HMS GAME: DR. BOGEN’S ARMY

Final Paper—May 7, 2015

Cerebral palsy (CP) is the most common motor disability in children, appearing in infancy or early childhood. According to the CDC, 1.5-4 in 1000 children are diagnosed with CP, which currently has no cure. HMS is a school specifically addressing the needs of this population. After a visit, it was discovered that games available were not adapted specifically for children with CP. As a result of their different cognitive abilities, these children lack crucial elements of normal childhood development such as motor skills, social interaction, and environmental control. Dr. Bogen’s Army is an automatic robot-enabled interactive game designed specifically to incorporate elements of play and environmental control for children affected with CP. The design process began with identifying an existing game that could be adapted to fulfill these clinical needs. We then determined types of specifications for success, such as performance based (accuracy, function reliability), clinical need (staff involvement, social interaction), and design (board size, robot communication). These specifications then guided our choices as we iterated through the design process to determine the ideal game board material, hardware, and software architecture. We met all of our initial specifications, producing a fully functional game, except those concerning the budget and game duration. We expect that further iterations of the game will significantly improve on these two specifications through incorporation of 3D printed buttons, which are cheaper, and improved robot capabilities.

W5: Yong Sik Cho, Jacqueline Chow, Cathryn Herbst, Jessica Monsman
Senior Design

1 INTRODUCTION

Cerebral Palsy (CP) is the most common motor disability in children (Data & Statistics for Cerebral Palsy, 2013). “Cerebral” means of the cerebrum which is the largest part of the brain where perception, imagination, thought, judgment, and decision take place (Boeree, 2003). “Palsy” refers to an uncontrollable movement of the body (palsy, 2014). CP often appears in infancy or early childhood but doesn’t progress over time and cannot be cured. Additionally, CP is a congenital disease with symptoms that begin to show before the age of three including lack of muscle coordination during voluntary movements, stiff or tight muscles, exaggerated reflexes, walking on toes or with one foot dragging, and a crouched gait. Some children also have CP as the result of direct brain damage during the first few months of life from bacterial and viral infections or head trauma (Boeree, 2003) (Hinchcliffe, 2007). According to the Centers for Disease Control and Prevention’s Autism and Developmental Disabilities Monitoring (ADDM) Network, about 1.5 to 4 per 1,000 children have been diagnosed with CP (Data & Statistics for Cerebral Palsy, 2013). CP impacts demographics differently and often involves other developmental disorders including epilepsy and autism (Hinchcliffe, 2007).

1.1 FORMS OF CEREBRAL PALSY

CP exists in different forms and different degrees of severity depending on the part of the brain that is damaged as a result of the disease. The four different forms of CP can be differentiated based on posture and brain damage. With Spasticity CP, the muscles are stiff with limited movement patterns as a result of damage to the movement areas and pathways of the cortex. Athetosis CP involves constant movement that is often uncontrolled with stiff and floppy muscles resulting from damage to the basal ganglia of the brain. A child who suffers from Ataxia CP, has muscles that shake upon movement which the child tries to stop by stiffening his/her muscles. This results from damage to the cerebellum. Finally, Hypotonia CP involves floppy muscles (Hinchcliffe, 2007).

1.2 Physical and Mental Impact of Cerebral Palsy

CP impacts children’s lives in different ways depending on the form and development process. Some children can lead normal lives without lifelong care and special assistance while other children might not even be able to walk on their own (Hinchcliffe, 2007). According to the ADDM, over 58% of children can walk without assistance, 11% of children can walk with a mobility device including a walker, and 31% of CP children could not walk (Data & Statistics for Cerebral Palsy, 2013). Children also suffer from damaged intellectual capacity, difficulty hearing, decreased sight, fear of movement, and difficulty processing muscle sensations (Hinchcliffe, 2007).

1.3 Cerebral Palsy and Movement

However, CP is not caused by problems in the muscles or nerves even though the symptoms appear to affect muscle and nerve function. CP instead is caused by damages to the brain that impact the child’s ability to learn to move. A child learns to move as early as pregnancy when the child is beginning to move within the womb. Children learn how to move from their experiences: the brain develops the motor control through adaptations of these experiences as the child learns different postures and how his/her body interacts with the environment during movement. CP damages this part of the brain that controls the child’s ability to learn to move. However even with this damage, the brain can still adapt to different movements but only those that the child actually experiences during development. For example, if a child is lying on his/her back, this is the movement that his/her brain will experience and thus the movement that his/her muscles will respond to during future movement. Thus, it is vital for the child to experience active, not passive, movements in order to fully develop his/her brain and change the way the child functions (Hinchcliffe, 2007).

1.4 Current Treatments and Research for Cerebral Palsy

Since CP cannot be completely cured, current treatment for CP involves a targeted approach to improve a child’s development and learning including physical, occupational, and speech therapy. Drugs are also used to control seizures, muscle spasms and pain. Surgery can also be used to correct abnormalities or ease muscle tension. Children also use medical devices including orthotic devices, wheelchairs, walkers, communication aids and voice synthesizers to help with everyday activities (Data & Statistics for Cerebral Palsy, 2013). Some researchers have also explored the benefits of virtual reality experiences for CP children although these still remain costly and do not address social concerns (Deutsch, 2008).

Historically, CP research has been focused on rehabilitation interventions that target the CP children's development abnormalities (Shikako-Thomas, 2012). However, rehabilitation research is becoming more focused on the quality of life for CP children. As researchers identify factors with worse or better quality of life, they are able to create programming and therapies to improve targeted inputs into quality of life. Participation and play are currently two factors that researchers have identified as areas of improvement for CP children's quality of life (Shikako-Thomas, 2012). Research is also focused on the different therapy options for CP children which include experiments to determine the best type of therapy for children with different forms of CP (Ketelaar, 2001).

1.5 Importance of Play in Childhood Development

Play and interaction with the environment are important elements of development for all children, including CP children, which helps improve the cognitive, physical, social, and emotional well-being of children (Ginsburg, 2007). In fact, play is so vital to childhood development that the United Nations High Commission for Human Rights has declared that play is a right of every child (UNICEF, 2012). Because of children's abilities to engage and interact with the world around them during play, children can be creative and imaginative while conquering fears and gaining physical, cognitive, and emotional strength. Furthermore, during undirected play like a board game, children learn how to work in groups, share, negotiate, and resolve conflicts. It is especially important for play to be child driven so that children can make their own decisions and engage with the environment around them. Play also allows children to adjust better to the academic setting and further develop their social skills (Ginsburg, 2007) (Majnemer, 2008). Play is even more important for those children who lack complete motor function. A lack of motor function often results in less experiences and learning opportunities and therefore results in cognitive impairment as the brain develops (Hinchcliffe, 2007) (Sinno, Charafeddine, & Mikati, 2013).

1.6 Clinical Need: Interactive Game Incorporating Elements of Play and Environmental Control

Current options for CP children to engage with their environment through play and games are limited. (Majnemer, 2008). However, we do know CP children are involved with fewer leisure, play, and game-like activities than peers (Ginsburg, 2007). Furthermore, the activities that CP children do partake in often do not involve a social or group setting. Traditional therapy for CP children strives to improve motor ability, but some therapies have been transitioning towards functional success and participation within groups through leisure activities (Ginsburg, 2007) (Majnemer, 2008). For CP children, assistive technologies provide an opportunity to improve current play options.

With the involvement of robotics, CP therapies are working towards more active engagement from the children with promising results. For example robotic therapy allows CP children to grasp and manipulate different objects while other games create an interactive table-top surface (Li, 2008) (Trafton, 2009). However, most games that CP children normally play during therapy have not been adapted to their needs and are often developed for a younger age group with lower cognitive ability. Currently, no solution exists because of the wide variety of skill levels of CP children. The children's favorite games at HMS School are Uno, video games, Pretty Pretty Princess. These games like Uno often require extensive staff involvement here the staff essentially plays the game on the child's behalf. Although the video games allow the children to play, these games lack a social and environmental control aspect. (Play, n.d.) (Special Perspectives: Cerebral Palsy, 2014).

After researching this topic and meeting with our adviser Dr. Bogen, we developed an interactive board game for CP children in order to meet the need of an interactive game incorporating elements of play and environmental control. Our current idea is to create a robotic version of the board game “Sorry!” that involves separate robot “game pieces” that will move around a game board. The CP children will be able to control the movement of their game piece through ability switches (or buttons) (TASH Ability Switches, n.d.). The goal of the game is to return your game piece back to your home without being knocked off the game board by another opponent. We developed the game with the goal of testing the game at the HMS School for Children with Cerebral Palsy in University City.

2 Goals and Objectives

The goal of our project was to build an interactive board game to increase cognitive development and social interaction among CP children through play. This would be a game that limits staff involvement while also engaging the CP children in a social, leisurely setting. As previously mentioned, currently, the CP children do not have a suitable game to play that has been specifically designed to meet their needs and to increase the elements of play in their daily lives.

To fulfill our goal and complete our project, we identified four main objectives. Our first objective is to create a game board design that the CP children can control on their own with minimal staff involvement. This involves creating a balanced game with enough automation that the game can essentially run on its own while also giving enough control to the CP children so that they are truly engaged with the game.

Our second objective is to create a dynamic game for all skill levels and number of players. Since the severity and ability of CP children widely varies amongst children at HMS and amongst CP children overall, a dynamic game will allow the game to be adjusted to different learning levels and types so that each player can receive maximum benefit from the game. This will mainly involve the integration of different types and numbers of ability switches.

The third objective is to engage the CP children with a higher level of spatial cognition. With our game, the CP children will have the ability to control something without special assistance that is outside of their physical reach. This is a sensation that most of them have yet to experience even though it is a vital aspect of development and motor control.

The fourth objective is to create a cognitive learning environment within a social setting. This will allow the CP children to interact with their environment in an active instead of passive manner while also developing their cognitive ability through the use of numbers, colors and strategy within the game design and game play.

Our final end product and game that we have called “Dr. Bogen’s Army” (DBA) includes a complete game board, score board, four robot game pieces, and a Master robot control. Through the design process and after consideration of our specifications, we designed our final project to meet our objectives and fulfill the clinical need.

3 PROJECT ILLUSTRATION

The project illustration highlights our first objective to create a game board design that CP children can control on their own with minimal staff involvement (Appendix A, Figure 1). This is shown by the use of our game (top pathway) to play a board game instead of the traditional route (bottom pathway) that involves staff assistance. The nature of the game is also modular with specification that can be varied (e.g. number of players, number of pieces) which makes it a dynamic game suitable for all skills levels and number of players. Also, by directly engaging with the physical world via button, the game strives to engage CP children with a higher level of spatial cognition in a social setting.

4 DESIGN AND DESIGN PROCESS

4.1 The specifications for our game were separated into three different categories: performance based, clinical need, and design focused. This was to address various critical parts of our game board. Performance based specifications are geared towards ensuring the logistical success of the game, clinical need specifications ensure that the objectives of the game in terms of medical application are addressed, and design needs are to tailor the game towards the specific setting at the HMS school (Appendix L).

4.1 DEFINING THE PROBLEM AND CONCEPTUAL DESIGN

After defining the clinical needs and deciding on specifications that will drive our project, we focused on the conceptual design of the project by looking at literature and evaluating existing treatments. We discovered that no current games targeted towards children with cerebral palsy fulfilled the specific clinical needs that we had chosen to focus on. However, the concepts were not new and could be found in games specific to children with other afflictions, such as autism. For example, a common type of game for autistic kids is the “cause-and-effect” toy, which directly addresses the need for increased spatial cognition. The need for cause and effect toys has been proven over and over in research literature. Piaget, the first psychologist who studied cognitive development, theorized that schemas were developed through play (Piaget, 1952). A schema is a series of linked mental representations that we use to understand to respond to the world. Specifically, Piaget theorized that children learn through play when it causes things to change in the environment around them. His theory still propagates in recent literature: Gordon and Browne recently declared that children learn how to manipulate the world around them through play (Gordon et al., 2014). However, most of the current cause and effect toys require more mobility than the general population of kids with cerebral palsy has. In addition, virtual cause and effect toys such as video games do not teach spatial cognition, as all objects involved are imaginary. Therefore, no true comparable games existed in the current array available to children with cerebral palsy.

