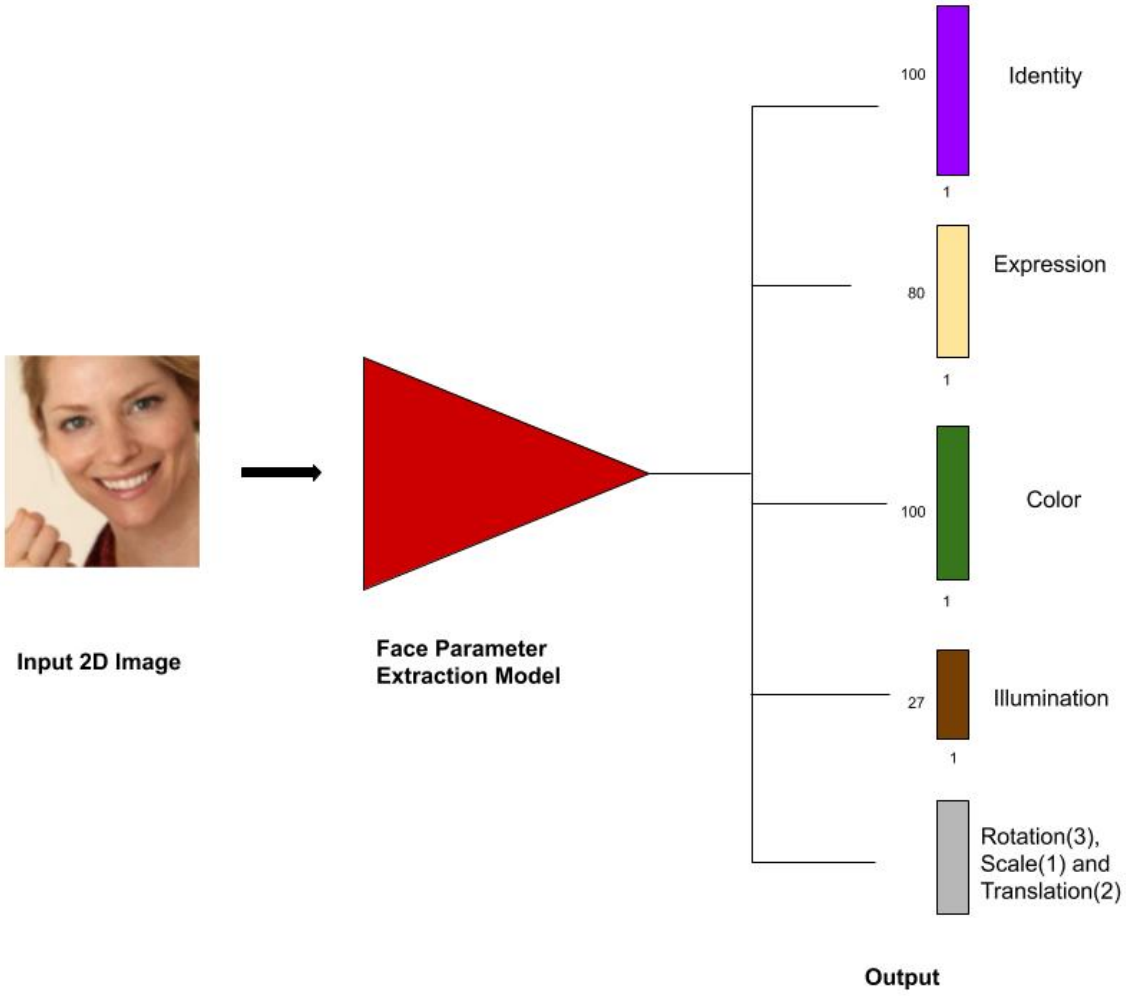
