

CSE467 Embedded System Final Project Report

Byeongchan Gwak, b.gwak@wustl.edu

Abstract

Embedded systems are microprocessor or microcontroller-based hardware and software systems designed to perform dedicated functions. [omni21]. Raspberry Pi and Arduino are the most basic tools to learn these embedded systems. The difference between the two systems is that Arduino is microcontroller board, while Raspberry Pi is a microprocessor-based minicomputer [hub21]. Raspberry Pi shows better performance for image and video work, so in this project, I will try to make a CCTV using Raspberry Pi. I am going to make a Motion Detector CCTV that records only when there is a movement of a person or an animal, rather than continuously recording CCTV.

Table of Contents

1. Introduction and project goals
2. Project specifications
3. Project hardware design
 - 3.1 Block diagram of a Motion Detector CCTV
 - 3.2 Schematics of a Motion Detector CCTV
 - 3.3 Photos of a Motion Detector CCTV
4. Project software design
 - 4.1 State diagram of a code
 - 4.2 Block diagram of a code
 - 4.3 Source code
 - 4.4 Compilation and console messages
5. System integration
6. System testing and results
7. Conclusions
8. References

1. Introduction and project goals

The goal is simple. In this project, we will make a motion detector CCTV using Raspberry Pi. I use the garage door instead of the front door because I don't wear shoes inside the house. But there are some problems. My wife is really surprised when the post office or courier knocks on the front door. To reassure her of her, I'm going to do this project where I'm recognizing who's coming and recording a video of her.

2. Project specifications

- a. Motion Detector CCTV should record only when a person or animal approaches within 3 meters.
- b. Motion Detector CCTV must be sealed in a box to keep the equipment inside safe.
- c. A camera should have true IP camera, 1080p or higher resolution.
- d. Recorded video must be remotely viewable.
- e. Recorded video must be at least 20 frames long.
- f. CCTV should indicate that a recording is in progress.
- g. If you run out of video storage space, free up space by erasing the oldest videos.

