A Video-based Virtual Instrument

Siyuan Wang sw5593 Zhuocheng Tao zt2246

1 Introduction

Our project is developing a video-based virtual instrument. The idea is inspired by an instrument called Theremin: an electronic musical instrument controlled without physical contact by the performer. The property that this instrument does not need physical contact is ideal as it does not change how to play it.

For the final decision, we decided to make a multifunctional instrument program not limited to Theremin. There will be three kinds of instruments: Theremin, “Air Key”, and keyboard. They will be introduced later in section 2--Program Description.

2 Program Description

2.1 Hand Tracking

To figure out the computer vision part, we take advantage of a Python library called “Mediapipe”. “Mediapipe” is a library developed by Google which has a function to identify the hand or body and return a relative coordinate of specific points. It can also show the landmarks of hands as the screenshot shows (which is optional):

As the developing period for the project is limited, we did not choose to train the model to identify

some complex, self-defined postures. Instead, we defined several restrictions of postures to use this program, which is common in most programs.

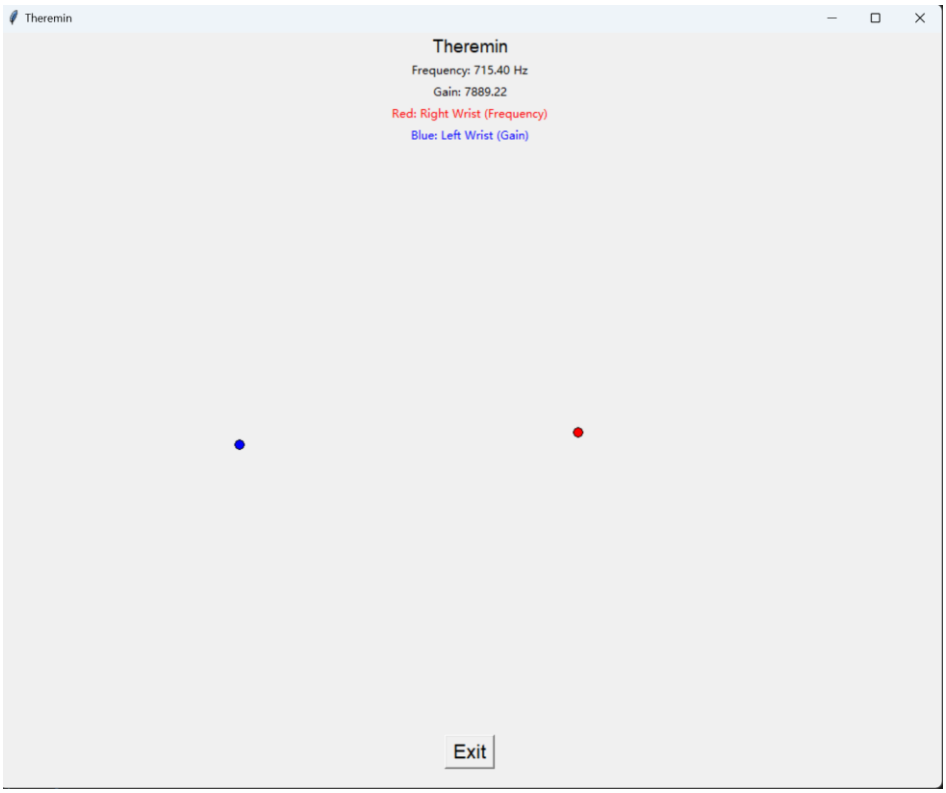
2.2 Theremin

To make sounds, we first need to know the operation of that instrument. The left hand is used for controlling the output volume, which will be called “gain” in the program. The right hand is used for controlling the frequency of the sound. The sound of Theremin is a sinusoidal function, so we can utilize some codes that we wrote earlier and use block processing.

Although current CPUs are quite fast, the loop in block processing still introduces noticeable delays, particularly in tasks like real-time video or hand-position tracking. These delays arise because the two types of processing are serially executed. To enhance performance and reduce these delays, we decided to make them parallel by using multi-thread techniques to make the processing of video and audio be done simultaneously. To develop a multi-thread program, we need to use a library called “threading” and separate the program into 2 functions, which are video and audio respectively.

