On-Screen Virtual Keyboard (OSVK)

Introduction

A virtual keyboard is a computer keyboard that a user operates by typing on or within a wireless- or optical-detectable surface or area rather than by depressing physical keys. Such a system can enable the user of a small handheld device. The term virtual keyboard is sometimes used to mean a soft keyboard, which appears on a display screen as an image map, and the user points and clicks on the map of keys to enter text. In some cases, a software-based keyboard can be customized. Depending on the host system and specific software, the user can use a touch screen or a mouse or a virtual method to select the keys.

Project Description

On-screen virtual keyboard (OSVK) is a simple hand gesturing character input system that is built on a computer vision-based library. The whole system is based on OpenCV and python libraries. A picture of a keyboard (figure 1.0) layout is displayed on a computer screen when the OSVK program run.

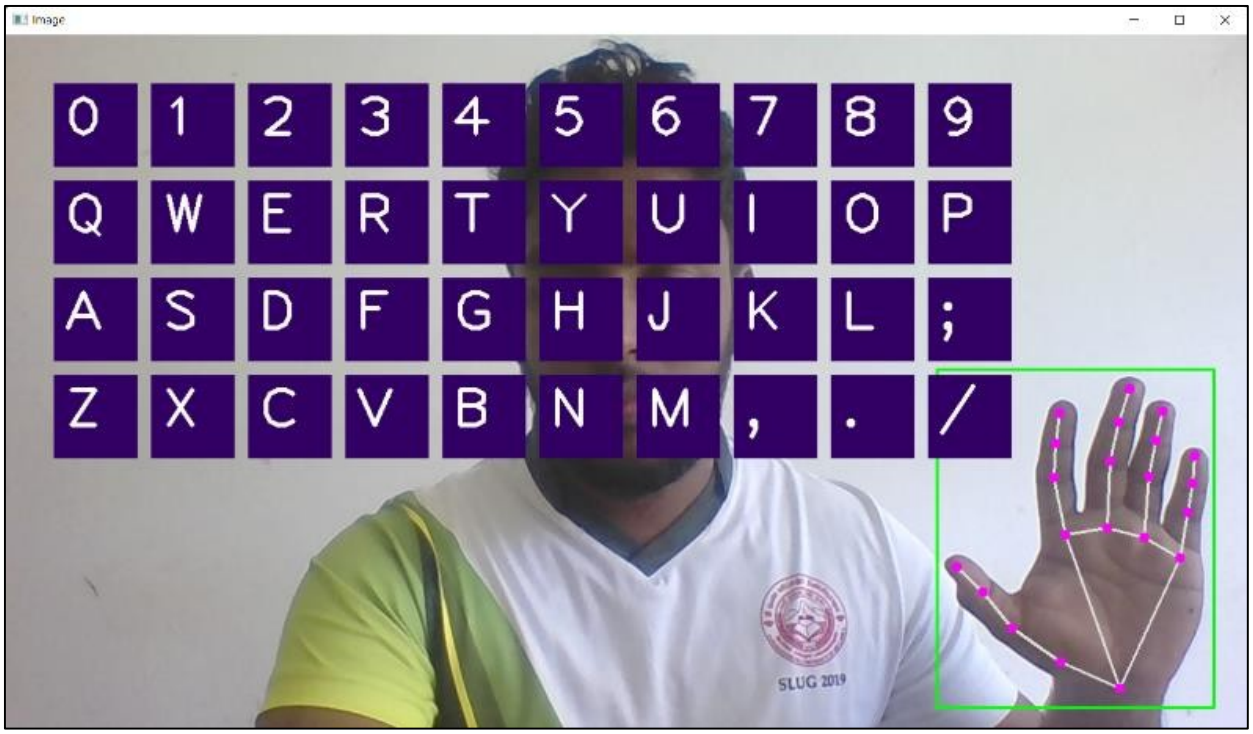

Figure 1.0 – Mapped keyboard layout & hand detect and track landmarks of hand by the mediapipe library