4.2 DETAILED DESIGN

4.2.1 Determining Materials: QFD Analyses

Next, we approached the details of our project design by using QFD analyses to evaluate the materials and platform for coding as well as deciding on the robots that would serve as the player pieces. QFD analysis was used to evaluate materials and the coding platform (Appendix B, Figure 2 and 3). To determine the type of robot to be used as the moving pieces, we evaluated several options also using QFD analyses. Our options included making our own robot using an Arduino Robot Kit, Sparki, an Omni wheel robot, or a Propeller Activity Bots Kit. However our analyses using the QFD table suggested that purchasing Sparki robots would be the most cost and time effective path to a complete project given the time constraints (Appendix B, Figure 4).

4.2.2 Adapting Sorry to Robotic Mechanisms

After deciding on using Sparki as the moving pieces, the game of Sorry was redesigned with the Sparki robots in mind. This was one of the most important steps in our design process of the game board as the main challenge was the maintenance of as much of the original Sorry board as possible while accommodating for the limited mobility of robots. Given that Sparki has 5 infrared sensors, we decided to use the middle three sensors to track a line that would go around the board (Appendix C, Figure 5). The line tracking methodology went through three different iterations to find the optimal balance in the tradeoff between speed and accuracy. In the first iteration, the line tracking code in Sparki’s original library was used. This code, however, was accurate for only a particular width of line, and had to be adjusted in order to adapt the game to more possibilities for the game board. In the second iteration, the line tracking code only tracked the left edge of any given line with one sensor: if the one sensor saw white, it would rotate towards the right, and if it saw black, it would rotate towards the left. In the last and final iteration, the line tracking code made use of all three middle sensors to eliminate the constant rotation. If two sensors saw black, the Sparki would move forward. However, if only one saw black, the Sparki would make adjustments accordingly depending on which sensor recorded values corresponding with white.

Because Sparkis cannot ‘jump’ over other Sparkis as in the real Sorry game, mechanisms to allow for individual robot movements without running into each other had to be created. There were two ideas that we ultimately chose from – in the first scenario, there would be an inner and outer lane for movements was created for Sparki to rest and move on, respectively (Appendix C, Figure 6 and 7). In the second scenario, each Sparki would have its own individual lane. This second idea was eventually discarded because it hindered future expansion of the game board. Because the full game of Sorry generally has four pieces per player and four players, and the intention was to develop the game board such that one day a future iteration of the project could achieve this ideal, the second idea was infeasible. Once “passing” methodology was established, the starting positions of Sparkis were then added to the corners of the board in order to allow Sparki to return and leave its ‘home’ base (Appendix C, Figure 8).

The game board also went through several different iterations. First, several different possible line widths were tested. Originally, electrical tape that was ¾ inch was used. However, it was quickly determined in tests that this width was too wide and would result in slower functions and/or lower accuracy. Thus, we redesigned the game board to incorporate ½ inch tape. Likewise, corners went through several iterations. In our initial design, corners were imagined to be rounded edges. However, in implementing prototypes, it was discovered that rounded corners were extremely hard to implement on the game board with electrical tape with any type of consistency. Therefore, sharp corners were implemented and the code had to be adjusted accordingly.

4.2.3 Master and Slave Robot Designations

Next came the design of the code. We designed the code from an object perspective, giving a “Slave” a library of certain movements that it could employ, with a “Master” robot figuring out the logic behind each move. With the Sparki and board materials set, we then proceeded to design the remainder of the deliverables which included a Master robot that would wirelessly send commands to several Slave Sparkis.

After theoretical dissection of the board game of Sorry, the robotic gameplay of Sorry was dissected into individual movements of the pieces such that the game could be played with only 10 distinct maneuvers. These 10 maneuvers were chosen to be the largest distinct unit of action that allowed the robot to perform all permutations of actions that the game rule allowed (Appendix D, Figure 9). This enabled Sparki to operate with the fewest lines of code possible which helps improve processing times and speed. For example, the simplest move for Sparki would be to move one square which involved following a straight line passing through three intersections. This movement was defined to be “move_fwd(3)” which enabled Sparki to move one square, or across three horizontal lines, in one line of code rather than having a smaller movement, such as moving across one horizontal line, repeat three times (Appendix D, Figure 10). The code for the Slave robot and the Master robot can be found in Appendices I and J respectively.

The Master robot was designed by custom-ordering an Arduino shield that allowed a standard Arduino Duo board to communicate with 6 different Bluetooth modules at the same time (Appendix D, Figure 11)**.** Four of these Bluetooth modules were set to communicate with each of the 4 Slave robots (Sparkis) on the board while the remaining 2 modules were set up to communicate with a scoreboard that reported the current player number as well as the dice roll. Four female mono-plug jacks were also included on the custom-made shield to connect to ability switches that would prompt the game (Appendix D, Figure 12).

4.2.4 User Interface and Packaging Design

Next came the design of scoreboard for visual feedback, incorporation ability switches for students to press, and a complete set of instructions manual that would explain the setup and mechanics of the game. In our original iteration of the board, we imagined the scoreboard display to be an LED monitor of some type. In the initial stages, this display was a monitor screen. However, we quickly discovered that this was unrealistic because Arduino could not easily communicate with a monitor screen. We ordered a Gameduino shield to develop images to display on a monitor screen, but found that operations were limited. Therefore, we revisited our original design and changed our plans to incorporate a Neopixel shield instead. (Appendix E, Figure 13).

In order to develop a scoreboard for the game, the size, material, and hardware that would be used in it had to be considered. Based off of the size of the game board and maximum distance that a player could be from where the scoreboard could be placed, initial dimensions of the scoreboard were determined. Next, the outline of the scoreboard was cut out of cardboard. On the front of the scoreboard, ‘PLAYER’ and ‘MOVE’ were written. Two squares were cut out next to each of these words to fit the Neopixel LED screens that would display the numbers 1-4, depending on the game situation. After the realization that a deeper scoreboard could be used to store some of the game materials inside of it, the width of the scoreboard was increased. Next, the final scoreboard design was 3D modeled in SOLIDWORKS. The scoreboard is a press fit box, with a hinged back. The pieces of the box were laser cut out of ¼” acrylic. The Dr. Bogen’s Army logo was designed and rastered into the back of the scoreboard. Finally, a strap was added to the top of the scoreboard for easy transport, and the rastered logo and text were painted white to make them more visible. Figures 14 and 15 of Appendix E are images of the front and back of the final scoreboard. Housing for the fragile Master hardware was created in a similar manner. The original design was created in SOLIDWORKS to fit the Master hardware. Additionally, different holes were strategically designed on the box to make room for cords. Using ⅛” transparent acrylic, a laser cut box with a hinged lid was created. The DBA logo was rastered on the lid. Figure 16 of Appendix E is an image of the Master box.

Ability switches (Appendix E, Figure 17) were also used because 90% of HMS students are able to operate at least one ability switch. While 3D printed ability switches (Appendix E, Figure 18) were considered at the beginning of this semester because lower costs, we decided to proceed with regular ability switches because of time constraints and also because it deviated from the main objective of our project.

Lastly, we created a detailed instruction manual (Appendix K) in order to aid future set up of the game. This detailed instruction manual was originally intended to only have set up instructions, but during the course of testing the game, we noticed that the staff frequently asked questions about how the game was played, and that certain other questions regarding troubleshooting were also raised. Thus, two new sections of the instruction manual were added.

5 Evaluations Performed and Results of Specification Testing

5.1 Performance Based Specifications

Robot Speed, Angle and Distance from Ideal Testing: After narrowing down materials based on the QFD table and sensor readings, we investigated other aspects of robot performance on each of the materials. Robot speed, as well as the average angle and distance from the ideal location at the end of a move were tested. Ideal location is defined as landing on the correct space on the game board facing perfectly forward and in the center of the black line. The sheet performed the best in the speed (2.96 +/- 0.071 cm/sec.) and angle testing (4.30 +/- 4.28 degrees) (Appendix F, Figure 20). Vinyl #1 and the tarp were consistently more accurate in terms of distance from ideal location than the other two materials. Ten trials were performed for speed, angle, and distance testing. Ten trials were chosen reduce error as much as possible, while still fitting our testing within our time constraints.

Angle Threshold Testing: After we determined the sheet to be the optimal material, we also conducted an Angle Threshold Test. The purpose of this test was to determine what the maximum angle displacement from the ideal ending location could be without causing the robot to go off course (Appendix F, Figure 22). The Sparki facing perfectly to the right on the line was the ideal ending location. The Sparki was strategically rotated counterclockwise to this position by 30 degree increments until the robot could no longer find its ideal location. This ideal location is found by the robot searching for the black line via its sensors. From the test, it was concluded that the robot could be up to 228 degrees counter-clockwise off of the ideal location before it would no longer be able to continue on with the game to complete the next move (Appendix F, Figure 21). From general game testing, it was clear that this maximum error would never be encountered throughout the course of movements on the game.

Processing Time: In this context, processing time is defined as the delay between when a player pushes his/her button and when the Sparki game piece starts to move. We used a time to measure this delay. Twenty trials were conducted to reduce testing error while still keeping in mind the time constraints of the project. After twenty trials, the average time processing time was determined to be negligible, as measurements had different averages based on the individual taking the measurement (determined to be more a factor of reaction time than actual processing time.

Function Reliability Testing: In order to determine functional reliability, we systematically went through all of the possible combinations of game movements that a robot might encounter throughout the course of a game (Appendix F, Figure 23). These scenarios were all tested to ensure that the code would work in all situations of the game. After this debugging stage, we played the game an additional five times. The game was determined to be fully reliable after these five successful completions of the game without any mistakes (~12 hours of gameplay).

5.2 DESIGN SPECIFICATIONS

Sensor/Contrast Testing: The Sparkis utilize a line following mechanism in order to maneuver about the game board. Optimal line following performance requires the greatest contrast between Sparki sensor readings on white material and the black lines that the robots follow. Each Sparki is equipped with five sensors. The average sensor reading across all five sensors was recorded for a total of 9 different white materials. The ideal contrast reading on the Sparki between black and white values is 1000. As can be seen in Appendix F, Figure 19, the tarp had the largest contrast value, 984.6 +/- 3.286 units. The bed sheet had the second highest contrast reading, 976.2 +/- 9.418.

Material QFD Table Evaluation: Other important areas of importance in game board material selection were also identified and considered in final game board material selection. We constructed a QFD table in order to compare the nine materials in each of the eight categories listed in the table (Appendix B, Figure 2). Each material received a score from 1-10 in the eight categories. The highest scoring material, the sheet, received a 63. After determining that the sheet was the highest performing material, we started the game board construction with this material.

Movement Time Testing: A Slave robot could be commanded to do five possible movements throughout the course of a game (Appendix F, Figure 23). It was important to incorporate the amount of time that each movement takes into the Master code, so that there is no lag between when a command is given and when the move is executed. We recorded the length of time it took for each of these moves to execute fully. We tested this for four trials with each of the four Sparkis. The last two columns of Appendix F, Figure 23 display the average and maximum observed lengths of time that each of the five moves took to complete. The amount of time that each move took was then incorporated into the Master code in the form of a delay.

Bluetooth Range: In order to determine the maximum range in which the Bluetooth connection could still function properly, the Master was incrementally moved farther and farther away from the Slave robots until the Slave robots could no longer receive the message that the Master was sending. When the Slaves received a message, they were programmed to move forward one space. Therefore, no movement was associated with a signal that was out of range. After testing, it was concluded that the Bluetooth signal worked up to a distance of 10.67 +/- 0.3 m away. The maximum Bluetooth was tested twice in two different scenarios, one with a free path and one with obstacles.

Battery Life: Throughout extensive game testing, the amount of time that each Sparki's batteries lasted was kept track of. The average battery life was 273 +/- 49 minutes.

