DeejAI - Melody generation assistant interfaced through gesture-controlled display

Objectives

- Define a gesture-controlled mechanism to control an application interface
- Define music generation functionalities for a melody generation assistant
- Develop a gesture-controlled interface to interact with the application

Work organization

Throughout the whole project, three main work axes can be identified. Each of them relates to a specific work area.

UX Research

The analysis of the users, definition of functionalities, basically anything related to the experience of the user while using the application, and the ways it interacts with the provided functionalities are scoped here.

UX research	Define users
	Define user needs
	Define requirements/functionalities
	Define information architecture
	Define interaction design
	Define components library
	Define interface

Table 1. Tasks of UX research

Table research

There is a need to properly identify and define an environment that aligns with the functionalities the application will provide as well as the limitations in hardware and budget. Given this, the following tasks are defined

Table research	Camera Projector calibration
	Hardware setup
	Object tracking
	Hand gesture recognition
	Define possible input methods

Table 2. Tasks of Table research

Model research

Strives toward achieving the application's main goal, providing a mechanism to generate a melody, and music according to certain inputs. This will go as an experimentation process required to assess the models defined to support properly the functionalities of the application.

Model research	Research SoTA
	Determine possible features(frequency domain)
	Models and datasets for selected features
	Implementation and testing

Table 3. Tasks of Model research

State of the art

Music generation research has a long history (dating back to the 1950s). Many methods have been tried, with various amounts of success. However, this task proves to be challenging even for nowadays AI. The main challenges are inconsistent evaluation techniques, problems with the long-term structure of the generated music pieces, and a lack of creativity.

Even objective evaluation of human-made music is difficult - we all have different music tastes. With computer-made music, we can measure the statistical properties of the song and compare it with the human-made ones, but that’s just a baseline. There are many different methods of measuring the performance of music generation models in the research papers, but the field doesn’t have a measurable benchmark yet. The most reliable method so far is a user study, which can be time-consuming and expensive to perform.

By the long-term structure of a song, we mean the main theme or melody of the song (for example, repetitive chords). The models either gradually start generating, what appears to us, as a completely different song, or they are still producing the same chords with no new inducement (which quickly become boring).

Data-driven learning algorithms produce a distribution that is close to the provided dataset. However, we want to hear something new - that equals an outlier of this distribution. Also, creativity is difficult to quantify, and therefore it is difficult to train a model that optimizes it.

Methods overview

Four main categories of methods have been experimented with for music generation:

- **Probability models** - they model music as a set of probabilities they have learned from the training dataset, which also means that they can only re-use patterns from the given data
    - Markov chains, Bayesian networks
- **Rule-based models** - often hand-crafted rules and algorithms, they are also style-specific
- **Evolutionary algorithms** - they have problem with defining the fitness function, also EAs work by combining pieces of information together, in music generation, it is troublesome to define these pieces
    - Genetic algorithm (ex. GenJam)
- **Neural networks** - nowadays the most used technique for music generation, from now on we will focus on them
    - RNN (LSTM), VAE, GAN, Transformer

Music generation tasks

Music generation can be divided into three levels of tasks: score generation, performance generation, and audio generation.

