Boston University
Electrical & Computer Engineering
EC464 Capstone Senior Design Project

User Manual

Submitted to

Thomas DC Little
8 St. Mary's Street
Room 426
(617) 353-9877
tdcl@bu.edu

by

Team 4
Wizard's Chess

Team Members

Jeff Leong jeefle@bu.edu
Belle Zhang zbelle@bu.edu
Andreas B. Papadakis apapadak@bu.edu
Jake Thornton thornt17@bu.edu
Devin Chen djchen@bu.edu

Submitted: 4/11/18

Wizard Chess User Manual

Table of Contents

Executive Summary

Technology moves at such a fast pace that if there are no proper use cases or adoption, it dies out quickly. The life cycle of emerging technology relies on proof of concepts that later lead to other applications, either commercial or academic.

Our project seeks to design and build a hands-free, voice-controlled robotic version of chess by utilizing emerging technologies such as augmented reality, IoT, indoor light positioning, swarm robotics and blockchain.

Voice control and mixed reality innovates in the field of human-computer interaction. IoT robots change how we generate and interact with data. Our indoor positioning system sets a groundwork for future development for more advanced systems where multiple entities are identified. Finally, blockchain enhances trust and data validity that a simple excel sheet or database cannot offer.

All of this is packaged in an approachable format with Chess. This encourages people of all ages to be exposed to these new technologies, spreading outreach and eventual adoption.

1 Introduction

Wizard's Chess was derived from the popular movie *Harry Potter*, where one would give chess pieces voice commands and a piece would move to a specific space. Born out of a love for Harry Potter, the idea of Wizard's Chess came to be. After considering the technical challenges of building such a system, our client concluded that boundaries would have to be pushed in order to make this game a reality. New technologies would have to be used and explored, and overall proves to a great technical challenge that is wholly possible. With this task in mind, a full project was conceived with the idea of being able to expose the masses to emerging technology in a simple-to-use, yet technically complex product.

The task ahead of us implements *indoor positioning technology* combined with *IoT, mixed reality, and blockchain* into a game of chess that non-technical people can enjoy and experience. The seamless integration between these technologies allows us to leverage the benefits of each and work in tandem to create a friendly user experience.

Each component chosen to solve this problem has its purpose. IoT allows us to have robots that are controllable and interactable. Using an *ESP32* microcontroller, we are able to have networked robots due to the onboard WiFi chip. These will function as physical pieces. This segways into indoor positioning technology using *Optitrack*, an IR camera based positioning system that that recognizes objects and their x, y, z positions in space. Using this system, we are able to make the robots self aware of mechanical error and location error. Optitrack is networked as it sends correction data to the robots for the robots to keep a straight line. Mixed reality, specifically, augmented reality was chosen in order to show the animations of chess pieces. By having an augmented reality overlay, we can also show the use of having a projected image versus a fully virtual image. Things are more tangible and spatial which can have applications further on. Finally, blockchain was chosen to be implemented in this project due to its function of data validity. Users can only change history and a game state - imagine a game's save file - by interacting with the game. *Ethereum* was chosen for its smart contract capabilities. This essentially functions as a normal contract with transactions and valid data binding however instead of a physical contract, this contract is between software components.

A special feature of our project is the use of a *Microsoft Hololens.* This device allows us to create our augmented reality environment. The Hololens is essentially a computer, allowing a full game to run on it while also projecting holographic images on specific areas. The Hololens' multiple cameras and sensors also allows it to sense locations in relative space. Therefore one can coordinate the Hololens field of view with physical components i.e. the robots. One could theoretically walk around through the playing field to get a different perspective. It's only caveat is that you could get too caught up and walk directly into your game! Please be aware of your surroundings!

The resulting product of our year's work is elegant, efficient, and fun. Anyone can enjoy this game, whether they have years of experience with these technologies or have never heard of them before. It is truly a hands-on learning experience. Any user can become aware of the functionality of these emerging technologies through an interactive game of chess.

