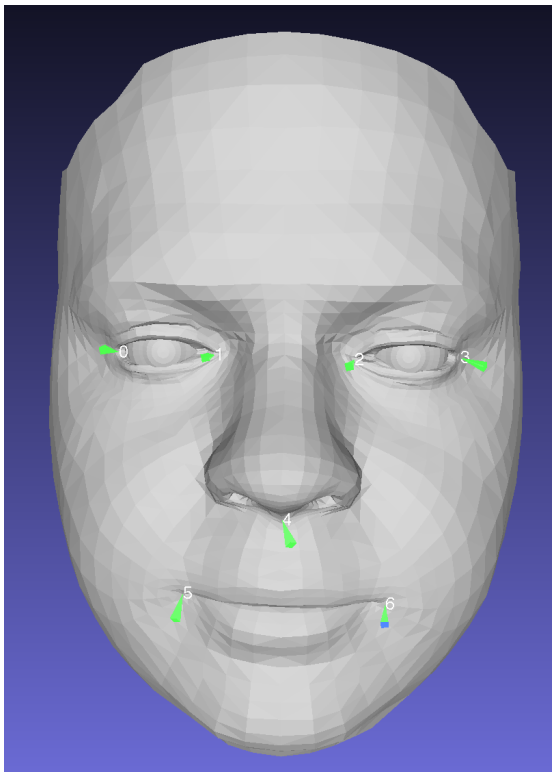
# Now Evaluation

Since we do not have ground truths for evaluating the model, we are going to assume that one of the FLAME scans for our faces is the ground truth. Moreover, we learned to set up the NoW Evaluation pipeline which has multiple steps to make it work.

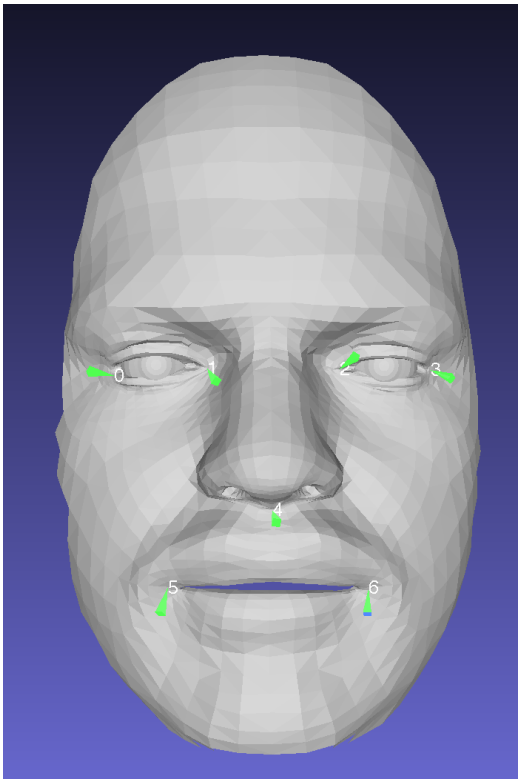
**Step 1: Setting up the ground truths:**

While building the ground truths we have to mark the seven key points that are required in the face alignment step. Every model will have the different topology of the mesh, and therefore, face alignment is required to get the closest match of the predicted and the ground truth mesh.

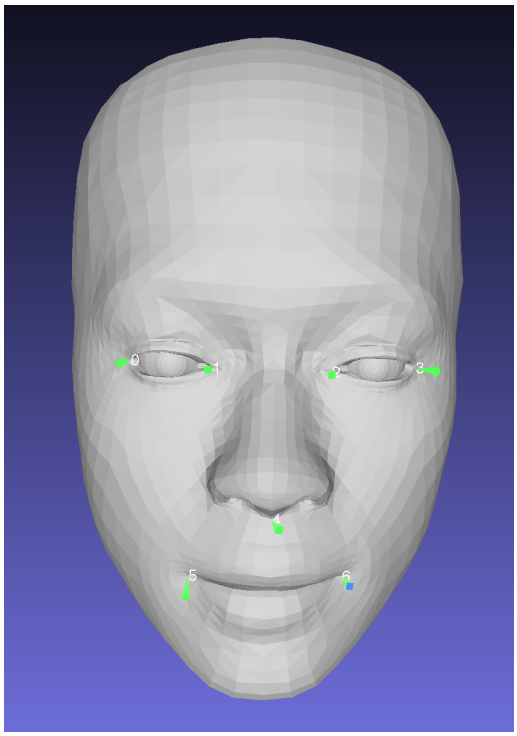
Following are the ground truths that we have assumed from one of the FLAME scans for doing the NoW evaluation:




**Kesav Ravichandran**




**Utkarsh Gupta**



**Yunquian Cheng**

**Step 2: Setting up the folder structure in the same format of NoW evaluation**

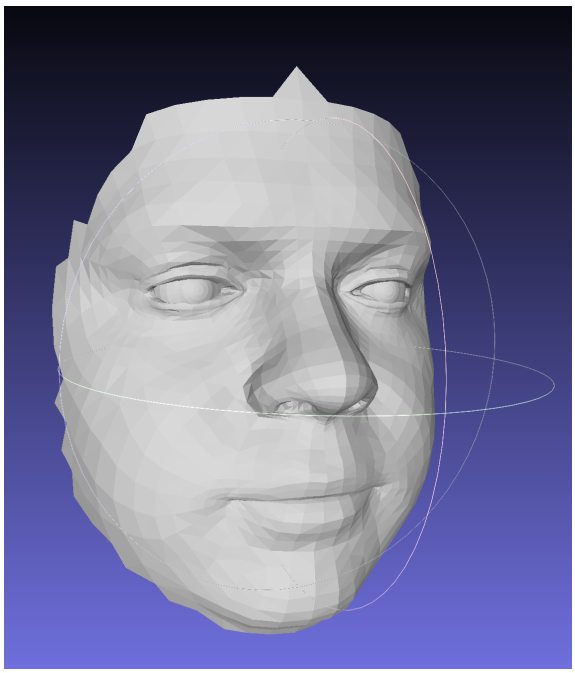
Following is the folder structure for the predicted meshes and landmarks:

**|---- kesav**
**|---- selfie**
**|---- IMG_0000.jpg**
**|---- IMG_0000.npy**
**|---- IMG_0001.jpg**
**|---- IMG_0001.npy**
**|---- utkarsh**
**|---- selfie**
**|---- IMG_0002.jpg**
**|---- IMG_0002.npy**
**|---- IMG_0003.jpg**
**|---- IMG_0003.npy**

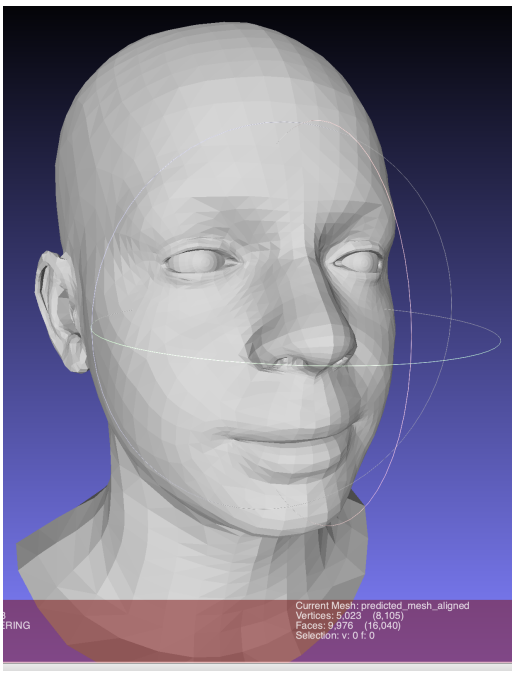
Following is the folder structure of the ground truth data:

**<Dataset_Root>**
**|---- imagespathsvalidation.txt**
**|---- scans**
**|---- kesav**
**|---- ground_truth_scan.obj**
**|---- utkarsh**
**|---- ground_truth_scan.obj**
**|---- scans_lmks_onlypp**
**|---- kesav**
**|---- ground_truth_landmarks.pp**
**|---- utkarsh**
**|---- ground_truth_landmarks.pp**

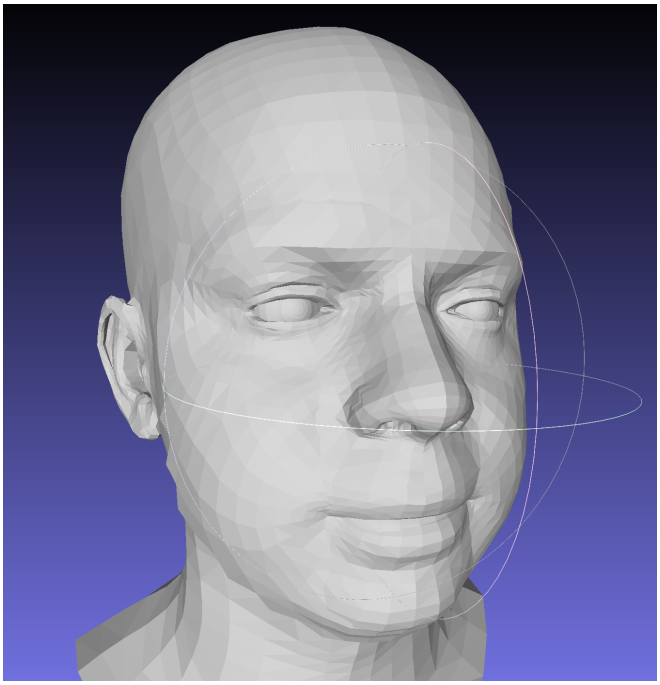
**Step 3: Check the predicted mesh alignment with the ground truth in the mesh lab**



**Ground Truth**




**Predicted Mesh**



**Rough Alignment**

**Step 4: Run the compute_error.py file to get the final results.**

# Conclusion

- Made an iPhone mobile application for facial 3D point cloud capturing
- By comparing results with baseline models from NoW challenge (MICA and DECA) we found that:
    - Our method (iPhone capture point cloud + FLAME) produces visually better results than MICA and DECA
    - Image-based Machine Learning algorithms will either not work at all or will perform suboptimally as compared to the 3d face reconstruction done using iPhone's depth data.
- Publish the iPhone mobile application in the Apple store.

# Discussion & Broader Impact

Our solution uses an iPhone as the data capturing device and only require one capture to generate a good 3D face model. Based on these characteristics, our method can provide a faster, more accessible, and consistent solution for 3D face model capturing up to a certain degree of accuracy. We hope that this work can help researchers capture more data and potentially lead to better solutions to 3D face model capturing

# Individual Contributions

Kesav - iOS application development
Utkarsh - FLAME, MICA, DECA model and NoW evaluation pipeline setup
Yunqian - Experiment design

# Code

cse-260-project

# References


Feng, Y., Feng, H., Black, M. J., & Bolkart, T. (2021). Learning an animatable detailed 3d face model from in-the-wild images. *ACM Transactions on Graphics (ToG)*, *40*(4), 1-13.

Zielonka, W., Bolkart, T., & Thies, J. (2022, November). Towards metrical reconstruction of human faces. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIII* (pp. 250-269). Cham: Springer Nature Switzerland

Li, T., Bolkart, T., Black, M. J., Li, H., & Romero, J. (2017). Learning a model of facial shape and expression from 4D scans. *ACM Trans. Graph.*, *36*(6), 194-1.

Creating Photo and Video Effects Using Depth - WWDC18

AVFoundation Programming Guide - Apple Developer

https://blog.csdn.net/u011574296/article/details/73658560

Intrinsic Matrix and Intrinsic Matrix Reference Dimensions

Streaming Depth Data from the TrueDepth Camera | Apple Developer Documentation