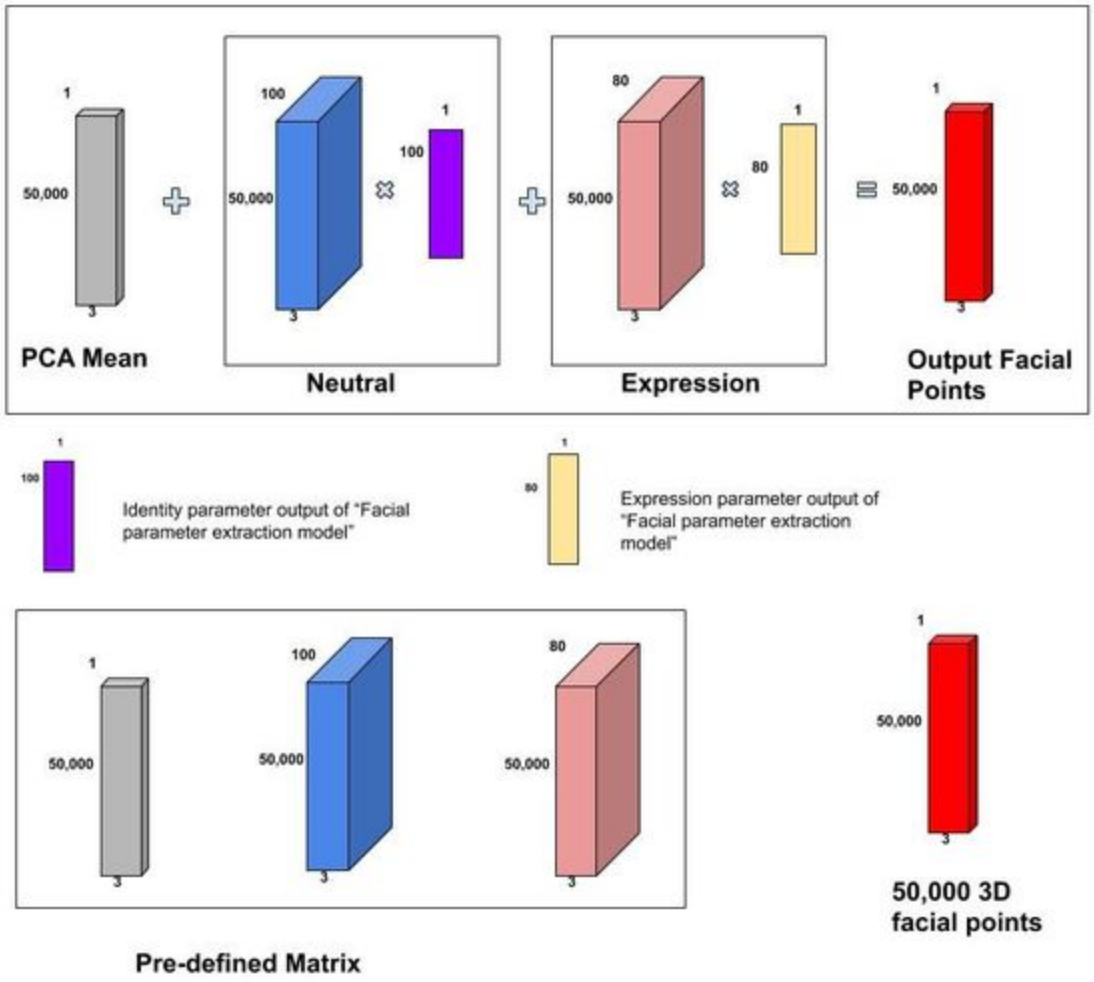
Face Parameter Extraction Model