2 System Overview and Installation

2.1 Overview block diagram

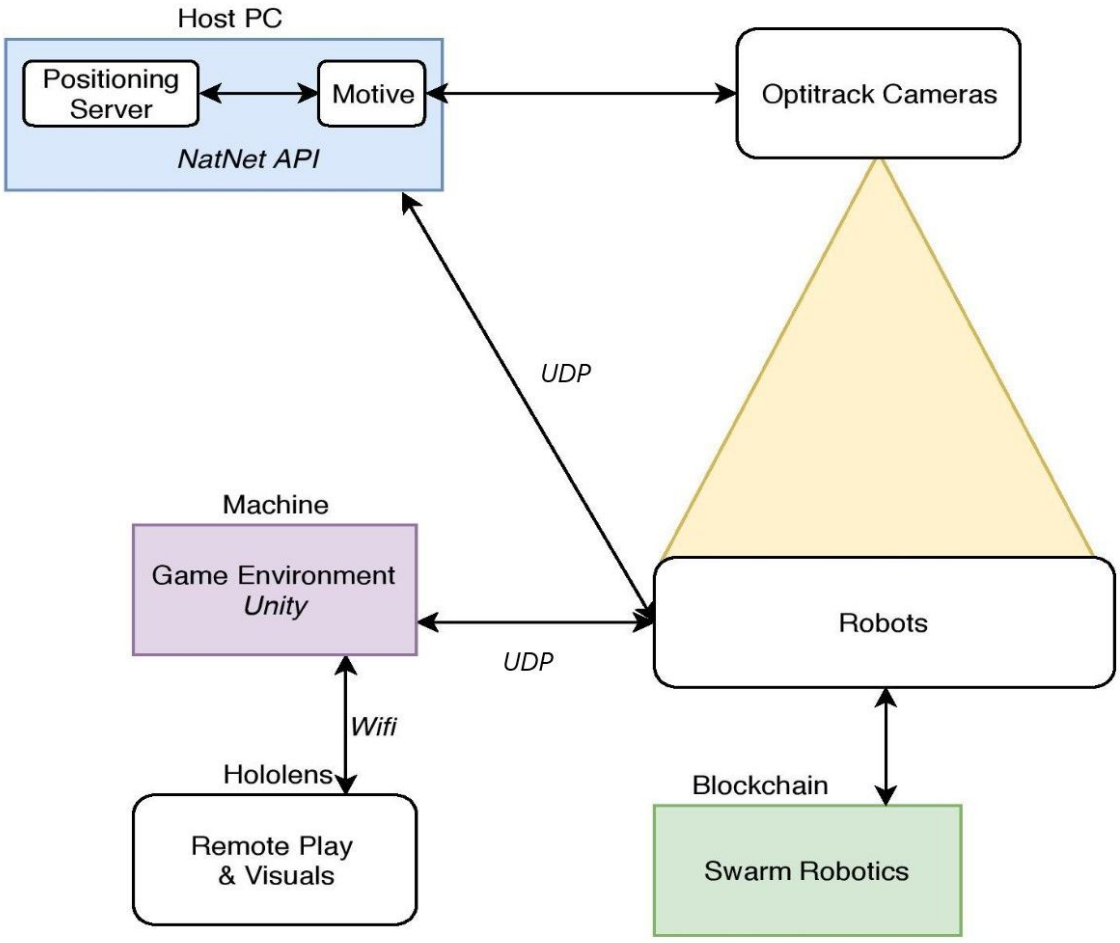


Figure 2.1 The game board is seen through the Hololens. The user then gives a voice command that is handled by the Hololens. That command communicates with Unity, the game environment, to move the robots which are tracked by Optitrack cameras. The cameras communicate to the robots in order to correct their course. Blockchain then saves a robots position once it stops moving. UDP connections are used to exchange data between hosts.

