Video Capture to Website Streaming

Option B: Development Project

Ryan McCormick, Pablo Corona, Drake Morgan

Due 12/2/2021

Computer Science,Colorado State University
Fort Collins, CO 80523
ryanbmcc@rams.colostate.edu
pcorona@rams.colostate.edu
drakeem@rams.colostate.edu

***Abstract*— This document provides a detailed account of our assignment and it's requirements. For our development project, we chose to implement a raspberry pi with camera and motion sensor functionality in order to stream a video to a website. When the motion sensor detects motion, the camera begins recording a video, streams that video to a website, and displays a series of images from the popular video game: Skyrim: Elder Scrolls V.**

I. Introduction

It is impossible to always be at home and on guard against intruders, especially with everyday duties such as work or school, and thus remote surveillance is becoming an increasingly popular method of keeping our homes and belongings safe. Remote surveillance is a method of capturing and streaming video to a portable device in order to view a specific area, such as a front patio, from a cellphone or laptop. In the below sections we will be discussing how we built our prototype and how the final product turned out.

II. Problem Characterization:

The problem of home intruders and unwanted guests is what we are attempting to solve with our surveillance prototype, which has motion sensor, video capture, and streaming capabilities.

III. Proposed Solution and Implementation Strategy.

In order to solve this problem, we must first gather the necessary hardware to implement our project. The main component is a raspberry pi, which acts as the motherboard for our prototype. The raspberry pi is responsible for handling the operating system, connecting the various components together, and sending power to different sections of the device. The secondary components of our prototype included the camera, which captures and saves video files, a set of speakers to play a sound when a video was being captured, an amplifier which helps the speakers integrate with the raspberry pi, and the motion sensor, which is capable of detecting movement, such as a person at a front door. The last component of the project is the software that unites the raspberry pi and the secondary components into one to provide functionality to our device. For this assignment, we decided to use several software libraries to provide quality and secure implementation. The libraries used are as listed below and are described later in the paper:

1. Condition Object - Thread Synchronization
2. Pi Camera
3. Http.server
4. io
5. Logging
6. Motion Sensor
7. Socket Server

In order to implement our project, we first began with the idea of using the camera to save a video file of 5 seconds upon detecting motion, and then stream that file to the website concurrently using threads. The problem that we encountered with this approach was that the website was unable to find the file on our raspberry pi operating system. We are unsure whether this was due to the file path being incorrect, or the fact that the file may have

been incomplete upon attempting to stream it to the website.

For our final implementation, we decided to follow a data structure that we learned how about earlier in this course, circular buffers. Since saving full 5 second videos to later be streamed to a website did not work, we decided to split up the problem further and only stream one frame after another of the video capture directly to the website. We thought that a circular buffer would best suit this problem, due to the fact that our program could technically run indefinitely with an unspecified number of frames that we would only need to use once.

IV. Limitations

There were a few limitations that we ran into while creating the Motion activated video streaming, primarily coming from the hardware which we were using.

1. The framerate of the camera is dependent on the resolution of the stream which we are creating. Creating a higher resolution stream results in a stream which is choppy and suffers from framerate.

2. Increasing the framerate of the camera over 60 frames per second caused the quality of the stream to also decrease. The images of the camera became red saturated and no longer worked.

3. The motion sensor simply begins with very sensitive motion meaning that potentially a falling leaf can trigger the sensor. while the range which the motion sensor is detecting can be tweaked the individual sensitivity cannot. The only means of doing so is through using a physical object to obscure the sensor which still only helped to stop false positives about fifty percent of the time.

4. Our amplifier that is required to run the speakers on our raspberry pi took up all of the pins on the board. If we wanted to be able to use the speakers it meant we could not use the camera and motion sensor. Since these two components are more important to the project we decided we would instead not be able to use the speakers and amp.

5. Picamera Can only run on one instance of camera at a time, meaning that while the camera is streaming mjpg to the website, no video can be recorded or saved on top of it. This meant we couldn't save the streamed "video" to the pi at the same time.

6. The website is not always operational, meaning that it is only active when the program is running. The ideal surveillance system would have little to no downtime.

7. In order to connect to the website, a user would need to be connected to the same wifi source. This again poses a problem as you would not need to watch a security camera while at home but instead when you are away. This can't be done due to how the website is built and run.

IV. Conclusions.

Ultimately the product is somewhat of a success. While we were able to implement the core functionality of the motion activated camera stream, some aspects which we intended to include were not implemented. Particularly being the implementation of the speakers which alerts when the stream has begun. While this is ultimately not a center of the project it would come to impact the marketability of our product. Similar products in the market at the current time contain more functionality for about the same cost.

Due to this we feel there are multiple key elements that we could improve upon in the future to make the product more marketable.

1. Many embedded surveillance systems have an app which can control the stream. This idea can be implemented to control the video stream through a mobile device. Adding functionalities such as, initializing the stream at any point, viewing and downloading stored surveillance video.