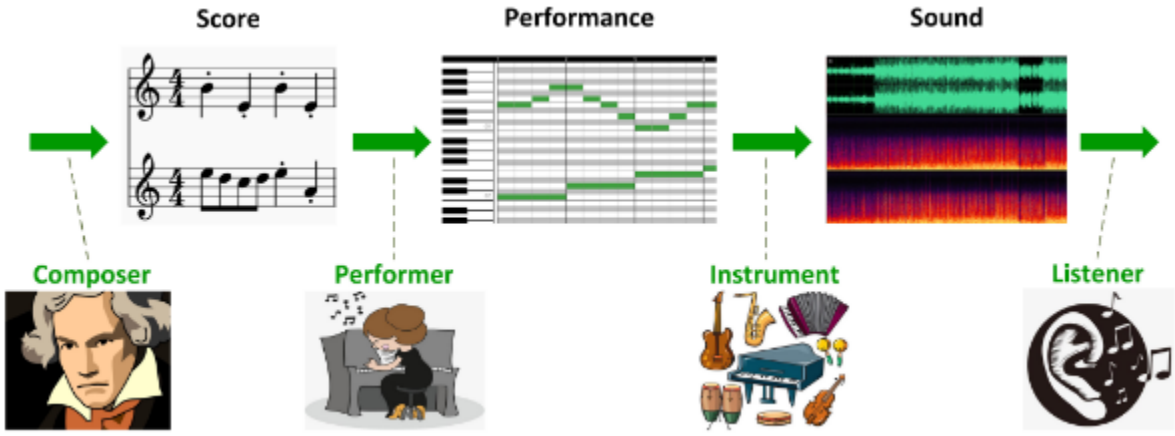

Figure 1. Process of music generation

Score generation

Process is similar to humans composing music. The output is a symbolic representation of music which includes information about a melody (sequence of tones for each instrument), tempo, and structure.

- **Monophonic** - melody for one instrument playing one tone at a time
    - RL-Tuner (2017) - sequential learning combining ML and RL, pre-trained RNN provides reward
- **Polyphonic** - one instrument playing multiple tones at a time

    - Music VAE (2018) - polyphonic music, good music reconstruction abilities, hierarchical decoder
    - MuseNet (2019) - transformer, based on GPT-2, 4 min tracks, 10 instruments, various styles, decoder predicts next token at each time step
    - PianoTree VAE (2020) - tree structure of music syntax (reflects hierarchical nature of music)
- **Multi-track** (monophonic or polyphonic) - multiple playing instruments
    - MusAE (2020) - reconstructs phrases with high precision, change specific attributes of songs

Performance generation

Adding information about individual musician style (a song played by two different artists doesn't sound the same), like tempo, timing, or dynamics

Audio generation

Generation of pure acoustic information in the form of waveforms or spectrum.

- Audio synthesis (WaveNet, DrumGAN)
- Singing voice synthesis (DCGAN-based, FastSpeech-based)
- Score-to-Audio
    - LSTM with the help of WaveNet-based audio generator
    - PerformanceNet (2019) - mapping between symbolic repr. of piano rolls and their audio repr., deep convolution model
    - Mel2Mel (2019) - polyphonic music, explainable embedding space

State-of-the-art models

- **MuseGAN:** Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment (2017)
    - Code: https://github.com/salu133445/musegan
    - Results: https://salu133445.github.io/musegan/results
    - Application: generate music from scratch, accompany given track
    - Trained on: Lakh Piano Roll Dataset (https://salu133445.github.io/lakh-pianoroll-dataset/)
- **Theme Transformer**: Symbolic Music Generation with Theme-Conditioned Transformer (2021)
    - Code: https://github.com/atosystem/ThemeTransformer
    - Results: https://atosystem.github.io/ThemeTransformer/
    - Application: finish the song from the given first few seconds
    - Trained on: POP909
- **SaShiMi**: It's Raw! Audio Generation with State-Space Models (2022)
    - Code: https://github.com/hazyresearch/state-spaces
    - Results: https://hazyresearch.stanford.edu/sashimi-examples/
    - Application: generate raw audio (piano, speech, music) from scratch, or conditionally