In this post we will be discussing about the workflow of “Face Parameter Extraction Model” in detail. As discussed on overall architecture, “Face Parameter Extraction Model” outputs 5 parameters namely “Identity”, “Expression”, “Color”, “Illumination” and “Rotation, Translation and Scale” (see image below). We will discuss how we use each of these parameters to reconstruct 3D facial points and how do we train the”Face Parameter Extraction Model”.



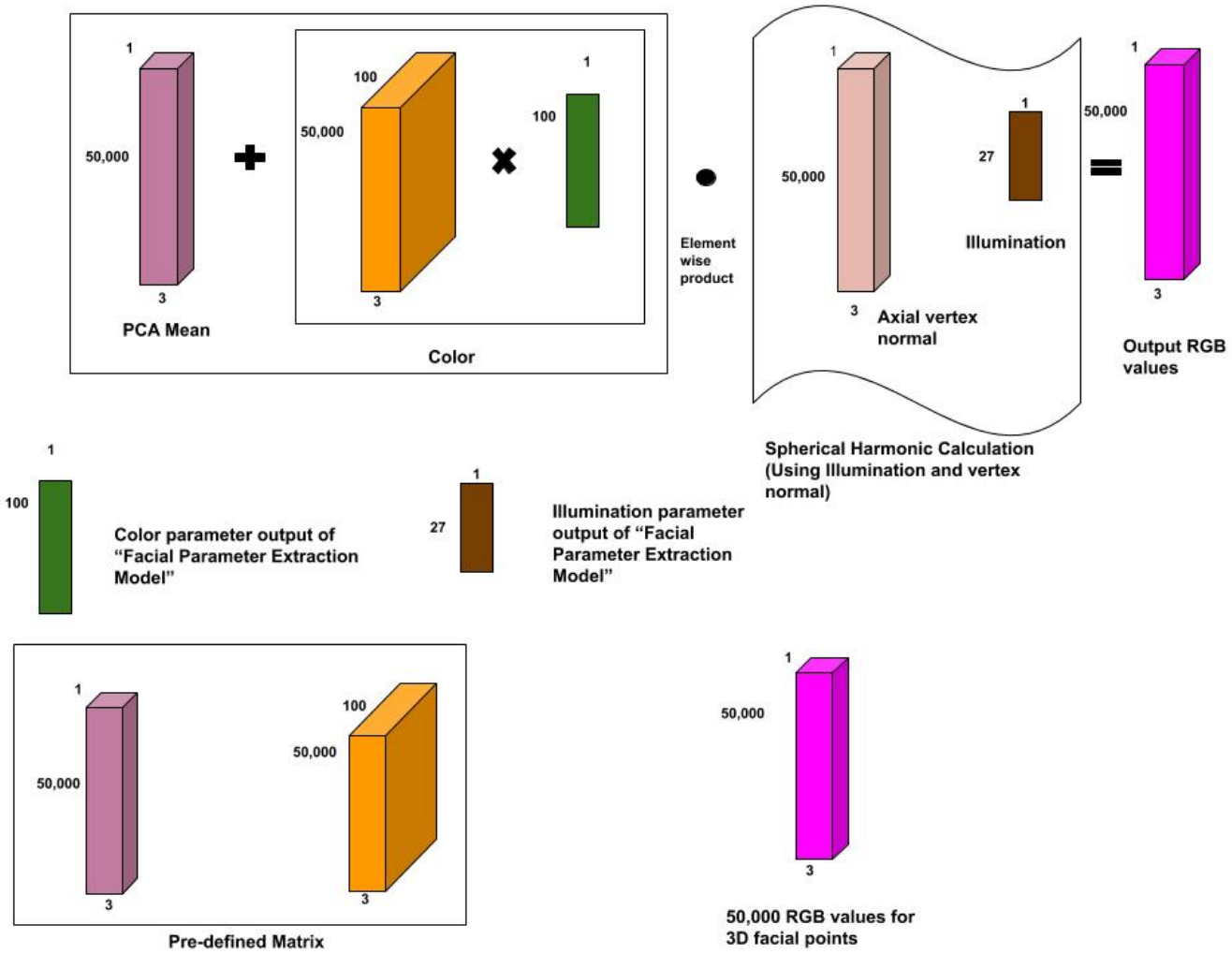
Identity and Expression

We use Identity and Expression parameter output of “Face Parameter Extraction Model” to construct 50,000 3D facial points. To construct that we use predefined matrix named “PCA_mean” and “PCA_neutral” along with “Expression_matrix”, we multiply “PCA_neutral” with “identity” matrix and add that with multiplication of “Expression_matrix” and “expression_vector”, we finally add that resultant matrix with “PCA_mean” to get 50,000 3D facial points. Below image gives better visual explanation of the calculation.



