System Design Document

For

Facial Analyzer for Rhinoplasty Surgery (Option 9)

Team members: Priscilla Carbo, Marvin Cazeau, Jared Curtis, Maree Kelly, Alessandra Oo

Version	Date
v1.0	16 September 2021
v1.1	28 September 2021
v1.2	26 October 2021
v1.3	30 November 2021
v2.0	02 February 2022
v2.1	06 March 2022
v2.2	08 April 2022

TABLE OF CONTENTS

SYSTEM DESIGN DOCUMENT

Overview
The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

1. INTRODUCTION

1.1. Purpose and Scope

The purpose of the Systems Design Document is to track the necessary information required to effectively define the architecture and system design for Facial Analyzer for Rhinoplasty Surgery v2.0. This is done to accurately detail and provide guidance for the development team of the system architecture implementation. Information such as the operating environment, system architecture, interface design, and Machine Learning techniques included will be found within this document.

1.2. Project Executive Summary

This section provides a description of the Facial Analyzer for Rhinoplasty Surgery (Option 9) project from a management perspective and an overview of the framework for which the conceptual system design was prepared.

1.2.1. System Overview

The goal for the Facial Analyzer Rhinoplasty Surgery project is to generate a database of realistic 3D faces through a GAN (Generative Adversarial Network) modeling software. Then, fit the patient to a cluster in order to provide the patient with nose options that are the most suitable for their face based on the cluster they were assigned to. These clusters of similarly structured patients will be generated from a database using multiple varying clustering algorithms. Lastly, the significant landmarks from these faces in the cluster will be used to suggest changes to the patient's nose.

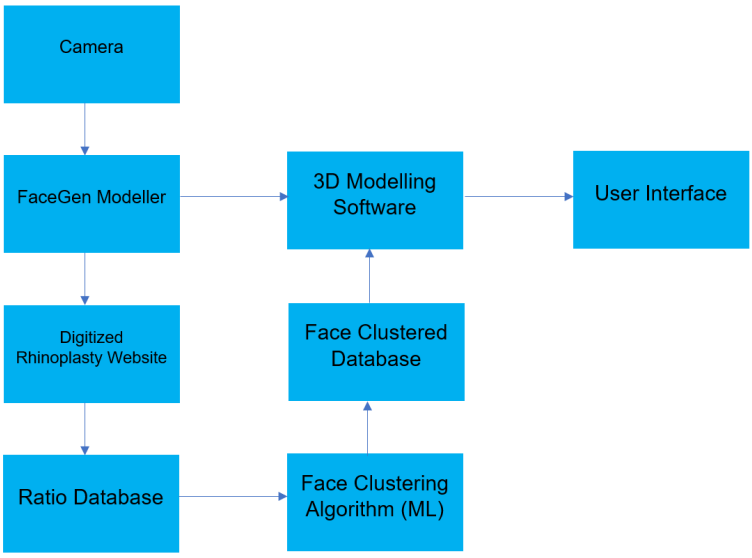

Figure 1: Overview of the methodology of the entire system

1.2.2. Design Constraints

There are a few notable design constraints for this system as of v2.1. Firstly, the system relied on an application called "Bellus3D" to produce 3D scans of the patient, which can only be downloaded using an iOS device with the iOS App store. Moreover, the application has been removed from the Apple app store since the end of December, 2021. Previous implementations of grouping faces had the KNN (K-Nearest Neighbors) algorithm which did not suit the goal of the software because it made less accurate facial clusters, so it has been replaced. FaceGen Modeler is now used for creating 3D faces to add to the database. In order to create these 3D models using this software (FaceGen Modeler), it needs to be downloaded on a device and 3 images of a person, front, left and right side, are needed, which is another constraint. Additionally, FaceGen puts a watermark on the 3D model, which can make plotting landmarks challenging.

1.2.3. Future Contingencies

Contingencies that may arrive in the future are in Unsuitable Machine Learning AI suggestions. Unsuitable machine learning (ML) is the possibility that when a suggestion, that is either bugged or off the set parameters, is made, the image created will be incorrect. This would require rigorous testing to identify areas of weaknesses in the code that caused this issue. Machine Learning Incompatibility is the use of an ML AI that does not fit the purpose of what we're trying to accomplish. The KNN algorithm having to be superseded in this project is an example.

1.3. Document Organization

This document is designed to give the reader an overview and in-depth look on the system design and the documentation to understand the system, process, and architecture. The following will include the system architecture, the different interfaces used in this project, hardware and software designs, and the system integrity.

