# The Many Faces of Microbial Communities

Senior Design
Spring Term Final Report

Thomas Albertine, Michael Phelps

## I. Abstract

CONTENTS

## II. Introduction

Our client, Dr. Jenna Lang, a microbiology researcher from University of California Davis requested that we create a tool that she could use to visualize microbial population data as human faces. This tool could potentially help microbiology researchers (as well as those from other fields) find patterns that aren't obvious using existing visualization techniques.

Currently, the standard visualizations are pie charts or heat maps. Unfortunately, Pie charts aren't effective when there are a large number of categories to represent because there are too many slices. Heat maps on the other hand, while still useful to see patterns, do not make proportions between organisms obvious the way pie charts do. Our tool takes advantage of the human brain's ability to recognize and interpret patterns in faces to visualize data, so users can find new patterns that they would normally miss.

Our team consisted of Thomas Albertine and Michael Phelps. We had no formal roles aside from "developer", although generally (but not exclusively) Thomas worked on the backend code, loading data files and generating models, while Michael worked on the frontend code, connecting the backend to a user interface (henceforth referred to as UI). Our client played a hands off role. We spoke with her early in the project to collect requirements and have contacted her to revisit requirements or to keep her aprised of how the project is going, but she generally gave usa lot of freedom regarding how to proceed.

## III. Original Requirements Document

### A. Requirements Document

The Many Faces of Microbial Communities Project

Team Name: ViewCrobe Software

Members: Thomas Albertine, Michael Phelps

*1) Introduction:*

*a) Purpose:* The theory behind the project is that, since there is a specific portion of the human brain dedicated to processing faces, visualizing microbial data as faces allows users to better compare the complex data. Therefore, the purpose of the project is to provide that functionality in a tool usable primarily by microbiologists, but secondarily by researchers in other disciplines.

*b) Scope:* FaceView will be a user-friendly tool for providing visualizations of microbial population data by representing data as 3d models of humans. Given a supported data file, FaceView will allow the user to select a subset of samples from the file from which to generate models based on a subset of the various populations in the file. After these models are generated, these models will be draw to the screen for comparison by users.

In order to enhance user-friendliness, sample subsets will be relatively small for ease of comparison, while population subsets will be significantly larger to allow for more comprehensive comparisons.

*c) Definitions, Acronyms, and Abbreviations:*

- Model–In the context of this project, "model" refers to a 3 dimensional representation of a human based on a subset of the input data. Since a given model is generated from a sample, in some contexts, the two terms may be interchangeable.
- Sample–In the context of an input file, a sample is a subset of the data representing one population. As an example, a sample may contain the number of colonies and which species they are associated with in a petri dish.