Game Duration: Throughout extensive game testing, the amount of time that each individual game lasted was kept track of. The range of game duration was from 47 minutes to 2 hours and 37 minutes. There was a rather large variation in game times due to the randomness of the game, so we chose to better encompass these values with a range designation instead of an average and standard deviation. In the event that lots of takeovers occurred after which robots were sent home, the game lasted much longer.

5.3 CLINICAL NEED SPECIFICATIONS

Social Interaction: Through our survey, we were able to determine a method to evaluate the social interaction of students during game play. This is based on survey results that examine the effectiveness of encouraging social development when compared to previous games played. Our game scored an average of 1.36 +/- 0.63 versus the initial observation of 1.83 +/- 1.04 from the other games played at HMS (where 1 is ideal game. Scale of 1 to 5). The relatively large standard deviations are due to the sample size because we were only able to receive five surveys for the previous games played and two surveys for our game. Staff at HMS School also provided written feedback via the surveys which was included in the game design (Appendix F, Figure 25). The staff rated our game better than the previous options because we provided a game that was truly adapted for the population that allows the children to interact with each other on their own.

Staff Involvement: The level of staff involvement was also determined in the same manner as social interaction. Both the previous games and our game had a staff interaction level of 1. This was determined by measuring the amount of time that the staff spent setting up, cleaning up, and helping with the game. This was combined with the results from questions regarding staff involvement on the survey.

Student Engagement: Student engagement also increased with our game from a level of 2.32 +/- 1.39 for previous games to 1.45 +/- 0.60 for our game. This was based on survey results that examine the level of engagement of children.

6 PERFORMANCE AND IMPACT

The specification of DBA were carefully outlined in such a way that, if all specifications were met, the game would meet our objectives. Fortunately, almost all specifications for the project were either met exceeded (Appendix G, Figures 27-29).

Our game did very well in the Performance Based specifications category. As a result, the robots are able to maneuver around the game board without error. The only shortcoming in this group was in the test results for Accuracy in Stopping. The tested value was found to be 0.43 +/- 0.11 cm. The specified performance was 0.53cm. This means that the robot movement meets this specification approximately 68% of the time. After testing the game by playing it, we determined that our listed specification was a more precise value than what was needed for the game to function properly. Therefore, overall success of the game was unaffected. In terms of processing time, the game performed extremely well. The specification was listed as 1.6 seconds and the actual processing time was on the order of milliseconds. Due to such an insignificant delay time, the players are able to easily associate the push of their buttons with the initialized movement of their robot game pieces. Finally, the speed of the robot was recorded as 2.96+/-.071 cm/sec. The specification was listed at 2.22 cm/sec. While we were within the specified range for robot speed, this is an area that our game could improve upon in the future. Faster robot movement would decrease overall game duration and would make the game more enjoyable. Currently, the movements are quick enough that the game is still fun, however, in the case of takeovers, the game pieces take a significant amount of time to return to their home bases.

It was essential that this group of specifications performed well because the clinical need that our game aims to address relies on the proper function associated with these specifications. Without proper game function and performance, the children at HMS school would not be able to play the game and thus we would not have met our clinical need of creating a board game incorporating elements of play and environmental control. With our game, the CP children are able to actively engage with their environment and increase cognitive learning in a social setting due to the automated design of the game. The staff at HMS were extremely excited to see the children playing our game and had asked if we could leave the game with them. Further developments include making a “DBA Kit” where other CP schools would be essentially able to make their own version of the game. With games like DBA, the CP population would be able to more actively engage with their environment and participate in more play and leisure activities--two elements of CP children’s lives that they are missing. The minimal staff involvement of our game also allows CP children to essentially play the game on their own so they would be able to play the game more often since constant staff involvement isn’t needed.

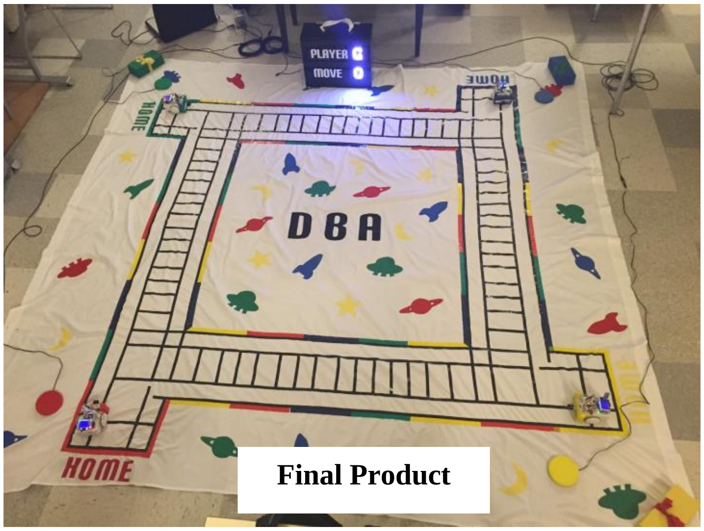
The only specifications for Design that were not met are cost and game duration. Overall, the project ended up costing $1,207. However, some of these costs were due to purchasing materials that did not end up being used in our final project. Taking into account all of these expenses that would no longer be necessary, our final cost could be reduced to $1,008. Additionally, the cost of the ability buttons can be decreased as well. A previous senior design group has created 3D printed ability buttons that can be sold as cheap as $15. Should the game be produced on a larger scale, costs would decrease even further. Game duration is highly variable for DBA. This is due to the nature of the rules of the game. In the event that a game consists of a large number of takeovers, the game can last quite a long time. However, when the game was tested at HMS, there were very few takeovers that occurred, and the game only took 47 minutes. Moving forward, the game duration has proven to be close to the listed specification in most cases. In order to ensure that the game ends within one hour, the game rules could be altered in such a way that winning becomes easier or that takeovers are less common. One solution could be that the game board could be expanded to have more spaces, to decrease the likelihood of takeovers. Additionally, as previously mentioned, robot speed could be enhanced to reduce game duration. As we originally planned when we laid out our specifications, our robots included 5 sensors and 2 wheels each, with Arduino as the platform. Using Bluetooth, the Slaves were able to communicate as far as 10.67 +/- 0.3 m away from the Master robot, which was more than sufficient for the purposes of our game.

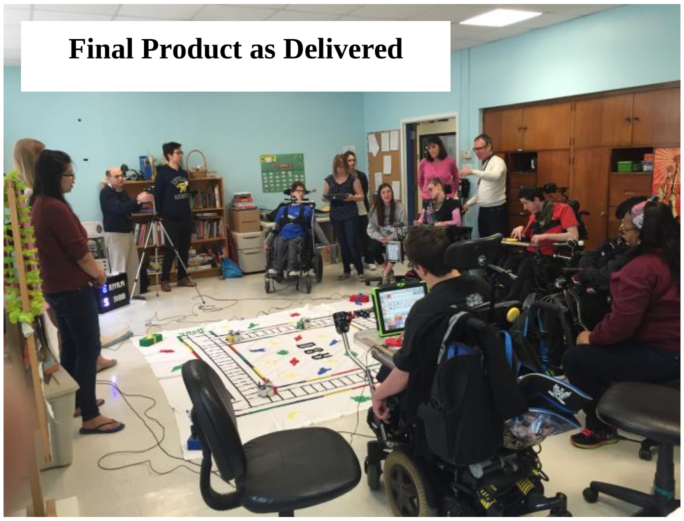
One area that will need further testing in order to determine whether or not the full range in which the specification lists is achieved is Age Range of Players. When the final game was tested at HMS, the age group that played the game were teenagers. Due to our limited access and the busy schedule of students at HMS, we were unable to test other ages of students. Due to the great success that we had in our first trial, we expect to see positive results for a number of age groups of students.

The final game allows for four players, as opposed to two, each with their own ability button. This increased number of players enhances the social interactions that the players can have throughout the course of the game. Sufficient battery life, as well as the specified width of game board lines and overall size of the game board also are attributed to the overall success of the game. Finally, the Outer Space theme of the game adds to the fun experience that each player has during game play.

7 FINAL PRODUCT

The final product is depicted below. Further details can be seen in Appendix N.


Final Product


Final Product as Delivered

7.1.1 Algorithm

A Master robot controls the game logic. Taking input from the users with cerebral palsy through the ability switches, the Master robot will display both the player number and the roll count for each player on the scoreboard. The Sparki robots will then be told how to navigate the game board.

8 BUDGET

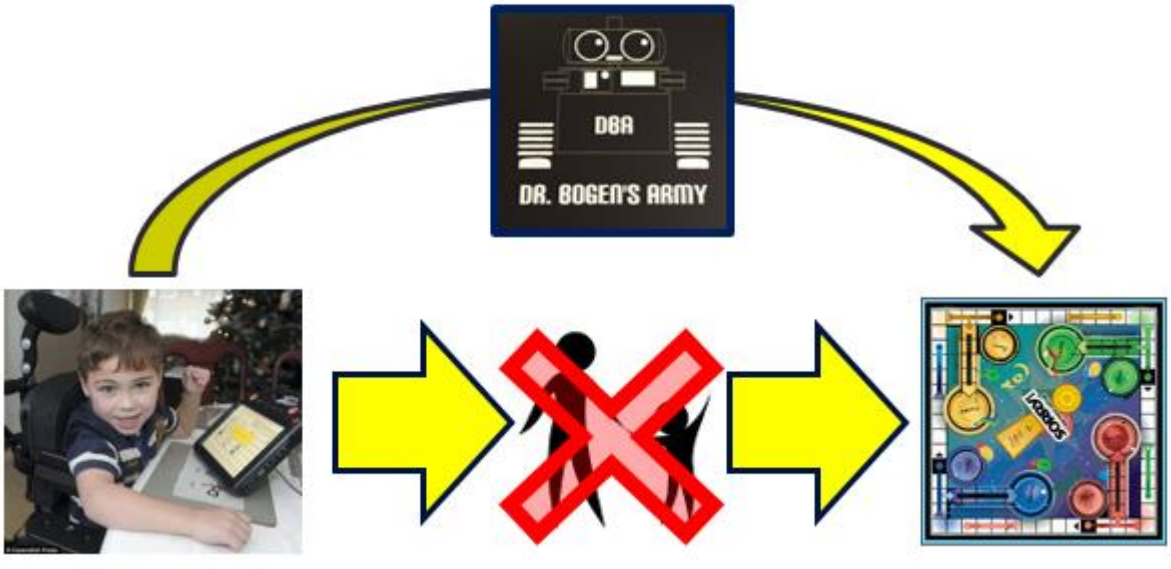
Our budget was developed throughout the design process in order incorporate both economically feasible items but also with a high degree of customizability. Our funds were provided by the Senior Design Budget as well as funds from Dr. Bogen's Lab. Further developments would include a standardization of materials so that the game could be reproduced by others with a lower cost. The total cost was of our project was $1192. See Appendix M for more specific details.

9 CONCLUSIONS AND SUMMARY

The goal of our project was to build an interactive board game to increase cognitive development and social interaction among CP children through play. This includes objectives to limit staff involvement and engage the CP children in a social, leisurely setting. The design of the game was achieved through iterations of concept testing and product development and is now fully functional. Testing of the game on site at HMS has proved its ability to allow unaided HMS students to interact with the physical world to play a complete board game simulated by robots that are controlled remotely with the use of ability switches. All of our specifications were met except game duration and cost. This means that the game is logistically playable, though the HMS school may have to use it in the after school program rather than during the day, and addresses all the clinical goals as stated above. Additionally, we were the first senior design project where the coordinators at HMS specifically reached out and requested for our game to be delivered to them, indicating that our game does in fact fulfill a need. Recommendations to improve the game in the future include using robots that are faster to address game duration, using 3D printed ability switches from a past senior design project to reduce costs, increasing the number of buttons for more advanced players, introducing a teamwork component by having multiple robots per color, adding sounds to the game, and adding movements of the robot eyes.