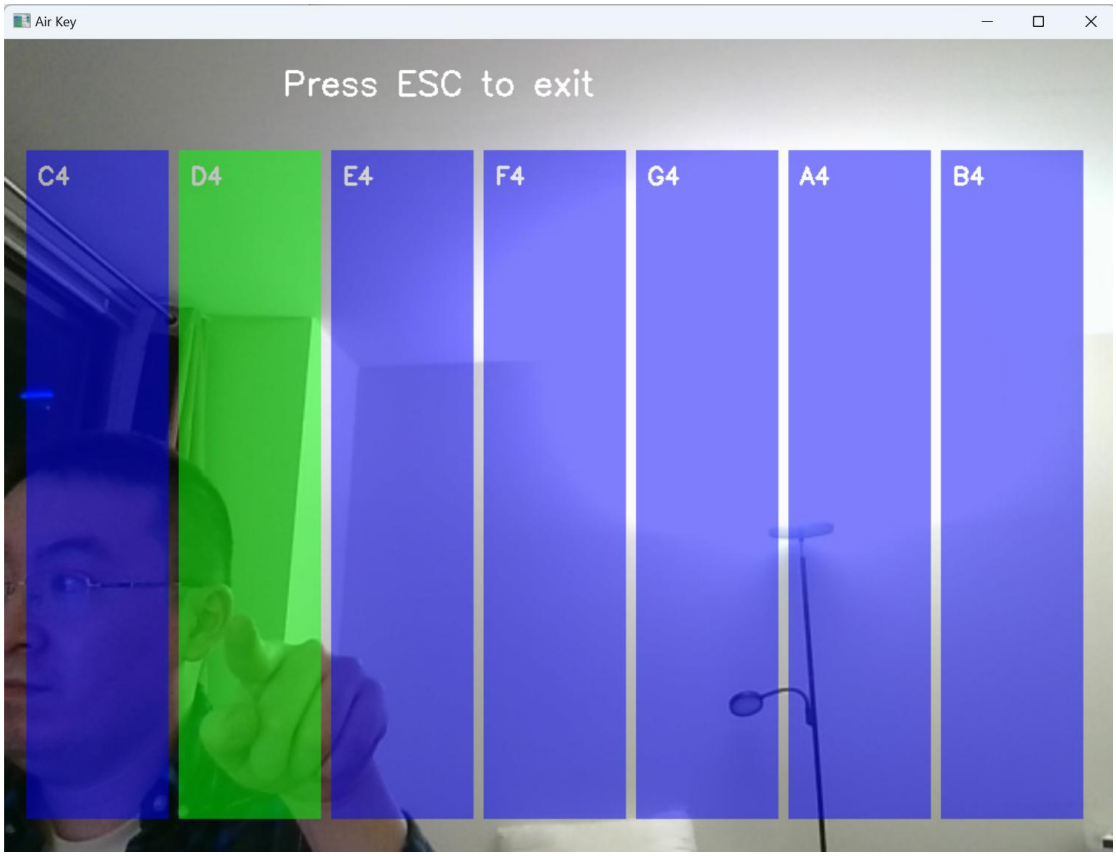
We do not want to display the raw video data for that section. So, to display the location of hands only, we used Tkinter. We got the X and Y coordinates from Mediapipe and used the coordinates to show points on a canvas. As we do not want a trace of movement, we also need to clear the canvas for each loop.

In the demo, you may find that there is a camera window. That is not contained in our final program. It is just for demonstration.



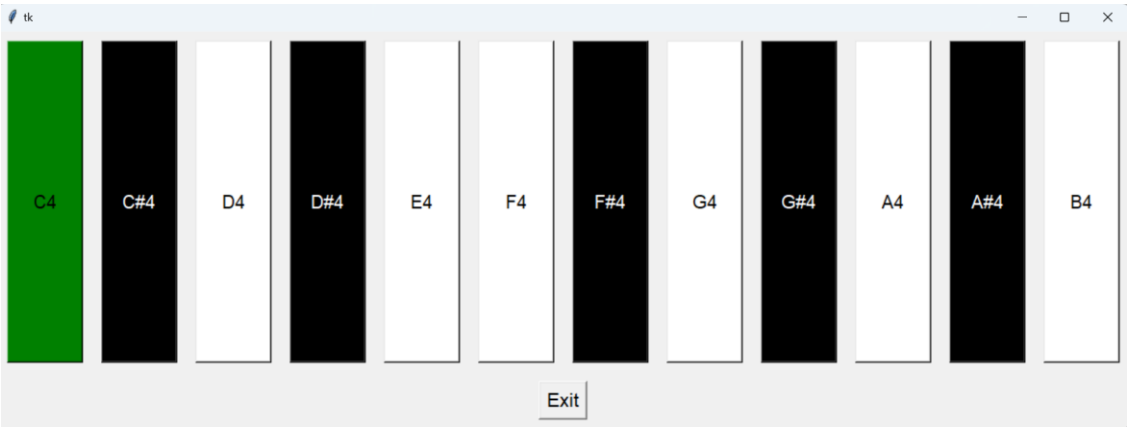
2.3 Air Key

Furthermore, we added another function for this video-based instrument. We used the idea of mapping the hand position to frequencies to play notes from C4 to B4, for example (maybe not the same as what we chose), if the finger has an X-coordinate between 0 to 0.2, then play the “C4” key. To judge if a key is pressed, we set a threshold of Z-coordinate we got by using Mediapipe. Also, to avoid the identification error, we set several thresholds including for press, release, and change keys. That way, it will not suddenly jump to other keys and mess up the sound. We named this instrument “Air Key”. When a key is “pressed”, it will turn green.