3. Project hardware design (include all figures, block diagram, schematics, hardware photos, etc.)

3.1 Block diagram of a Motion Detector CCTV

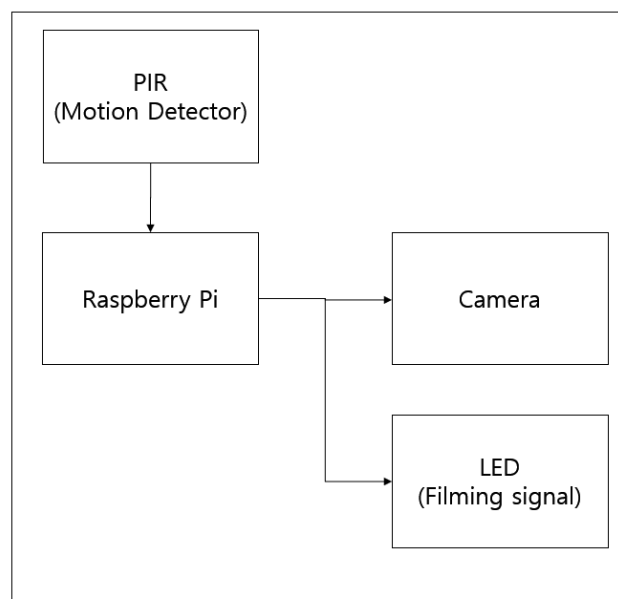


Figure 1: Block diagram of the project

3.2 Schematics of a Motion Detector CCTV

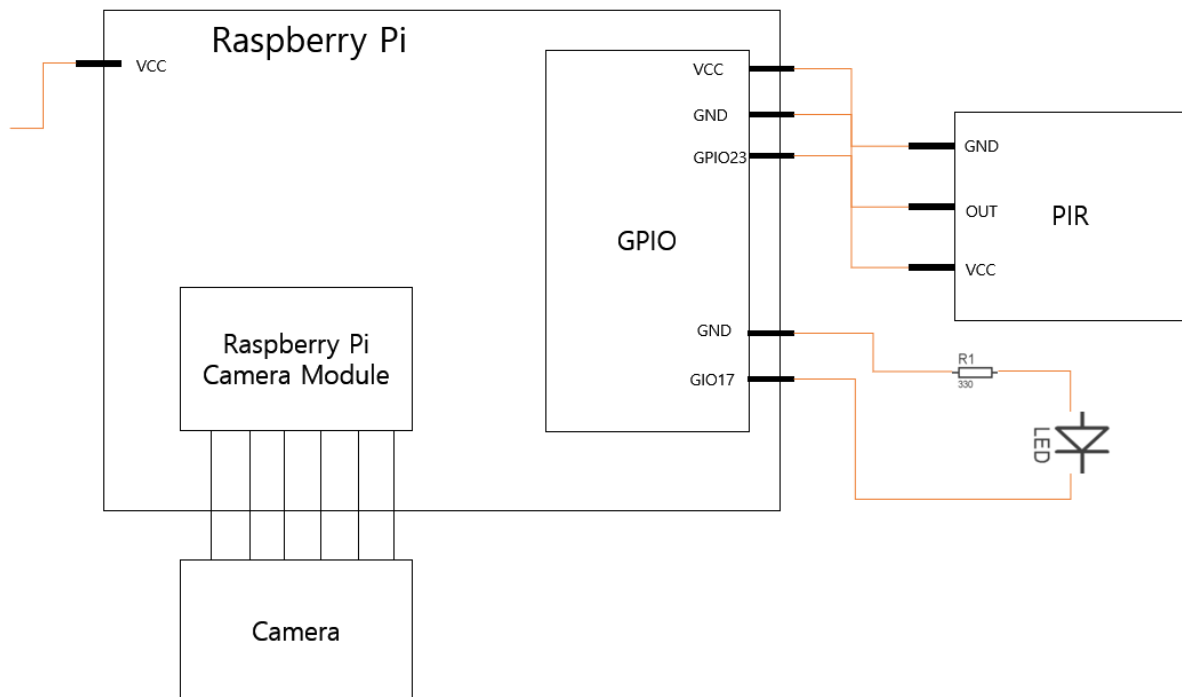


