

*2*

****

Technical Design Document

**Assembled by:**

Jackson Oberkirch

Micheal Swiger

Timothy Pease

Connor Underwood

Seth Guerra

Matt Jung

3/3/20

**Table of Contents**

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. **Overview**    1. Software Tools |  |  | | | |
| 1.2 Coding Guidelines |  |
| * 1. Technical Proficiencies |  |
| 1. **Python**    1. Open CV |  |  | |  | |
| * 1. OSC |  |  | |  | |
| **3. Max**  3.1 Data Processing |  |  | |  | |
| 3.2 Sectional Development |  |  | |  | |
| 3.3 Sampling |  |  | |  | |
| * 1. Front-End |  |  | |  | |
| 1. **Delivery**    1. Live Installation    2. Website |  | |  | |
| **5. Milestones** |  | |  | |
| * 1. Vertical Slice (Alpha) |  |  | |  | |
| * 1. Red-Stick (Beta)   5.3 Final Website |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |



**1. Overview**

**1.1 Software Tools**

**Trello**

Trello is an online cork board that allows users and managers to create tasks, notes, and projects with deadlines and other features that allow for easy delegation of assignments. The Pet Sounds team uses Trello for project management; dividing up the project into smaller tasks allowed the team to quickly and efficiently delegate work and meet deadlines. Trello will aid us in our daily scrum meetings and will be used to organize our team meetings.

**GitHub**

GitHub is a massive online code repository run by Microsoft and can be freely used to publish  code. Pet Sounds utilizes this to share working code between developers and track changes made by each individual.

**PyCharm**

PyCharm is an integrated development environment designed for coding in Python. It provides many tools that allow Python to be easily run, including support for third-party libraries. This support allows the use of OpenCV and OSC that Pet Sounds takes advantage of.

**Discord**

Discord is an online text and voice communication platform. Originally built for gaming, it provides an ideal environment for sharing documents and discussing ideas. Discord is used considerably with the creation process to organize meetings and ideas when the team could not be together in person.

**1.2 Coding Guidelines**

The team stuck to contemporary standard coding guidelines that dictate consistent naming conventions, logic structures, and clean code. Beyond this the team behind Pet Sounds practiced agile development. Agile purports a cyclical production management strategy by which tasks are continually created, implemented, and evaluated. The team utilized this to its fullest extent, performing a scrum at every meeting to ensure proper project development speed.

**1.3 Technical Proficiencies**

The Pet Sounds team is comprised of four Computer Scientists and two Musicians. Despite the listed concentrations, the entire team had experience in programming and almost all had worked with Python in the past. Additionally, the musicians brought crucial knowledge of Max 8 and common data protocols that drastically increased what could be done within the scope of the project. The plethora of Python programming experience combined with musical talent provided a proper balance of technical and artistic proficiencies to create Pet Sounds.

****

**2. Python**

**2.1 OpenCV**

      OpenCV is the defacto open source computer vision library for python. Its documentation is extremely fleshed out and the library as a whole is very flexible. The library was discovered through research into similar digital media projects such as “Fish Plays Pokémon” that made use of it. The computer vision pipeline with OpenCV involves object recognition with the use of a constructed haar cascade, imagine resizing and manipulation, and parameter generation.

**2.1.1 Object Recognition**

OpenCV provides several tools for simple pattern recognition, motion detection, and object recognition. Pattern recognition allows OpenCV to look for multiple identical objects within a frame based on a source image given a certain threshold of accuracy. Pattern recognition ended up being discarded due to the potential complications resulting from the enviable multiple directions the fish could be facing, which pattern recognition fails to allow. Motion detection involves using a comparison algorithm involving the last few frames that the capture has taken in. Thus given a threshold of differential OpenCV decides what is considered motion. Despite how flexible of a solution motion tracking would be, it would lead to false positives from other things in the frame (such as seaweed swaying the tank). Therefore object detection, which is detection based on a single positive source image, was chosen as the default method of detection. Object recognition through OpenCV allows for the use of a provided or generated Haar Cascade, which is essentially the object that is meant to be tracked.

**2.1.2 Image Manipulation**

OpenCV supports several different methods and tools for modification of the capture. One of the most important forms of manipulation is converting the capture to grayscale. This is a simple form of image normalization that removes a large amount of noise from the image. The other image manipulation method that provides vital to the project is image scaling and drawing.  Due to the computation complexity of object detection, a low resolution video capture is necessary. Through the provided image scaling methods with OpenCV, we are able to reduce this rasterization bottleneck and ensure there are minimal frame drops. We are also able to draw simple geometry onto the frame such as bounding boxes and circles over the detected object.

**2.1.3 Haar Cascade**

Cascade classification is a sub-branch of object recognition/classification within OpenCV.  We are able to use already created cascades and feed them into OpenCV manually. However, due to how unique our object is, we intend to create a unique haar cascade. This is accomplished through a process of gathering thousands of negative images (images that do not contain at all the object in question), and a single positive image of the object we want to track. Through OpenCV, we are able to superimpose the reference positive image on top of each negative image with various degrees of rotation, tilting, and scaling. After the very time consuming machine learning execution, OpenCV produces a single haar cascade in the form of an XML file. We are using scaled down images of goldfish as the reference image for generating the test cascade, and will use multiple pictures of our fish from multiple directions to generate the final cascade.

