INVISMAZE - Project Proposal

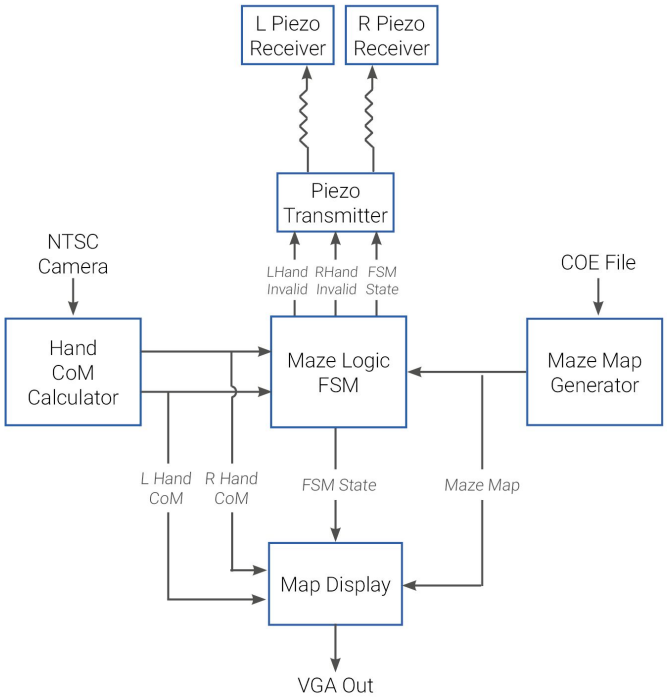
Steph Pavlick and Libby Zhang

1 Introduction and Motivation

Our project is to create an interactive, virtual-reality maze game. The game will be for a single player and will require an indoor, open space of at least 5 x 8 feet. The player starts at a marked edge of the open space, and must navigate through a virtual maze to the finish. The layout of the maze will be created and contained in the memory of the labkit, and a camera located on the ceiling above the play space will be used to project the player's location onto the virtual maze grid. The goal of the game is for the player to navigate the virtual maze to get to the finish.

The player navigates through the selected maze by wearing two gloves equipped with piezo buzzers that go off when the player ventures into an invalid region of the virtual maze. If the player ignores the signal and either stays in the invalid region or walks through the invalid region, the game is over. A camera above the course tracks the two gloves, which will both be a different bright color to allow for color recognition by the camera, sending the glove locations to the FPGA, which will also display the player's progress on the monitor, overlaid with the actual maze course for any spectators. The player may select a timed mode where a countdown timer dictates how long they have to reach the center and win the game

2 System Design



2.1 Design Goals

Our design choices revolve around making the game easy to play for the user and making our system robust and responsive to user movements. We want the game to ultimately be easy to learn to play and to be quickly responsive to the player's movement.

2.2 Module Details

2.2.1 Hand CoM Calculator

This module calculates the center of mass of the player's right and left hands in the camera frame. The player will be wearing two brightly colored, distinct gloves – bright blue on the left hand and bright orange on the right. This allows the calculator to differentiate between the two hands and output the hand location relative to the frame.

A stretch goal will be to calculate the body location based off of the distance between the two hands. If the two hands are very close together, then the body is directly behind the two hands. If the hands are separated, then the body is more likely to be between the two hands.

2.2.2 Maze Map Generator

This module combines the physical boundaries of the play space, the player's hand locations, and the selected maze map all into one. The generator scales the maze map (which demarcates invalid and valid locations) to the physical maze boundaries. It then also converts the hands' CoM locations from camera frame locations to map locations.

Prior to the implementation of the Maze Corner Locator (a stretch module), the Maze Map generator assumes that the boundaries of the board is the camera's frame.

2.2.3 Maze Logic FSM

The Maze Logic FSM operates the core of the gameplay, determining which of the 6 states the system is currently in and outputting the appropriate user signals. The six states consist of: NewGame, Standby, Playing, Warning, YouWin, and YouLose. NewGame is the default state, and the remaining five are all different states during which the game is being played. The Maze Logic FSM controls the haptic feedback to the player and other UI features to make a user-friendly game.