Figure 2: Schematics of the project

3.3 Photos of a Motion Detector CCTV



Figure 3: Appearance of the Motion detector CCTV

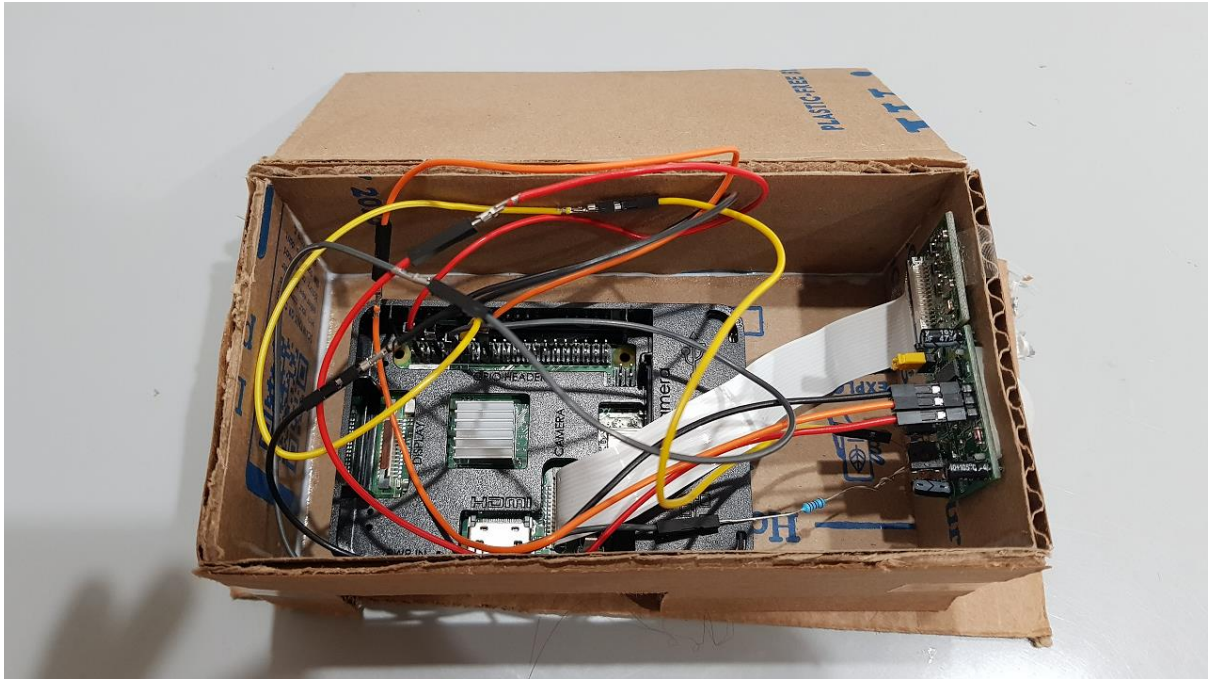


Figure 4: Inside of the Motion detector CCTV

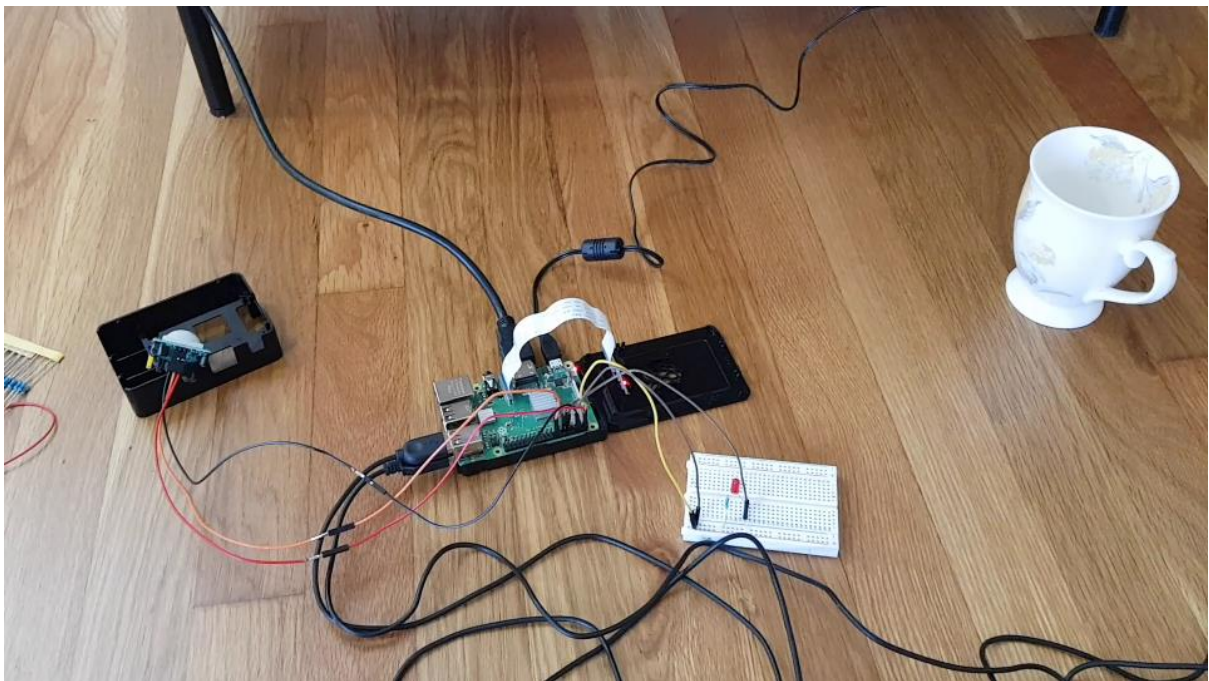


Figure 5: Unit test of the system