You should show your one hand to your web camera, no matter if it is left or right. after that system will recognize your hand and detect landmarks of your hand. (Figure 7.0)

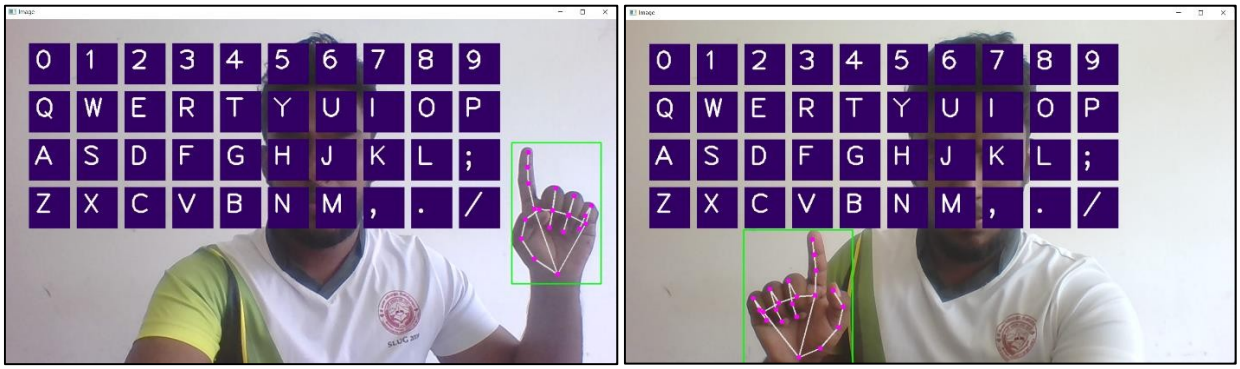

Figure 2.0 – Mapped keyboard layout and hand & landmarks detected

Figure 1.0 shows that mapped dark purple keys. Edge of each and every square keep track whether the middle finger tip and index finger tip points are inside their square edges or not. Because that's the way how to keep track which key is active and which key is pressed.

When you move your index finger tip through the mapped keys, the color of the key change to pink color. Which means that the key is active. At that point, if you bring your middle finger tip as well. The system track distance between your index finger tip (8) and middle finger tip (12). If the distance is less than 50, the system detects key is pressed. Then the button color will change to green color. When it is detected as a pressed key it will give the sound of the character via speaker. Because of the voice assistant feature included as well.

