A Single Camera Virtual Keyboard

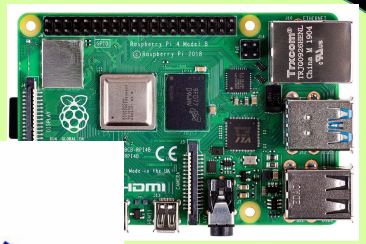
Peter West, Kevin Connell

Introduction

- Motivation
  - We aimed to create a virtual keyboard capable of creating a gesture controlled keyboard on any surface using one web camera. This could be expanded into any camera using AR to virtually have a keyboard anywhere.
- Goals
  - Create a virtual keyboard embedded on a microcontroller using hand gestures
  - Use one webcam or camera to capture the visuals
  - Create a readable character device or utilize a key-out system to render keyboard key presses
  - Calibration phase to map the keys the user's liking (could also choose what keys the user would like)
  - Typing phase showing resulting key presses after the calibration process where the user can type like a normal keyboard

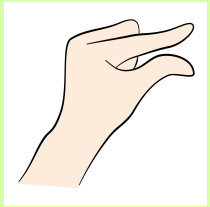
Overview:

- **Materials**:
  - BeagleBone Rev C or Raspberry Pi 4
  - Logitech C920 Webcam
  - Laptop/Monitor (Ethernet or HDMI)
  - Example keyboard layout (printed paper)
- **Software:**
  - OpenCV
  - MediaPipe
  - Pyautogui or kernel module
  - Other various config libraries
  - Python virtual environment -> embedded system (performance concern)
- **Set up:**
  - Webcam plugged into microcontroller positioned above reference keyboard layout (or no layout)
  - Calibration Phase: Follow key calibration steps to map keys using a pinch gesture
  - Typing Phase: Type by positioning index finger to a mapped key and using a pinch gesture



OpenCV

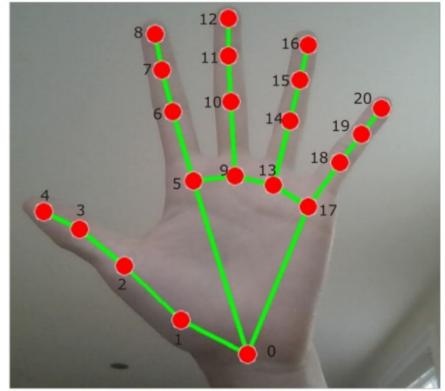
MediaPipe

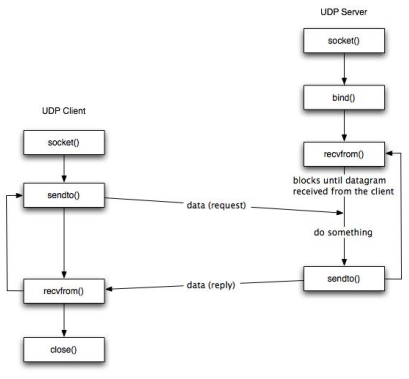


-

Methods:

- Webcam streams live video feed through FFmpeg and v4l2
- Utilizes a UDP socket connection to send the video stream through WiFi and minimize latency (or HDMI)
- Video feed data is piped to python script through Pyautogui
- MediaPipe is used to detect and track hand movement and to detect keystrokes
- Opencv is then used to determine key positions and display the virtual keyboard to the user

