Department of Mechanical and Mechatronics Engineering

Autonomous Tic-Tac-Toe Playing Robot

A Report Prepared For:
Department of Mechanical and Mechatronics Engineering
University of Waterloo
200 University Ave W.
Waterloo, Ontario, N2L 3G1

Prepared By:
Group 58
Kamin Andres Cheung Zhang, Hanut Pandey, Kristoff Stefeo Sahadeo and Victor Sung
Candidate for Bachelor of Applied Science, Honors Mechanical Engineering
200 University Ave W.
Waterloo, Ontario, N2L 3G1

July 25, 2018

Table of Contents

List of Figures

List of Tables

Summary

This report details an engineering design project undertaken by a group of first-year Mechanical Engineering undergraduate students. The goal of the project was to design a device that plays a game of Tic Tac Toe against an opponent autonomously while being constructed with parts from the Lego Mindstorms EV3 and Tetrix kits. The Engineering design process was adhered to by the group in producing the final design, to ensure that the best design was developed in the end. During the conceptual design phase of the design process, four different conceptual designs were developed using a morphological matrix. In order to select the best design, the group judged the concepts based on the decision matrix, which outlined various characteristics that the projects should have, and selected the best design which suited these characteristics. Different designs were developed, prototypes were created, and new causes of failures emerged. This allowed the group to iterate through various designs until the final successful robot was developed. Ultimately, a dual rack and pinion system, coupled with a ball dispensing system was chosen and the final prototype featured this design. The project featured the implementation of various sensors that worked together to produce a single functioning unit that played Tic Tac Toe autonomously with users. All the motion in the final prototype was also powered by Lego motors, and mechanisms which translated rotary motion to other forms. Many parts were fabricated both in the Engineering Student Machine Shop in E5 and WatiMake and were assembled and integrated with Lego by the group to produce the final design. To control the Lego EV3 robot, RobotC was used. The group faced a considerable challenge with the software as initially, the code was too resource intensive and the Lego EV3 was unable to run it. The group then worked towards optimizing the code for the Lego EV3 and tried to save as much memory as possible. Eventually after a lot of effort, the software finally worked with the Lego robot. Once the software and the mechanical design were completed, the group began testing the software on the robot to check the accuracy of the various sensors. Several test codes were developed to test the robot and to demonstrate the Engineering Design specifications. In the end, the project succeeded and fulfilled all the software and mechanical design requirements. However, a few recommendations to further improve the final design have been made.

1.0 Introduction

Artificial intelligence (AI) has become popular for a long time. Recently, AI has been seen in many practical applications such as self-driving cars and virtual assistants in smartphones devices. It is clear that the technology will take a prominent role in the future. This project's main goal is to develop a Lego robot that uses AI to play a game of Tic - Tac - Toe against a person, that is, an autonomous Tic-tac-toe playing robot. Tic-tac-toe is a two-player game. The two players take turns putting marks (X and O) on a 3x3 board. The player who first gets 3 marks in a row (vertically, horizontally or diagonally) *wins* the game, and the other *loses* the game. If neither player is able to form a row of 3, the game results in a draw. Due to the game's simplicity, it is often used as a testbed for artificial intelligence algorithms. To create the robot with Lego parts, the engineering design process will be followed to ensure the completion of the project. The engineering design process is a series of steps that engineers follow to create a functional product. Hence, this project will increase familiarity with the engineering design process, and the implementation of AI in products.

2.0 Needs Analysis

Once the project description was introduced in the class, the members of group 58 began working towards selecting an appropriate project topic. Several brainstorming sessions took place where different ideas were brought forward by the group members. The project should have sufficient complexity to provide a significant challenge. All the team members were inclined towards the idea of recreating a classic two player game and having a computer program play against a human opponent. After several suggestions from the group members, Tic-Tac-Toe was collectively selected as the game of choice. The group members felt that recreating Tic-Tac-Toe would be a significant challenge both in terms of software and mechanical design. After the project topic was finalized, the next step was to identify the prime design requirement of the project which would be represented with a needs statement. The needs statement formulated by the group for the project is as follows:

"*A need exists to create a device that plays a game of Tic Tac Toe against an opponent autonomously while being constructed with parts from the Lego Mindstorms kit"*

Once the real need of the project was identified and formalized, the functional requirements and the constraints of the project were identified. This was done as a part of the first assignment

on Needs Analysis, where the non-functional, functional and constraint requirements were identified. Using these requirements, the Engineering Design Specifications were derived.

Table 1. Initial Engineering Design Specifications.

No.	Characteristic	Relation	Value	Units	Verification Method	Comments
1	Cost	<	75	CAD	Analysis	
2	Mass	<	3	kg	Test	This is to verify whether the device is portable or not. Use Balance
3	Winning Percentage	=>	40	%	Test	Formula to calculate winning percentage: WP% $=\frac{(Wins + 0.5 * Ties)}{Total\ Games} *$ 100%
4	Time per Move	<=	15	seconds	Test	Use timer
5	Number of Fabricated parts	=>	5	pieces	Analysis	

As the group progressed through the project, more entries were added, and some existing entries were altered accordingly. The mass of the project, for example, was increased from 3 kg to 10 kg as the group members realised that their initial estimate of the project's final weight was too low, and the finished project will be heavier as various metal and wood parts would also be used in addition to the Lego Mindstorms parts. Another entry that was altered was the Winning percentage of the robot. This was replaced by the Losing percentage as an ideal game of Tic Tac Toe always results in a draw. This made the Losing percentage a more accurate specification as compared to the winning percentage.

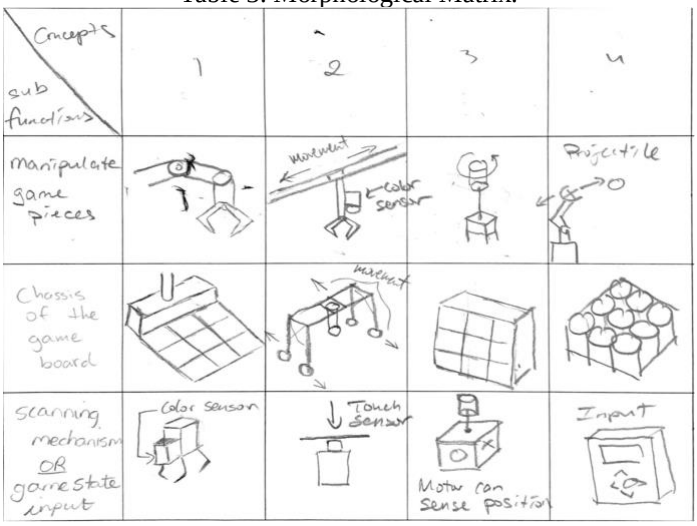
Table 2. Final Engineering Design Specifications.