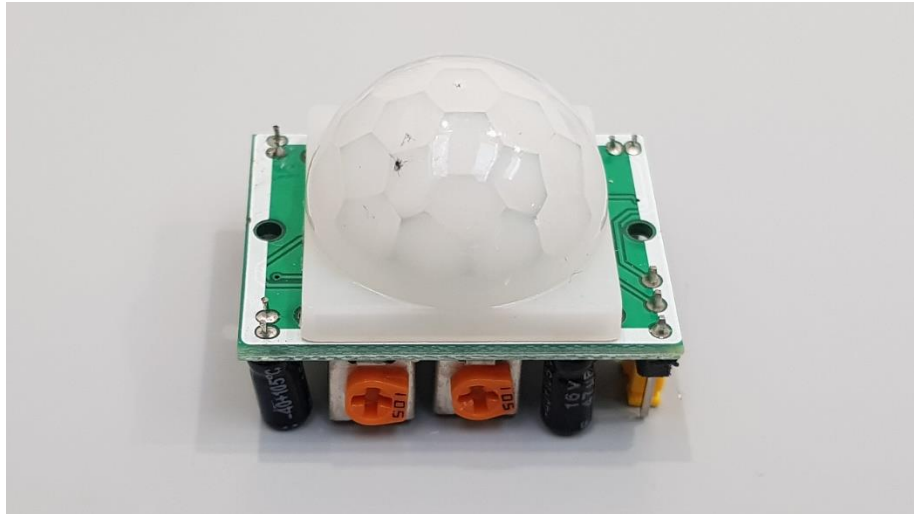


Figure 6: A PIR module(Infrared sensor)

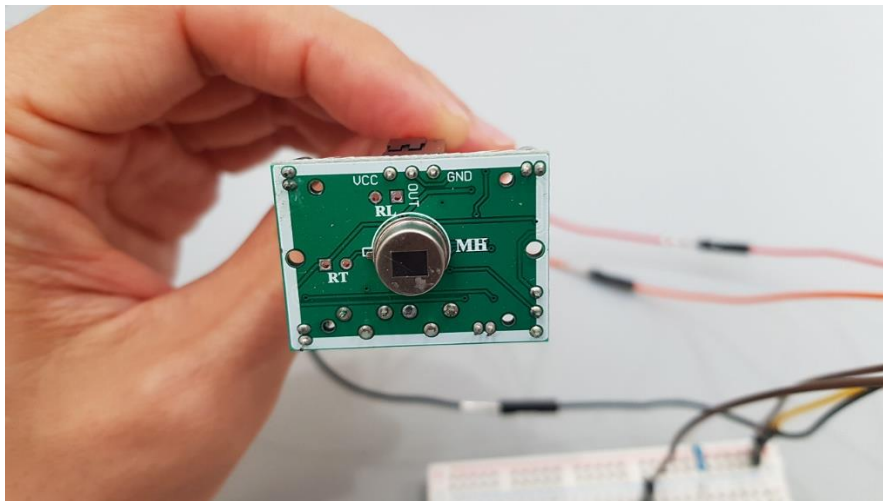


Figure 7: A PIR module without a cap

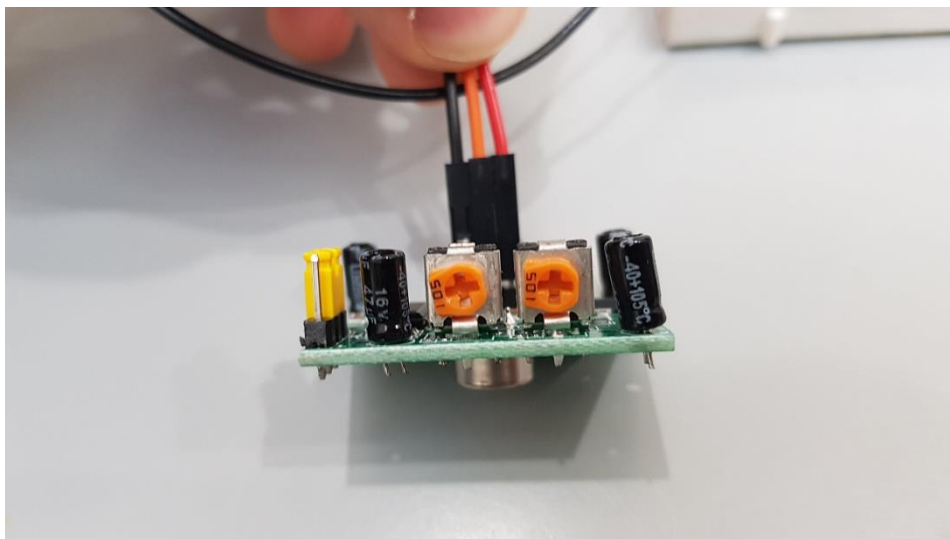


Figure 8: PIR Sensitivity Controllers (Distance on the left, timing on the right)