2.2.4 Map Display

This module translates the information produced by the Maze Map Generator module to be compatible with display on the monitor. Additionally, it also implements alpha-blending depending on the state of the FSM (e.g. if in Warning state, this module would overlay a semi-transparent orange layer over the screen indicating that the player is in an invalid area).

2.2.5 RF Transmitter and Haptic Feedback

This module transmits a message over radio from the FPGA to the gloves worn by the player. This message contains the enable command for the piezo buzzer, which would signal that the player is in invalid area, and the flash/color commands for the LEDs.

On the receiver end, the buzzer and the LEDs will be controlled by 555 timers that are enabled and disabled by the RF signals.

2.2.6 Stretch Modules

Audio Player
Additional user experience and user feedback module. This module will parallel the monitor display of the game so that the player and the spectators have an aural component which indicates when the player is danger, when the player loses the game, and when the player wins the game. This module will use pre-recorded sound bites stored in memory and playback via an external speaker.

Total Play Timer
This timer records the total time of play of the current game. It starts the moment the player enters the Start region and the Maze FSM transitions to the “In Play” state, and it stops counting when the FSM transitions into either the “Lose Game” or “Win Game” states. If it transitions into the latter, the time is displayed but not saved; if the former, the time is displayed and saved in the current session’s records.

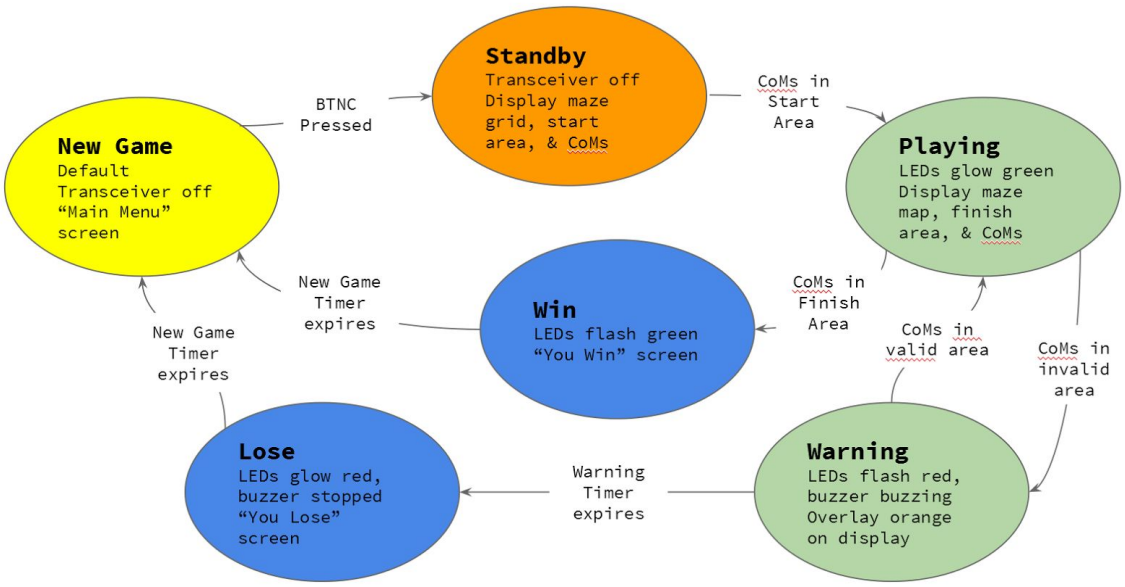
High Scores
The high scores of each session are saved in local memory and displayed when requested. A “session” is defined as the time from when the FPGA is turned on to when it is turned off. So, the high scores are not saved in a hard drive, just in RAM.

Maze Corner Locator
The implementation of this module is a stretch-goal. This module identifies the four corners of the “board” from the camera feed to auto-calibrate and scale the maze map. The boundaries of the “board” would be demarcated by black gaffer’s tape, which this module would identify via center of mass calculations and determine NE, NW, SW, and SE corners of the rectangular board.