No.	Characteristic	Relation	Value	Units	Verification Method	Comments
1	Cost	<	75	CAD	Analysis	
2	Mass of Product	<	10	kg	Test	Determined with a Scale
3	Time per Move	<	60	seconds	Test	Determined with a Timer. Start timer when the user press the touch sensor. End timer after the robot places a ball.
4	Accurate placement iof the chips on the board.				Demonstration	Validation will come through the robot displaying the accurate placement of the balls on the board.
5	Time per Game	<	500	seconds	Test	Determined with a Timer
6	Losing Percentage	<	5	%	Analysis	$L.P\% = \frac{\#\ of\ Games\ Loss}{Total\ of\ Games} * 100$
7	Ease of Use	>	2	1-5 Scale	Test	A rating of one means the game can only be played by people with high IQ. A rating of five means the game can be played by anyone who is between 10 to 75 years old.
8	Cheating Detection				Demonstration	
9	Accurate detection of chips on the board				Demonstration	Validation will come through the robot displaying the accurate detection of the balls on the board.
10	Autonomously play a game of tic-tac-toe against a person				Demonstration	

3.0 Conceptual Design

A morphological matrix was used to create 4 different design concepts for the project. The matrix consisted of three different sub-functions. These three sub-functions were the manipulation of game pieces, the chassis of the game board, and the scanning mechanism. In addition of the sub-functions, the matrix also consisted of 4 different concepts for each sub-function. The morphological matrix is shown in Table 3.

Table 3. Morphological Matrix.



Four different design concepts were devised solely using morphological analysis. The following figures demonstrates the four design concepts.

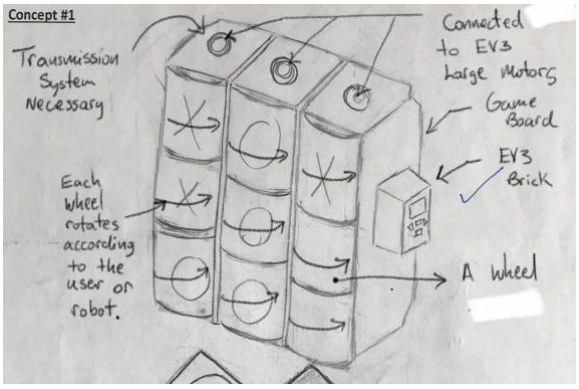


Figure 2. Conceptual Design #1.

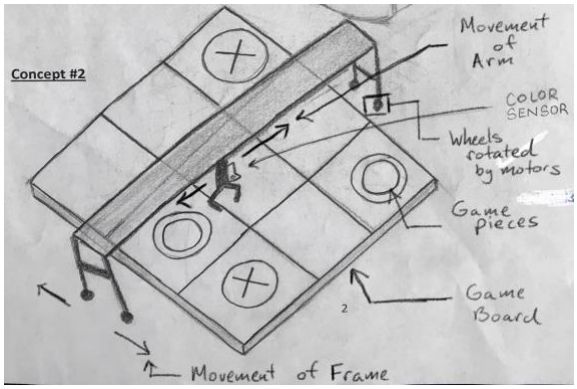


Figure 1. Conceptual Design #2.

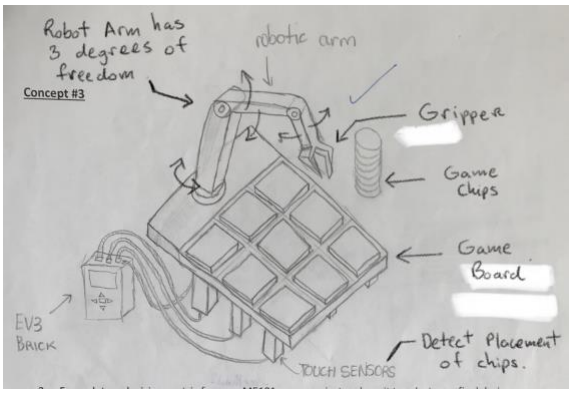


Figure 4. Conceptual Design #3.

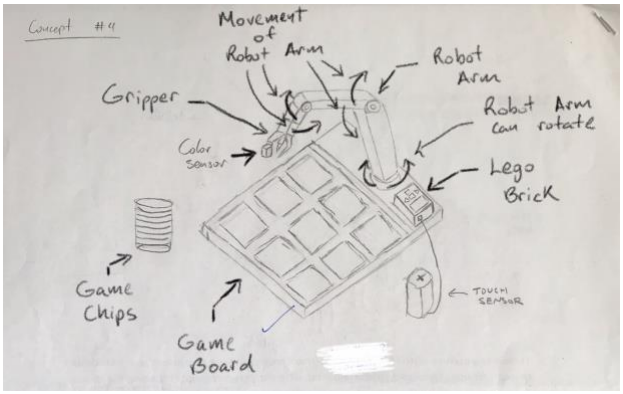


Figure 3. Conceptual Design #4.

Due to the many different design concepts,a decision matrix was formulated to select the final design concept. The decision matrix included 5 different criteria. These criteria were reliability, cost, manipulation of fabricated pieces, portability, and ease of user input. The decision matrix is shown in Table 4.

Table 4. Decision Matrix.

	Sketch of Datum (Concept # 3	Sketch of Concept #2	Sketch of Concept #1	Sketch of Concept #4
Reliability	0	-1	+1	0
Cost	0	-1	0	+1
Manipulate fabricated pieces	0	0	0	0
Portability	0	-1	0	+1
Ease of User Input	0	0	-1	0
Total	0	-3	0	12

The decision matrix compared the different design concepts with a datum, which is design concept #3 as shown in Fig. #. Through this comparison, the design concepts were carefully evaluated and examined to properly rank all the concepts. The ranking of the design concepts was based on whether a design concept was better or worse than the datum. The design concept is ranked +1 if it is better than the datum, and -1 if it is worse than the datum. However, if the design concept is neither better nor worse than the datum, then it is ranked 0. For example, design concept #2 was ranked -1 in cost because it was more expensive to create design concept #3 than design concept #2. After an extensive ranking process, it was determined that design concept #4 was the best among all the other concepts because it had the highest total score. The total score was obtained by summing all the rankings

4.0Mechanical Design

4.1 Chassis and Frame Design

The most important decision, as it regarded to the chassis and frame design for the project was to fix all the parts to a single 61 cm x 61 cm wooden board. The material of the board chosen was a medium-density fibre board as it offered good tensile strength, while being lightweight and affordable and easily obtained. This base was critical since our project relied on multiple pieces being fixed at various positions. During operation of the autonomous tic tac toe robot, it was vital that accuracy was maintained, as inaccuracy would result in the incorrect placement of the playing pieces. To maintain this accuracy, the various parts of the robot was fixed to a single board, various sensors and motors needed to reference each other, and a zero-position was established. Refer to the image below to attain a visual perspective of the base board.