**2.1.4 Generated Parameters**

From the object recognition using the constructed haar cascade, we will gain our two initial parameters, the x and y, at the center of the detected object. The rest of the parameters that will be involved in the music generation will be derived from these two initial points. This includes the animals position relative to a grid, the velocity of movement, direction of movement, edge overlap, and lastly if the animal has ceased movement for a certain period of time.

**Math:**

[ X1 / (n) , Y1 / (n) ] = position on grid, where n is the size of a grid section.

[ X2 - X1 / absl(X2 - X1) , Y2 - Y1 / absl(Y2 - Y1) ] = direction of the fish’s movement

sqrt(  (X2 - X1)^2 + (Y2 - Y1)^2 ) ) / # of frames between X1 and X2 = velocity of fish

V2 - V1 / # of frames between V1 and V2 = acceleration of the fish

**2.2 Open Sound Control (OSC)**

OSC is a protocol for sending messages to computers and multimedia devices across a network. OSC is formatted so that each message can easily be parsed by the receiving device. The structure of the OSC message provides for an incredibly versatile and accessible message in which the user can define and route any parameter in the form of integers, floating point numbers, and even strings. Because the messages are sent as bundles, they occur almost simultaneously. These aspects all make OSC the perfect protocol for sending data from our Python script to Max 8. To make use of OSC, we have to use User Datagram Protocol (UDP) to send OSC packets through a client. Since we are doing this all on one machine, we can use the localhost 127.0.0.1 IP address and specify an open port where we send the packets. The library Python-OSC provides the ability to easily bundle packets of data together, create an accessible data structure, and create a client server.

**2.2.1 User Datagram Protocol**

User Datagram Protocol (UDP) is a network protocol that does not require a “handshake,” or a confirmation between anyone using a network. In our case, error checking and confirmation are not crucial for the delivery of the OSC packets between the Python script and Max 8. Max will simply listen for UDP messages on the local host and specified port. In this case the IP address is the local 127.0.0.1 on port 5005. The only thing UDP needs is an IP address and a specified port number, which is much more appropriate for the scope of our project than other protocols like Transmission Control Protocol (TCP), which requires a confirmation to set up end-to-end communication.

* + 1. **Local Host**

In order to set up a client to send OSC packets, we need to use the 127.0.0.1 IP address and specify a port number (5005) on which the packets will be sent. In order to do this, we include a section in our Python script that makes use of the functionality of Python-OSC to simply create a client on the local host at our specified port. In python, we create a server that contains the address and port number. Then, we send a message 30 times a second, that contains the x and y coordinate information inside the UDP packet.

****

**3. Max**

**3.1 Data Processing**

The data sent to Max is derived from the coordinate position tracked by OpenCV. In the Python script, the data is preprocessed to obtain positional data related to the coordinates. We intend to send the raw X and Y position, the quadrant data, the velocity, and a stop command into Max, which will process the data using objects such as scale, change, and select. The data will determine musical qualities such as pitch, note length, and quantization value, and it will also be used to develop sections through the work by triggering checklists.

**3.2 Sectional Development**

Max will also be used to generate sections throughout the work, which will create interest for the listener. By manipulating and retaining coordinate data in objects such as counter, select, and call, we can trigger sectional changes throughout the piece. The different sections will entail cycling through various ambient soundscapes, changing the applied special effects, and varying the rhythmic and harmonic content of the piece. A large part of this will be determined by sample use.

**3.3 Sampling**

Max also has the capability to store and playback samples in objects such as buffer~ and play~. Using this feature, we will create ambient backgrounds upon which the fish will improvise a melodic line. In keeping with the vaporwave aesthetic, the samples will be derived from short cassette tape loops created and sampled from old television advertisements.

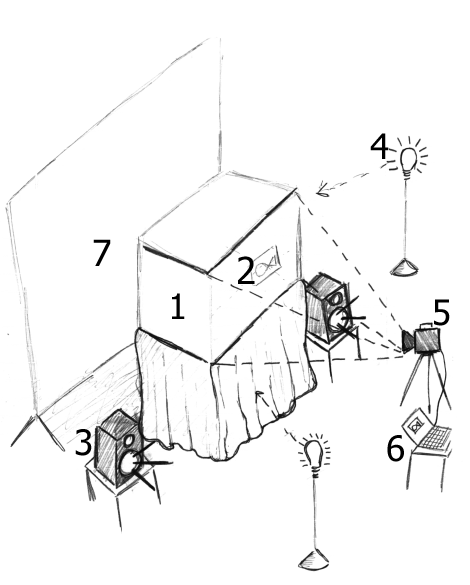
**3.4 Front End**

In the end, the “presentation mode” feature of Max will be utilized to create a simple user-friendly interface that clearly explains and exhibits the data being processed by Max. Using objects like panel and slider, the values sent to Max will be clearly visible to the audience. Additionally, the presentation will include the option to turn the patch on and off using the object ezdac~ as well as the option to change the output volume using gain~.

