

Fleeting Signals

Media Technology MSc, Leiden University
Sound, Space and Interaction course
Dewi Becu & Reyhan Danisan

In this report, we will briefly document the development of the final project for the Sound, Space, and Interaction course. We will detail the concept, development, and realisation of a Python and Plugata-based system that uses hand gesture recognition to generate immersive soundscapes.

Concept

Our project's concept revolves around the interactive creation of an auditory landscape through the use of specific gestures, each linked to a unique sound that symbolises 4 different elements:



Infinity

Water



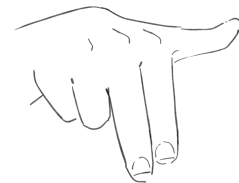
Malevolent

Thunder



Maho

Wind



Fuga

Fire

“Maho” for wind, “Infinity” for water, “Malevolent” for thunder, and “Fuga” for fire.

The users are able to manipulate the prominence of these sounds by repeatedly making the corresponding gesture; the more a gesture is held up, the more dominant its sound becomes.

Part of the background noise adds a layer of suspense, increasing in intensity as a cue for the user to switch gestures. Failing to change the gesture in time results in the cessation of all sounds, followed by a brief pause before the soundscape restarts. Initially, only the sounds that are introduced by the user's gestures are audible. For example, if the gesture “Maho” is not shown, its sound will remain absent until the gesture is made. Only after the gesture has been shown will it gradually be played in the background until the user holds the gesture for a longer time, making it more prominent.

Development

The development of our interactive sound system began with the integration of Python's MediaPipe library to track hand gestures. These can be found on:

- <https://developers.google.com/mediapipe>
- <https://github.com/kinivi/hand-gesture-recognition-mediapipe>.

This initial step involved training the model for the system to recognise specific gestures that corresponded to the elemental sounds. The model would record our hand, linking data to the different joints of your hand in order to map it. After this, it gets rid of the images leaving behind a lot of data, which can be used by anyone, regardless of environment or hand differences, because of this acquired data. Following the successful implementation of the hand gesture recognition, we used OSC (Open Sound Control) communication to integrate the data of the gestures with PlugData.

After we got the gesture recognition down, we started developing the logic of the project. A few main points of focus behind the logic were:

- Singular tone pitches that go up and down in volume according to each gesture
- The same system is utilised for the elements
- The suspense game, in which the suspense builds over time insinuating that the user has to change gestures
- The chord speed works primarily on the object "drunk", however depending on how long you keep a gesture, it will either speed up or speed down the chord
- Specific gestures will move the chord progression up or down
- The core logic behind the phasing and intertwining of all these layers comes to volume control. The system keeps track of how long and how many times you have used a specific gesture, and this data/info will be used in the "random" object, to output the numbers that will be used for the volume. For example: the wind's total gesture time and appearance can lead to the volume being at 50%. A maximum range of 50 is put in the "random" object, which could end with wind having an influence of 23%.

Following this we implemented the elemental sounds which would make up the remaining part of our project. The patches for these sounds were inspired by the code provided by Andy Farnell in his code examples. These can be found on <http://aspress.co.uk/index.html>. After tailoring the sounds and values to reflect our vision, the project was mostly finished aside from the final changes. A few patches from https://www.evdh.net/media_technology/sound_space_interaction/ was also utilized, such as the plucking sounds and OSC communication.

Realisation

During the realisation phase, our concept and development came together, forming an actual functioning system. We had to do a lot of debugging and testing to ensure that the gesture detection was accurate and responsive in different types of settings. This involved excessive training in different positions to ensure the right data was extracted. After acquiring the data for the gestures with Python and Mediapipe, the connection to Plugdata was successfully implemented.

Each elemental sound was adjusted for volume, clarity and harmony within the overall soundscape to achieve an immersive auditory experience. Adjustments were made based on iterative feedback to ensure the transitions between sounds were smooth and that the buildup of the suspenseful background noise was effectively used to give the users the cue to move on to another gesture at the right moment. By making use of the plucking sounds and the chord progression we created multiple levels of depth within the output, keeping the creation novel and interesting throughout the whole experience.

Setting up the space did not have many requirements, as our project is adaptable to different types of environments. It only needs a laptop that can output sound to run the project, together with a camera. During the setup for the demonstration of our project, we used one laptop, one built-in webcam, and two speakers that were present in the classroom. You do not need a working screen for this project, as it is obsolete.

In order to register the gestures, a person has to stand in front of the camera and alternately make one of the four gestures. The interactive dialogue between the system and the user is based on the hand gesture the user makes, and therefore, the system reacts with elemental and/or chord sounds. Often, the system nudges the user to change their hand gesture, influencing the system to create other sounds.

Group work

It was a good learning opportunity and experience to connect the hand recognition in Python to PlugData, since we have never done anything with Python and machine learning before. It was challenging to try and understand how to work with Python and get everything to work accordingly.

During this project, Dewi worked on Mediapipe and the core logic of Plugdata. Reyhan worked on gathering the proper sounds that would use the core logic and produced the written works.