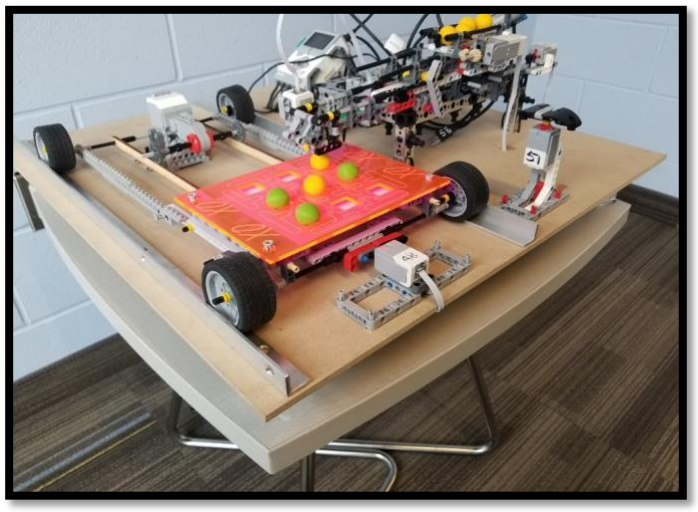


Figure 5. Autonomous Tic-Tac-Toe Robot's Structure.

The functionality of the robot involves resetting the position of both rack structures (board and playing piece dispenser) after each move, by running the motor until the touch sensors are touched. This position is used as the reference position. Another purpose of the base board was to offer a medium for cable management. Holes were drilled in the board and wires were managed below the board.

The base board was supported by four L-shaped Aluminum supports, approximately 6 cm tall. This lifted the entire base platform above the surface it was placed on. This allowed for easy cable management. This increases the aesthetic beauty of the design by hiding all the

unattractive wires from the user, as well as separated the wires from the moving parts of the design was prevents the obstruction and entangling of the wires with the various kinetic mechanisms in the robot, which would result in mechanical failure. The wires were secured with clear tape, as its adhesive power was adequate for this purpose.

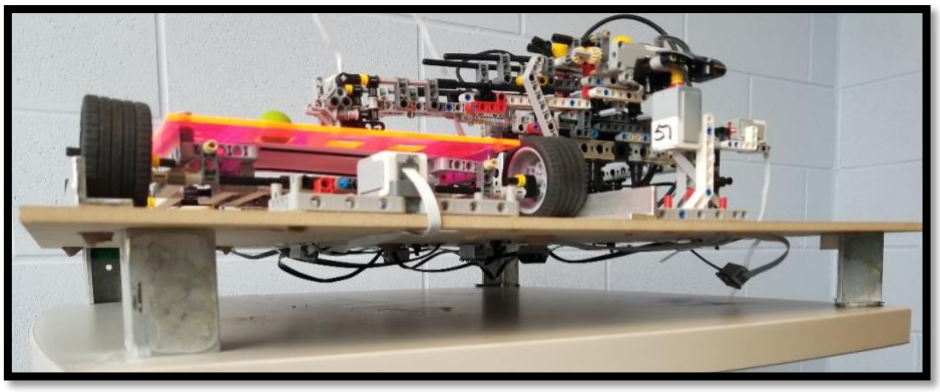


Figure 6. Robot Support Structure.

The holes through which the wires were channelled through were filled using light brown polymer clay which matched the colour of the board, to improve the overall aesthetics of the design, as well as, to secure the wires firmly into place. See the image below to gain a visual representation of the plastered effect described above.

Figure 7. Base Board Hole Fillings.

Most components were secured to the board by drilling holes in strategic locations and inserting Lego pins inside. These pins were attached to the Lego construction for the associated components. The result was a stable and accurate placement construction. The EV3 brick stand was built so that the brick would be at 60° angle with respect to the base board. This allowed for easy cable management, while being at a convenient angle for users to read the display and interact with the buttons on the brick. The various touch sensors secured directly to the base board, including the user input button, were also secured using the construction indicated

above. Tetrix pieces were implemented in this design, primarily to support the secondary rack and pinion structure. This rack moved the ball dispenser into appropriate positions, but also mobilized the scanner in the direction perpendicular to that of the motion of the Tic Tac Toe board, facilitating the two-axis motion needed to scan and dispense the pieces accordingly. This structure was secured to the board using black zip ties. This method of securing the structure was very effective as it allowed for negligible movement of the construction in any

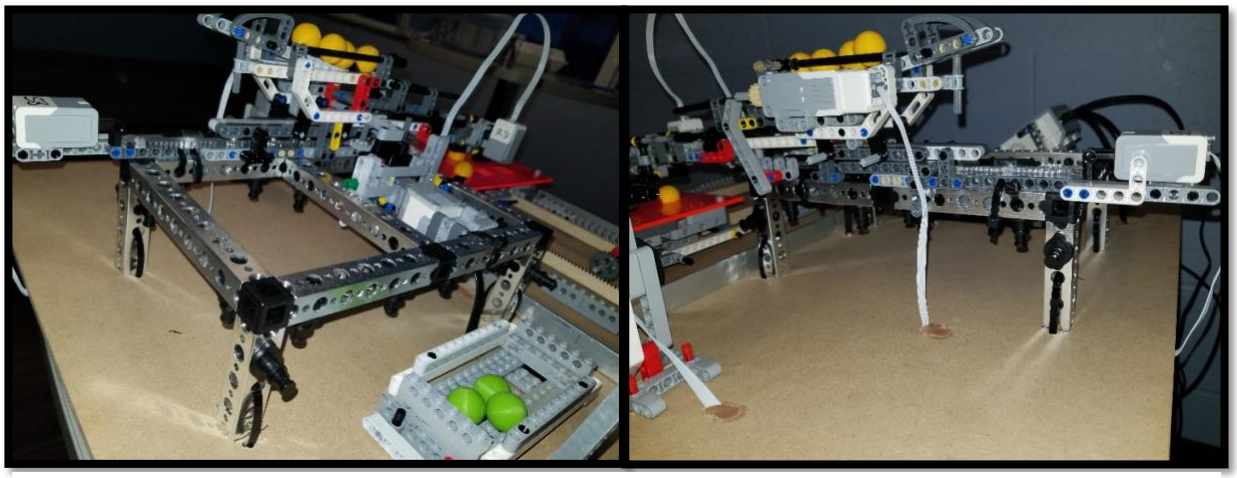


Figure 8. Tetrix Construction.

direction even while under operation, making it ideal for the rack system. Refer to the figures below that depicts the final structure of this component.

4.2 (Motor/Drivetrain Design)

The final design for this project involved the use of three (3) motors, namely, two large motors and a medium motor, as provided by the Lego Mindstorms EV3 kit.

4.2.1 Lower Primary Rack and Pinion Structure

The Tic Tac Toe board platform was constructed with mostly Lego parts as the racks alone did not provide enough structural support to keep the entire design stable. As seen in Figure 1 above, the structure was reinforced using Lego pieces, to prevent destabilization or deformation during operation. The game board was mounted to the structure by cutting four reference holes and fitted with Lego pins which holds the board securely. The platform was driven by a large Lego motor fixed between two racks and a single shaft drove both racks simultaneously. A rack and pinion was incorporated because it locked the board when the motor is turn off which minimizes error. A large motor was chosen as it offered the power required to easily drive the relatively large playing board component. It also contained the

feature to directly drive two racks simultaneously, as it could accommodate a single shaft running through the motor, a feature not available in the medium motor. The figure (10) below shows this motor in the final design.

