Puzzle Module for a Mobile Escape Room

ECE 445 Design Document

Colin Flavin, Helen Swearingen, Nick Russo

Group 33

TA: Eric Liang

1. Introduction

1.1. Objective

Champaign-Urbana Adventures in Time and Space, a local escape room company, is producing the "Adventure Crate," a portable escape-room-in-a-box device. For this device, they need a series of "puzzle modules," which are smaller puzzles or games that the players must solve in order to progress through the experience. These puzzles should be novel and entertaining, with an emphasis on teamwork and communication within the group.

1.2. Background

Escape rooms are a form of entertainment that has been on the rise in popularity over the past few years. Groups of people, often up to eight, work together solving themed, time-sensitive puzzles in sequence to complete a "mission" and escape the room [1]. Rooms are usually set in a particular location that people visit to play the game. Few escape rooms currently utilize any motion-tracking technology, even in escape rooms fixed in a location. There are few escape rooms that have the capability to move around easily, let alone any that utilize motion tracking. Typical escape room puzzles might involve the finding and placing of objects in correct locations, solving passcodes, or other similar games, but few to none of them involve the motion of the player themselves.

Our group plans to build one of the larger modules for the Crate. This module is an "invisible maze," which requires a player to follow a path that they cannot see, directed by another player who views the path to follow on an LCD screen on another side of the Crate (note: no one is physically inside the Crate. The Crate simply houses the electronics for different puzzles during the escape room experience, and interfaces via outward facing displays). We will track the location of the maze-walker using the AprilTag computer vision system. The maze-walker will wear a backpack that has the AprilTag tag (similar to a QR code) on it that will be tracked using a USB webcam mounted on the main Crate. The path walked will be compared to the correct path as shown on the screen, and the screen will update in real-time with the location of the maze-walker. The path will have 1ft of tolerance on either side of the correct path as "correct." 1ft outside of that will be a "warning" zone, and any farther outside of that will count as exiting the border of the maze. When the maze-walker is on the correct path, we will have RGB LEDs on the backpack stay green. In the warning zone, they will turn yellow and there will be vibration motors that go off. When you exit the border, the LEDs will flash red, and the motors will vibrate more intensely. At this point, you need to reset to the start and try again. The processing of the maze and location will be done on a series of Raspberry Pi's. There will be an additional Raspberry Pi located within the backpack that will communicate with the main control Pi's and control the LEDs and vibration.

1.3. High-Level Requirements

- The location of the player in the maze must be tracked with a USB webcam using the Apriltag system within a precision of 1 foot or less.
- A maze must be automatically generated and displayed via a small monitor to the player outside of the maze.
- Players must be notified when maze walls are crossed via a wifi signal sent from the game processor to a wearable device. The player in the maze should be notified via haptic feedback on the wearable device, and the player outside of the maze should be notified on the monitor.