Color and Illumination

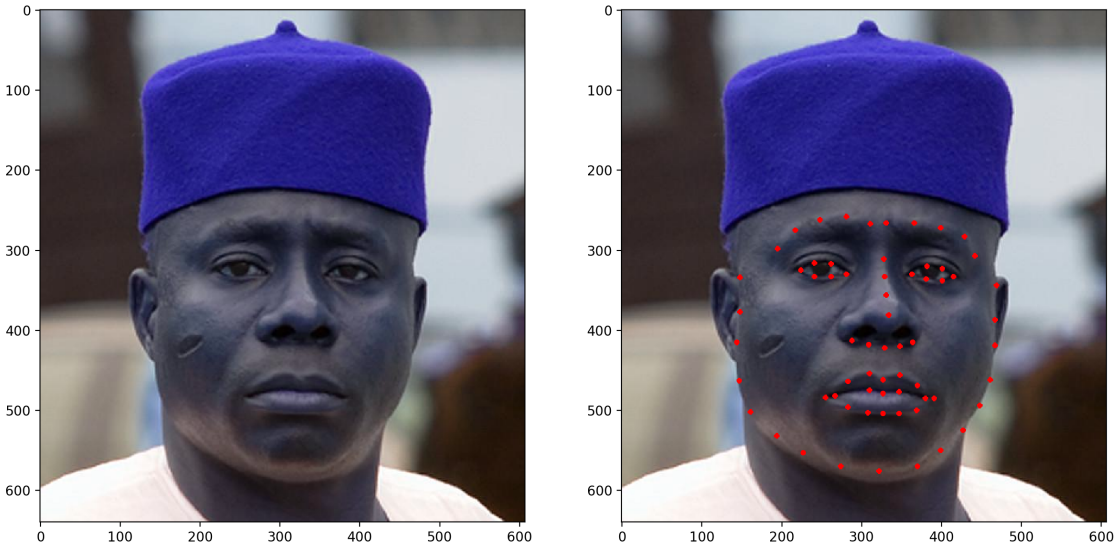
Color is used in similar manner as we use identity and expression, we multiply color with “PCA_color” and added with “PCA_mean” (different from “PCA_mean” for 3D facial points generation”). Illumination parameter is used little differently, more about use of illumination parameter can be found on “`secondorderSH` function defined at “decoder.py” of “UnsupervisedDeepFaceReconstruction” https://github.com/embodyme/UnsupervisedDeepFaceReconstruction . The output of this operation is 50,000 RGB values of those corresponding 50,000 3D facial points (Check `get_pca_colour` function definition on decoder.py for more information)



The output 50,000 3D facial points and 50,000 RGB values are used to create 3D face which can be further used to create 2D image using "Rotation, Translation and Scale" parameters and further to train the network.

Training Data Visualisation

Training data for training of "Face Parameter Extraction Model" is 2D RGB image of different person with various expressions. We've tagged 68 feature vertices on each of these 2D RGB images. Below is an example of an example training data.



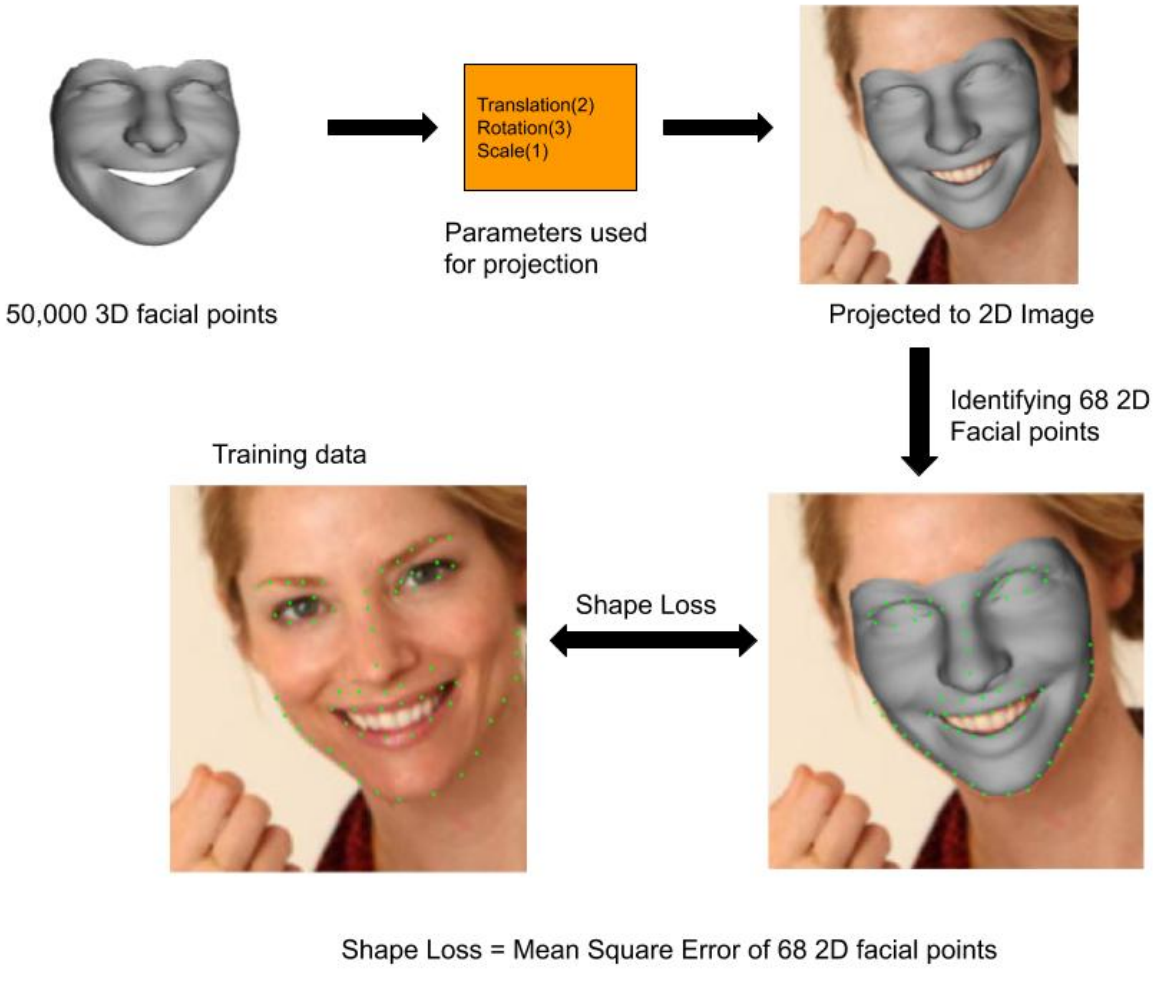
Rotation, Scale and Translation output produced by the “Face Parameter Extraction Model” is used to project the 3D facial points to 2D image, without those parameters there would be no way to know what view to project the 3D facial points.