The gear used as the pinion had a diameter of 2cm (, and thus provided a desirable speed to torque ratio, when driving the system.) The entire platform was mounted on four Lego wheels of diameter 5.5cm. These provided adequate elevation, as well as, reduced friction necessary to facilitate the smooth and accurate movements required for this robot. To ensure the board only traveled in one direction, Aluminum rails were installed on the base board and secured using wood screws. The rails also add structural rigidity to the base board. Refer to Figure 1 above to gain a visual perspective of this design.

4.2.2 Upper Secondary Rack and Pinion Structure

The colour sensor and ball dispensing mechanism were driven using a second rack and pinion system. Once again, this system was preferred over a free rolling system, as it offered more precise positioning, as it reduced effects due to inertia. The rack was fabricated using clear acrylic. Holes were placed strategically along the rack so that Lego shafts would fit at various points. They offered a way of fixing Lego parts to the rack. The teeth of the rack were also designed to be compatible with the Lego gears provided. The rack system was driven using the second large Lego motor attached to the Tetrix support structure. This motor was secured using a smart integration of Tetrix and Lego pieces. Although the Lego pins were not an ideal fit into Tetrix, when multiple support points were used, the motor became very stable, and this construction proved sturdy.

A large motor was also chosen for this system as it involved the motion of the second heaviest component in the project, the ball dispenser and color sensor mount. This system operated similarly to the motor driving the primary rack, however, only one rack was driven in this set up. The gear driving the rack was the large Lego gear and was approximately 4cm in diameter. The set up was oriented in such a way that the face of the driving gear was parallel to the surface of the base board. Refer to the image (11) below for more details.

The rack itself slid on the flat surfaces of fixed Lego pieces. Lego pieces were also used to act as a guiding mechanism for the rack. As one can imagine, the rack did not fit perfectly inside the Lego guiding structure. To fix this problem, zip ties were cut, and glued together, and fixed on the inner guiding rails of the Lego track using tape, to reduce the tolerance of the

cross-sectional area allocated for the rack. The result was a very stable rack that could only move in a single direction. The guiding structure was secured to the Tetrix support structure

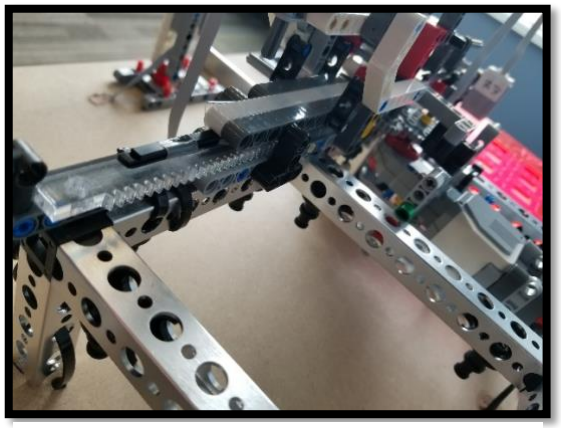


Figure 9. Secondary Rack.

using black zip ties. The gear used to drive the rack had an effective diameter of approximately 4cm. A second smaller gear was also attached, which ensured that the rack moved in one direction, without chance of pivoting. Refer to the image below for more details.

The ball dispensing system incorporated a medium gear that drove a rotating mechanism, with three angled arms. As the motor turned the arm 120°, a single ball would be dispensed. The balls were stacked at an angle to the horizontal, and placed within guiding rails, therefore the stored gravitational potential energy facilitated the mechanism to be self-feeding. To place the playing pieces, the secondary rack would extend out a pre-calibrated distance, and the dispenser would release a single ball. Due to the angle, and the relative elevation of the balls, they would roll down the rails, until they reached the end of the rack and drop into the according position. This system proved to be sufficiently accurate for our design, as the accuracy was tested and verified. A medium motor was utilized in the ball dispensing mechanism. The medium motor provides sufficient power required to rotate the dispenser fan.

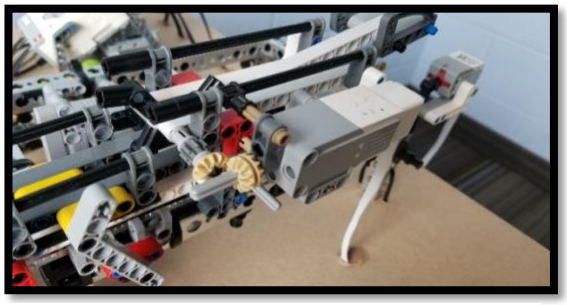


Figure 10. Medium Motor.

Accuracy was required in terms of encoder count. The medium motor proved to be the ideal solution for this scenario. A bevel gear was used to translate the motion by 90° so that the medium motor could be placed along the length of the rack structure which allowed the components could fit in a more tightly packed design. Finally Refer to the figure below for a view of the medium motor in this design.

4.5 Sensor Implementation

The final design for this project incorporated the use of touch sensors, motor encoders, a colour sensor and timers. Multiple touch sensors were used in this project, and each served a crucial role. The first touch sensor was fixed on the base board. It acted as a physical reference point for the primary rack mechanism. Similarly, another touch sensor was mounted at the back of the secondary rack system as reference for the upper rack mechanism. When the rack touches the touch sensor, the motor is was turned off, and this point acted as the reference point. Having these two reference points is important to ensure the robot's accuracy. The motor encoders are returned to the reference point after every major movement. The last touch sensor was implemented as a user input button. This button would be pushed once the user is finished making their move which would begin scanning the board to determine the move made. Refer to the figures below for the various touch sensors used in the project.

Motor encoders were used extensively in this project. The encoder built into the large motor driving the primary rack system was used to calculate and move the game board specified distances when performing various tasks and the motor encoder in the second large motor used to drive the secondary rack system was utilized for the same purpose. Finally, the last motor encoder implemented in this project was that used in the medium motor. The ball dispenser has to turn 120 degrees at a time to dispense a single ball.

The colour sensor was attached to the secondary rack system, and was used to scan the game board state, after the player made a move. The game involved using both yellow and green playing pieces. The robot would play using yellow playing pieces while the user used green playing pieces. The colour sensor was used to distinguish these colours when scanning and worked with the encoders and touch sensors to determine which spots were occupied by the playing pieces, who played them, whether the move made was valid, and whether someone won, lost or drew.

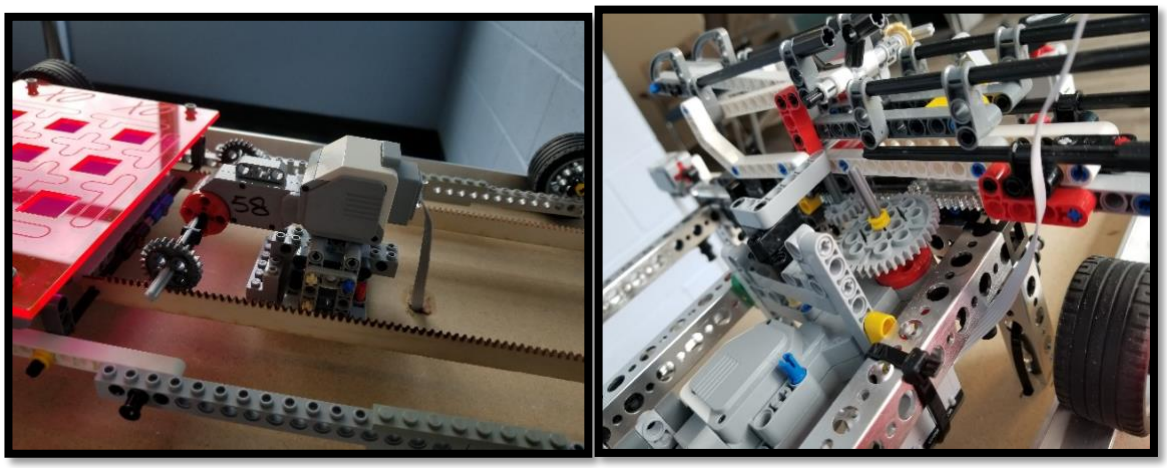


Figure 11. Large Motors.