2.2 User interface.

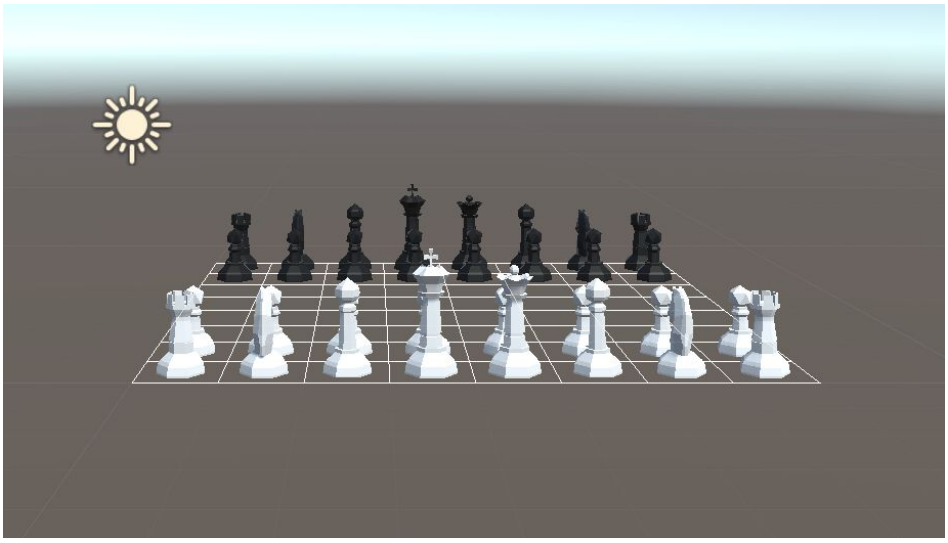


Figure 2.2 A Chess Game similar to this screen capture from Unity can be seen by the user through the Hololens. When a voice command is given, the pieces move in both physical and virtual space.

2.3 Physical description.

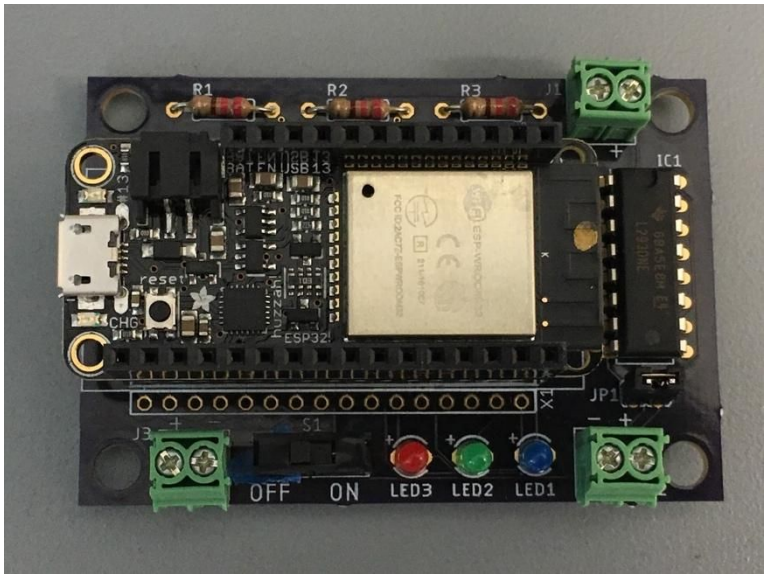
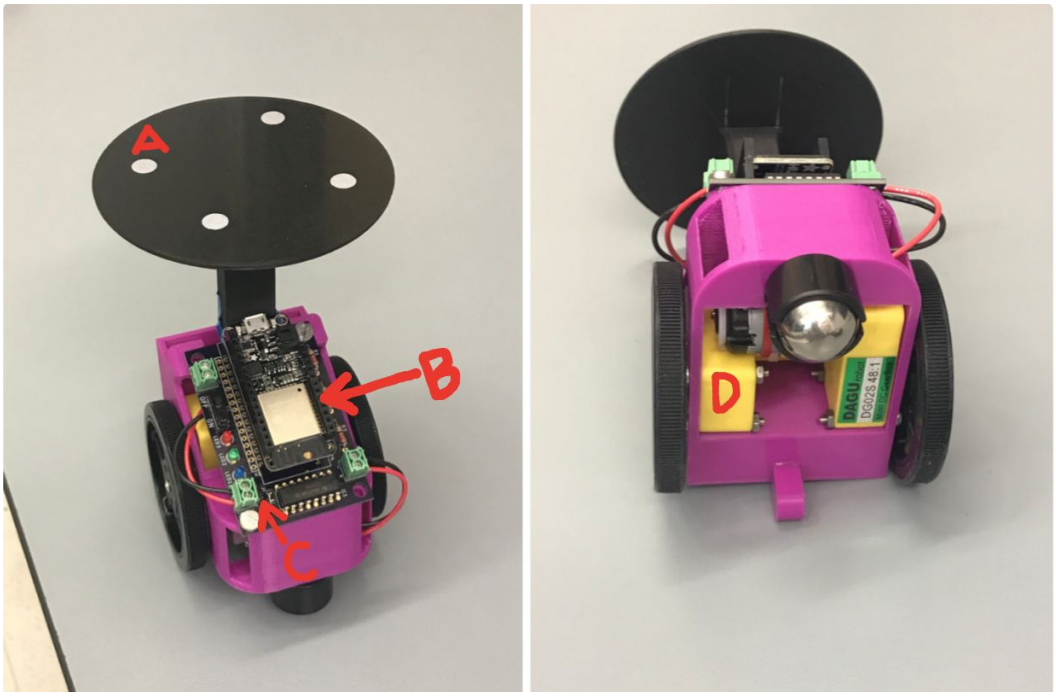