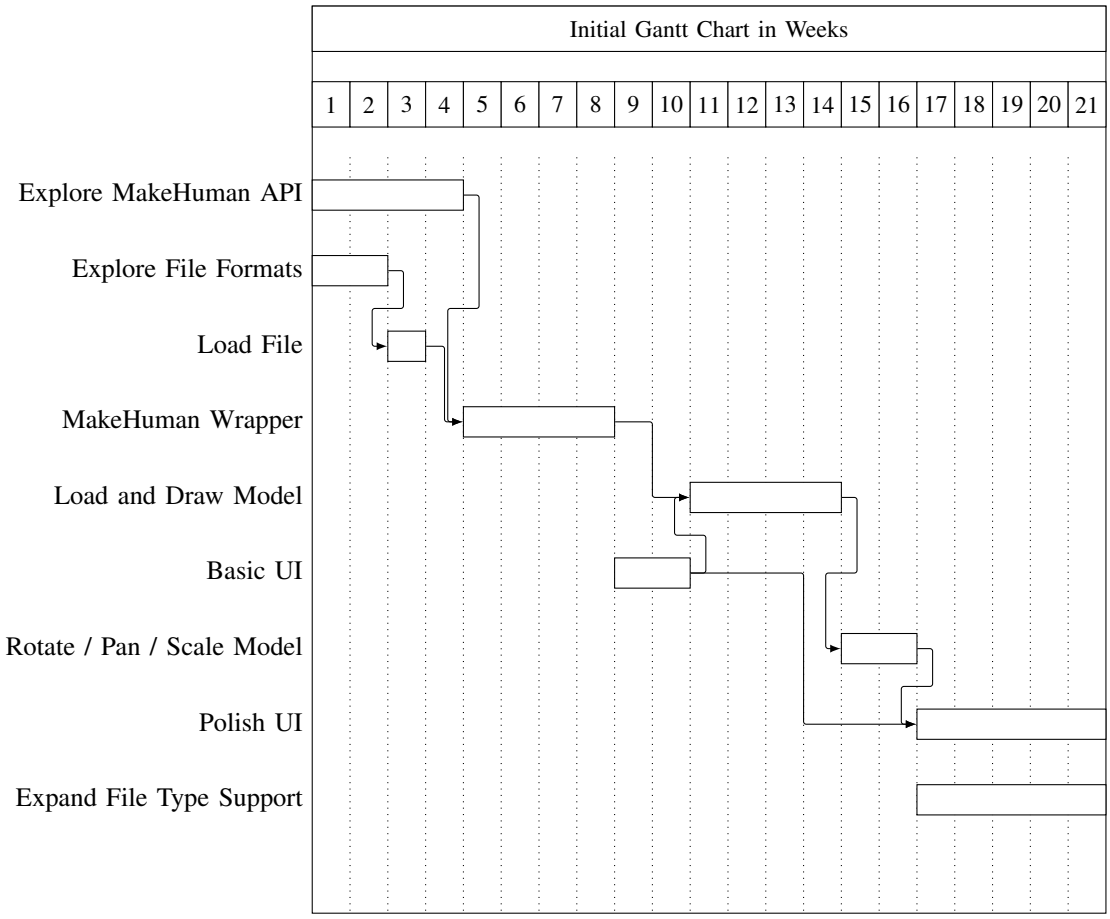


Fig. 1. This is the original Gantt chart for the project.

- Common Camera Angles–A set of configurations for the observer of a 3d scene that are used often. Such a set would include a front view, a profile view, and a three-quarter view at least.
- Visualization Profiles–Data files may include many samples and many organism populations. It may contain more populations than there are supported models or model parameters. Therefore it may become necessary to ignore some samples and some organism populations. Furthermore, the user may want to reassign the model parameter that corresponds to a given organism's population to better emphasize differences in that organism among the samples. However, manually modifying which model parameters correspond to which organisms and which samples should be ignored could be time-consuming to do each time. Therefore it may be expedient to create a profile of these settings, which could be saved and loaded later.
- Model Parameters–Values that can be passed in when generating a model in order to influence the outcome. For example, nose length might be a model parameter.
- Data Parameters–Values that are read from the data. These can be associated with model parameters to make the visualization reflect the data. For example, the population of E. Coli in a set of samples, which may be tied to the nose length in the finished model.

*2) Bibliography:* [1] "Data Formats," [Online]. Available: http://www.metagenassist.ca/METAGENassist/faces/Docs/Format.jsp. [Accessed 6 November 2015].

[2] “The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information,” Psychological Review, vol. 63, pp. 81-97, 1956.

*3) Overall Description:*

*a) Product Perspective:* This project interacts with several other components. It reads and interprets supported data file formats. The minimal acceptable set of supported formats includes only the tab separated “Classic QIIME OTU” format [1], as it is sufficient for use. Expanding the set of supported file formats is a stretch goal.

Additionally, we include a UI component to facilitate user interaction. This contains several components.

- A screen in which the user will direct the application to a data file, a visualization profile which allows the user to configure which attributes are associated with which model features, and optionally a metadata file, containing groups of samples to aid in selection.
- A screen in which thumbnail images of the models will be drawn. This screen would also allow users to add and remove models from a group, and select models to be compared in more detail.
- A screen in which the selected models will be drawn. At this point, the user would be able to rotate, pan, and zoom around the models in order to better compare their features.

We recognize that certain additional features, while not absolutely necessary for a minimal functional product, would enhance usability, and therefore we add, as stretch goals, shortcuts to certain common camera angles, some way to save and load visualization profiles, and a way to export comparison images. Via our UI, a user from our target user group should be able to use our tool to produce their visualization in 5 minutes or less with approximately 2 minutes or less of initial instruction.

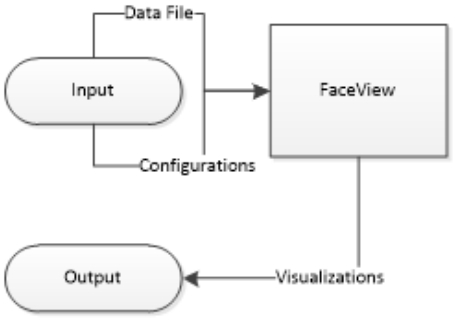


Fig. 2. A Functional Diagram, demonstrating the interactions between the user and the application.

*b) Product Functions:* The project should support the following functions

- The project should allow the user to select a subset of models to be drawn larger. Note that, in order to allow the user to easily identify similarities and differences in the drawn models, the number of models that can be drawn to the screen, and thus samples that can be represented at once, is limited to 6, since the maximum number of things a person can process at a given time is $7 \pm 2$ [2].
- The project should also support a thumbnail view to select models to view in more detail, and to add and remove models from groups.
- Similarly, the project should allow the user to select a subset of organisms to associate with certain module parameters, as those are limited by the number of model parameters that can be modified.
  - Furthermore the user should be able to choose which supported model parameters correspond with which organisms in the data file. This allows the user to allocate more obvious model features to the organisms he or she is most interested in.