4. Project software design (include all state diagrams, other block diagrams, code, compiling, simulation if you have any, etc)

4.1 State diagram of a Motion detector CCTV

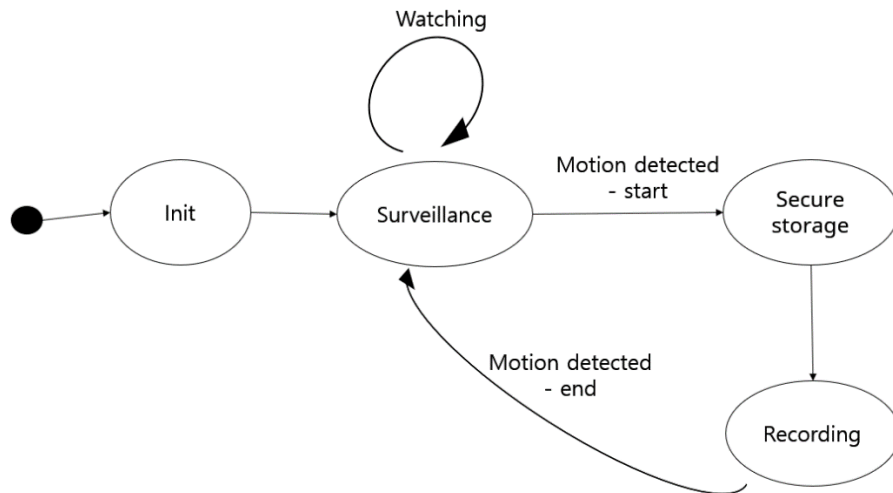


Figure 9: State diagram of the source code

4.2 Block diagram of a code

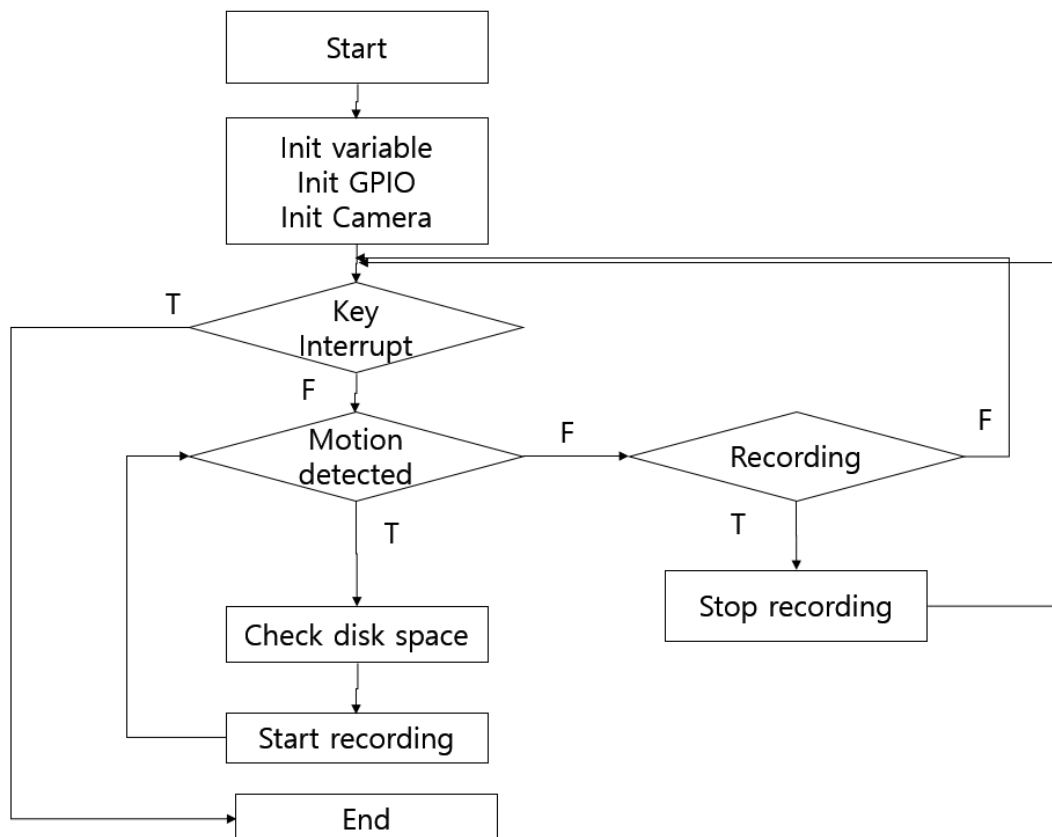


Figure 10: Block diagram of the source code

4.3 Source code

Final.py

```
from gpiozero import MotionSensor
from picamera import PiCamera
import RPi.GPIO as GPIO
import time
import os

PIR_PIN = 23 # pin for PIR which is motion detector
LED_PIN = 17 # pin for LED which shows PIR is working
MIN_DISK = 100 # minimum disk space to record video in MB

def getDiskSpace():
    st = os.statvfs(".")
    return int(st.f_bavail * st.f_frsize / 1024 / 1024)

def checkEmptySpace():
    remainSpace = getDiskSpace()
    print('\nAvailable space in Mega Bytes: ', remainSpace)

    # if the disk space below MIN_DISK,
    # and then delete oldest file to free up the space
    while remainSpace <= MIN_DISK:
        list_of_files = os.listdir('.')
        oldest_file = min(list_of_files, key=os.path.getctime)
        print('this file will be deleted: ', oldest_file)
        os.remove(os.path.abspath(oldest_file))

        remainSpace = getDiskSpace()
        print('\nAvailable space in Mega Bytes: ', remainSpace)

# init
startRecording = False # toggle purpose for recording
printOnce = True # toggle purpose for debug printing

filepath = '/home/pi/detector/recording_'
fileext = '.h264'

# initGPIO
GPIO.setmode(GPIO.BCM) # set GPIO in BCM mode which
GPIO.setwarnings(False)
GPIO.setup(PIR_PIN,GPIO.IN) # set 23 pin for PIR input
GPIO.setup(LED_PIN,GPIO.OUT) # set 17 pin for LEF output

# initCamera
```

```