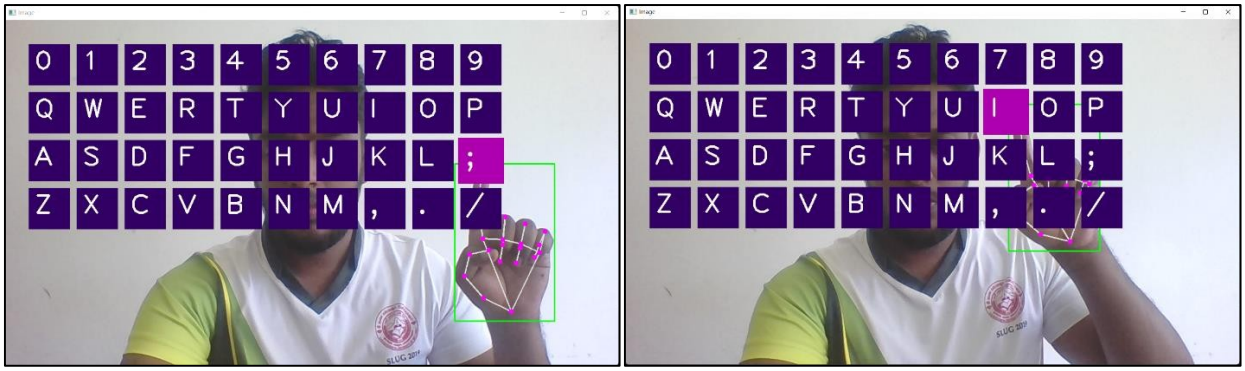

Figure 4.0 – Shows active button when finger moved

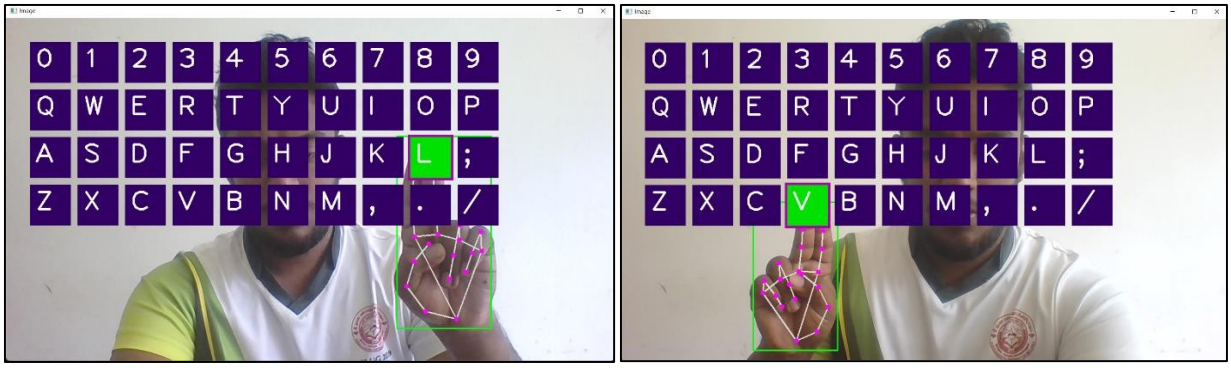

Figure 5.0 – shows the status of the button pressed

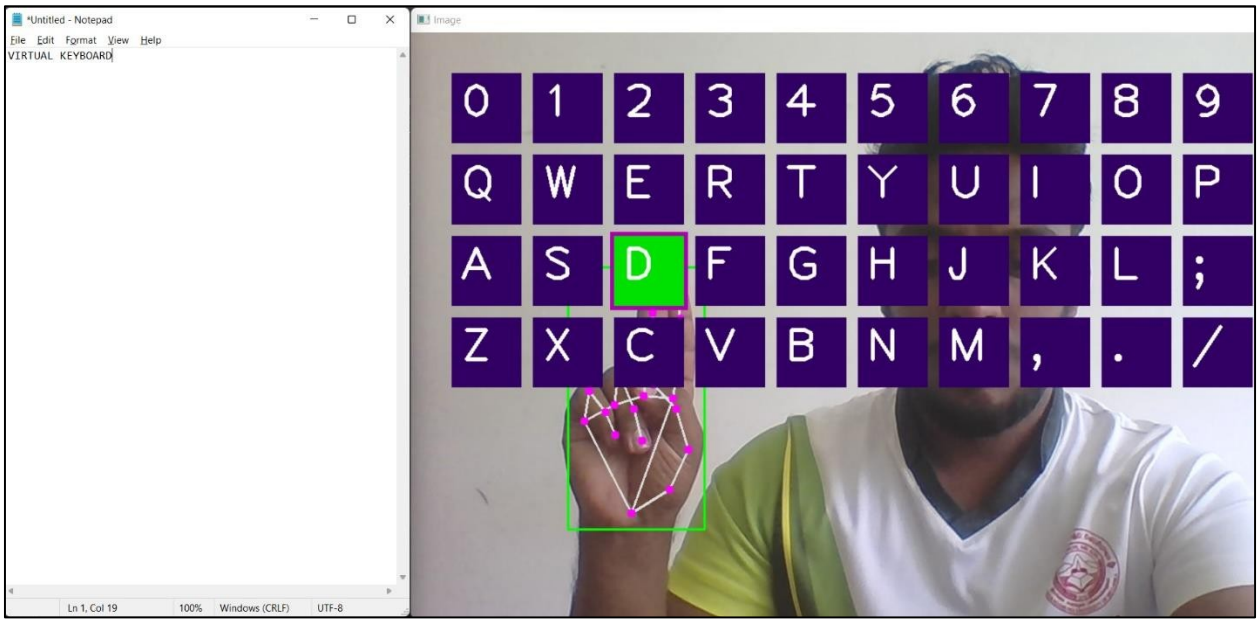

Figure 6.0 – Type text in notepad using OSVK system

Methods used

This whole project is based on python language, and many python libraries use to integrate some features such as voice assistant, control mouse keyboard, mathematical calculations. Used the **mediapipe** python library to find hands and detect the landmarks of hands. This library allows extra functionalities like finding how many fingers are up or the distance between two fingers. It also provides bounding box info of the hand found. Figure 2.0 shows all landmarks detect in **mediapipe** library on the hand.

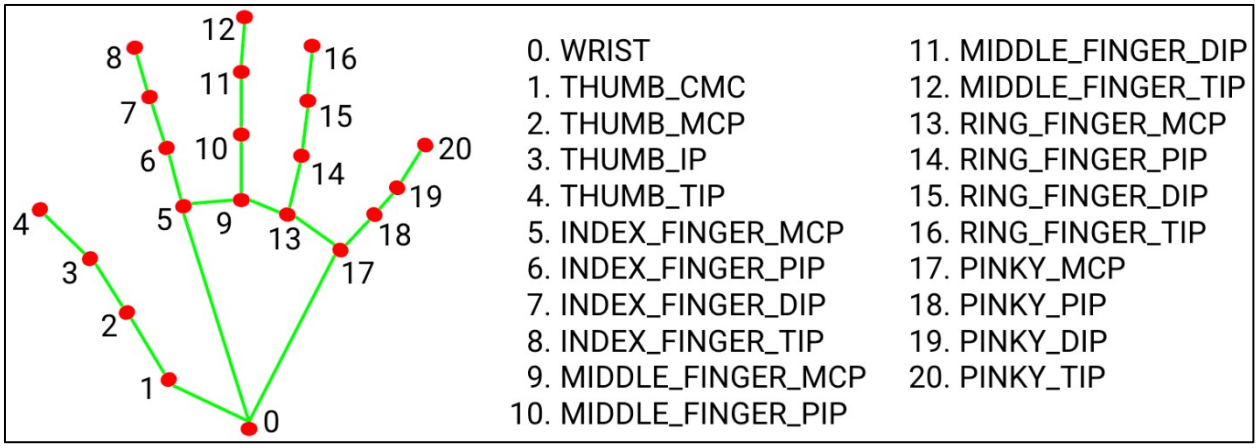

Figure 7.0 – 21 Hand landmarks

pyttsx3 is a text-to-speech conversion library in Python which is used to integrate voice assistant feature to conversion character, which is print pressed key. Its sound can be customized as a male sound or female sound and also can change voice level and rate. Can save voice as a mp3 file as well.

cvzone python library also used to develop this because it helps to write the program easily.

Not only those two libraries but also many libraries which commonly used in python are used to develop this project. PyCharm IDE is used to write this program.

Discussion

When developing this project, some difficulties have occurred. Such as measuring the distance between the index finger tip and middle finger tip. Because if hand close to the camera it measures significant distance value and if hand there away from the camera it measures the small distance between both index figure tip and middle finger tip. Therefore, sometime keys are not detected as a pressed key due to that issue. To solve that issue the range of hand and landmarks detect towards the camera and away from the camera should give constant range.

In this system detect only one hand if both hands appeared. Therefore, this should be developed to detect both hands and landmarks on both hands, allow to type using both hands as well.

Reference

[1] mediapipe. 2021. MediaPipe Hands. [online] Available at: https://google.github.io/mediapipe/solutions/hands.html [Accessed 30 December 2021].

[2] PyPI. 2021. pyttsx3. [online] Available at: https://pypi.org/project/pyttsx3/ [Accessed 30 December 2021].

[3] Computer Vision Zone. 2021. Advance Computer Vision with Python - Computer Vision Zone. [online] Available at: https://www.computervision.zone/courses/advance-computer-vision-with-python/ [Accessed 30 December 2021].