APPENDIX A: PROJECT ILLUSTRATION

Figure 1: Project Illustration



APPENDIX B: DESIGN QFDS

Figure 2: QFD (Materials)

Material	Durability	Erasibility	Waterproof	Tearing	Price	Ability for Expansion	Ease of Storage	Creases	Final
Vinyl 1	8	9	10	9	1	7	8	7	59
Vinyl 2	8	9	10	9	1	7	8	7	59
Vinyl 3	7	8	10	9	1	7	7	8	57
Vinyl 4	7	7	10	9	1	7	7	8	56
Vinyl 5	8	7	10	9	1	7	8	5	55
Vinyl 6	8	8	10	9	1	7	8	5	56
Tarp	10	9	10	9	8	8	4	2	60
Poster board	4	4	0	2	10	4	1	1	26
Sheet	8	4	5	9	10	9	10	8	63

Figure 3: QFD (Microcontroller)

	Cost	Open Source	Ease of Use	Features
Arduino Uno	10	10	10	5
BeagleBone Black	5	3	5	10
Raspberry Pi	8	8	7	8

	Importance
Cost	5
Open Source	8
Ease of Use	10
Features	5

	Total Score
Arduino Uno	255
BeagleBone Black	149
Raspberry Pi	214

Figure 4: QFD (Robot)

	Customizability	Cost	Functionality (Sensors)	Size	Speed	Communication (Bluetooth)	Setup Time
Arduino Robot Kit (DIY)	8	3 (197.98)	1 (no sensors included)	4	7	5 (Bluetooth compatible)	3
Omniwheel Robot	3	8 (118.99)	1 (no sensors included)	6	3	5 (Bluetooth compatible)	10 (ships already assembled)
Propeller ActivityBots Kits	6	7 (139.99)	4 (some sensors included)	10	7	5 (Bluetooth compatible)	4
Sparki	5	5 (149)	10 (large variety and number of sensors included)	10	4	10 (Bluetooth included)	10 (ships already assembled)

	Importance
Customizability	10
Cost	8
Functionality (Sensors)	10
Size	7
Speed	4
Communication (Bluetooth)	9
Setup Time	10

	Total Score
Arduino Robot Kit	245
Omniwheel Robot	303
Propeller ActivityBots Kits	339
Sparki	**466**

Appendix C: Adapting Sorry to Robotic Form

Figure 5: Line Tracking Mechanism

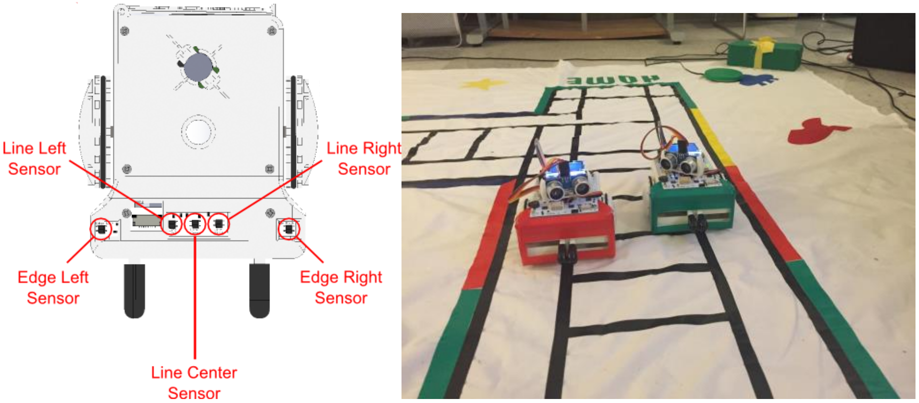


Figure 6: Passing Lane Mechanism

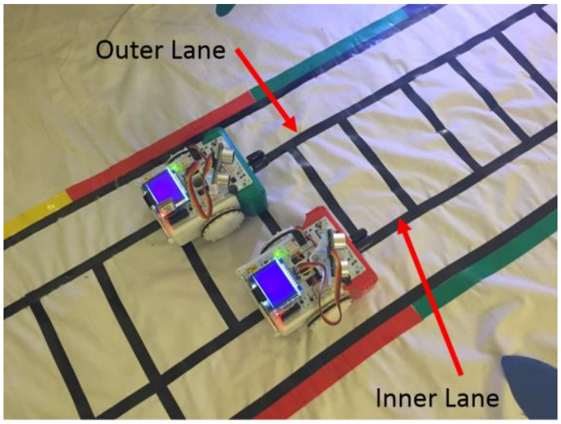


Figure 7: Resting Spaces on the Game Board

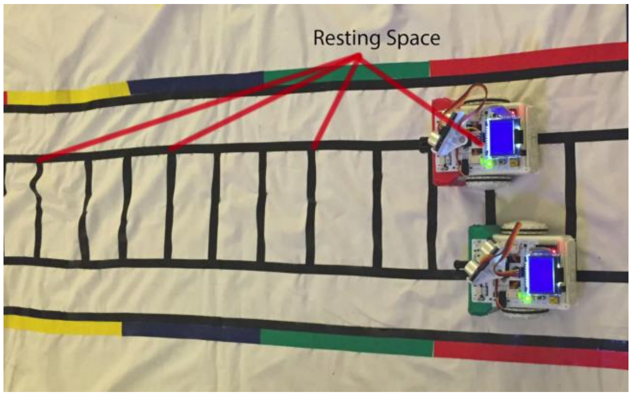
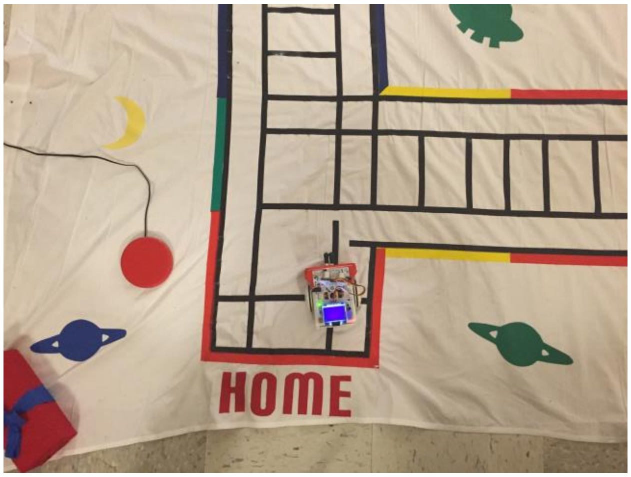


Figure 8: Home on the Game Board

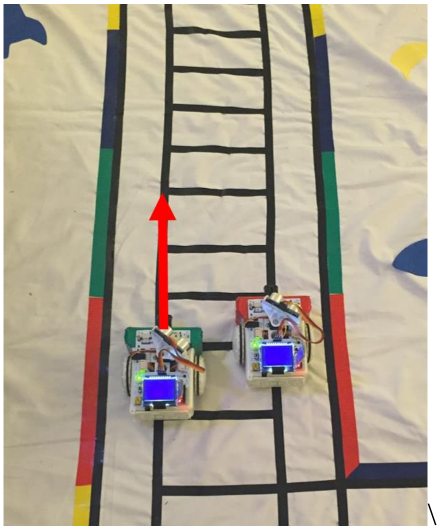


APPENDIX D: MASTER AND SLAVE ROBOT DESIGN

Figure 9: 10 Discrete Slave Functions

Slave Function Case	Function
A	Move Forward 3
B	Move Out
C	Move In
D	Move Corner
E	Dead – Go Home
F	Dead – At Home
G	Leave Castle Takeover
H	Leave Castle
I	Win – At Home
J	Win – Go Home

Figure 10: Move Forward 3



\

This specific move is move forward 3.

Figure 11: Master Robot, with Custom Bluetooth Shield

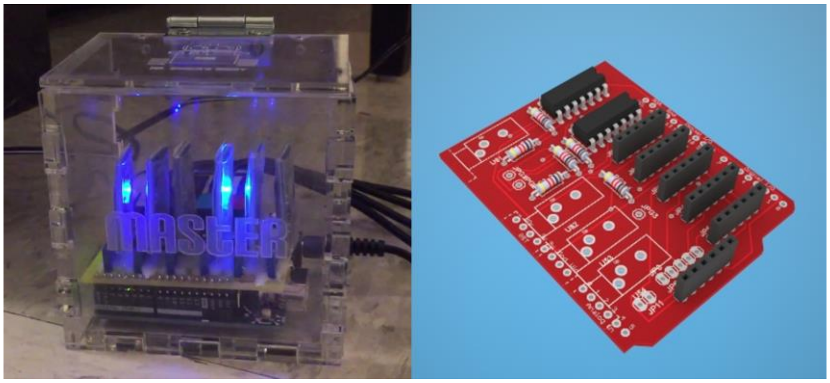
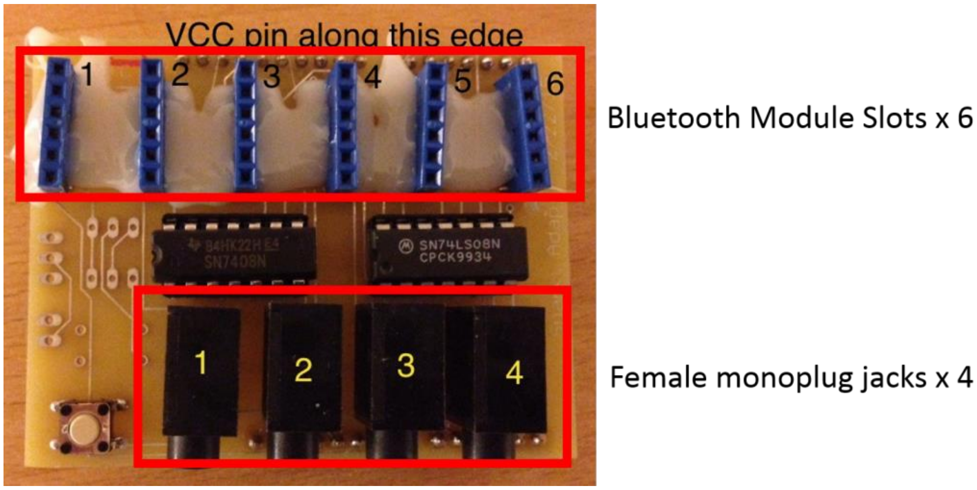


Figure 12: Custom Bluetooth Shield



APPENDIX E: SCOREBOARD DESIGN

Figure 13: Gameduino Shield (left) and Neopixel Shield (right)