3 Implementation

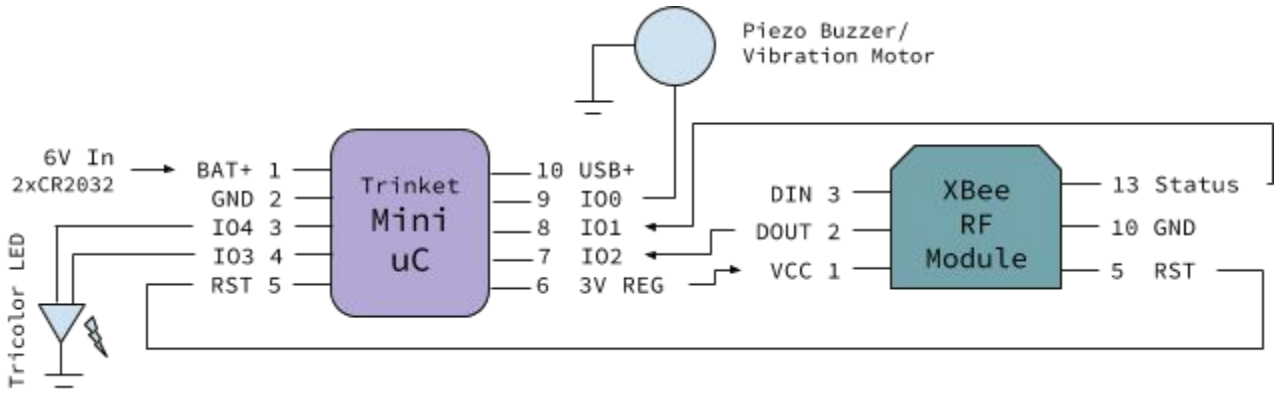
3.1 Finite State Machine Diagram

Implementation of our game will rely heavily on the use of a finite state machine that will dictate various signals between modules. This finite state machine will be implemented as follows:



3.2 RF Transmitter and Haptic Feedback

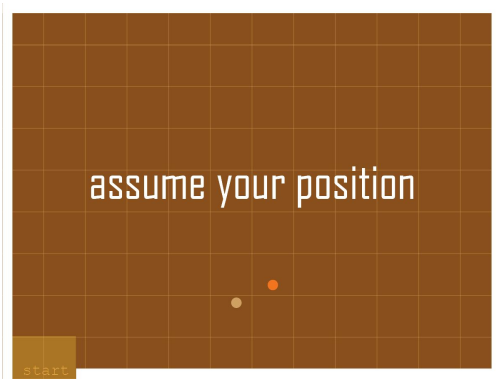
The circuitry for the electrical components on the glove are shown below:

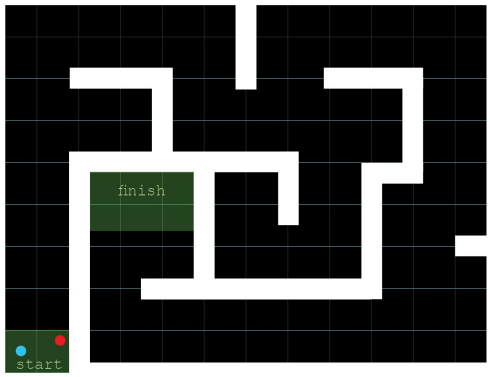


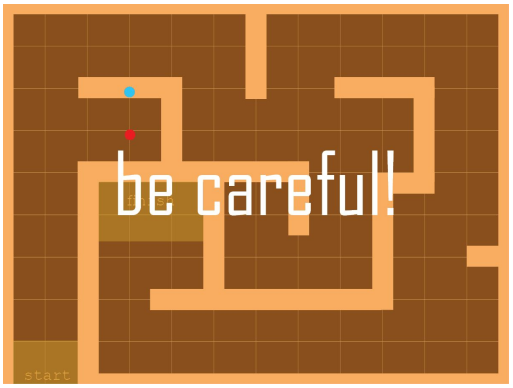
4 User Interface

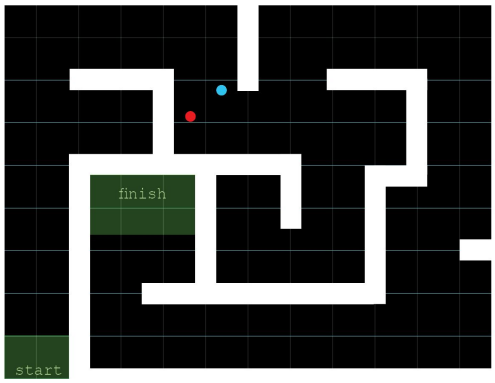
The player navigates through the three menu options using the up and down buttons, and selects the desired option. “How to Play” and “High Scores” options display static text. The player can return to the main menu from either of these two pages by pressing the left button.