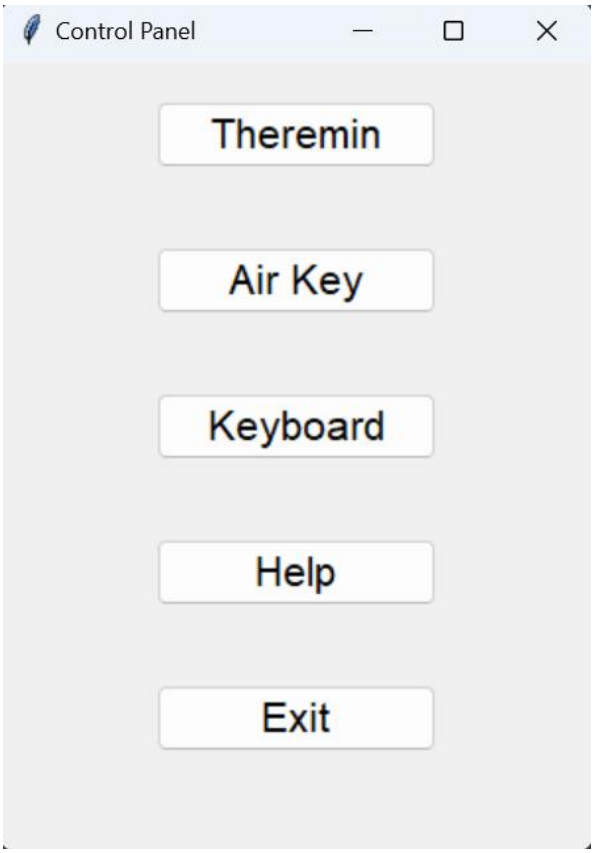
2.4 Keyboard

Also, we added a keyboard play function with GUI that shows which key is pressed, which looks like the black and white keys as a piano. It is mostly based on the demo from the course, so it is just an extra function. **Please click the popped-up window of the keys to avoid this program is not selected which may cause the program not to respond to your keyboard.** The effect of this function is shown below:

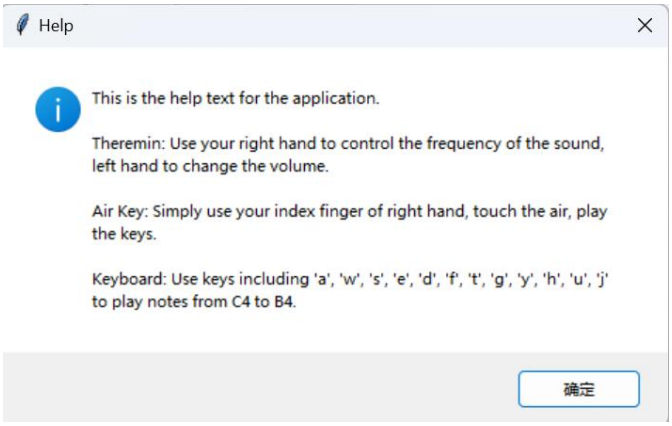


2.5 GUI

To manage these functions, we designed a simple GUI to let users select which kind of instrument, that we introduced, they want to play. The GUI is shown below:



To let new users be easier to use this program, we also wrote a brief “help” function by using a message box. The effect of “help” is shown below:



3 Conclusion

The program we designed works as we expected. Our project successfully demonstrates the potential of combining advanced computer vision and audio processing techniques to create an innovative, video-based virtual instrument.

Using Google's Mediapipe library for hand tracking proved to be a pivotal decision, enabling precise and real-time interaction without developing complex algorithms. Our decision to address processing delays through multi-threading techniques ensured a smoother and more responsive user experience, essential for the authenticity and enjoyment of playing a musical instrument.

There are still more improvements that can be made. The error of identification is still can be observed although it is reduced. Also, for “Air Keys”, there is another idea that we use the surface of the table as the keyboard, which is also can be seen in the advertisement of Meta Quest 3. In our project, as we do not have a separate webcam, we cannot do it now. For the keyboard, we did not design a function that can let us control the length of sounding and it cannot identify multiple keys at the same time to play a chord.