    - Trained on: Beethoven, YouTubeMix, SC09
    - Compared to: SampleRNN, WaveNet
- **FIGARO**: Generating Symbolic Music with Fine-Grained Artistic Control (2022)
    - Code: https://github.com/dvruette/figaro
    - Results: https://soundcloud.com/user-751999449/sets/figaro-generating-symbolic-music-with-fine-grained-artistic-control
    - Application: description-to-(symbolic)-sequence (ex. descr: note density, instruments, chords)
    - Trained on: LakhMIDI
- **Jukebox**: A Generative Model for Music (2020)
    - Code: https://github.com/openai/jukebox/
    - Results: https://openai.com/blog/jukebox/
    - Application: generates music based on a given genre, artist, and lyrics; prompt with own track
    - Trained on: created their own
    - Notes: on a V100, it takes about 3 hrs to fully sample 20 seconds of music - from our own experiments, this model is too slow for our needs

Interactive Table

The interactive table is not a novel idea, many have designed and developed different interactive table systems since the 1990s. Wellner [1] proposed a DigitalDesk system that can interact through paper. In its DigitalDesk system, a camera and a projector is placed over a table where the camera reads images on the table, a computer detects where the user's finger is pointing and then projects rendered images onto the table surface. Wellner developed a few applications with the DigitalDesk system, including a calculator that reads numbers from printed paper, a translator that translates French words on a printed paper to English words, and a paper painter which can draw digital paintings on a paper surface.

Later on, PlayAnywhere was introduced by Wilson [2] at Microsoft Research. It also includes a camera and projector but they are placed on the tabletop instead of over the table. The whole hardware system is more compact and thus more mobility compared to DigitalDesk where everything is fixed mounted. In this project, Wilson compared different setup solutions for digital-desk using a projector.

The top-down, rear, and table-top are three common approaches for projector vision systems. The top-down approach builds the projector and camera to be mounted overhead above the table. This installation often requires a precise adjustment of the projector to vertical alignment with the table. The distance of the projector to the surface can be adjusted to the required throw length of the projector and the focus length of the camera. The area of the projected screen is adjustable with the throw distance of the projector. The farther the distance between the projector and table surface the larger the projected area will be.

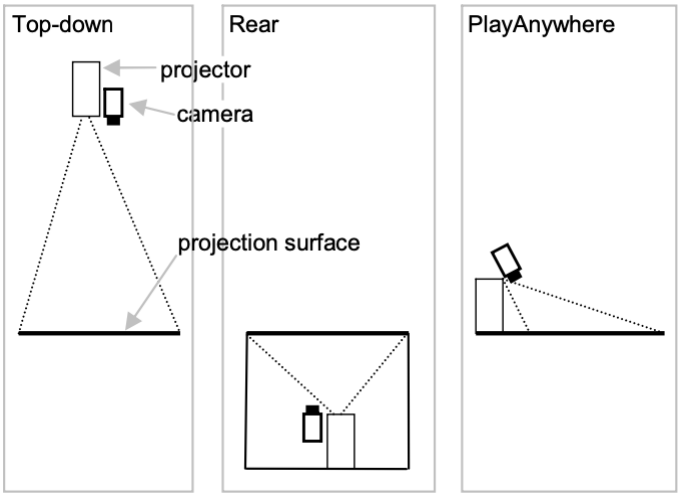

Figure 2. Three projector-vision solutions

The rear solution places the projector and camera under a special semi-transparent surface. It is difficult to capture a high-resolution image through the surface which will be used for object detection. A dedicated table is also impractical for the scope of our project. Another drawback is that the requirement of short-throw length and focus length is not easy to satisfy with ordinary projectors and cameras. The area of the projected screen is fixed which limits the UI.

The third solution is popular in recent years as we see many commercial projectors installed in tabletop mode (Hachi Infinite M1). This is the easiest and mobile way among the other setup solutions. And it does not require calibration since the distance and angle of the projector and the camera to the table surface is fixed. However, it requires an ultra-short throw projector.

For the scope of our project, the top-down and table-top solutions are ideal, and due to the long throw distance of the projector we had, we are choosing the top-down solution for our hardware setup.