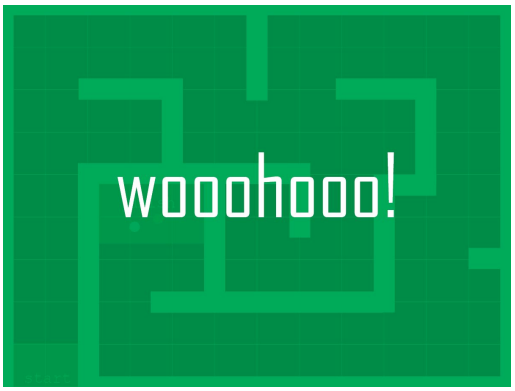
When the player selects new game, the game waits for the player to go the demarcated start location. Only when they are in the start location will the selected maze appear on the computer screen for spectators. When the player is in the correct location, the maze will appear and spectators can follow the player’s progress through it. Occasional warnings will flash on the screen (i.e. “Be Careful!”) if the player is “in a wall” for too long and too deeply.











5 Intermediate Testing and Milestones

These are the milestones we wish to individually test and the order we believe we must complete them. There are separate milestones in each individual module and also milestones to be reached to integrate the modules together.

5.1 Hand CoM Calculator

- Display CoM as seen by camera directly onto VGA monitor
- Ensure that CoM calculation is accurate, can distinguish between two hands (even when they are close together)
- Ensure that the calculations are quick enough (>60 Hz refresh rate) so that rapid movement can be detected
- This will be used in the Standby state of the FSM

5.2 Maze Map Generator

- Input different maze maps, scale and show on display
- Overlay selected maze map with hand CoM locations (see Maze Map Display)
- With the CoM, ensure that the corners of the frame correspond with the maze corners

5.3 Maze Logic FSM

- Build timer module for countdowns and countups
- Test each state in software simulation
- Create a virtual maze game that controls the hand locations throughout the maze
    - BTNU and BTNR for right hand forward and backward
    - BTNL and BTND for left hand forward and backward

5.4 Map Display

- Create each screen and overlay and test individually using switches to call up different screens (independent of state)
- Integrate with Maze Logic FSM to make sure displays correct screens and overlays

5.5 Haptic Transmitter

- Build and test the receiver electronics
- Simulate FSM states via switches
- Integrate with Maze Logic FSM

6 Resources

We will be using two 6.111 labkits running in parallel for our project. We will also be using two NTSC cameras hanging from the lab ceiling to achieve an 8 foot by 5 foot space to play the game. Each NTSC camera will be hooked up to a separate labkit.

In terms of component resources, we will be purchasing the following:

3	Digi Multipoint RF Modules XB24-API-001
2	Adafruit 3.3V Trinket Mini uC
2	Vibration Motors
2	Tricolor LEDs
4	CR2032 3V Coin Cell batteries
2	2xCR2032 Battery Holder

7 Timeline

Blue - Steph, Red - Libby Green - Both

	10/26	11/2	11/9	11/16	11/23	11/30	12/7
Proposal and Planning							
Camera/Object Recognition							
Map Generator and Display (UI)							
Maze logic FSM							
Haptic Transmitter							
Integration and Debugging							

8 Conclusion

Ideally, we project the implementation of our project will be straightforward with a few major challenges. The largest hurdles will most likely be communicating camera data between the two labkits, sending commands reliably to the Piezos to provide haptic feedback, and timing for retrieving the data from ZBT memory.

Overall, we hope to not only deliver a 6.111 final project that integrates many of the things we have studied in class, but also a fun game to play!