Software Overview

Year: 2022 Semester: Fall Team: 12 Project: R.A.C.H.E.L.

Creation Date: ­9/9/2022 Last Modified: September 10, 2022

Author: Bartosz Stoppel Email: bstoppel@purdue.edu

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Software Overview** |  | x2 |  |  |
| **Description of Algorithms** |  | x2 |  |  |
| **Description of Data Structures** |  | x2 |  |  |
| **Program Flowcharts** |  | x3 |  |  |
| **State Machine Diagrams** |  | x3 |  |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** |  | x2 |  |  |
| **Formatting and Citations** |  | x1 |  |  |
| **Figures and Graphs** |  | x2 |  |  |
| **Technical Writing Style** |  | x3 |  |  |
| **Total Score** |  | | |  |

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

*Relevant overall comments about the paper will be included here*

1.0 Software Overview

For this project we will not be writing any firmware for any of our peripherals. We will be going to use the default firmware on all of our devices: STM32 microcontroller, laptop, depth/color sensor, and projector. We will be using different software to interact with every devices’ firmware.

* STM32 Microcontroller

The microcontroller will have software that we will be writing for it that will be able to read inputs from all the peripheral sensors and communicate with the laptop. The MCU will handle ball bounces by reading data from the ADC channel and discerning it from all other noises the piezo sensors could pick up. This will be important as a paddle hitting the table or the table moving mid-point could potentially trigger a ball bounce in our system. The microcontroller is also handling the inputs from a 4x4 button matrix that will allow users to adjust the score, start/finish games and potentially pick game modes. The MCU will have to process the button presses and prevent ghosting/masking of presses. Once the microcontroller is done processing the inputs, it will send a UART signal to the laptop with ball bounce information and keypad user inputs.

* Laptop

The laptop receives the UART signal in its serial port. If a ball bounces, it’s location will be estimated by the camera that needed OpenNI2 to interface with. From there, the video feed becomes handled by our blob detection algorithm. The location of the ball goes through the decision tree in Fig. 2. If the ball bounce is passable (a valid bounce), then the laptop will proceed with adding whatever sprite/image the minigame requires into a frame buffer. The frame buffer will be what is displayed by the projector.

* Minigame

We had ideas for two minigames that could be played with table tennis. Both are opposites of one another. For one minigame, we would create a circle around every part of the table where the ball hit. The player must avoid those circles if they don’t want the other player to score a point. The circles could get progressively larger the longer the round takes. The other minigame has a circle on the opposite side of the table before a player hits the ball. The player needs to aim it into the circle in order for the volley to keep going. If the ball lands outside a circle, then a point is scored. Likewise, the further the players are into the game, the smaller the circle could get.

2.0 Description of Algorithms

* Blob Detection Algorithm

This is the algorithm that will handle image processing of pin-pointing the ping pong ball from the camera feed. When we were first trying to implement this, we were planning to use the 30 FPS depth sensor on our ASUS Xtion Pro Live camera. However, after talking to our TA there was some thought in using the color feed instead. The color sensor goes up to 60 FPS and does not require as much post image processing. Therefore, we will most likely proceed with the color sensor for identifying where the ball is.

OpenCV will be the library that we will be using in order to ease the process of detecting the ball. OpenCV is a computer vision and machine learning library that will be used as a framework for our project [1]. As of now, we have made a test data set that had white circles with various background. Without using the camera, we ran our OpenCV code to see if it would detect those white circles, but it did not. Once we get the software to detect virtual table tennis balls, then we can move onto using it with the camera feed. From there, the software would have to give us the centroid of the ball so we could use it as a location. This location is then sent to the microcontroller as previously noted.

* Table Calibration

This may be more of a stretch-goal, however, I will still include it in this software overview. So instead of setting up the camera to be perfectly aligned to the table, we will try to write software that takes into consideration of where the ball is relative to the table. There are many ways to accomplish this. We could use OpenCV to detect the table shape from the camera feed. From there we can relatively position the ball within the table. Another option is to display bright green/red from the projector, adjust the projector’s image to be illuminated only on the table and from there we use blob detection on the green/red square on the table. This would be a nice feature to add, but it remains on the backburners until we get a functioning game first.

3.0 Description of Data Structures

OpenCV is the only software library we will be using for its computer vision framework. It will make image processing lightyears easier as they already have a built in color differentiation filter. We can set it to filter out the color of the ball, mind the projector displaying onto the ball also.

In our UART protocol from the MCU to the laptop, the packets will include a bit if a button is pressed, 4 bits containing which button was pressed and one bit if a bounce was detected. From the laptop to the MCU, there will only be request packets and reset packets. Reset packets will reset the MCU and request packets will ask for the data packets mentioned before. We are using request packets as this may be better for the refresh rate of the displaying the images from the projector. The laptop will request data, the MCU will send it, the laptop will process it, and then display it.

4.0 Sources Cited:

*[1] OpenCV “OpenCV: Home”* [*https://opencv.org/*](https://opencv.org/) *(retrieved 9/10/2022)*

Appendix 1: Program Flowcharts

Shape

Description automatically generated with medium confidence

Fig 1. Flow Chart

Diagram

Description automatically generated

Fig 2. Decision Tree