Figure 2.3 The PCB design uses an ESP32 microcontroller, a power switch, indicator lights, an H-Bridge motor, and a driver jumper



Figures 2.4 and 2.5 A fully built robot with the PCB attached. Pieces are rendered virtually on top of it. A) Markers for Optitrack Positioning B) ESP32 Microcontroller C) PBC D) Motor

Figures 2.6 and 2.7 An overlay of the user's view of two robots through the headset.

2.4 Installation, setup, and support

In order to use the Optitrack cameras, Motive, a program complementary to Optitrack, needs to be running in the background. Cameras are calibrated using a wand with IR reflectors and the Motive project. The robots must be placed according to the markers placed on the floor (or however the user wishes to scale the board). In order to run SampleClient.cpp, all external dependencies (given by NatNetSDK) must be in the same directory. SampleClient.cpp must be opened in Visual Studio 2017, due to the dependencies of NatNetSDK. Once running the code, a figure window will show up, displaying confirmation of data interactions, acting as a server.

When opening the game in Unity, the user must make sure that the list of chess pieces is declared in the main game object. The pieces are listed in order of their starting positions on the board, beginning with the White Rook. Once completed, the game must be exported to the Visual Studio Solution.

In File > Build Settings, change the platform to Universal Windows Platform. The SDK should be set to Universal 10, the target should be HoloLens, and the UWP Build Type should be D3D. In Player Settings > Other Settings, set the Scripting Backend to .NET, and in Publishing Settings ensure that InternetClient, InternetClientServer, PrivateNetworkClientServer, and Microphone are enabled.

Build and then create a new folder for the exported app. Open the app folder and the created Visual Studio solution. Once open, change the target to Release and x86. To deploy to a mixed reality device over Wi-Fi, change the deployment target to Remote Machine and open the Holographic Remoting Player app on the Hololens. Enter the IP address of the Hololens and change Authentication Mode to Universal. At this point, the game should be visible through the HoloLens. After the Remoting Player is closed, the game is still accessible as an app on the Hololens.

To set up the blockchain, one just has to verify WiFi connection. The blockchain is running on a separate PC with a smart contract deployed. A user does not have to download any software in order to have their robots/game interact with blockchain.

3 Operation of the Project

3.1 Operating Mode 1: Normal Operation

In a normal operating mode, a user will interact with the game as follows:

1. The user provides a voice command with the start position and end position of a chess piece and move. These commands correspond with algebraic chess notation with coordinates “a” to “h” and 1 to 8. A sample command would be: “a1 to c3.”
2. Users will alternate moves with an opponent in a similar fashion to step 1.
3. The voice command “reset” will return the game to its initial starting stage.
   *This feature is yet to be implemented

In this mode, a user will only have the option to interact with the game using the commands as seen above.

Normal consequences of user actions include misheard commands not recognized by the Speech Manager.

3.2 Operating Mode 2: Abnormal Operations

Abnormal operations include a robot running out of battery, robots interfering with each other’s movements, power outages to networked components.

To exit this state of abnormal operations, the game should be shut down by calling the Hololens menu to return to home and close the game app. It may then be set up again as described in section 2.4.

3.3 Safety Issues

It is important to be mindful of your surroundings when using the AR headset. The equipment in Wizard’s Chess is fragile, and it is easy to lose one’s bearings in virtual space. For maximum safety, players or bystanders should maintain at least an arm’s length away from each other and any robots or other equipment (i.e. Tripods, Wires).

Due to the complexity and amount of parts in this project, users must pay attention to wires as a tripping hazard.

4 Technical Background

Unity:

The chess pieces are assigned to begin the game at their starting positions, at which point a Piece class is used to store the location, IP address, and port numbers of the associated robot. When a voice input is detected, the command is parsed to obtain the start and end positions of the chess piece. To make a move, the script searches through a list of the robots' locations, finds and connects to the chess piece that corresponds to the start position, and sends that robot end coordinates. While the robot moves towards its destination, the robot sends its real-time coordinates back to Unity, and the animated chess piece uses those coordinates to update its position on the board.