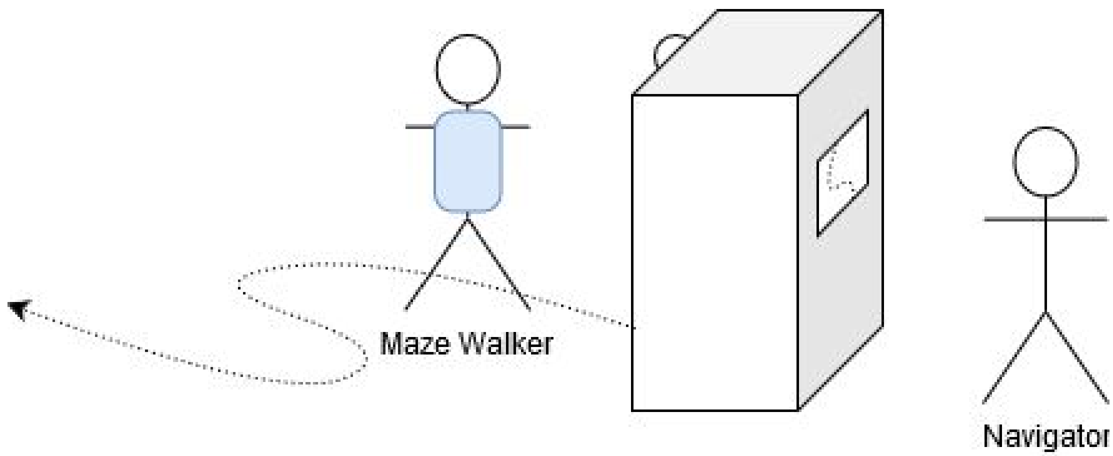

Figure 1: How the game runs

1.4. Game Rules

The portable escape room will be brought into an empty and placed in the center of it. It should be able to be plugged in, and then it should start the escape room. Other modules outside of the scope of this project will be played by the player in the room. As each module ends, it starts the next module.

When the maze module is up, it will automatically generate a maze according to the room size. It will then direct a player to wear a backpack with the haptics and Apriltag on it. One player will then use the small monitor with the maze information on it to direct the player in the maze to the exit. If any maze walls are broken, the players must restart the maze. When the maze is completed, the module starts up the next module in the escape room.