camera = PiCamera()          # Initial and create Pi camera

GPIO.output(LED_PIN, False)  # before start turn off the LED
try :
    while 1:
        if GPIO.input(PIR_PIN) : # if PIR is detected
            if printOnce == True :
                print('\nMotion Detect CCTV is working - Gotcha!!' +
time.strftime("%Y%m%d_%H:%M:%S"))
                print('Recording is progress',end='')
                printOnce = False

            print('.',end='')

            if startRecording == False :
                checkEmptySpace()          # check disk space

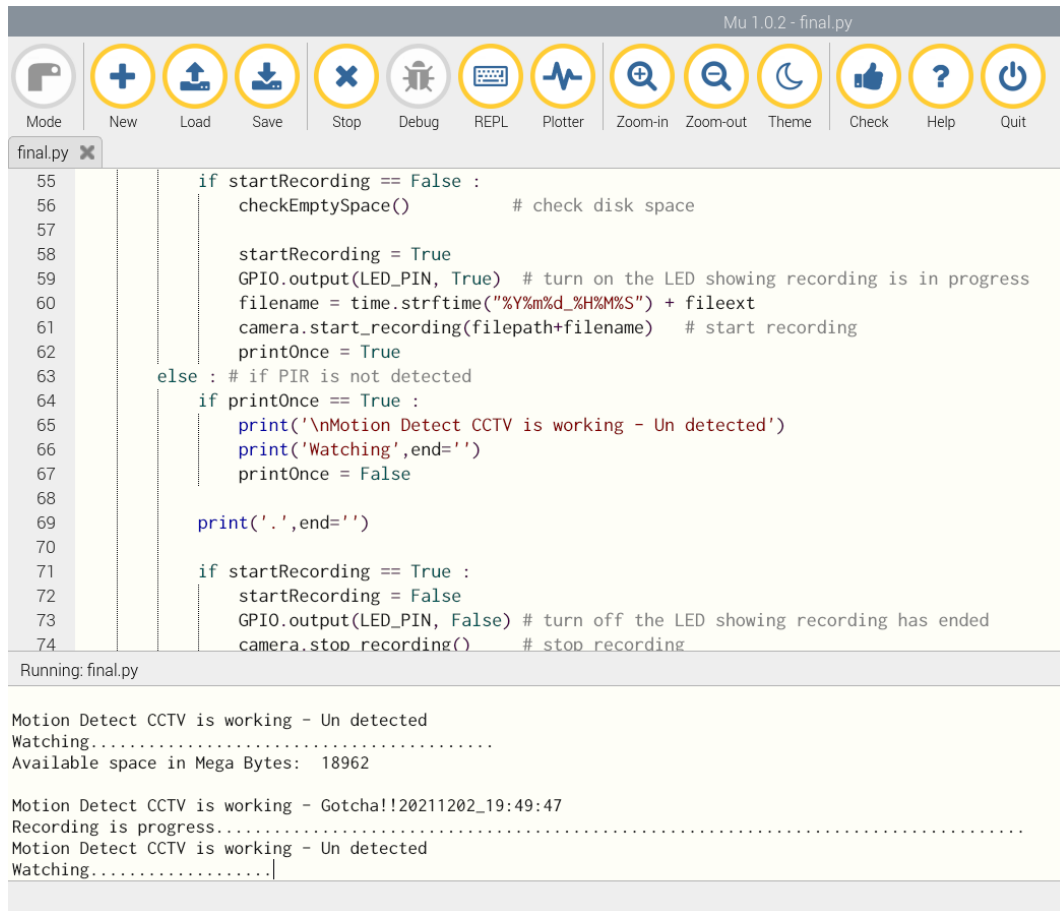
                startRecording = True
                GPIO.output(LED_PIN, True) # turn on the LED showing
recording is in progress
                filename = time.strftime("%Y%m%d_%H%M%S") + fileext
                camera.start_recording(filepath+filename) # start recording
                printOnce = True
            else : # if PIR is not detected
                if printOnce == True :
                    print('\nMotion Detect CCTV is working - Un detected')
                    print('Watching',end='')
                    printOnce = False

                print('.',end='')

                if startRecording == True :
                    startRecording = False
                    GPIO.output(LED_PIN, False) # turn off the LED showing
recording has ended
                    camera.stop_recording()    # stop recording
                    printOnce = True

                time.sleep(0.1)
except KeyboardInterrupt :
    print('Motion Detect CCTV is interrupted!')
finally :
    GPIO.output(LED_PIN, False)          # turn off the LED when exit
    GPIO.cleanup()                       # cleanup the GPIO when exit
    print("CCTV end")
```