Lastly, two timers were utilized in this project. One timer was allocated to determine the entire length of a single game, from the time the user indicated whether he/she wanted to play first or second, to the time the final game state was scanned and conclusions about the game was drawn. Another timer was used to give the user 60 seconds to make his/her move before reminding them to play. It would then give the user another 60 seconds to make their move, and if no input was detected, the program would reset the board then end.

As shown, a number of sensors were used in this project, and they all worked coherently to result in a robot that functioned as expected when it came to autonomously play a game of Tic Tac Toe with someone.

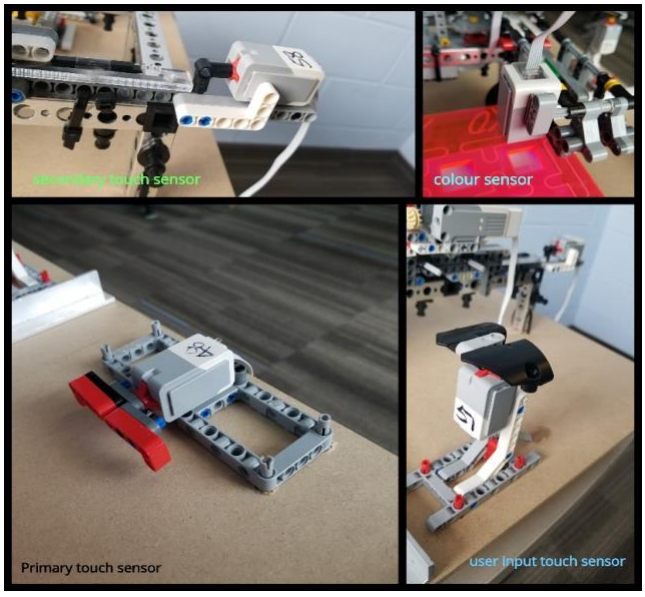


Figure 12. Various Sensors.

4.6 Fabrication

When constructing the physical mechanical parts of this project, several parts were fabricated using the equipment present at the Engineering Student Machine Shop (ESMS) in E5, as well as WATIMAKE in the University of Waterloo.

Firstly, during the prototyping stage, an initial concept game board was laser cut in WATIMAKE. This board was not used in the final project design because it was not the correct dimensions. A playing piece prototype was 3D printed which was not used, but future iterations were created based on this piece, with improvements such as reduced size, and varying colour. The smaller playing pieces used in the final design were 3D printed using the large printers at WATIMAKE. Without these pieces, the game could not be played. A number of the prints failed, but this may have been due to the small size of the playing pieces, as well as their spherical shape.The final game board was laser cut from ¼ inch thick coloured acrylic material in WATIMAKE. Refer to the image below of the game board. The racks included in this project were made from both wood and acrylic. The two wooden racks and the acrylic racks that were used in the primary rack system and the secondary rack system were both fabricated using the laser cutter in WATIMAKE.

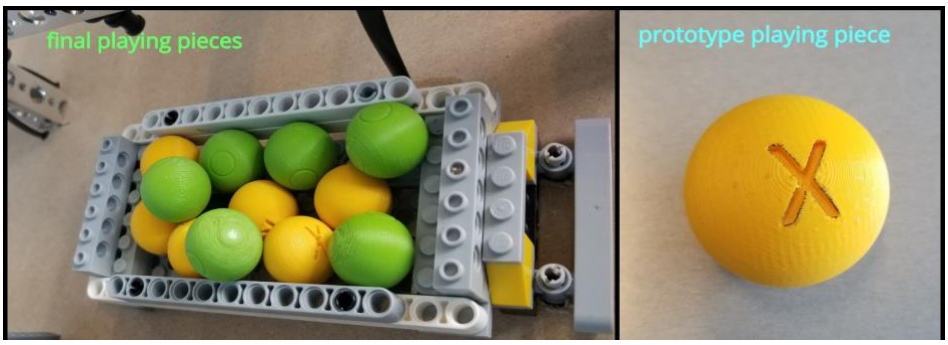


Figure 13. Playing Pieces.

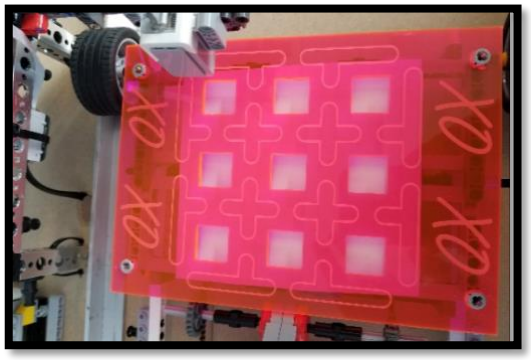


Figure14. Game Board.

The base board was purchased and some of the holes were drilled using the drill press in ESMS, however the dimensions of the board made it impractical to drill some of holes using this method. Instead, most of the holes in the board were drilled using a hand drill and were smoothened using sandpaper. The aluminum supports, and guiding rails were machined in ESMS using both the horizontal bandsaw and the sanding machine. The supports were fastened to the base board using superglue and the guiding rails were mounted using wood screws.As shown, several parts were fabricated for this project, and without them, the robot would not have been functional.

4.7 Initial Prototypes and Why they Failed

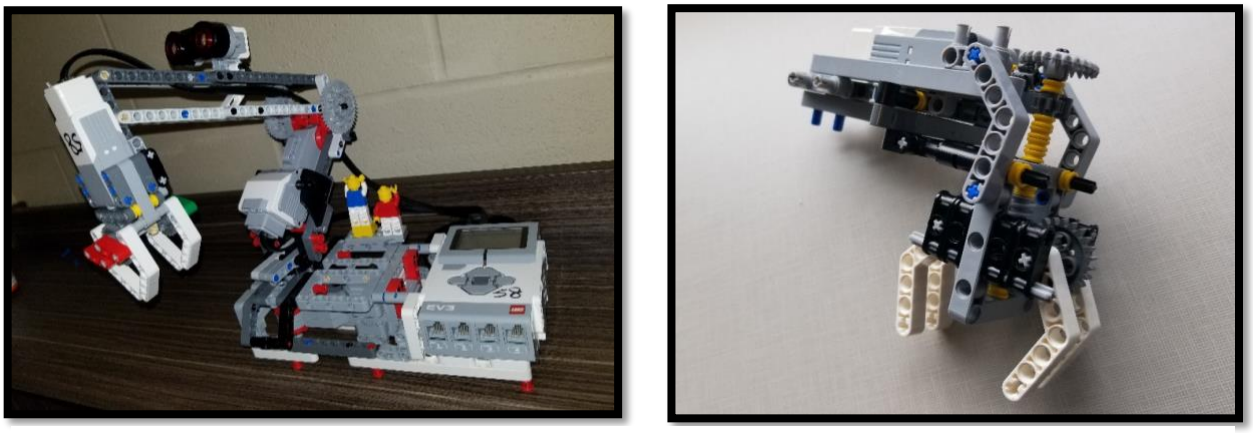


Figure 14. Prototypes.