- The project should, using the data files and visualization profile, generate models and draw them to the screen.
- Once these models are drawn to the screen, the user should be able to rotate, pan, and zoom the views. The thumbnail view is exempt from this requirement, as that would make initial screening difficult.

*c) User Characteristics:* Our intended users are researchers who are analyzing population data, specifically microbial population data. Such users are highly educated, but do not want to have to use unnecessarily complex or difficult tools.

*d) Assumptions and Dependencies:* We assume that Windows will be installed on the machine running the project. We also assume a version of Python 3 is included.

*4) Specific Requirements:*

*a) Sample Subset:* The project should allow users to select a subset of at most 6 samples to compare in detail. This is to improve user-friendliness. Restricting the number of models shown may reduce distractions, allowing users to better compare certain samples.

*b) Grouping:* The project should allow users to organize samples into groups for easy profiling of features or comparing subsets of the data. Users should be able to add or remove samples from a given group.

*c) Organism Subset:* The project should allow users to select a subset of organism appearing in the samples. This subset is limited by the number of model parameters which we can support, which should be at least 150.

*d) Model Parameter Assignment:* The project should allow users to specify certain model parameters to correspond to certain measurements in the data file, such as the population of a certain microbe corresponding to eye shape. This is to improve user-friendliness by allowing users to match parameters used in an existing result, or to allow certain features to be made more obvious.

*e) Classic QIIME OTU Format Support:* Initially, the project should support the tab separated "Classic" QIIME OUT format for data files. It is human readable as it looks like a table when viewed in a text editor, which makes it a good choice for a proof of concept. Expanding file format support is a stretch goal.

*f) Transforming Data:* The project should transform the data from the data file into a format that can be used to generate the models. This allows us to produce a standard format for model generation, which allows us to expand supported file formats.

*g) Producing a Model:* The project should produce a 3d human model based on the parameters from each selected sample. For each model generated in the batch, the same data parameter corresponds to the same model parameter. For example, if the population of a certain microbe in a certain sample corresponded to the straightness of the produced model's nose for that sample, then the population of the same microbe in a second sample would correspond to the straightness of its model's nose. This process should not take longer than 30 seconds.

*h) Drawing Models:* The project should draw the produced models to the screen such that each of the models are viewable at the same time. This is to facilitate the user comparing models.

*i) User Controls:* The project should allow users to rotate, pan, and zoom around the models in order to better compare them. All models should rotate, pan, and zoom.

*5) Additional "Stretch" Goals:*

*a) Provide a Web UI:* In addition to the local version, this would provide a web UI which would invoke the component that generates models on the server. These models would then be sent to the client browser

to be presented to the user. This stretch goal would necessitate that the project be portable enough to run on a Linux server.

*b) Export Images for Later Comparison:* This feature would allow users to not only compare samples in the same file, but also to compare to files compared earlier. However, this would allow the user the opportunity to associate different data parameters with different model parameters between the two results, which could lead to misleading results.

*c) Save and Load Visualization Profiles:* This feature would allow users to save and load visualization profiles, configurations of which samples and organisms from a given file are to be used when generating the model, and which organism corresponds to which feature.

*6) A – An Explanation of Prerequisites Apparent in the Preliminary Gantt Chart:*

- Exploring File Formats – This is time dedicated to choosing a data file type and understanding how to interpret one of those files. It has no dependencies.
- Explore MakeHuman API – This is time dedicated to learning how to use the MakeHuman project to create Human Models programmatically. It has no dependencies
- Load File – This task is to write code to load a data file, and translate it into a standard format. Its prerequisite is to understand a file format.
- MakeHuman Wrapper – This task is to write a wrapper around the MakeHuman project so that we can send the parameters included in the standard parameter format into the project to generate a model. Its prerequisites are to explore the MakeHuman project and to generate the set of standard parameters.
- Basic UI – This task is to create a basic UI through which the user will interact with the other components. Obviously, this will need to be connected to those components so one will be a prerequisite of the other. However, this one has no prerequisites because we need to be able to start somewhere.
- Load and Draw Model – This task is to be able to load and draw a model generated by the MakeHuman project. It's prerequisite is a UI to draw it in.
- Rotate, Pan, Scale Model – This task is to be able to apply the specified transformations to the model, allowing the user to better examine it. It's prerequisite is to be able to load and draw the models.
- Expand File Type Support – This stretch goal is to support more file types. As it is a stretch goal, we would not work on it until we have satisfied the rest of the goals, therefore its prerequisite is to be finished with the rest of the project, represented in the Gantt chart as tasks that are otherwise at the end of a prerequisite chain.
- Polish UI – This stretch goal is to make general cosmetic and user friendliness related improvements to the UI. As another stretch goal, its prerequisites are the same as the other's.