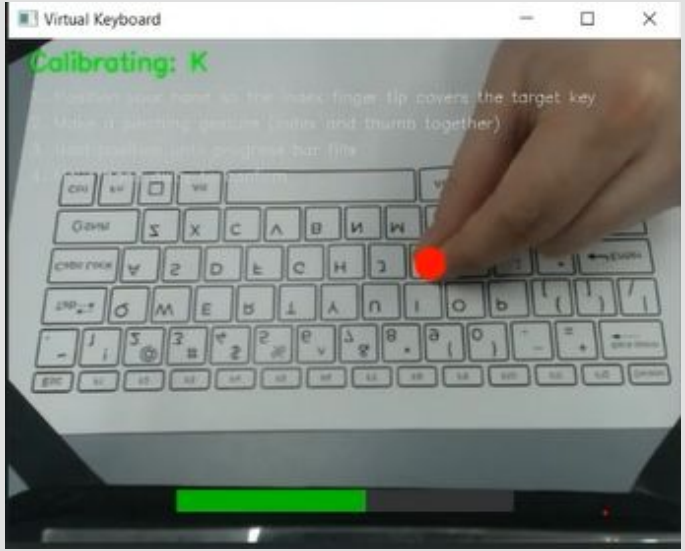




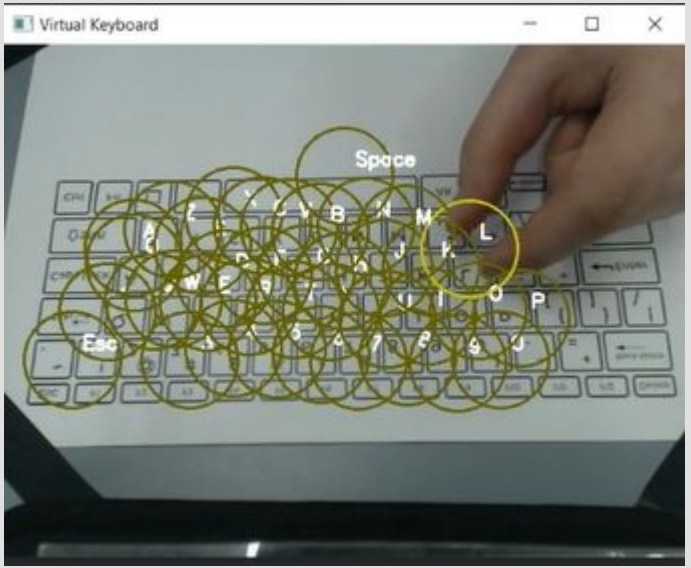
Challenges

- Getting the BeagleBone running with eduroam
- Libraries were difficult to install/not optimized for microcontroller
- Calibration system went through several different iterations
- Camera feed latency and image quality issues
- TCP vs UDP
- Establishing piping for camera feed directly to keyboard process
- Translating Python to C++ for BeagleBone implementation