Initial design was a robotic arm that moves playing chips with the relevant markings (X or O). See the figure below for an image of the initial design.

Prototype for this design was not technically feasible

Reasons for failure include:

- Structural Integrity
- The length required for the arm to reach could not be properly supported using Lego pieces alone.
- The design's precision relies entirely on the motor encoders, which could not be zeroed as is not possible to implement reference positions using the available sensors with the initial design.
- Because the design was only supported by the Lego motors, the robotic arm would oscillate and vibrate for a short period after the motors have been stopped.

The team realized that a new design was required:

- The second design included the dual rack and pinion system with a claw.
- The claw would grab the playing pieces from one location and drop it in the desired location.

However, upon prototyping this design was not practical for the following reasons:

- Lacked the necessary accuracy to manipulate the game pieces
- The robot claw included a medium motor and a colour sensor which is a very heavy component for the rack to support. Deformation of the rack occurred. See the figure below for an image of the claw that was built and tested.
- Additionally, the claw required a complex mechanism to feed the balls which was not possible because of the space constraints.

5.0 Software Design and Implementation

5.1 Introduction

The team identified that a need exists in this project for software to control the hardware components, implement the game logic of tic-tac-toe and artificial intelligence to play tic tac toe while running on the available computation and memory resources of the Lego EV3 brick.

The software implementation goals for this project can be broken down into three main sections: hardware control, game implementation and game play decision.

Hardware Control

The main functions the software must perform in hardware control are:

- Localization.
- Observe the physical state of the game.
- Allow the program to make physical moves on the game board.

Game Implementation

The implementation of the game and game logic must:

- Keep track of game state.
- Ensure cheating is not allowed.
- Display the state of the game to the user.

Game Decision

- Decide to optimal move to make based on current game state and return instructions to hardware control to execute the move.

5.2 Design considerations, constraints and assumptions

Based on the data gathered by the team in the ME 101 sensor lab, the team expected some inaccuracies from the Lego sensors. The software design should assume some sensor inaccuracies. Since sensor inaccuracies are inevitable, the software should try to minimize the effects of sensor error and be able to work with some level of error to ensure that it can perform in multiple scenarios. To combat sensor inaccuracy, the team ensure to test different possible sensor readings as well as resetting the sensor values frequently to prevent error buildup.

The team decided that the entire program should run on the Lego EV3 brick without any additional computation done off-board. The EV3 has a TI Sitara AM1808 (ARM926EJ-S core) @300 MHz processor and main memory of 64 MB RAM with MB Flash[1] which is considerably less processing power and less memory compared to any personal computer released in 2018.

Based on the engineering specifications, the design of the program should insure that the bot never loses against the human player.

5.3 Design Process

References [1], [2], [3], [4], [5], [6], [7] and [8] were used for the research for this program.

5.3.1 Data structure Design

The data structure used to keep track of the game state is a 2D array with a dimension of 3 by 3 in a struct. Since the game of tic tac toe is played on a 3 by 3 grid, the most natural representation for the game state was to use a 3 by 3 array. Based on the team's research, it was determined that passing arrays into functions in RobotC is not supported and a struct was needed to work around this limitation.

Due to the physical orientation of the board and EV3 display, the internal representation of the board has flipped row and column indices. To account for this difference, the scanning function flips the indices of the scanned cells. This insures that the internal storage of the game state is in the correct orientation.

Some variables were initialized as bytes to limit the memory used. During initial testing, the program often crashed due to memory issues and initializing values that did not need the storage size of int as bytes seemed to help with the memory issues.

5.3.2 Algorithm Design

Minimax with Alpha-Beta pruning was selected as the algorithm to implement the artificial intelligence. Minimax is an artificial intelligence algorithm that can be applied to two player games such as tic-tac-toe, checkers or chess. These games can be classified as zero-sum games since one player wins (+1) and the other player loses (-1) or the game results in a tie.

In a nutshell, minimax searches recursively for the best move that leads the maximizing player to win or not lose. Note that game search when using regular minimax is essentially a depth first search. The algorithm simulates the game by employing a maximizer (the player) and minimizer (the opponent). The maximizer attempts to reach a state that gives the highest possible score while the minimizer will make moves that minimize the score. It considers the current state of the game and the available moves for that state and then recursively simulates that game tree until it finds a terminal state (win, draw or lose). The terminal state is then assigned a score by a static evaluator function which evaluates the terminal state and assigns a score. For tic tac toe, the static evaluator is hard coded to recognize one of the 8 possible winning positions (3 horizontal, 3 vertical and 2 diagonal) and a win returns a score of 10, loss a score of -10 and tie is a score of 0.

After the first implementation of normal minimax, it is determined that the Lego EV3 does not have enough memory to run all the evaluations and recursive calls that are required by minimax. The program repeatedly crashed due to memory issues. Therefore, it was decided to optimize the algorithm with alpha-beta pruning. Alpha-Beta is an optimization method for the minimax algorithm that can be used to decrease the number of nodes that are evaluated by the minimax algorithm. Alpha-beta are numbers that represent the “fail-safe” option (worst case scenario) of the maximizing player and the minimizing player respectively. As minimax progresses down the game tree, the alpha value is updated at maximizing nodes and the beta values are updated at the minimizing nodes. The optimizing happens when alpha is greater than beta. This represents that the maximizer’s score is higher than the minimizer’s score. Since the minimizer doesn’t want this to happen and will play to avoid this in reality, the subtree of that move does not need to be considered and is pruned. This saves computation

both on minimax evaluations and leaf node static evaluations. It is important to note that while in testbook minimax examples, the leaf node scores are just numbers, in reality, those scores are obtained by a static evaluator function. In complex games such as chess, this static evaluator can take significant resources to run and any optimization that can reduce the number of static evaluations is a significant saving.

5.3.4 User Interface Design

The user interface in this project was printed to the EV3 brick's display and also included a touch sensor. The game board and any relevant prompts that informed the player on how to play the game was printed to the screen. The touch sensor was used to indicate when the user makes a move and also to inform the robot that the board was reset if the user wanted to player additional games. The team experienced some issues printing out certain messages on the screen. It is unclear what the cause of the issues was but is most likely a hardware bug or a memory problem.