2. Design

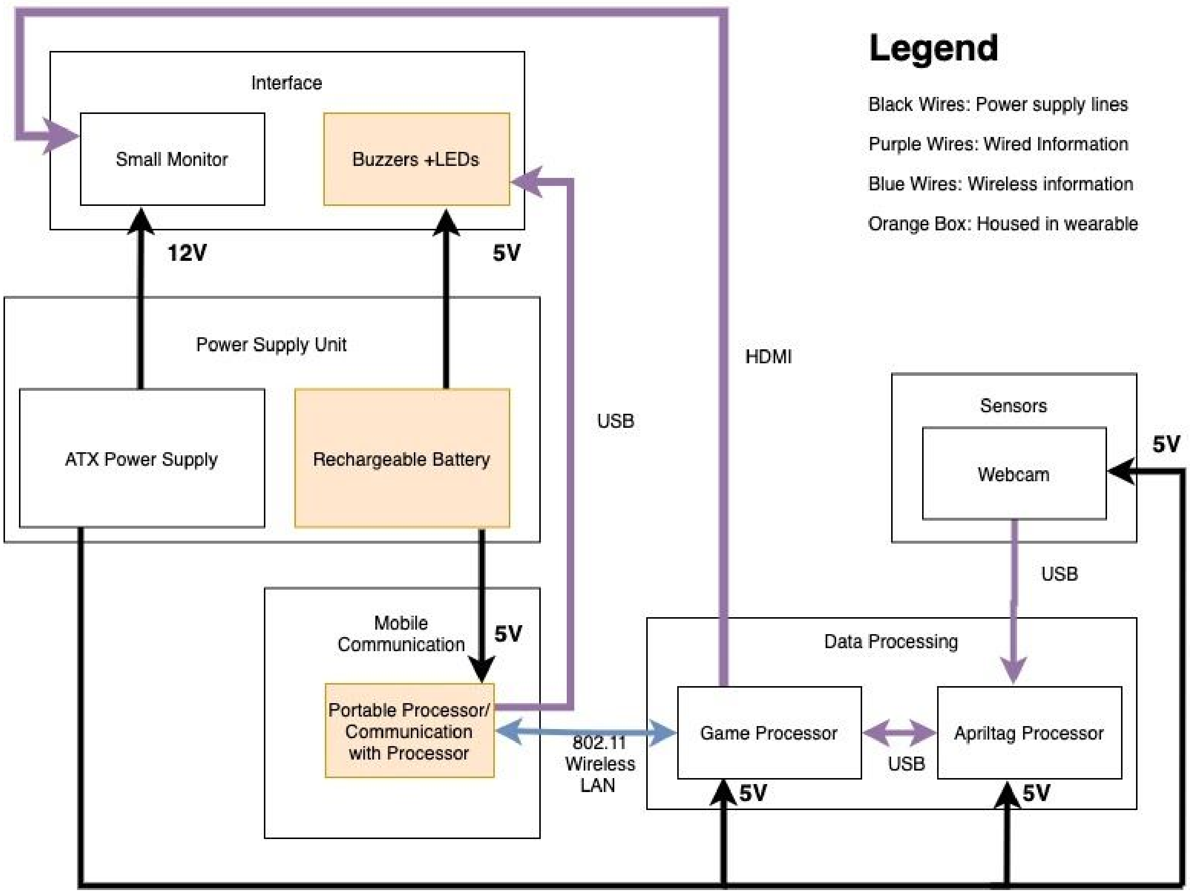

Figure 2: Block Diagram

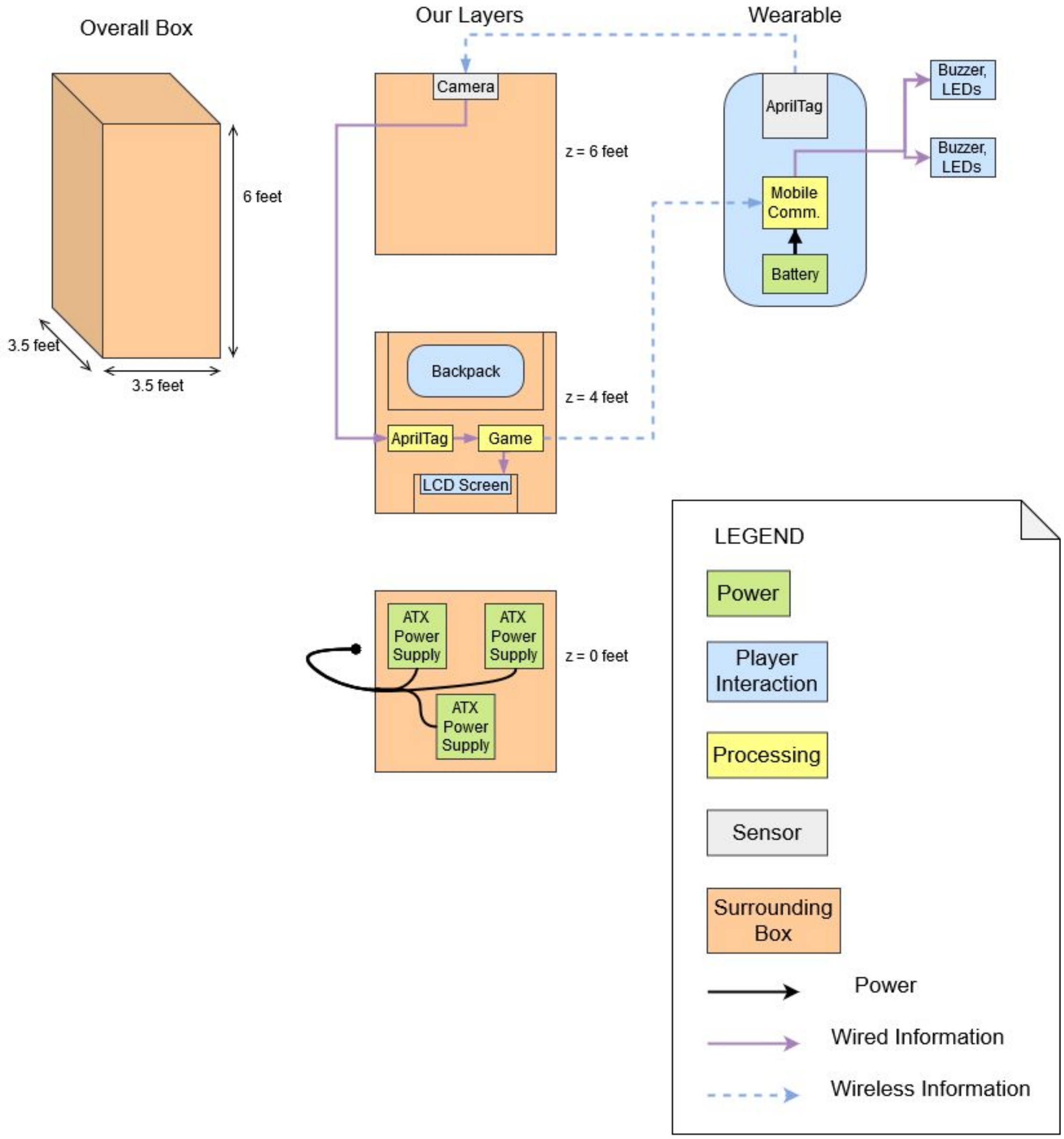

Figure 3: Physical Diagram

There are five main aspects to this project. First, everything must be powered by two separate power supplies. Sensors are also required, in the form of a webcam to search for the player in the maze. The data both to create the game and to process the camera feed to calculate the player's location must be processed in two separate processing units. Then, this data must be communicated to the player's wearable device in the case that they cross a boundary. Finally, the game must have an interface to display the maze to other players.