4.4 Compilation and console messages



The screenshot displays the Mu Python IDE interface. The top toolbar includes icons for Mode, New, Load, Save, Stop, Debug, REPL, Plotter, Zoom-in, Zoom-out, Theme, Check, Help, and Quit. The main editor window shows a Python script named 'final.py' with the following code:

```
55     if startRecording == False :
56         checkEmptySpace()          # check disk space
57
58         startRecording = True
59         GPIO.output(LED_PIN, True) # turn on the LED showing recording is in progress
60         filename = time.strftime("%Y%m%d_%H%M%S") + fileext
61         camera.start_recording(filepath+filename) # start recording
62         printOnce = True
63     else : # if PIR is not detected
64         if printOnce == True :
65             print('\nMotion Detect CCTV is working - Un detected')
66             print('Watching',end='')
67             printOnce = False
68
69             print('.',end='')
70
71         if startRecording == True :
72             startRecording = False
73             GPIO.output(LED_PIN, False) # turn off the LED showing recording has ended
74             camera.stop_recording()    # stop recording
```

Below the code editor, the console output for 'Running: final.py' is shown:

```
Motion Detect CCTV is working - Un detected
Watching.....
Available space in Mega Bytes: 18962

Motion Detect CCTV is working - Gotcha!!20211202_19:49:47
Recording is progress.....
Motion Detect CCTV is working - Un detected
Watching.....|
```