5.4 Flow Chart

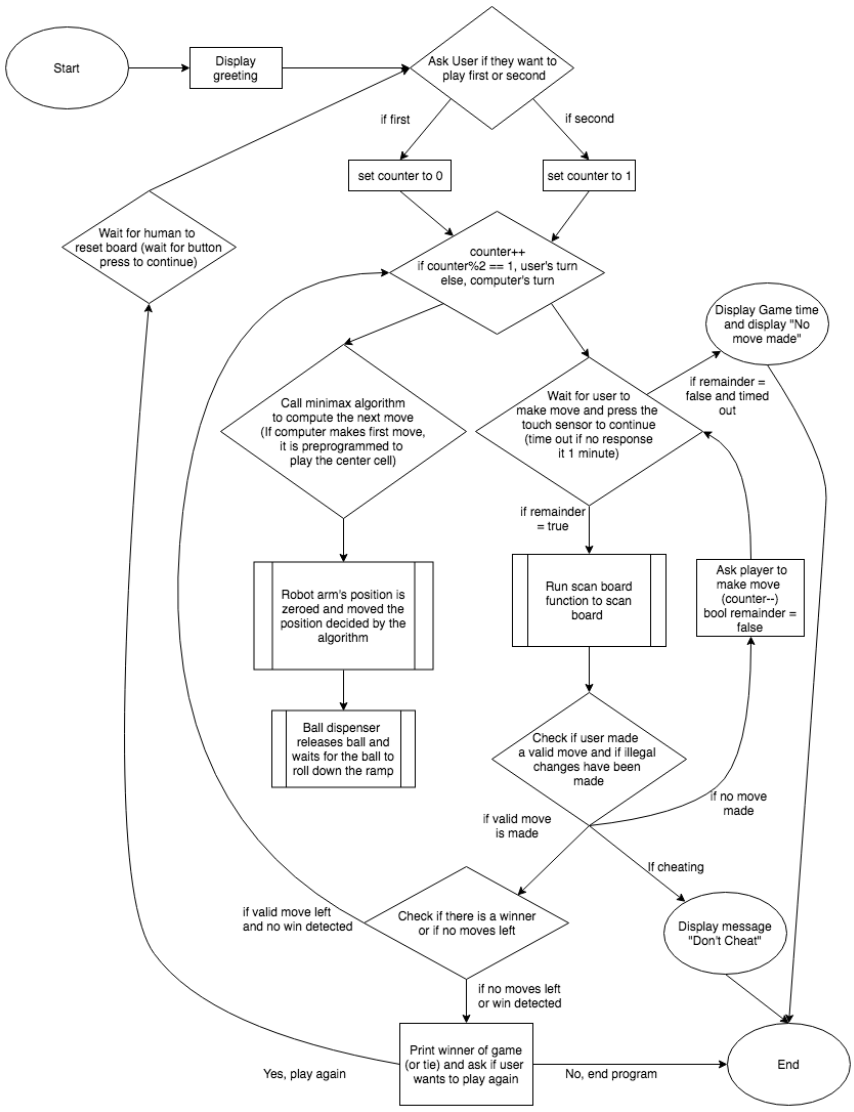


Figure 15. Flow Chart.

5.5 Testing

Since RobotC is a C-based language, it was decided to implement the minimax algorithm in C for development and testing purposes before moving it to RobotC for the final implementation. After debugging of the algorithm, itself was complete, it was tested on the EV3 with a display interface.

After the physical robot was built, the team wrote multiple test functions to test individual mechanical functionalities of the robot. This included separate functions to test the accuracy of the ball dispenser, board and the arm. There is also a separate function that was used to dispense a ball into each board cell to ensure that the robot could accurately place a

move in each cell on the board. The team also did some experiments to determine the different values that the colour sensor might read in normal lighting conditions when scanning the green balls, yellow balls or an empty cell to provide some level of redundancy. Then the scanning functionality's accuracy was validated since the accurate scanning of the board is vital for the robot to play the game.

6.0Verification of Design

The group's revised Engineering Design Specification sheet consists of 10 entries. The first specification listed in the document was the cost of the project. This cost excluded the price of the Tetrix and Lego kits and included everything that the group spent towards buying/fabricating parts for the project. The group spent a total of 47 Canadian Dollars on the project, most of which was spent on 3D printing and Laser cutting of parts. The group easily satisfied the specification, which was set at 75 Canadian Dollars. The second specification was the mass of the finished product. This was specified as 10 kg. The project was measured with a mechanical scale in E3. The scale was zeroed with a wood block that held the project on the loading pad. The project was then loaded on top of the wood block and the total weight was recorded. The final weight of the project was calculated by taking the total recorded weight minus the weight of the wood block.

The third specification was the time required by the robot to compute and play it's move. This was specified to be as 60 seconds as the group felt a longer play time would make the user lose interest in the game. This specification was verified by testing the finished project. The robot was tested 3 times and it consistently made the move within 20 seconds.

The fourth specification on the Engineering design specification sheet is the accurate placement of the playing pieces in the board. To verify this specification, one of the members of the group created a test program in which the robot continuously dispenses the playing pieces into the 9 slots on the board. This program was run on the finished project 5 times and observations were made by the group members. The ball dispensing arm exceeded the expectations of the group members and only failed once during the testing, proving itself to be highly accurate.

The next entry on the list was the time per game. This was verified by testing the finished project several times. The project's code uses the built-in timer in RobotC and

displays the total time after the game is finished. A normal game usually lasted for 140 seconds. The robot waits for 60 seconds to let the user play his/her move and even if the user decides to use the full 60 seconds for every single turn, the game still lasts for about 300 seconds.

The sixth specification was the losing percentage of the robot. The algorithm of the robot's program is mathematically unbeatable, and in the group's experience, the robot never lost a game.

The next specification was about the ease of use of the robot. This describes how easy it is for someone using the robot for the first time to understand and adapt to the robot. It was measured on a 1-5 scale where a rating of 1 would indicate that the game is really difficult to follow and a rating of 5 would mean that the user adapts to the game very easily. The group conducted a survey during the demo day and asked for feedback from students and members of the teaching staff using the robot for the first time. The ratings given by the first 10 users were recorded and the average was calculated to be 4.5/5, indicating that the robot was really easy to use.

The eighth item on the list was the robot's ability to detect cheating. The robot's program uses 2 different 2D arrays where it stores the old state of the game board and the new state of the game board after the user plays his/her move. This helps to keep track of the pieces on the board and the robot can detect if the user tries to change the location of any existing pieces on the board. The program also detects cheating if the user places 2 pieces at once. During the testing of the finished project, the robot accurately detected cheating every single time. During the project demo day, a lot of people tried to beat the robot by cheating but were caught by the robot without fail.

The second to last specification on the list was the accurate detection of the playing pieces on the board. The robot uses a color sensor, attached to the ball dispensing arm, to scan and detect the playing pieces on the board. The color sensor was fairly accurate in sensing the chips and identifying whether the chips belonged to the robot or the user. Once in a few games, the color sensor did malfunction and did not detect the last piece placed by the user. This was observed only a few times and the issue was fixed by telling the robot to scan the board again, using the touch sensor.