2.1. Power Supplies

Power supplies are required for both the units in the main housing for the processors and in the wearable device.

2.1.1. ATX Power Supply

The ATX Power supply converts the outlet AC voltage to DC rails at 5, 9, and 12V. The escape room we're working with has asked for several DC rails so this serves both our module's needs as well as the rest of the box's modules.

Requirement	Verification
Must consistently supply DC rails at 5±0.5V, 9±0.5V, and 12±0.5V.	A. Check each voltage with a voltmeter to ensure it remains within tolerance

2.1.2. Rechargeable Battery

The wearable device also must have a power supply to power the processor communicating with the game processor, as well to power the haptic feedback device. This needs to last for at least an hour, but ideally it could last for a whole day. The battery must be able to power the Raspberry Pi within the wearable. The Raspberry Pi we are using should take in 5V, and needs variable amounts of current to power everything [2]. For most of the time, this should be low, since nothing happens in the wearable for most of the time. However, when triggered, there must be power supplied to LEDs and ERM motors, so current will spike. Due to the spikes, the battery should be able to handle currents up to 2.4A.

Requirement	Verification
Battery must supply 5±0.5V at max 2.4A for at least one hour.	A. Measure the open-circuit voltage and ensure it doesn't surpass 5.5V with a voltmeter B. Use resistor load to force 2.4A from the battery, check with an ammeter C. Leave for an hour and ensure power specifications can still be supplied by checking voltage and current with voltmeter and ammeter
Battery must continue to function after being dropped at least 5 feet	A. After checking it works within the voltage and current specifications, drop the battery from 5 feet and recheck functionality as above

2.2. Data Processing

Data processing is extremely important to this project. Information about the maze and the player's location is constantly being updated, checked and sent through the system. All image data captured by the webcam must be passed into a processor to be sent through the Apriltag libraries for processing. From this, data concerning the player's location in respect to the camera is created. Due to some concerns about the amount of processing power required to run this in real time, an entire Raspberry Pi is dedicated to image processing.

Afterwards, the player's location data must be incorporated into the game. This is done through another processor that is responsible for the general control of the module. At the beginning of gameplay, this processor generates a maze based off of room size. From there, as it receives location data, it maps that onto the maze and displays that onto a monitor. It then calculates if the player collides with a maze wall, and communicates any collisions to both the monitor and the wearable device.

2.2.1. Game Processor

The maze is rendered, and as information is fed in concerning the player's location, the processor calculates if any boundaries are crossed, and communicates that to the interfaces.

Requirement	Verification
Must be able to communicate via iEEE standard 802.11 wireless LAN connection	A. Establish connection between the two Raspberry Pis B. Program dummy data block to one Raspberry Pi C. Send data block to the other Raspberry Pi D. Compare to make sure they are same
Software must be able to render a maze	A. Write maze software B. Display a generated maze to an LCD screen

2.2.2. Apriltag Processor

Information is fed in from the camera to a Raspberry Pi processor. Using an AprilTag library, the location from the camera is calculated, then fed into the game processor.

Requirement	Verification
The processor must be able to calculate a user's location from the camera accurate to ±1 foot	A. Setup and calibrate the computer vision software B. Print an Apriltag C. Measure various distances away from the camera and place the Apriltag in the camera view D. Check that the distance the system calculates is within the tolerance of 1 foot.

2.3. Interface

During gameplay, the player should see feedback throughout. Without updates on player location in respect to the map, it would be nearly impossible to complete successfully. In addition, without any feedback that a maze boundary was crossed, the game has no real purpose. Therefore, the maze and the player's current location will be displayed on a monitor. At the beginning of gameplay, the maze will be rendered and displayed onto the monitor. As the game processor gets updates on the player's location within the maze, their location in respect to the maze should also be placed onto the monitor. In the case that the player crosses a boundary, the haptic feedback system on the player in the maze should be turned on, and a notification on the monitor should be displayed.