Hardware setup

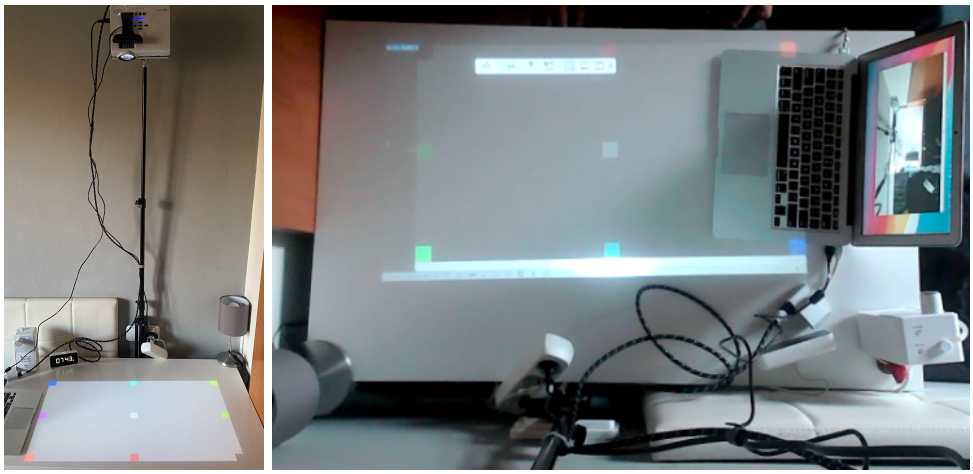

Figure 3. The top-down projector-camera(Left) and Camera View(Right)

For our setup of projector and camera, we used a tap to bind a web camera with a projector facing downwards to the table surface. The camera and projector are mounted on a tripod that stands aside from the table. The projector connects to a computer using an HDMI cable while the web camera connects to the computer with a USB cable. The following table shows the current hardware specifications and may be changed in future development.

Hardware	Specifications
Projector	Throw Distance: range from 1.4 to 4 meters Adjustable focus Lumens: 5500 Resolution: 720P
Camera	Resolution: 2M pixels, 1080P Focus distance: range from 0.3 to 5 meters Viewing angle: 85 degree
Tripod	Adjustable height: range from 0.5 to 1.5 meters

Table 4. Specifications of hardware, including projector, camera, tripod

Calibration

After setting up the hardware, a calibration is necessary to fix the problem that the image captured from the camera is larger than the projected screen as in Figure 2. The purpose of calibration is to transform the position in the camera view into the projected screen. As shown in Figure 3, the projected screen is in a distorted shape, we mark the position of objects inside the projected screen captured from camera view as P_{world}, and the related position in the projected screen as P_{proj}. A simple way is to use a mapping function to transfer each corresponding pixel from camera view to projected screen. Malik and Laszlo [3] proposed a solution for image rectification with equitation $P_{proj} = H_i^{-1} P_{world}$, where P_{world} represents pixel in world space and P_{proj} represents the pixel in the project screen and H_i^{-1} means the mapping matrix.