****

**4. Delivery**

* 1. **Live Installation**

****

1. **Fish Tank**

The fish tank will be a standard sized, rectangular aquarium, around 20in x 12in x 10in. The tank will be set up to have the least amount of objects as possible, in order to allow the camera to focus on only the fish.

1. **Fish**

The fish itself will be an average goldfish that can be obtained from pet stores. The fish’s front, back, and sides will be used to set up haar cascades for image recognition before the event starts.

1. **Speakers**

There will be two stereo speakers placed adjacent to the fish tank hooked up to the computer, allowing for a surround sound listening experience.

1. **Lighting**

There will be standard LED lighting diagonally positioned to the fish tank. These lamps facing towards the fish tank will allow for optimal lighting to make it easier for the camera to capture the required footage.

1. **Camera**

The camera we will be using is a Logitech HD 1080p webcam. It records at 30 frames per second. It will most likely be attached to the top of the laptop, facing the fish tank.

1. **Computer**

The computer will display the footage of the fish along with a motion capture program pulled up on the screen for the audience to look at the program in action in real-time.

1. **Backdrop**

The backdrop for the installation will be a flat, mono color curtain to further assist in providing a background that assists in creating a flat background that makes it easier to capture the fish with the camera.

* 1. **Website**
     1. **HTML / CSS**

The site will be a simple HTML page that has a header bar and a scroller that can be used to switch animal compositions. When a different animal is selected the page will fetch the related composition from the YouTube repository and begin playing it for the user. The website will have links to the YouTube videos inside the html and allow users to play multiple videos at once and easily swap between the “instruments” of different animals.

* + 1. **YouTube**

YouTube will be used to host all music compositions created throughout the project. YouTube serves as the ideal hosting spot as it has a good interface of its own and can be easily embedded into other web pages. YouTube allows us to save on server costs as well. Pet Sounds uses this to its advantage and links a display of a user’s selected desired composition. The user will have the option both on the Pet Sounds website and YouTube of playing any and all pet sounds.

* + 1. **Syncing system**

The website syncing system will consist of multiple buttons that allow the user to select specific “instruments” of specific animals and add them together to create compositions. The videos will load simultaneously using an AJAX request. All videos will be at the same time, thus playing the composition.

****

**5. Milestones**

**5.1 Vertical Slice (Alpha)**

The vertical slice will demonstrate that the technology can successfully be implemented. Perhaps most importantly, it will show that the different programs can all communicate with each other. The alpha will use OpenCV to track and generate the positional data from a preliminary face-tracking program. Additionally, the Python script will pre-process some aspects of the data so that related data can be obtained, such as the quadrant location of the user’s face. This data will then be formatted and sent to a local client using Python-OSC. This data will then be received using the unperceived object in Max, and there it will be manipulated to control a few different parameters of music. In order to test for the full functionality of the final design, the alpha Max patch will test that the pitch and note duration can change, that samples can be manipulated and played back to create sectional development, and that a user-friendly presentation can be created.

* 1. **Red Stick (Beta)**

The Red Stick Festival is an annual event hosted by Lab for Creative Arts & Technologies (LCAT) in Baton Rouge that aims to show projects that aims to “demonstrate the linkage between creativity and technology”. There are numerous demonstrations at the event that exemplify this goal. Speakers and workshop leaders visit to participate in the event from high profile companies such as Disney, Pixar, DreamWorks, Digital Domain, and Sony Imageworks. Red Stick features both a diverse art display and a chance for students to demonstrate a commanding knowledge of digital media art and technology.

The Pet Sounds Red Stick demonstration will feature the fish tank installation described above. Attendees of Red-Stick will be able to view a live composition of Pet Sounds generated from a real fish. The installation consists predominantly of the tank with moving fish and speakers playing generated sound. Audience members will also be able to see some of the live processing that occurs in order for the computer to generate the music. The sound and captured footage from the entirety of the event will be archived on the Pet Sounds site for future and repeat listeners.

* 1. **Final Website**

The final manifestation of Pet Sounds will be a website that hosts the Red Stick demonstration as well as a variety of other compositions from different animals. A user will be able to change videos and animals to hear a unique composition from each. The site will use YouTube to host and store these videos in advance as the generation process would be too much for the average user’s computer to process. Pet Sounds fans will be able to relive their favorite pet’s composition on the site for years on end.

****

**6. Resources**

**6.1 Python**

[**https://www.python.org/**](https://www.python.org/)

**6.2 OpenCV**

[**https://opencv.org/**](https://opencv.org/)

**6.3 Open Sound Control**

[**http://opensoundcontrol.org/**](http://opensoundcontrol.org/)

**6.4 Max 8**

[**https://cycling74.com/**](https://cycling74.com/)