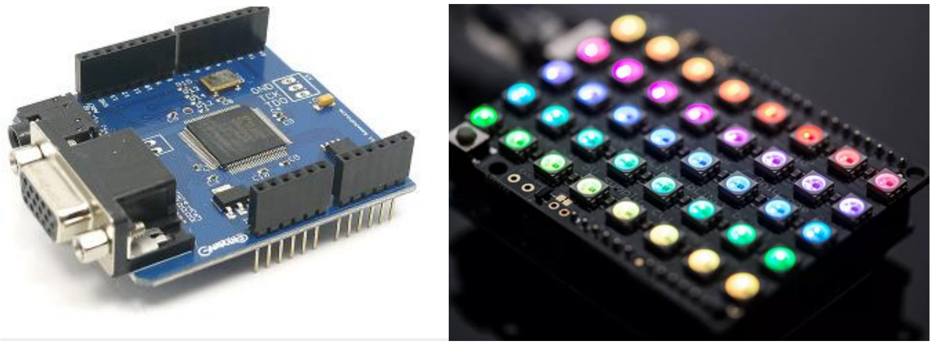


Figure 14: Front of Scoreboard

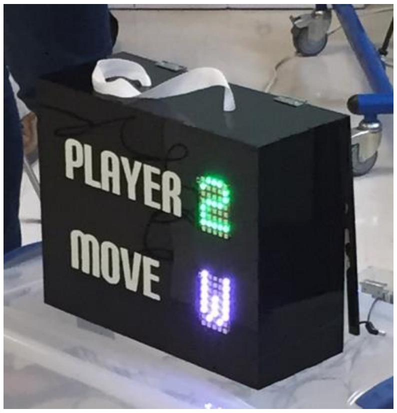


Figure 15: Back of Scoreboard

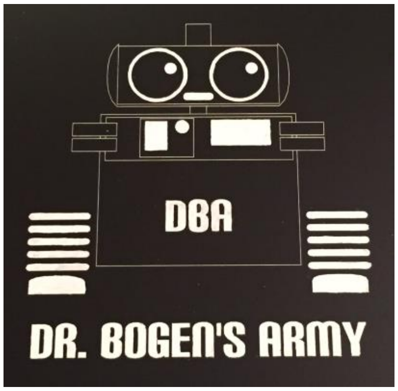


Figure 16: Master Robot Casing

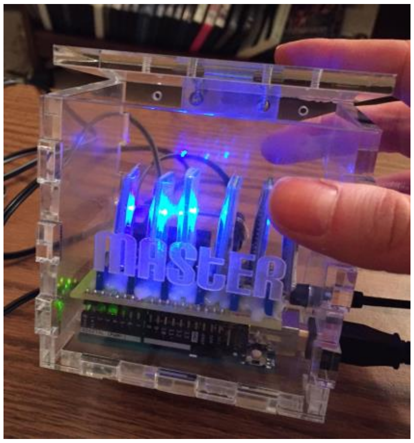


Figure 17: Commercial Ability Switches

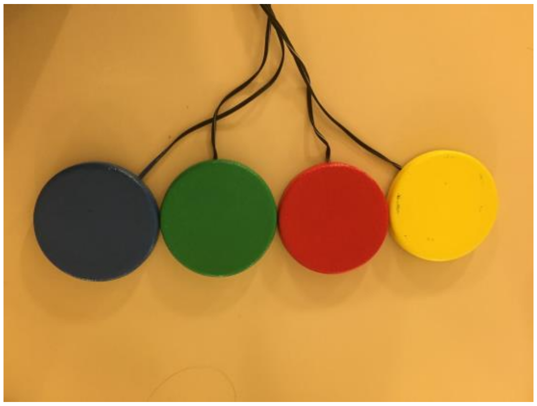
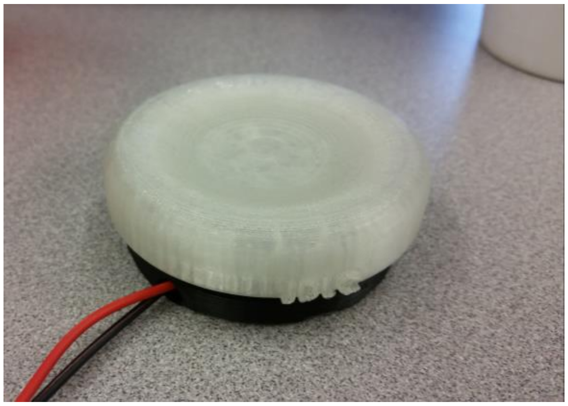


Figure 18: 3D Printed Ability Switches



APPENDIX F: EVALUATIONS PERFORMED AND RESULTS OF SPECIFICATION TESTING

Figure 19: Material Sensor Readings

Material	Sensor Reading
Vinyl #1	962.41 +/- 13.05
Vinyl #2	969.44 +/- 6.73
Vinyl #3	915.23 +/- 64.63
Vinyl #4	917.47 +/- 54.34
Vinyl #5	971.18 +/- 4.29
Vinyl #6	969.86 +/- 4.71
Tarp	984.64 +/- 3.29
Poster Board	966.56 +/- 2.95
Sheet	976.21 +/- 48.94

**Perfect contrast between white material and black line is 1000*

Figure 20: Robot Speed and Average Angle and Distance from Ideal

Material	Average Speed (cm/sec)	Average Angle (degrees)	Average Distance from Ideal (cm)
Vinyl #1	2.21 +/- 0.04	7.25 +/- 2.41	0.38 +/- 0.048
Vinyl #2	2.26 +/- 0.05	7.35 +/- 3.00	0.53 +/- 0.211
Tarp	2.24 +/- 0.10	6.55 +/- 4.79	0.38 +/- 0.197
Sheet	2.96 +/- 0.07	4.30 +/- 4.28	0.43 +/- 0.111

Figure 21: Angle Threshold Testing Setup

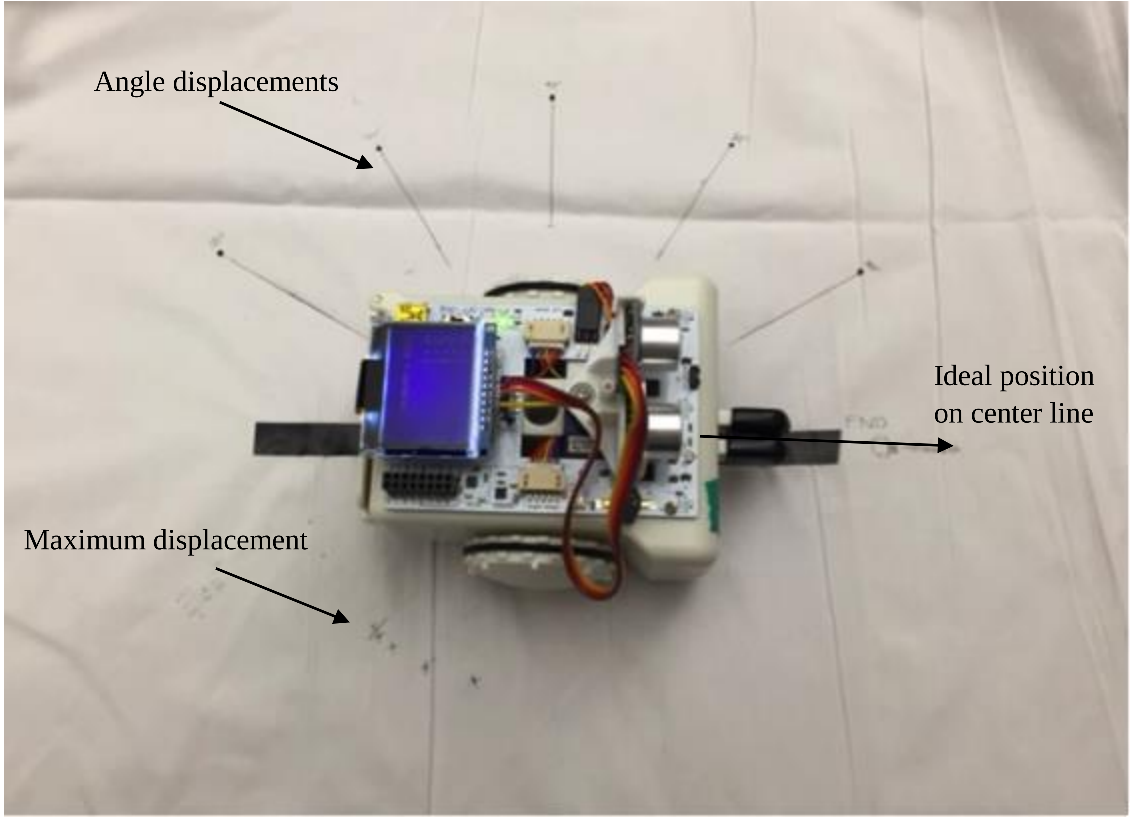


Figure 22: Angle Threshold Test

Angle to Left of Center Line (degrees)	Test Number	Ability to Re-Center on Line
0	1	yes
30	2	yes
60	3	yes
90	4	yes
120	5	yes
150	6	yes
180	7	yes
210	8	yes
220	9	yes
225	10	yes
226	11	yes
227	12	yes
228	13	yes
229	14	no
230	15	no

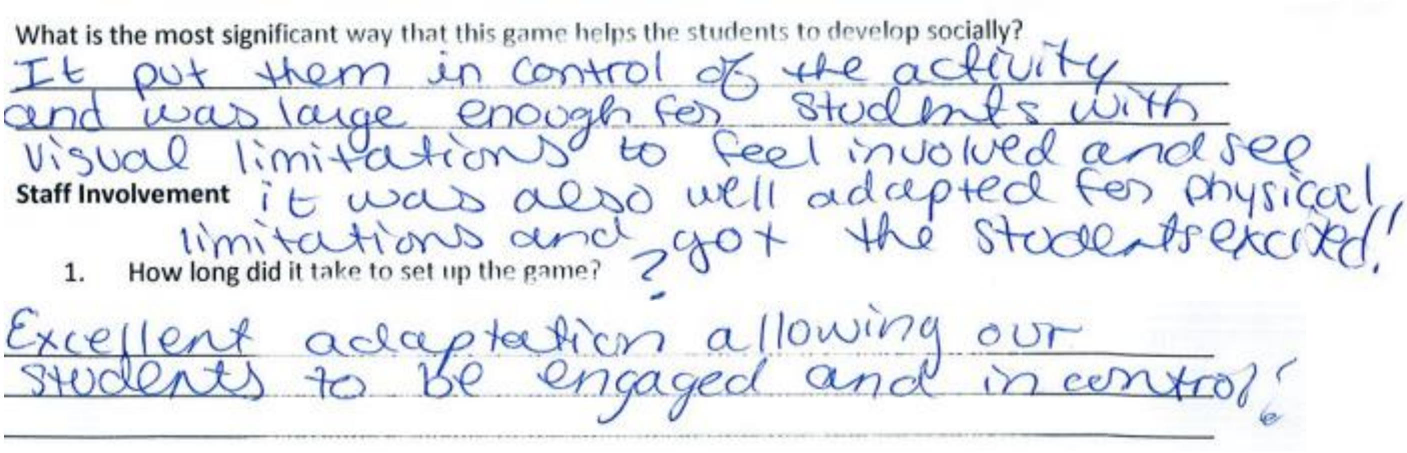
Figure 23: Robot Movement Timing

Master Command	Slave Function	Average Move Time (sec)	Maximum Move Time (sec)
a	move_fwd(3);	11.75 +/-1.00	13.15
b	move_fwd(1);	3.84 +/-.41	4.81
c	move_out();	4.67 +/-.59	5.58
d	move_in();	12.13 +/-.52	12.91
e	move_corner();	23.46 +/-.69	24.61

Figure 24: Possible Movement Combinations

Possible Combinations
move_out(); --> move_fwd(3);
move_out(); --> move_fwd(1);
move_fwd(3); --> move_in();
move_fwd(1); --> move_in();
move_fwd(3); --> move_corner();
move_fwd(1); --> move_corner();
move_corner(); --> move_in();
move_corner(); --> move_fwd(1);
move_corner(); --> move_fwd(3);

Figure 25: Written Staff Feedback



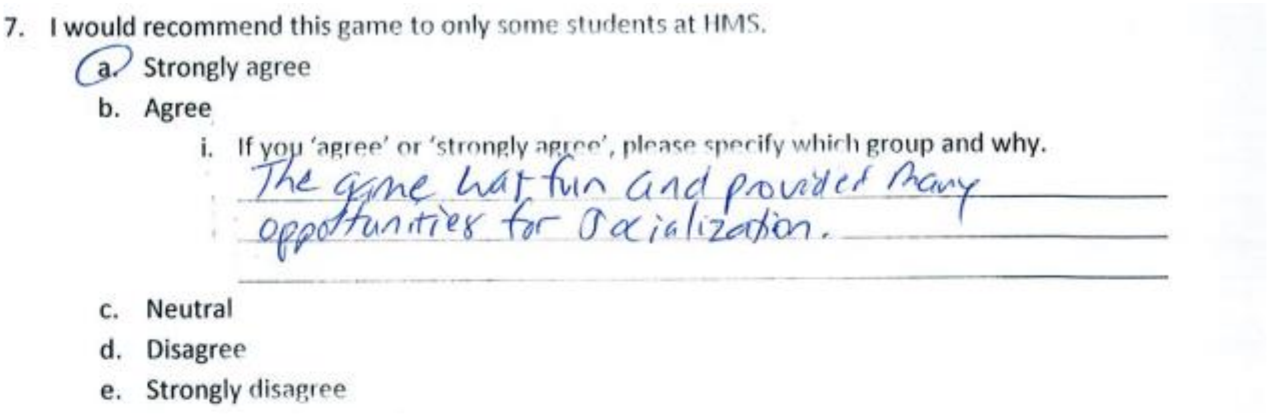
What is the most significant way that this game helps the students to develop socially?