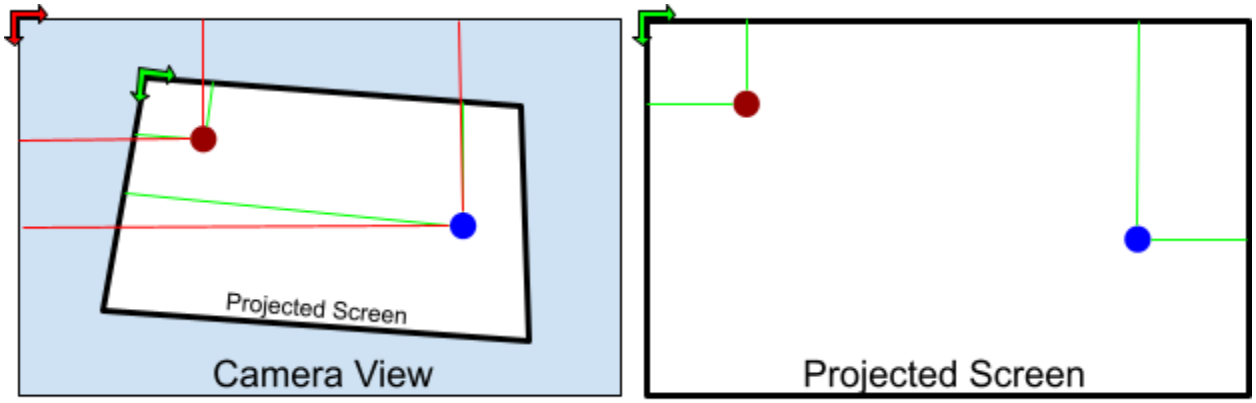

Figure 4. The camera view of the projected screen and the origin projected screen

Hand Gesture detection

A natural way to interact with the projector-vision system is using hand and fingers. Wilson [2] proposed a shadow based tracking algorithm to detect fingertip action. Based on the shape of shadow casted by the finger, the system can detect common user input like touch and hover. Alternatively, Letessier and Bérard [3] using automatic thresholding and shape filtering to detect fingertips.

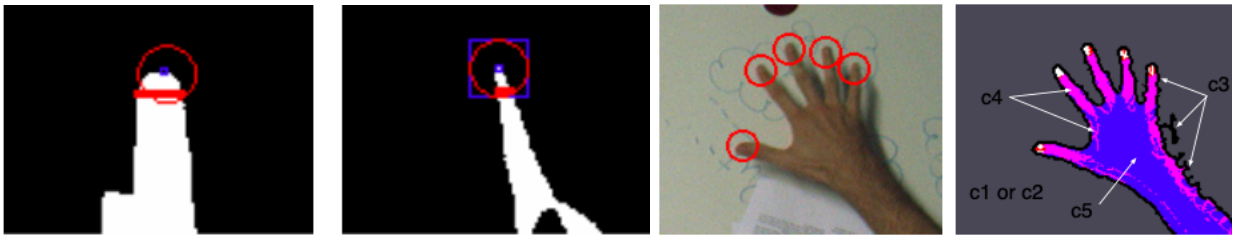

Figure 5. The shadow based fingertip tracking (Left) and shape filtering fingertip tracking(right)

For hand gestures, Wu and Balakrishnan [4] lists 10 distinct hand gestures including tap, double tap, flick, catch, flat hand, vertical hand, horizontal hand, tilted horizontal hand, two vertical hands and two corner-shaped hands.

Our approach will combine both classical shadow and shape filtering based techniques and recent RHO(Recon- structing Hands and Objects) technique [5].

Challenges

Given the different axes, the project has, it is important to be aware of the challenges that might come along the way.

- Synchronize I/O, table interface, and models to provide the desired functionalities
- Model definition carries a heavy load in the task. Being able to properly discriminate against the models that better fit our goal is crucial.
- Limitations in budget and hardware can hold the progress when developing the table setup
- Designing an effective and reliable way to recognize user inputs. For example there could be a difference between hovering over an option and actually intending to click it.
- Gathering a user pool to get feedback can be a hard task by itself, given the scoped nature of the application.

References

[1] P. Wellner, "*Interacting with paper on the DigitalDesk*". Communication of the ACM vol 36, n. 7, 1993, p. 87-96.

[2]A.D. Wilson,"*PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System*",UIST '05 Proceedings of the 18th annual ACM symposium on User interface software and technology ,2005.

[3]J. Letessier, F. Bérard, "*Visual Tracking of Bare Fingers for Interactive Surfaces*" UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology, 2004.

[4]S. Malik, J. Laszlo,"*Visual Touchpad: A Two-handed Gestural Input Device"* ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces, 2004.

[5]Z. Cao, I. Radosavovic, A. Kanazawa, and J. Malik, "*Reconstructing Hand-Object Interactions in the Wild*", arXiv e-prints, 2020.