

# Realtime 4220 Project Report

## Motion Detection and Alert System

### ABSTRACT:

This project uses a server-client model and software interrupts as its application to real time embedded computing. Fundamentally, it is a motion detection system that alerts the user over Twitter or through SMS messaging when motion is detected. It also sends a picture of the motion event. Its applications to everyday use include as a home monitoring system, a doorbell, or a wildlife observation unit.

### INTRODUCTION:

This project uses a Raspberry Pi development board, a Logitech webcam, and an Internet connection to achieve motion detection and user alert. The objective is to create a system that can monitor a room or area without direct user interaction, alerting the user whenever it detects motion. The motivation is to create a system that can be used, for example, to monitor one's room when away. If one has roommates, it would be interesting to know if they ever enter your room when your not there.

### BACKGROUND:

The method that I am replicating is the use of the Motion software. This software constantly takes pictures at a user-defined rate and compares two sequential ones to determine if there was motion. The common method for detecting motion, as far as I can tell, is to use a PIR motion detector as the catalyst for picture taking. This project accomplishes the same goal using less hardware. Again, an example of when this project might be used is as a doorbell or as a wildlife observation unit. However, my project is best suited as a room monitoring unit, as it alerts the user whenever motion is detected, instead of passively storing motion related files.

### PROPOSED METHOD:

The way this project works is by taking pictures of an area and checking them to see if motion is detected. When motion is detected, an interrupt (if not in name then in its effect) occurs that uploads the picture to a server. The handler function, called on-motion-event.py, will be called with a command line argument of 0. This project uses Imgur as the server for picture upload. Then, the picture is grabbed from the server and sent to the user-defined phone number. It also uploads the picture to Twitter. The Motion software detects the webcam and uses it as its image producer. This program waits 5 seconds after a motion detection event to upload the picture, and will send out two pictures, one to Twitter and one through SMS. Motion will continue to

take pictures and store them on the Rpi until no more motion is detected. Eventually, the Motion software will “wrap” around and start overwriting its own stored image files if the motion even is too long. Once motion is no longer detected, it will call the same function as when motion was detected, except with a command line argument of 1. This will simply clear the folder containing image files.

I looked into using threads to concurrently upload the picture to twitter and the imgur server, but I could not get it to work. As it stands it first sends the SMS and then uploads the picture to twitter. However, since the overhead is low there would not be a significant speed increase by including threaded programming in this application. Perhaps if I implement a video upload instead of a single picture, using two threads to upload would be more beneficial.

#### EXPERIMENTS AND RESULTS:

To test this application, I sent the pictures to my phone and also created a twitter account. Here is the twitter account, it is accessible to the public:

<https://twitter.com/mizzouRealtime>

I ran the code many times, to ensure that the folder was cleared upon event end, to ensure that the correct picture was sent to both Twitter and SMS. I managed to get SMS working as well, as will be shown in the video.

#### DISCUSSION AND RESULTS:

My results were consistent with what I expected. I managed to get both the Twitter and the SMS portions working. I used a variety of sources to implement my code, which are credited in the readme. I combined several different API's to create a project that accomplishes an entirely new function, namely user alert as well as a timeline storage of motion events. Twitter works as a timeline, as the user can scroll down and see when motion was detected. The first problem I had was using the Raspberry Pi. I could not ssh into it without a monitor and keyboard to edit the config files, and I did not have a router. So I was blind at first as to the IP address of the Pi. I fixed this problem by using school resources. The second problem was that the handler function needs to be called as root from Motion, otherwise the Motion software does not have permission to execute the file. I learned how to use API's for my own goals, how to generate keys and tokens to allow access to my Twitter and Imgur accounts. I also learned how to program a Raspberry Pi to do what I want it to do. I learned how to send data through several different places until it gets to where I want it to go. Finally, I

gained a more complete understanding of software can be implemented using careful, step by step planning.

#### APPENDICES:

Code is attached, as well as a readme

#### FINAL THOUGHTS:

This is not part of the report, just a simplified version of everything if you don't want to read through my entire report to figure out how the program works. I also want to say here that if I somehow didn't credit someone in my code, that it was unintentional. All of my sources and the code I combined can be found in the links at the bottom of the README. I also allude to the sources of my code in the comments in the actual code. I am not sure what the citation guidelines are for code, but everything I used was publicly accessible online.

#### Simple Explanation of Project:

1. Motion software runs
2. When motion is detected, call on-motion-event.py
3. on-motion-event.py looks at folder where images of motion are stored, and after five seconds, uploads the most recent of them to twitter and sends that same image through SMS
4. Once motion is no longer detected, it empties the folder where images are stored and continues scanning the room. The Motion software is running during this entire process.