2.3.1. Small Monitor

The monitor displays the map and the player's location within the maze. If the player in the maze crosses a maze wall, a notification will be displayed. It gets this data from the game processor. To function successfully, it must be bright enough for players to easily be able to see it without strain. We are planning to use an off the shelf screen so that all requirements are outside the scope of this project.

2.3.2. Haptic Feedback Device (buzzer + LEDs)

When a player crosses a maze wall, they should get haptic feedback and be flashed lights to notify them. It gets activated by a processor on the wearable that is communicating with the game processor in the main housing.

Requirement	Verification
LEDs must be visible from at least 5 feet away with a drive current of 25mA+-5%	A. Adjust resistors to ensure 25mA current per LED, test with an ammeter B. Measure 5 feet away from an LED C. Ensure the LED is clearly visible from that distance
Vibration from the ERM motors must be easily felt through the backpack with a drive current of 125mA+-5%	A. Adjust resistors to ensure 125mA of current per motor, test with an ammeter B. Place motors in the four corners of the backpack C. Activate motors while wearing the backpack to see how easily they are felt
Device must continue to function after being dropped at least 5 feet	A. Drop haptic feedback device from 5 feet and ensure it is stable and functions properly as before

2.4. Communication

When the player in the maze crosses any boundaries, it is necessary to notify them. However, the wearable does not hold any information about its location, nor does it have any information about the maze. In turn, it must constantly be communicating with the game processor so that when any maze boundaries are crossed, the player is notified. When the game processor calculates that the player has crossed a maze boundary, it should send a signal stating that the player has crossed the wall. The wearable's processor should then receive that signal, and set voltage to high on all the haptic devices to turn them on, effectively notifying the player.

2.4.1. Wifi Processor on Wearable

The wearable must have a WiFi communication with the game processor so that it can be notified when the haptic feedback must be activated.

Requirement	Verification
Processor needs to activate haptic feedback when signalled to	A. Send a signal from the game Raspberry Pi B. Ensure the wearable Raspberry Pi receives the signal C. Check the voltage out to the LED and Buzzer are correct
Processor must continue to function after being dropped at least 5 feet	A. Drop the wearable from 5 feet B. Repeat the steps from the first requirement

2.5. Sensors

A camera is required to and track the location of the player via Apriltag. This is an extremely critical portion of our project. If the camera is not functional, there would be no way to track the player in the maze. The camera needs to communicate via USB with a Raspberry Pi, feeding image data to be processed by the Apriltag computer vision software. It's important that the resolution of the camera is good enough so that Apriltag codes can be detected reliably from the far wall of the play area.

2.5.1. Camera

The camera should be at a resolution high enough to accurately see the Apriltag on the player, and steadily communicate via USB connection to its associated processor.

Requirement	Verification
The camera resolution needs to detect an Apriltag from 20 feet away	A. Set up the camera and computer vision system B. Place an Apriltag 20 feet away from the camera C. Ensure the camera can detect the Apriltag