Figure 11: Console message after a running

5. System integration

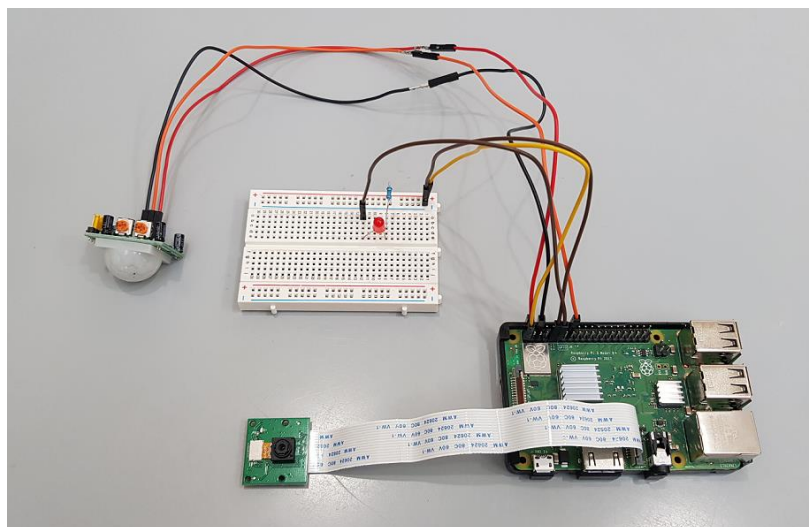


Figure 12: Whole integration of the system

6. System testing and results



Before capturing a motion -
LED is off



After capturing a motion -
LED is on and recording is in progress

Figure 13: Motion detector CCTV is working when my kids is in front

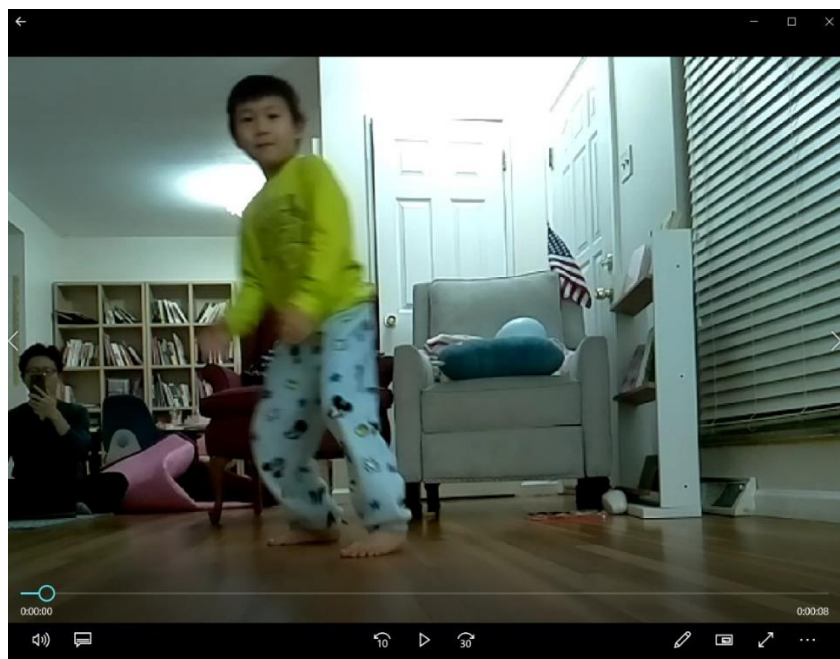


Figure 14: Capture of a real recording file

7. Conclusions

In this project, I implemented the most basic motion capture CCTV using Raspberry Pi. Although the Raspberry Pi is smaller than a fist, it has shown that its performance is sufficient for image processing. What I felt while doing this project is that it is important to fully understand the parts you want to use. When performing the unit test, the result value of the PIR module is always 1, so I found out that VCC and GND were swapped. And it would have been difficult to predict the result correctly if you did not read the manual carefully about the part that adjusts the sensitivity or the jumper setting part. Even with other embedded projects, I have come to realize that a complete understanding of the basic components is the key to project success.

8. References

[hub21] Electronicshub, “What are the differences between Raspberry Pi and Arduino?”, <https://www.electronicshub.org/raspberry-pi-vs-arduino/>

[omni21] Omnisci, “Embedded Systems”, <https://www.omnisci.com/technical-glossary/embedded-systems>