It put them in control of the activity and was large enough for students with visual limitations to feel involved and see it was also well adapted for physical limitations and got the students excited!

Staff Involvement

1. **How long did it take to set up the game?**

Excellent adaptation allowing our students to be engaged and in control!



7. I would recommend this game to only some students at HMS.
   [✓ a.] Strongly agree
   b. Agree
      i. If you 'agree' or 'strongly agree', please specify which group and why.
      The game was fun and provided many opportunities for Socialization.
   c. Neutral
   d. Disagree
   e. Strongly disagree

APPENDIX G: PERFORMANCE AND IMPACT

Figure 27: Performance Based Specifications

Specification	Promised (Specified)	Delivered	Met/Unmet
Accuracy in Stopping Distance	0.53 cm	0.43 +/- 0.11	✓
Accuracy in Stopping Angle	228 degrees	4.30 +/- 4.28	✓
Function Reliability	100% success	100% success	✓
Processing Time	1.60 seconds	< 10 milliseconds	✓
Speed of Robot	2.22 cm/sec	2.96 +/- 0.07	✓

**Actual performance from average of 10 trials*

Figure 28: Clinical Need Specifications

Specification	Promised (Specified)	Delivered (Actual Performance)	Met/Unmet
Social Interaction	1.83 +/- 1.04	1.36 +/- 0.63	✓
Staff Involvement	1 +/- 0.00	1 +/- 0.00	✓
Student Engagement	2.32 +/- 1.39	1.45 +/- 0.60	✓

**promised specifications are based off of currently available game results obtained via a survey*

Figure 29: Design Specifications

Specification	Promised (Specified)	Delivered (Actual Performance)	Met/Unmet
Robot: Sensors	5	5	✓
Robot: Method of Communication	10.00 meters	10.67 +/- 0.3 meters	✓
Robot: Wheels	2	2	✓
Robot: Platform	Arduino	Arduino	✓
User: Age Range of Players	1st-8th grade	13-17 years old	*further testing required
Game Logistics: Cost	$1070	$1,207 (actual) $1,008 (*not including expenses due to items not used in final project)	✓-
Game Logistics: Number of Players	2	4	✓
Game Logistics: Number of Robots per Player	1	1	✓
Game Logistics: Game Duration	60 minutes	47 minutes to 2 hours and 37 minutes	✓- (met at HMS)
Game Logistics: Number of Utility Buttons per Player	1	1	✓
Game Logistics: Battery Life	60 minutes	273 +/- 49 minutes	✓
Game Board: Width of Line	< 2.2 cm	1.27 cm	✓
Game Board: Board Size	305 cm x 241 cm	203.8 cm x 203.8 cm	✓
Game Board: Theme	Princesses and Princes	Space theme	✓

Appendix H: Original Survey

HMS Senior Design Survey

Date: ___________

Name of Game: ______________

Number of Students: ______________

Students

Please describe the age group/classification of the group of students that played the game.

____
____
____

Social Interaction

1. The students communicated verbally or through gestures with each other during the course of gameplay
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
2. The students benefited socially by playing this game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
3. The observers (if any) benefited socially by observing the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
4. The students had fun playing the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree

    e. Strongly disagree

5. Games for more than one player are useful for the students at HMS.
    a. Strongly agree
    b. Agree
    c. Neutral
    d. Disagree
    e. Strongly disagree
6. The students showed elements of teamwork and further developed their teamwork skills.
    a. Strongly agree
    b. Agree
    c. Neutral
    d. Disagree
    e. Strongly disagree
7. Games for more than one player are more beneficial than single player games to the students at HMS.
    a. Strongly agree
    b. Agree
    c. Neutral
    d. Disagree
    e. Strongly disagree
    f.
8. This game involves: (check all that apply)

    ☐teamwork

    ☐negotiation

    ☐conflict resolution

    ☐sharing

What is the most significant way that this game helps the students to develop socially?

____

____

Staff Involvement

1. How long did it take to set up the game?
    a. ______________ minutes
2. How many staff members helped to set up the game?
    a. ______________
3. How many times did the staff have to intervene during the course of the game (aside from setup/cleanup)?

   a. ______________

4. Were the students able to successfully complete the game?
   a. Yes
   b. No
5. If ‘yes’, were the students able to successfully complete the game without staff involvement?
   a. Yes
   b. No
6. If ‘no,’ why were the students unable to successfully complete the game?
   a. ____
      ____
      ____

If staff involvement was required to successfully complete the game, why did the staff intervene?

____

____

____

____

Engagement

1. This game was appropriate for this group of students at HMS.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
      i. If you ‘disagree’ or ‘strongly disagree’, please explain why.
         ____
         ____
         ____
2. The students were engaged in the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
3. I would recommend this game to all students at HMS.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
4. The students could maintain focus throughout the entire game.

   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree

5. The students understood the goal of the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
6. The students were able to have the feeling of controlling something within their environment.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
7. I would recommend this game to only some students at HMS.
   a. Strongly agree
   b. Agree
      i. If you ‘agree’ or ‘strongly agree’, please specify which group and why.
      ____
      ____
      ____
   c. Neutral
   d. Disagree
   e. Strongly disagree
8. The students wanted to win the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
9. The students reacted when something negative or positive happened to them during the game.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
10. The students reacted when something negative or positive happened to another player during the game.
   a. Strongly agree
   b. Agree
   c. Neutral

   d. Disagree
   e. Strongly disagree

Additional comments regarding student engagement in the game:

____
____
____

Overall

1. The students have played this game before.
   a. Yes, approximately _______ times
   b. No
2. This game will be played again in the future at HMS.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
3. I would recommend this game to other facilities similar to HMS.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree
4. I would recommend this game to another person/group.
   a. Strongly agree
   b. Agree
      i. If ‘agree’ or ‘strongly agree,’ to whom? ______________________________
   c. Neutral
   d. Disagree
   e. Strongly disagree

Additional comments regarding the game overall and/or the experience for the students:

____
____
____

What other games are currently available to the students at HMS <u>that require</u> staff involvement?

____

____

____

What other games are currently available to the students at HMS that do not require staff involvement?

____

____

____

In your experience, what are the most popular games amongst the students at HMS?

____

____

____

Are there any features/components to a game that you feel would benefit the students at HMS that is not already available to them currently?

____

____

____

Logistical

1. All students could see the gameboard/components of the game.
    a. Strongly agree
    b. Agree
    c. Neutral
    d. Disagree
    e. Strongly disagree
2. All students could hear the components of the game (if applicable).
    a. Strongly agree
    b. Agree
    c. Neutral
    d. Disagree
    e. Strongly disagree
3. This game could be played in a number of classes at HMS.
    a. Strongly agree
    b. Agree
    c. Neutral

   d. Disagree
   e. Strongly disagree

4. This game would be easy to transfer to another location.
   a. Strongly agree
   b. Agree
   c. Neutral
   d. Disagree
   e. Strongly disagree

APPENDIX I: SLAVE CODE

```
/*******************************************
* HMS Sorry Game (SLAVE v4)
********************************************/

#include <Sparki.h>
const int threshold = 500; //line sensors thereshold.
const int linewidth = 1.27; //width of line on gameboard
const int sensorsDistance = 4.3+linewidth/2; //distance [cm] between the line sensor and the wheels
centers plus half the width of the black line.
const int turnSpeed = 40; //used to turn the robot over an external center of rotation.
const int outOfTheLineAngle = 60; //used to go outside the line and then find the branches.

bool  edgeLeft = false,
lineLeft = false,
lineCenter = false,
lineRight = false,
edgeRight = false;

String state = "undefined";

void setup()
{
 Serial1.begin(9600);
 //indicates to the user that the program started:
 sparki.beep(440, 300);
 delay(300);
 sparki.beep(880, 500);
 readSensors();
}
void loop()
{

 //receives command from remote computer
 if(Serial1.available())
 {
   int inByte = Serial1.read();
   sparki.print((char)inByte);
   sparki.updateLCD();
   switch((char)inByte)
   {
   case 'a':
     move_fwd(3);
     delay(1000);
     break;
   case 'b':
     move_out();
```

```
    delay(1000);
    break;
  case 'c':
    move_in();
    delay(1000);
    break;
  case 'd':
    move_corner();
    delay(1000);
    break;
  case 'e':
    dead_go_home();
    delay(1000);
    break;
  case 'f':
    dead_at_home();
    delay(1000);
    break;
  case 'g':
    leave_castle_takeover();
    delay(1000);
    break;
  case 'h':
    leave_castle();
    delay(1000);
    break;
  case 'i':
    win_at_home();
    delay(1000);
    break;
  case 'j':
    win_go_home();
    delay(1000);
    break;
  }
  sparki.clearLCD();
  sparki.updateLCD();
 }
}

void readSensors()
{
 //each sensor is 1 if reading white, and 0 if reading black:
 edgeLeft =  sparki.edgeLeft() > threshold;
 lineLeft =  sparki.lineLeft() > threshold;
 lineCenter = sparki.lineCenter() > threshold;
 lineRight =  sparki.lineRight() > threshold;
 edgeRight = sparki.edgeRight() > threshold;
```

```
}

void showSensorsAndState()
{
 sparki.clearLCD();

 sparki.print("eL = ");
 sparki.println(edgeLeft);
 sparki.print("lL = ");
 sparki.println(lineLeft);
 sparki.print("lC = ");
 sparki.println(lineCenter);
 sparki.print("lR = ");
 sparki.println(lineRight);
 sparki.print("eR = ");
 sparki.println(edgeRight);
 sparki.println();
 sparki.println(String("state = ") + state);

 sparki.updateLCD();
}

bool robotIsNotCentered()
{
 //when centered, lineLeft reads white, lineCenter reads black, lineRight reads white;
 return !(lineLeft && !lineCenter && lineRight);
}

bool isThereALeftBranch()
{
 //edgeLeft reads black:
 return !edgeLeft;
}

bool isThereARightBranch()
{
 //edgeRight reads black:
 return !edgeRight;
}

void centerRobot()
{
 while(robotIsNotCentered())
 {
   readSensors();
   showSensorsAndState();
 }
 sparki.beep(); //beep when centered
 state = "centered";
 showSensorsAndState();
}

//void turnLeft()
//{
//  state = "turn left";
//  sparki.moveForward(sensorsDistance);
//  sparki.moveLeft(outOfTheLineAngle);
//  sparki.moveLeft(); //turn left until the robot is centered.
//  centerRobot();
//  sparki.moveStop();
//}

//void turnRight()
//{
//  state = "turn right";
//  sparki.moveForward(sensorsDistance);
//  sparki.moveRight(outOfTheLineAngle);
//  sparki.moveRight(); //turn right until the robot is centered.
//  centerRobot();
//  sparki.moveStop();
//}

void turnHardLeft()
{
 state = "turn h_left";
 showSensorsAndState();
 sparki.moveLeft(outOfTheLineAngle);
 readSensors();
 sparki.moveLeft(); //turn left until the robot is centered.
 centerRobot();
 sparki.moveStop();
}

void turnHardRight()
{
 state = "turn h_right";
 sparki.moveRight(outOfTheLineAngle);
 sparki.moveRight(); //turn right until the robot is centered.
 centerRobot();
 sparki.moveStop();
}

void followLine()
{
 sparki.moveStop();
 state = "follow line";
 readSensors();
 showSensorsAndState();
 if (lineLeft && lineCenter) //white white
 {
  sparki.motorRotate(MOTOR_LEFT, DIR_CCW, 100);
 }
 else if (!lineLeft) //black black
 {
  sparki.motorRotate(MOTOR_RIGHT, DIR_CW, 100);
 }
 else if (lineLeft && !lineCenter) // white black
 {
  sparki.moveForward();
 }

 delay(200);
}

void move_fwd(int i)
{
 state = "move_fwd";
 showSensorsAndState();
 int counter = 0;
 while(counter<i)
 {
  readSensors();
  while(edgeRight)
  {
   readSensors();
   followLine();
  }
  sparki.moveStop();
  sparki.beep(700,300);
  sparki.moveForward(linewidth*.5);
  while(!edgeRight)
  {
   readSensors();
   followLine();
  }
  counter++;
  state = String(counter);
  showSensorsAndState();
 }
 sparki.moveStop();
 sparki.moveForward(sensorsDistance-linewidth);
}

void move_out()
```

```
{
 state = "move_out";
 showSensorsAndState();
 turnHardLeft();
 centerTurnLeft();
 move_fwd(1);
 turnHardRight();
 centerTurnRight();
}

void move_in()
{
 state = "move in";
 showSensorsAndState();
 turnHardRight();
 centerTurnRight();
 move_fwd(1);
 turnHardLeft();
 sparki.moveStop();
 centerTurnLeft();
}

void move_corner()
{
 state = "move_corner";
 showSensorsAndState();
 move_fwd(3);
 readSensors();
 turnHardRight();
 while(edgeRight)
 {
   followLine();
 }
 sparki.moveStop();
 sparki.moveForward(sensorsDistance);
}

void dead_go_home()
{
 state = "dead_go_home";
 showSensorsAndState();
 move_fwd(3);
 turnHardLeft();
 move_fwd(1);
 sparki.moveRight(180);

}
```

```
void dead_at_home()
{
 state = "dead_at_home";
 showSensorsAndState();
 turnHardLeft();
 move_fwd(1);
 turnHardLeft();
 move_fwd(2);
 sparki.moveRight(180);
}

void win_go_home()
{
 state = "win_go_home";
 showSensorsAndState();
 move_fwd(2);
 turnHardLeft();
 move_fwd(1);
 sparki.moveRight(180);
}

void win_at_home()
{
 state = "win_at_home";
 showSensorsAndState();
 turnHardLeft();
 turnHardLeft();
 move_fwd(1);
 turnHardRight();
 turnHardLeft();
 move_fwd(1);
 sparki.moveRight(180);
}

void leave_castle_takeover()
{
 state = "leave_castle_to";
 showSensorsAndState();
 move_fwd(1);
 turnHardRight();
 move_fwd(1);
 turnHardLeft();
 move_fwd(1);
}

void leave_castle()
{
 state = "leave_castle";
 showSensorsAndState();
 move_fwd(2);
 move_in();
}


void centerTurnLeft()
{
 state = "turn left center";
 showSensorsAndState();
 sparki.moveLeft(); //turn right until the robot is centered.
 centerRobot();
 sparki.moveStop();
}

void centerTurnRight()
{
 state = "turn right center";
 showSensorsAndState();
 sparki.moveRight(); //turn right until the robot is centered.
 centerRobot();
 sparki.moveStop();
}
```