2.6. Schematic

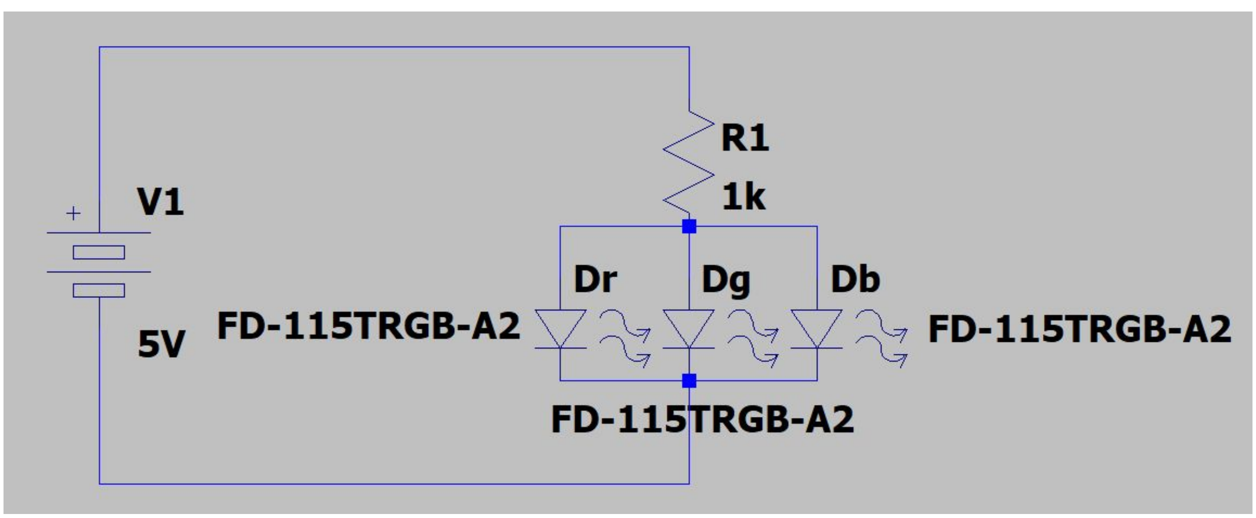

Figure 4: LED Schematic for 1 Diode

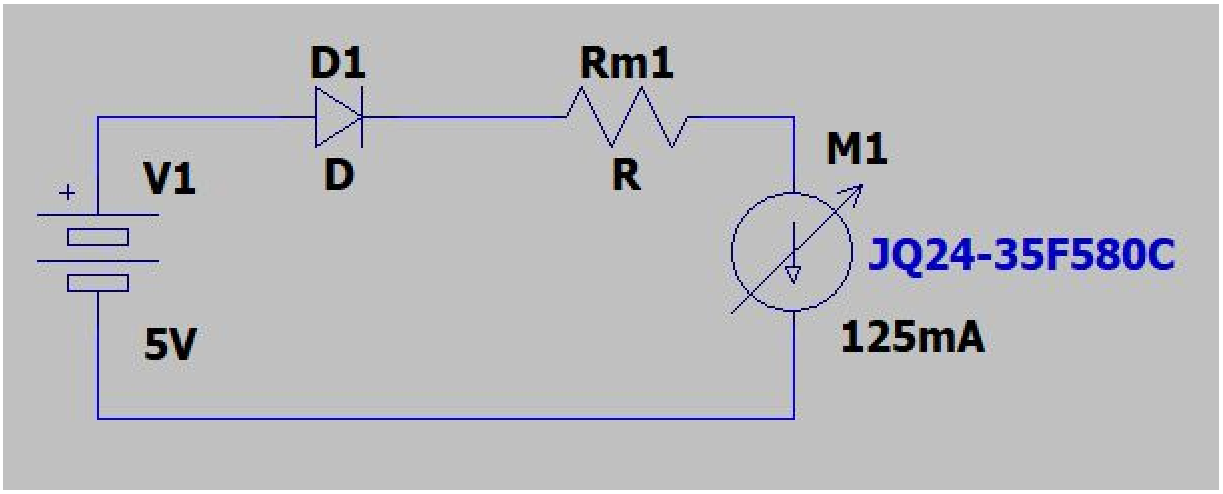

Figure 5: Motor Schematic for 1 Motor

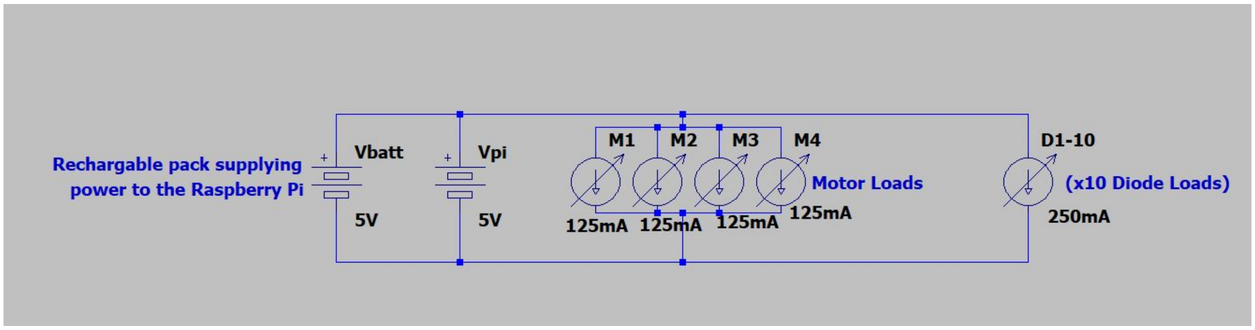

Figure 6: Full Wearable Schematic

2.7. Softwar

Software is an extremely important aspect of this project. We need to take in constant data from the camera, process it to find the location of the player in the maze, and then compare it to the maze to make sure no walls were crossed. Then, in the case someone crosses a maze boundary, a notification must be sent to all the players in the game.

2.7.1. Computer Vision

The brunt of the processing power goes into the computer vision portion of the project. Since we are using the Apriltag system, the libraries are all online and open source. For this project, we chose to use OpenCV to access the Apriltag libraries.

2.7.2. Game Mechanics

A maze must also be rendered for the gameplay. During setup, the distance to the back and side walls will be set. From that information, we know the boundaries of the room and render the maze accordingly. The main game will then take the location data of the player from the computer vision, and then compare it to the map data. If between data points, the player crosses a boundary, the player must go back to the starting position and restart the maze.

2.8. Tolerance Analysis