2. Speakers are a common functionality on similar types of systems. Thus implementing a speaker functionality to output pre-recorded audio files or live message streaming as well

3. The stream suffers from low resolution in order to maintain framerate which can be improved upon in order to create a higher quality product.

4. In order to catch an intruder with our current prototype, the user must keep a constant eye on the surveillance system, as there is currently no alert when the system is triggered. To fix this, we could improve the functionality of the motion detection to send text messages to a cell phone number when motion is detected. That way, the user could do other activities and only look at the camera system when needed.

V. Description of Libraries

I. Condition object- Thread Synchronization

The condition object is mainly used as a solution to the producer and consumer problem. The producer in our case is the video frames produced by the camera device, while the website streaming server acts as the consumer of those frames.

II. Pi Camera

The Pi Camera Library is used to capture video footage from the physical camera on the raspberry pi, and translate this footage into files on the system to be displayed on the server.

III. Http.server

The use of the Http.server library is to set up the server using our .html page layout to format the look and elements of the http website.

V. io

The io library provides implementation for reading, writing, and handling files efficiently. In our project, the io library is used to save and write video files to the machine to be later streamed to the website server.

V. Logging

The logging library is used to track when certain events, such as motion or camera capture, occur. We use this library to get information about when the video files get sent to the server and when the .html file gets displayed on the website.

I. Motion Sensor

The motion sensor library is responsible for providing functionality to the motion sensor device. The methodology that the motion sensor uses resembles a polling system, or heartbeat. The code checks the motion sensor to see if a high value is received, indicating sufficient movement. If movement is detected, a True signal is sent back to the code.

I. Socket Server

The Socket Server library is used to partition the threads of the nodes, which are the server and the raspberri pi, as well as the communication between the two nodes.

VII. Time

We used the Time library to synchronize and buffer certain events in order for our program to run smoothly. The main use of the Time library was to use sleep() to set a period of time for our camera to capture video footage.

References

R. Barnes and R. runs R. P. Press, “VNC: Remote Access a Raspberry Pi,” *The MagPi magazine*. [Online]. Available: https://magpi.raspberrypi.com/articles/vnc-raspberry-pi. [Accessed: 03-Dec-2021].

“Condition object - thread synchronization in python,” *Studytonight.com*. [Online]. Available: https://www.studytonight.com/python/python-threading-condition-object. [Accessed: 22-Nov-2021].

cytrontech. “Raspberry Pi Security System Using Pi Camera and Motion Sensor.” *YouTube*, YouTube, 8 Oct. 2020, https://www.youtube.com/watch?v=yHWT1SiH-iw.

D. Jones, “Picamera¶,” *picamera*. [Online]. Available: https://picamera.readthedocs.io/en/release-1.13/. [Accessed: 22-Nov-2021].

E. The Computer Guy, “Raspberry pi - how to install an operating ... - youtube,” *Youtube*. [Online]. Available: https://www.youtube.com/watch?v=kPz8DaUfi34. [Accessed: 03-Dec-2021].

“Http.server - HTTP servers¶,” *http.server - HTTP servers - Python 3.10.0 documentation*, 22-Nov-2021. [Online] Available: https://docs.python.org/3/library/http.server.html. [Accessed: 22-Nov-2021].

“IO - core tools for working with streams¶,” *io - Core tools for working with streams - Python 3.10.0 documentation*. [Online]. Available: https://docs.python.org/3/library/io.html. [Accessed: 22-Nov-2021].

“Logging howto,” *Logging HOWTO - Python 3.10.0 documentation*, 22-Nov-2021. [Online]. Available: https://docs.python.org/3/howto/logging.html. [Accessed: 22-Nov-2021].

“Motion Sensor,” *2. Basic Recipes - GPIO Zero 1.6.2 Documentation*. [Online]. Available: https://gpiozero.readthedocs.io/en/stable/recipes.html#motion-sensor. [Accessed: 22-Nov-2021].

“Raspberry pi 4 getting started,” *Doovi*. [Online]. Available: https://www.doovi.com/video/raspberry-pi-4-getting-started/BpJCAafw2qE. [Accessed: 03-Dec-2021].

“Socketserver - A Framework for network servers¶,” *socketserver - framework for network servers - Python 3.10.0 documentation*, 22-Nov-2021. [Online]. Available: https://docs.python.org/3/library/socketserver.html. [Accessed: 22-Nov-2021].

“Time - Time Access and conversions¶,” *time - Time access and conversions - Python 3.10.0 documentation*, 22-Nov-2021. [Online]. Available: https://docs.python.org/3/library/time.html. [Accessed: 22-Nov-2021].

“Use raspberry pi camera from python,” *YouTube*, 28-Jan-2021. [Online]. Available: https://www.youtube.com/watch?v=RPZZZ6FSZuk&%3Bt=53s. [Accessed: 02-Dec-2021].