The last specification was about the robot autonomously playing a game of Tic Tac Toe against the user. The robot satisfied this specification as during the game, no human intervention is required by the robot to ensure the game goes on to completion. A touch sensor is placed near the user's position so that the user can interact with the robot and let the robot know when the user has made his/her move. All the playing pieces required by the robot are preloaded before the game begins.

7.0Project Management

The work-breakdown structure for this project consisted of three different major deliverables. These major deliverables were assignments, robot project, and report.

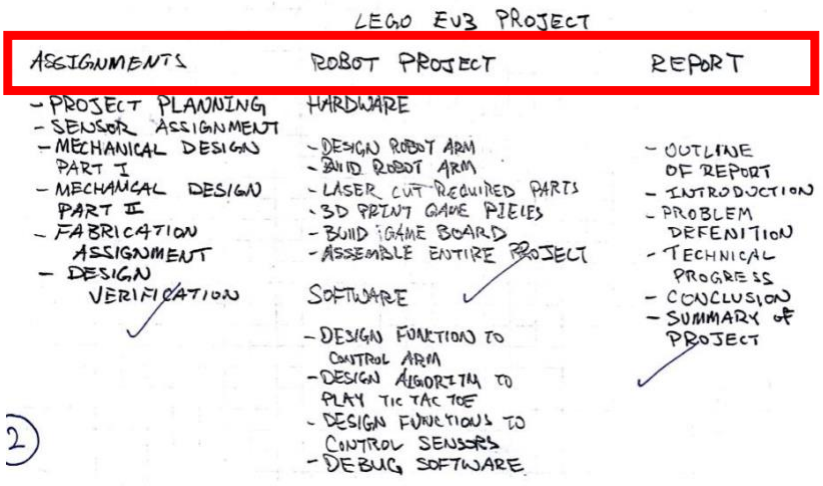


Figure 16. Work-Breakdown Structure.

These major deliverables consisted of many subtasks that must be completed to complete the major deliverable. For example, the sub-tasks for assignments were the sensor assignment, fabrication assignment and many more as shown in Fig. 18.

At the start of the project, the subtasks were evenly distributed to the individuals in the team. This will allow every person in the group to contribute to the project. However, in the middle of the project, the group concluded that it was not feasible to build the final design concept that was selected in the concept design phase of the design process. Therefore, the project schedule was revised to reflect the situation of the team. A noticeable alteration in the project schedule was the hardware section of the robot project. Although few of the subtasks

in the hardware section were completed, the change in the design concept forced the group to do those sub-tasks again. One of these subtasks was laser cutting the game board for the project. Another example is the building of the robot arm. The robot arm was an essential component in the project since it was responsible of placing the game pieces in the correct position of the game board. However, the change in the design concept meant that it was necessary to completely rebuilt the arm as mentioned in the mechanical design section.

Additionally, the change in concept design also changed the distribution of subtasks in the team. Due to time constraints, the work was not evenly distributed as initially planned. For example, one member did more programming than the others. In fact, the tasks were distributed based on the skills and knowledge of the team members. For example, the software for the robot was supposed to be evenly done. This allowed the team to successfully two of the major deliverables in the project, which were the robot project and the assignments.

One of the main differences between the project plan and the actual schedule was the final report for the project. The plan was to start the report at the beginning of July. Due to the unexpected change of concept design, the report could not be started after demo day for the robot project. Another difference is the programming of the robot because the plan was to finish it by the end of the first week in July. However, the programming was only completed three days before demo day due to the complexity of the code.

8.0 Conclusion

The design group underwent a comprehensive engineering design process to come up with a proper need analysis, a list of engineering requirements, and several design concepts. Additionally, a work-breakdown structure and project plan were created to ensure the group could complete the project by demo day. By using the decision matrix, a final design concept was selected. However, the final design concept was changed due to technical difficulties. The new design concept was not only technically feasible, but it also fulfilled all the engineering requirements set by the group. The software was successfully implemented to support the mechanical design, which allowed the project to fulfill the need analysis. Although the final prototype was a great success during demo day, there are still a number of things that could be done to improve its mechanical design and software.

9.0Recommendations

9.1 Mechanical Design Recommendations

There are a few things that could have been better on this project. The rack used to drive the ball dispensing arm was made out of acrylic. The acrylic rack was structurally very weak and couldn't take a lot of weight. The acrylic also exhibited a lot of flex when the arm was extended to its maximum length. The acrylic rack was used as it was already fabricated and available to the group, which saved a lot of time. Given more time, a similar design made out of steel could've been used to reduce flex.

Once the acrylic rack is replaced, the distance between the Tic Tac Toe board and the tetrix structure can be increased as the new rack, without the flex, would be able to extend further when need and retracted further back to not block the board. This would allow the user to more freely place the ball on the board, as with the current design, a part of the ball dispensing arm hangs over the board and covers up a cell on the game board.

9.2 Software Recommendations

Given more time, an emergency stop button could be implemented. Based on the team's understanding, this feature would have to be implemented at a lower level since concurrency cannot be directly implemented in RobotC. It is also possible to implement a timer for resetting the arm and board. This could account for the possibility that the arm or board were possibly missing. If touch sensor is not triggered in 5 seconds, assume that the arm or board is not connected or not attached and stops the motors to save power before stopping the program.

References

[1]"Lego Mindstorms EV3", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Lego_Mindstorms_EV3. [Accessed: 25- Jul- 2018].

[2]Lecture 6: Search: Games, "Lecture 6: Search: Games, Minimax, and Alpha-Beta | Lecture Videos | Artificial Intelligence | Electrical Engineering and Computer Science | MIT OpenCourseWare", Ocw.mit.edu, 2018. [Online]. Available: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/lecture-6-search-games-minimax-and-alpha-beta/. [Accessed: 25- Jul- 2018].

[3]Mega-Recitation 3: Games, "Mega-Recitation 3: Games, Minimax, Alpha-Beta | Mega-Recitation Videos | Artificial Intelligence | Electrical Engineering and Computer Science | MIT OpenCourseWare", Ocw.mit.edu, 2018. [Online]. Available: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/mega-recitation-videos/mega-recitation-3-games-minimax-alpha-beta/. [Accessed: 25- Jul- 2018].

[4]"Tic-tac-toe", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Tic-tac-toe. [Accessed: 25- Jul- 2018].

[5]"Alpha–beta pruning", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning. [Accessed: 25- Jul- 2018]. [6]"Minimax", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Minimax. [Accessed: 25- Jul- 2018].

[7]"Tic Tac Toe: Understanding the Minimax Algorithm", Never Stop Building, 2018. [Online]. Available: https://www.neverstopbuilding.com/blog/2013/12/13/tic-tac-toe-understanding-the-minimax-algorithm13. [Accessed: 25- Jul- 2018].

[8]"Minimax Algorithm in Game Theory | Set 3 (Tic-Tac-Toe AI - Finding optimal move) - GeeksforGeeks", GeeksforGeeks, 2018. [Online]. Available: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-3-tic-tac-toe-ai-finding-optimal-move/. [Accessed: 25- Jul- 2018].

Appendices