The most critical portion of this project is the distance tracking via Apriltag. Without accurate distance tracking, the game would become unplayable. Hence, we need to take a further look at the tolerance in the Apriltag.

Apriltag's accuracy depends on the camera's resolution, the angle of rotation of the Apriltag, and on the distance of the Apriltag code from the camera. As the Apriltag gets further from the camera, it's accuracy in computing the distance from the camera decreases exponentially. For a 600x600 pixel camera, by 60 meters, accuracy went up to +-5 meters. For this project, we require a tolerance of less than one foot, which would be satisfied with the 600x600 camera at distances less than 40 meters, or around 130 feet [3]. For most of our project's applications, this limit is satisfactory. Lower resolution cameras have tended to decrease the detection accuracy of the Apriltag system [4], so a higher resolution, but slightly more expensive camera must be used for this project.

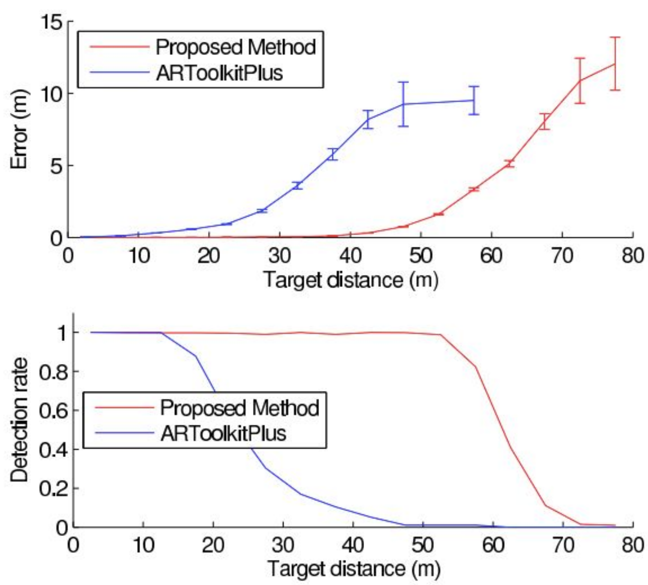

Figure 7: Error and Detection Rate for Apriltag (red) and ARToolkitPlus (blue) [3].

In the gameplay rules for this puzzle module, we are planning on adding a stipulation that the player inside the maze should be facing away from the puzzle box at all times, so the angle of the Apriltag code should not be an issue. However, the computer vision software can still detect

Apriltag codes reliably to up to an 80 degree rotation, and mostly error free to 70 degrees. By 80 degrees, the error rate jumps to around +-2 degrees when detected [3].