Robots:

Design:

The design of the robots is made to be as compact as possible while maintaining a low cost per robot. The right angle of the motors allowed it to be oriented in a way to not compromise the length or width of the robot. The ball caster tilts the robot back to make space for the motors but a small appendage at the back of the case prevents it from tipping over. For the Optitrack marker stand, it is held in by one of the two screws securing the PCB as well as a small nub which aligns it along the edge. This stand allows for a flatter surface to create rigid body definitions from.

Hardware:

The hardware consists of a 3D printed frame and Optitrack marker holder, two DC motors, a ball caster, batteries and a custom PCB. The frame is printed in two parts with the main frame and a small appendage glued to the back. Overall for the parts for one robot, the printing time is around 7 hours with an average printing speed of 55mm/sec. The custom PCB has a few extra features such as extra pinouts for the microcontroller, three indicator led lights, on/off switch and the ability to separate the motor and h-bridge logic voltage or connect them with a jumper.

Software:

The main core of the robot is the UDP server and client, HTTP client, and the PID computation. The UDP server is primarily used to communicate between the robot, Hololens and Optitrack. The HTTP client is used to push data onto the blockchain and the PID controller calculates the adjustments the robot needs to move in a straight line. The other features in the software include calculations for approximating a turn and detecting when the robot reached its destination.

Optitrack:

Optitrack acts as a server. It waits for a robot to connect to it, as well as receives the final coordinates of where the robot needs to go. Optitrack then begins to read the X and Y positions of the robot (defined as a rigid body in a Motive project) and sends it to the robot at every frame until the robot gets to its final position. It then stops connection and waits for another connection. These connections are handled through UDP.

Blockchain:

A private, centralized Ethereum blockchain was chosen in order to facilitate speed of block mining, ergo data writing. This blockchain implementation acts like a read only database for any outside user. Only a robot may write to this blockchain. Because of the immutability of blockchain, users may not tamper with data. Robots will post data to the blockchain at their final position.

Connections:

Hololens to Robot:

This is done using the DatagramSocket() low level class. Due to publishing settings on the Hololens, any high level class such as UDPClient cannot be used, and thus usage of Asynchronous methods with DatagramSocket() is needed. The type of software replicates a UDP connection, meaning data is being transferred as fast possible, and latency for initiating communication is reduced.

Robot to Optitrack and Unity:

This is done with the UDP client on the robot. All addresses and ports are registered within all the robots and robots will only communicate with Optitrack to transfer coordinates of where it will go. Optitrack then responds by sending coordinate data to the specific robot that sent the message. The robot sends data the same way to Unity but it sends its own coordinate data until it reaches the desired destination and then it sends a message to Unity to update the game logic.

Robot to Blockchain:

This is done with the HTTP client on the robot. The robot will take its position data and send a JSONRPC string to the blockchain address and smart contract that will then write the data.

5 Cost Breakdown

Project Costs for Production of Beta Version				
	Quantity	Description	Unit Cost	Extended Cost
1	1	Microsoft HoloLens	$5000	$5000
2	34	Motors	$5	$170
3	34	PCB + wires	$4	$136
4	136	Batteries	$1	$136
5	34	Wheels	$5	$170
6	34	ESP32 Microcontroller	$20	$640
7	3	3d Printing Filament	$15	$45
Beta Version-Total Cost				$6297

The main expense of the project is the Microsoft HoloLens. The headset was contributed by the client for this project, though the device normally costs $5000. Our customer also supplied the ESP32 microcontrollers.

The main constraint of this budget was constructing thirty-two robots plus two extra, each costing less than $20 to produce, not including the provided microcontrollers.

For the purposes of this project, free versions of Unity and Maya were utilized in addition to Visual Studios Enterprise and Motive. Other software used was open source. all in all negating the cost of software.

6 Appendices

6.1 Appendix A - Specifications

Requirement	Value, range, tolerance, units
Board Dimensions	1m x 1m - 2m x 2m
Robot Diameter	Maximum of 10cm
Robot Position Error	±2cm
Robot Velocity	Minimum of 0.1m/s
Cost of each Robot (PCB, motor, wheels)	Maximum of $20

6.2 Appendix B – Team Information

Devin Chen
Worked primarily on the hardware, software and design of the robots.

Jeff Leong
Worked on implementing blockchain on IoT. Future consultant at Accenture.

Andreas B. Papadakis
Worked on Optitrack and robot integration, along with networking for all components.

Jake Thornton
Will be working as a Technical Sales Engineer for Texas Instruments after graduation.

Belle Zhang
Worked on Hololens graphics and Unity interaction with the robots