Shape Loss and Color Loss

In this section we will discuss how we get the loss function which is used for training of the model. For training of the model 3 different loss functions are used i.e. shape loss, color loss and regularization loss. We are familiar with regularization loss (if not follow this link) so we will directly jump to discuss shape loss.

Shape loss

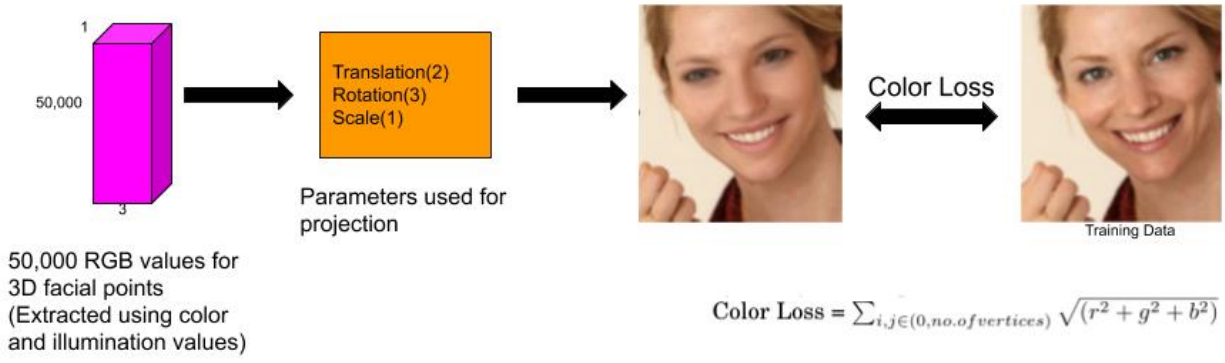
Shape loss is calculated using the 50,000 3D facial points and rotation, scale and translation parameters. We project 3D facial points to create 2D image using rotation, scale and translation parameters. We later identify 68 feature points on created 2D image (since we already have 3D facial points, its easy to identify corresponding 68 feature points).



Now we have 68 feature points of created 2D image and 68 feature points of ground truth image, shape loss is calculated by mean square error of the distance between corresponding feature points. See the picture above for more details.

Color Loss

Color loss is somewhat similar to that of shape loss but with a little variation. To get 50,000 RGB points for corresponding 50,000 3D facial points we use illumination and color parameter of “Face Parameter Extraction Model”. Getting 50,000 RGB points from illumination and color parameter discussed above, we further project those RGB values to 2D image using rotation, scale and translation parameters (similar to how we project for shape loss) .



We calculate color loss by calculating the mean square error of RGB values throughout the image (not just the feature points like we did on shape loss). Loss function is mentioned above in the image. We also calculate regularisation loss using the output face parameters so that their values doesn’t go too high (general idea of regularisation loss, check `get_regularization` on decoder.py for more information).

We use 3 loss functions (shape loss, color loss and regularisation loss) to train the “Face Parameter Extraction Model”. Next we will discuss about the “Mouth Interior Synthesis” which is the next step after generating facial parameters in Xpression App.

Link to discussion about “Mouth GAN” is here https://embodyme.atlassian.net/wiki/x/BwA8F .