3. Costs

Our development costs are estimated at around $40/hour, 10hours/week, for 3 people. We consider there are 10 weeks left in the course to complete the project. This neglects weekly TA meetings, and meetings with the escape room, this comes to $12,000 in labor costs.

$$3 \times \$40/hour \times 10\ hour/week \times 10\ weeks = \$12,000$$

Part	Cost
(Raspberry Pi 3) x 3 (provided by CATS)	$35 x 3 = $105
ERM Motor x 4 (Jinlong Machinery & Electronics, Inc. JQ24-35F580C -- Digikey)	$5.46 x 4 = $21.84
LEDs (Shenzhen Fedy Technology Co., LTD FD-115TRGB-A -- adafruit)	$10
HDMI Monitor (LONCEVON -- Amazon)	$52.99
ATX Power Supplies x 2 (Apevia ATX-VS450W -- Amazon)	$23 x 2 = $46
Anker PowerCore Battery (Anker -- Amazon)	$25.99
USB Webcam (ELP -- Amazon)	$45.99
Total	$307.81

We are only building one prototype, so the total materials cost should be $307.81. Overall, this implies a total cost of $12,307.81.

4. Schedule

Date	Nick Russo	Colin Flavin	Helen Swearingen
2/27	Research communication standards between necessary devices	Finish schematics and start PCB design	Communicate with CATS to get parts ordered Design and begin fabrication of the Crate and wearable mockups
3/2	Coding Apriltag software	Finish PCB design and start designing casings for wearable device parts	Initial placement of hardware modules in the Crate
3/9	Game mechanics software	Finish casings for wearable device parts and get casings made	Initial wearable backpack design (functional and aesthetic)
3/16	Break	Break	Break
3/23	Software for wearable electronics	Assemble wearable electronics	Assemble wearable backpack and electronics
3/30	Establish WiFi connection between wearable and game processor	Rework any parts that need it and get them reordered	Communicate with CATS to arrange an in-progress demo Secure parts in the Crate and wearable backpack
4/6	Catch up if not on schedule	Catch up if not on schedule	Catch up if not on schedule
4/13	Final software testing and debugging	Final hardware testing and debugging	Final placements in the Crate and wearable backpack
4/20	Prepare final presentation	Prepare final presentation	Prepare final presentation
4/27	Finish final report	Finish final report	Finish final report

5. Safety and Ethics

There are a few safety issues with our project. Safety of the players must be our biggest concern at all times. According to the IEEE code of ethics, we must “avoid injuring others...by false or malicious action” [5]. Rechargeable batteries have the potential for combustion if overcharged or overheated. Since this is indoors and use is somewhat limited, overheating should not be a large concern. In addition, it must be sturdy enough to deal with the potential misuse of the device by players. Our housing should be built to take this into consideration.

Another main concern with our project is our use of video. We use video to identify player’s location, but with any video, privacy is a concern. We do not save any video, however we should make consumers aware that cameras are used in this escape room.

Although we do deal with wall power, it goes straight into ATX Power supplies, which are built to deal with all potential issues we would have.

The height and weight of the escape room box also could be potential concerns. At 6 feet tall, the box has the potential to tip over. In addition, things may fall off of the box and injure players during play. We will work with CATS to weight down the box and securely mount everything to the escape room box.

6. References

[1]"What is an Escape Room?", *The Escape Game*, 2018. [Online]. Available: https://theescapegame.com/blog/what-is-an-escape-room/. [Accessed: 27- Feb- 2020].

[2]"Raspberry PI 3 model B+ Released: Complete specs and pricing - nixCraft", *nixCraft*, 2018. [Online]. Available: https://www.cyberciti.biz/hardware/raspberry-pi-3-model-b-released-specs-pricing/. [Accessed: 28- Feb- 2020].

[3] E. Olson, "AprilTag: A robust and flexible visual fiducial system", *https://april.eecs.umich.edu*, 2011. [Online]. Available: https://april.eecs.umich.edu/media/pdfs/olson2011tags.pdf. [Accessed: 28- Feb- 2020].

[4] "AprilTags", *Robotics Knowledgebase*, 2020. [Online]. Available: https://roboticsknowledgebase.com/wiki/sensing/apriltags/. [Accessed: 28- Feb- 2020].

[5] "IEEE Code of Ethics", *Ieee.org*, 2016. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 13- Feb- 2020].