APPENDIX J: MASTER CODE

```
/********************************************
* HMS Sorry Game  (MASTER v5)
*********************************************/

#include <SoftwareSerial.h>
SoftwareSerial BTSerial(8, 9); // RX | TX

int currentPlayer = -1; //indicator for current player’s number (0,1,2,3)
int currentRoll = 0; //current dice roll
int homePosition[] = {24, 6, 12, 18}; //array of home positions, will never change
float currentPosition[] = { -0.5, 5.5, 11.5, 17.5}; //array of current positions, will be updated
bool checkTarget = false; // 1 if next box has target in it, 0 if not
int checkTargetNumber = 0; //sparki number of target (0-3)
int targetMovesLeft = 0; //moves left to get home for target
bool almostDone[] = {false, false, false, false};
bool firstRoll[] = {true, true, true, true};

//delay times in milliseconds
int movefwd3 = 11533;
int moveout = 12128;
int movein = 11690;
int movecorner = 23247;
int deadgohome = 12500;
int deadathome = 23500;
int leavecastletakeover = 12533;
int leavecastle = 23223;
int win_at_home = 23500;
int win_go_home = 23500;

void setup()
{
 Serial.begin(9600);
 BTSerial.begin(9600);
 DDRC = DDRC | B00111111;  // Sets A0-A5 as outputs
 pinMode(4, INPUT_PULLUP); //input #1
 pinMode(5, INPUT_PULLUP); //input #2
 pinMode(6, INPUT_PULLUP); //input #3
 pinMode(7, INPUT_PULLUP); //input #4
 for (int i = 1; i < 7; i++) {
   sendChar(i, 'p');
   delay(500);
 }
 Serial.println("ready to start game");
}
```

```
void loop() {
 delay(100);
 currentPlayer = (currentPlayer + 1) % 4; //update current player number
 Serial.println("ready to roll dice");
 //while (digitalread(currentPlayer + 4) == 1) {} //currentPlayer is 0-3, digitalRead is 4-7; prompt button
 currentRoll = random(1, 5); //roll dice

 String stringOne = "Player ";//Serial check to see if player pressed to roll dice
 String stringTwo = stringOne + (currentPlayer + 1) + " rolled a " + currentRoll;
 Serial.println(stringTwo);

 updateDisplay();

 if (firstRoll[currentPlayer])
 {
   if (currentRoll > 2) //first roll has to be 3 or higher to get out of castle
   {
     Serial.println("entered initial move test");
     homeCheckTargetAvailable(); //check if target next to home (@ceiling check)
     Serial.println("finished homechecktargetavailable");
     //while (digitalread(currentPlayer + 4) == 1) {} //prompt button to move
     Serial.println("button is pressed");
     if (checkTarget) //if target is next to home
     {
       Serial.println("sending target home");
       sendTargetHome();
     } else {
       sendChar(currentPlayer + 1, 'h'); //if no target is next to home send leave castle
       delay(leavecastle);
     }
     currentPosition[currentPlayer] = currentPosition[currentPlayer] + 0.5; //update position
     firstRoll[currentPlayer] = false; //update current player first roll
     Serial.println("exit initial move test");
   }
 } else if (almostDone[currentPlayer]) {
   Serial.println("entered almostDone");
   if (currentPosition[currentPlayer] + currentRoll < homePosition[currentPlayer] + 1) { //if roll is not finishing move
     move();
   } else if (currentRoll == 1 && currentPosition[currentPlayer] == homePosition[currentPlayer]) { //if currentplayer is at home and player rolled 1
     //while (digitalread(currentPlayer + 4) == 1) {} //prompt button for winning
     sendChar(currentPlayer + 1, 'i'); //send win at home
     Serial.println("win_at_home sent");

     firstRoll[currentPlayer] = true; //reset target firstroll
     currentPosition[currentPlayer] = homePosition[currentPlayer] % 24 - 0.5; //update currentplayer position
     almostDone[currentPlayer] = false; //update almostDone to false
     currentRoll = 100; //adjust display to report WIN if currentRoll = 100
     updateDisplay();
     Serial.println("blueberry");

   } else if (currentPosition[currentPlayer] + currentRoll == homePosition[currentPlayer] + 1) { //going to win after moving currentRoll
     //player wins (code needs fleshing out: win_castle when it gets to homePosition)
     for (int movesLeft =  homePosition[currentPlayer] - currentPosition[currentPlayer] - 1; movesLeft > 0; movesLeft--)
     {
       //while (digitalread(currentPlayer + 4) == 1) {}//prompt button for each move_fwd
       moveOne(currentPlayer);
       updatePosition(currentPlayer);
     }
     //while (digitalread(currentPlayer + 4) == 1) {}//prompt button for win_go_home (last move)
     Serial.println("sending win go home");
     sendChar(currentPlayer + 1, 'j'); //send win_go_home

     firstRoll[currentPlayer] = true; //reset target firstroll
     currentPosition[currentPlayer] = homePosition[currentPlayer] % 24 - 0.5; //update target position
     almostDone[currentPlayer] = false; //update almostDone to false
     currentRoll = 100; //adjust display to report WIN if currentRoll = 100
     updateDisplay();
     Serial.println("blueberry");

   }
 }
 else {
   Serial.println("normal move");
   move();
 }
 Serial.println("end of turn");


 String stringA = "Current positions:";

 for (int i = 0; i < 4; i = i + 1) {
   stringA = stringA + "  "+ currentPosition[i] +"|";
 }
 Serial.println(stringA);
 Serial.println("-------------------");


}
```

```
void sendChar(int sparkiNum, char letter) {
 PORTC = PORTC & B11000000; // Clear A0-A5. Direct write to PORTC.
 PORTC = PORTC | (1 << (sparkiNum - 1)); // Put a "1" in a single position, A0-A5, to enable that
bluetooth module.
 BTSerial.write(letter); // Send the character.
}

void move()
{
 checkTargetAvailable();// checks if there is a target, if there is, extracts target number
 Serial.println("checkTargetavailable done");
 //while (digitalread(currentPlayer + 4) == 1) {} //button prompt to move out
 Serial.println("button pressed");
 sendChar(currentPlayer + 1, 'b'); //send move_out to currentPlayer
 Serial.println("move out sent");
 delay(moveout);//delay by move_out duration

 for (int movesLeft = currentRoll - 1; movesLeft > 0; movesLeft--) //currentRoll - 1 because last
move can be takeover or normal moveOne
 {
   //while (digitalread(currentPlayer + 4) == 1) {}  //button prompt to move
   Serial.println("button pressed");
   moveOne(currentPlayer);
   updatePosition(currentPlayer);
 }

 //while (digitalread(currentPlayer + 4) == 1) {}  //button prompt for last move
 if (checkTarget) { //if there is target, send target home (final move and movein for currentPlayer
embedded in sendTargetHome)
   Serial.println("sending target home");
   sendTargetHome();
   Serial.println("target arrived at home");
 } else { //if no target, moveOne normally
   moveOne(currentPlayer);
   updatePosition(currentPlayer);
   sendChar(currentPlayer + 1, 'c'); //send move_in
   Serial.println("movein sent");
   delay(movein);//delay by move_in duration
 }

 if (homePosition[currentPlayer] - currentPosition[currentPlayer] <= 4 &&
homePosition[currentPlayer] - currentPosition[currentPlayer] >= 0) {
   almostDone[currentPlayer] = true; //set almostDone to true if near home
 }
}

void updateDisplay()
{
 sendChar(5, (char)(((int)'0') + currentPlayer + 1)); //update LED display #1 with player #
 while (currentRoll == 100) {
   sendChar(6, 'W'); //update LED display #2 with ‘W’ (infinite loop to denote end of game)
   delay(3000);
 }
 sendChar(6, (char)(((int)'0') + currentRoll)); //update LED display #2 with currentRoll
 Serial.println("updated scoreboard");
}

void checkTargetAvailable()
{
 for (int x = 0; x < 4; x++) {
   if (currentPosition[currentPlayer] + currentRoll == currentPosition[x]) {
     checkTarget = true;
     checkTargetNumber = x;
   }
 }
}

void homeCheckTargetAvailable()
{
 for (int x = 0; x < 4; x++) {
   if (ceil(currentPosition[currentPlayer]) == currentPosition[x] || ceil(currentPosition[currentPlayer])
== currentPosition[x] - 24) {
     checkTarget = true; //target exists in square next to home
     checkTargetNumber = x; //target sparki number 0-3
   }
 }
}

void sendTargetHome()
{
 if (homePosition[checkTargetNumber] - currentPosition[checkTargetNumber] > 1 ||
homePosition[checkTargetNumber] - currentPosition[checkTargetNumber] < 0) //if target is not
already at home
 {
   sendChar(checkTargetNumber + 1, 'b'); //send move out to target sparki
   delay(moveout); //delay for move_out duration
   moveOne(checkTargetNumber);
   updatePosition(checkTargetNumber);
   if (homePosition[checkTargetNumber] - currentPosition[checkTargetNumber] > 0) { //calculate
moves to get home (-1 bc last move is dead_go_home)
     targetMovesLeft = homePosition[checkTargetNumber] - currentPosition[checkTargetNumber] -
1;
   } else {
     targetMovesLeft = homePosition[checkTargetNumber] + 24 -
currentPosition[checkTargetNumber] - 1;
   }
   if (firstRoll[currentPlayer]) { //currentPlayer moves
     Serial.println("sending leave castle takeover");
     sendChar(currentPlayer + 1, 'g'); //send leave_castle_takeover
   } else {
     moveOne(currentPlayer);
     updatePosition(currentPlayer);
     sendChar(currentPlayer + 1, 'c'); //send move_in
     Serial.println("currentPlayer finished takeover");
   }

   for (int tMovesLeft = targetMovesLeft; tMovesLeft > 0; tMovesLeft--) //move target back to home
   {
     moveOne(checkTargetNumber);
     updatePosition(checkTargetNumber);
   }

   sendChar(checkTargetNumber + 1, 'e'); //send dead_go_home before final home square
   Serial.println("dead_go_home sent to target");

 } else if (homePosition[checkTargetNumber] - currentPosition[checkTargetNumber] == 1) {

   sendChar(checkTargetNumber + 1, 'b'); //send move out to target sparki
   delay(moveout); //delay for move_out duration
   sendChar(checkTargetNumber + 1, 'e');
   Serial.println("dead_go_home sent to target");

   moveOne(currentPlayer);
   updatePosition(currentPlayer);
   sendChar(currentPlayer + 1, 'c'); //send move_in
   delay(movein);//delay by move_in duration
   Serial.println("currentPlayer finished takeover");

 } else if (homePosition[checkTargetNumber] == currentPosition[checkTargetNumber]) {
   sendChar(checkTargetNumber + 1, 'f');
   Serial.println("dead_at_home sent to target");
   delay(deadathome);
   moveOne(currentPlayer);
   updatePosition(currentPlayer);
   sendChar(currentPlayer + 1, 'c'); //send move_in
   delay(movein);//delay by move_in duration
   Serial.println("currentPlayer finished takeover");
 }
 firstRoll[checkTargetNumber] = true; //reset target firstroll
 currentPosition[checkTargetNumber] = (int)homePosition[checkTargetNumber] % 24 - 0.5; //update
target position to home
 almostDone[checkTargetNumber] = false; //update almostDone to false
 checkTarget = false; //update checkTarget to false;
```