Results



Calibration Phase



Typing Phase

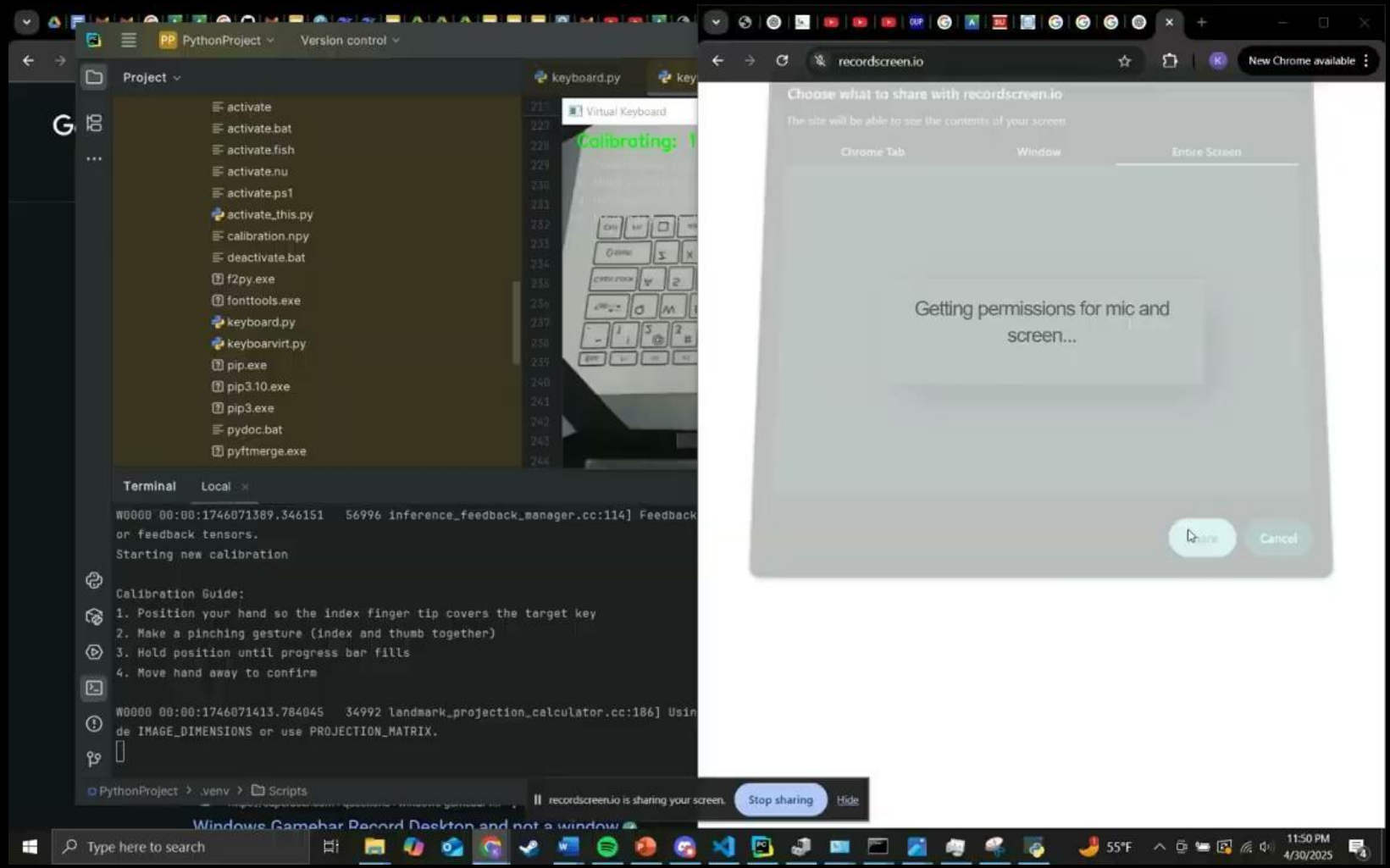
Future Steps:

Switching over to a more power microcontroller such as a Raspberry Pi 4

Full Implementation on microcontroller and limit need for an additional external device (Full vs Hybrid)

Kernel modules to enable direct control over a laptop through the virtual keyboard

Thank you!


PythonProject
Version control
Project
activate
activate.bat
activate.fish
activate.nu
activate.ps1
activate_this.py
calibration.npy
deactivate.bat
f2py.exe
fonttools.exe
keyboard.py
keyboarvirt.py
pip.exe
pip3.10.exe
pip3.exe
pydoc.bat
pyftmerge.exe
keyboard.py
Virtual Keyboard
Calibrating:
Terminal
Local
W0000 00:00:1746071389.346151 56996 inference_feedback_manager.cc:114] Feedback
or feedback tensors.
Starting new calibration
Calibration Guide:
1. Position your hand so the index finger tip covers the target key
2. Make a pinching gesture (index and thumb together)
3. Hold position until progress bar fills
4. Move hand away to confirm
W0000 00:00:1746071413.784045 34992 landmark_projection_calculator.cc:186] Usin
de IMAGE_DIMENSIONS or use PROJECTION_MATRIX.
PythonProject
.venv
Scripts
recordscreen.io
New Chrome available
Choose what to share with recordscreen.io
The site will be able to see the contents of your screen
Chrome Tab
Window
Entire Screen
Getting permissions for mic and screen...
Share
Cancel
recordscreen.io is sharing your screen.
Stop sharing
Hide
Type here to search
55°F
11:50 PM
4/30/2025