1.4. Glossary

Rhinoplasty (*noun*)	Plastic surgery on the nose usually for cosmetic purposes
GAN (*abbrev.*)	Generative Adversarial Network - A specific type of machine learning framework that emphasizes learning and regularizing training data sets to output or generate new examples that are similar and/or comparable to the original dataset.
ML (*abbrev.*)	Machine Learning - Subset of AI that includes methods of autonomous analytical model building.
AI (*abbrev.*)	Artificial Intelligence - Intelligence algorithms that mimic human and natural behavior
UI (*abbrev.*)	User Interface

2. SYSTEM ARCHITECTURE

2.1. System Software Architecture

2.1.1.1. User Interface

There are a few distinct interfaces the user will interact with. The user will use 'FaceGen Modeler' once they obtain 1 front view and 2 side view 2D .jpegs of the patient's face in order to get a .obj, .mtl, and a .jpg file. These files are then used in the Digitized Rhinoplasty web application to generate the desired JSON file. Moreover, the user will utilize these generated files (obj, mtl, jpg, JSON) in Blender to model the 3D face and facial landmarks to manipulate the patient's nose.

2.1.1.2. Database

2.1.1.3. The surgeon collects the 1 front view and 2 side view 2D jpegs of the patient's face using a camera.

2.1.1.4. The scan is then sent as an .obj, .mtl, and .mtl file to the Digitized Rhinoplasty web application.

2.1.1.5. The FaceGen GAN software will be used to increase the faces in the database.

2.1.1.6. The Digitized Rhinoplasty Application is used to plot the points on the patients'/volunteers' faces manually. After positioning the specified points on the nose the JSON files are exported from the web application.

2.1.1.7. The JSON files of the patients are then utilized to populate the database for the clustering model.

2.1.1.8. The JSON files are read by read_in.py using Pandas, after the code is run, the user is prompted to enter the file name on the console. After entering the file name of the desired file, the program creates data frames and generates a .CSV file of the final data frame into the same directory where the file is being run from. The naming convention uses the patient's original file name and "_df" added at the end.

3. HUMAN-MACHINE INTERFACE

This section provides the detailed design of the system and subsystem inputs and outputs relative to the user/operator. With each component of the system, outputs in certain applications/models serve as inputs to other applications/models, which is outlined below.

3.1. Inputs

3.1.1. Each user, with assistance from the Option 9 project team member, will input their 2D .jpegs of their face into the FaceGen Modeler software.

3.1.2. The input to the Digitized Rhinoplasty Web application is the scanned image as an .obj, .jpeg and .mtl file.

3.1.3. On the Digitized Rhinoplasty Web application, the system operator will rename the files with patients' names and point significant landmarks on the said face manually.

3.1.4. The Blender application takes in 3 files: The exported JSON file from the Rhinoplasty web application for both the dataPointsFile_path and the newDataPointsFile_path, the patient's obj file for the faceObjectFile_path, and the patient's Scaninfo folder for the meshFile_path.

3.2. Outputs

3.2.1. The GAN model outputs the scanned face into an .obj, .mtl and .jpeg file.

3.2.2. After the landmarks are implemented manually, the Rhinoplasty Digitized Web application outputs a JSON file with landmarks.

4. DETAILED DESIGN

4.1. Hardware Detailed Design

4.1.1. The hardware component needed for the Digitized Rhinoplasty project is an iOS computer or Windows device that is configured for Python version 3.0 or later.

4.2. Software Detailed Design

4.2.1. The GAN model will receive input as a .jpg and output .jpg, .mtl, and .obj of Patient's face to be used in Rhinoplasty Digitized Web application.

4.2.2. The Rhinoplasty Digitized Web application will output a JSON file of the patient's face with the significant landmarks and respective measurements.

4.2.3. The user then uploads the JSON file into the read_in.py, which will add the patient to the database and/or set the patient as the current one; as well as generate a CSV file from the JSON file.

4.2.4. The software will already have all of the users data uploaded and the measurements after going through the algorithm.

4.2.5. The clustering algorithm will be used to group a patient's 2D face with a group of similar faces that resemble their own.

4.2.6. The user will choose one of these patient's noses depending on the similarities they have with the database.

4.2.7. Blender will allow a surgeon to alter the nose of the patient's face utilizing the landmarks and ratios.

4.3. Internal Communications Detailed Design

First, the current patient's csv file is read containing the significant facial ratios used in the clustering model. The next step in the algorithm is to compare the patient's face to the database of face ratios to assign the patient to an appropriate cluster, where the patient will be able to choose a select few faces that they believe most accurately resembles them. These selected faces are then used by a python script to suggest changes to the patient's nose.

5. EXTERNAL INTERFACES

5.1. Interface Architecture

The external system used by the Facial Analyzer for Rhinoplasty Surgery project system is an excel file used to generate its database of faces. Users are able to access the face database by utilizing a program such as Excel, Notepad, Visual Studio Code, etc.

5.2 Interface Detailed Design

The interface utilizes the command prompt to run the python files.