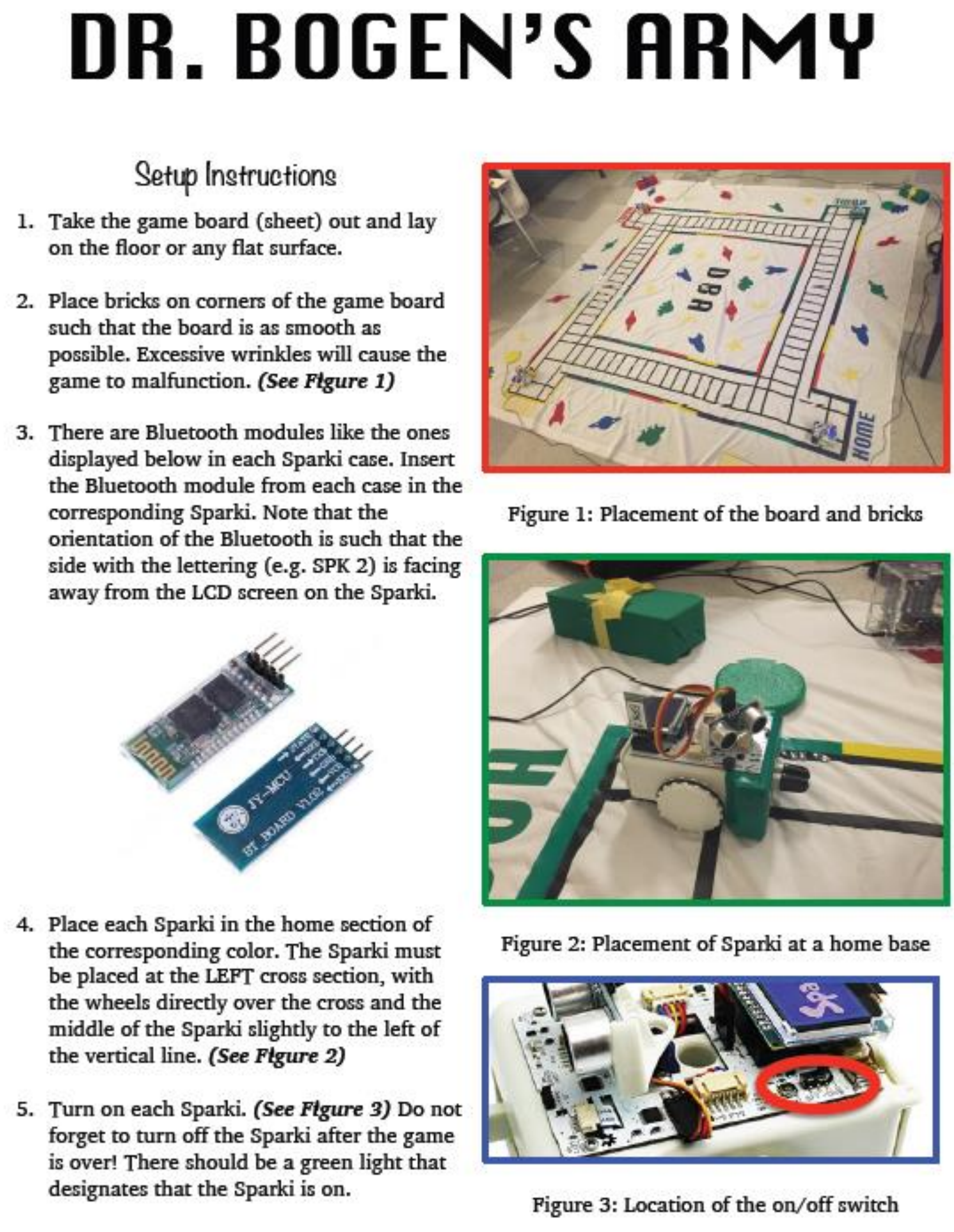
```
}

void moveOne(int player) {
 if ((int)(currentPosition[player] + 1) % 6 == 0) {
   sendChar(player + 1, 'd'); //send move_corner
   Serial.println("move corner sent");
   delay(movecorner); //delay move_corner duration
 }
 else {
   sendChar(player + 1, 'a'); //send move_fwd
   Serial.println("move fwd sent");
   delay(movefwd3); //delay move_fwd(3) duration
 }

}

void updatePosition(int player) {
 if ((int)(currentPosition[player] + 1) % 24 == 0) {
   currentPosition[player] = 24; //if next pos is 24, state 24 (not 0)
 } else {
   currentPosition[player] = (int)(currentPosition[player] + 1) % 24; //otherwise, give number 1-23
 }
}
```

APPENDIX K: INSTRUCTION MANUAL



DR. BOGEN'S ARMY

Setup Instructions

1. Take the game board (sheet) out and lay on the floor or any flat surface.

2. Place bricks on corners of the game board such that the board is as smooth as possible. Excessive wrinkles will cause the game to malfunction. ***(See Figure 1)***

3. There are Bluetooth modules like the ones displayed below in each Sparki case. Insert the Bluetooth module from each case in the corresponding Sparki. Note that the orientation of the Bluetooth is such that the side with the lettering (e.g. SPK 2) is facing away from the LCD screen on the Sparki.

4. Place each Sparki in the home section of the corresponding color. The Sparki must be placed at the LEFT cross section, with the wheels directly over the cross and the middle of the Sparki slightly to the left of the vertical line. ***(See Figure 2)***

5. Turn on each Sparki. ***(See Figure 3)*** Do not forget to turn off the Sparki after the game is over! There should be a green light that designates that the Sparki is on.

Figure 1: Placement of the board and bricks

Figure 2: Placement of Sparki at a home base

Figure 3: Location of the on/off switch

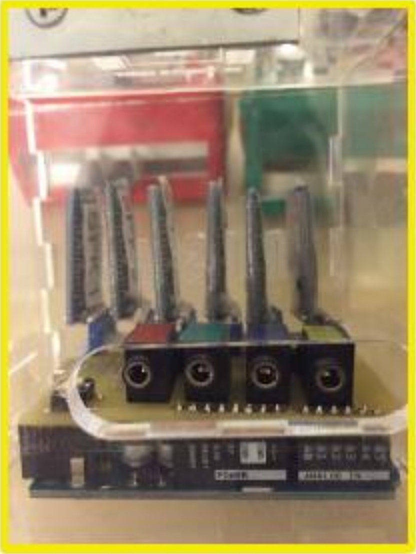


Figure 4: Color coded locations to plug in buttons

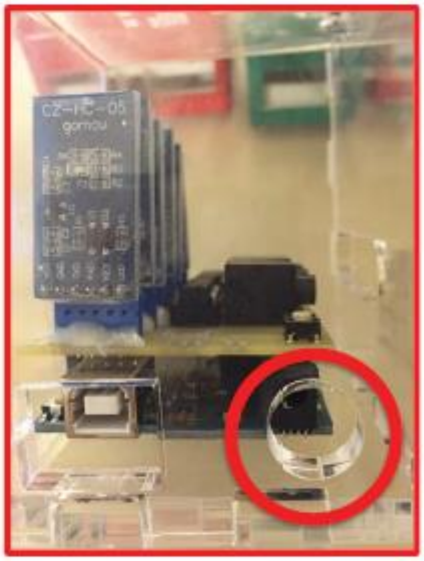


Figure 5: Location of plug for power

6. Plug in each button to the Master box according to the color codes. ***(See Figure 4)***
7. The Master box must also be plugged into power. ***(See Figure 5)***
8. Place the scoreboard somewhere where all participants can see it. Plug scoreboard into power. ***(See Figure 6)***
9. Give each participant a button. Each player will have one button to control his or her color-coded robot.

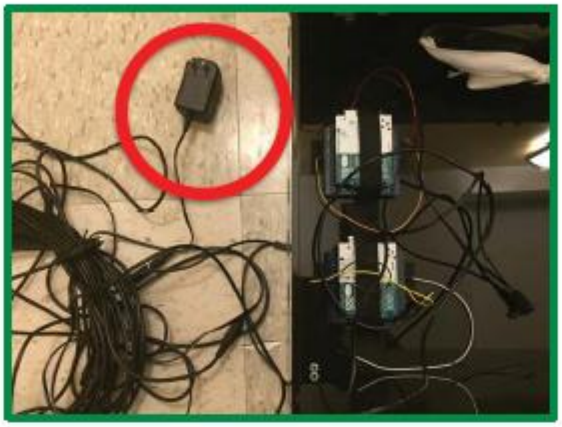


Figure 6: Inside of scoreboard - wiring to power

Game Instructions

The game begins with all players at their home base. Each player will interact with the game only through the press of a button. To win this game, the participant's Sparki must travel around the game board to return back to its respective home base.

The player must roll a 3 or higher in order to leave the home base. If a number lower than 3 is rolled, it will automatically be the next person's turn. This applies both to the start of the game and to whenever a player is sent home through a takeover.

When a player is on the game board, they will first press the button to roll the dice and to move their Sparki into the passing lane. To move forward one square, they will then press the button again. For example, if the first button press rolled a six, the participant would then press the button 6 more times to complete their move.

When the player gets within 4 moves of their home base, they must roll the exact number or smaller to advance. For example, if a player is 3 squares away from their home base, they can roll either a 1, 2, or 3 to proceed.

Takeovers

If the current player rolls a number and ends on a space that already has a player in it, the stationary player is sent home and must start from scratch again. The current player advances and remains in the square.

Winning

Once a player wins, the scoreboard will designate so with displaying a "W" instead of a number of moves.

FAQ

Charging the batteries

It is advised that batteries be charged between uses. A charger is included in the box, as depicted below

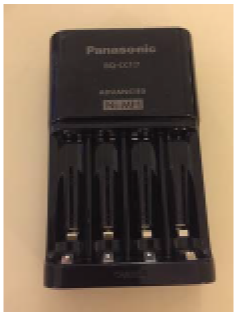


Figure 7: Charger for batteries

The robots are not working. What should I do?

1. Check if the wireless modules are connected correctly. Both the ones in the master box and on the Sparkis should be flashing blue intermittently.
2. Check if each robot has operating batteries. If the batteries have started to die, the LED screen on the individual batteries will be dimmer than usual. If the batteries are completely dead, the green light on the robot will not turn on.
3. Check that the connections of the buttons to the extension cables and to the master box are secure.

The robot didn't do what it was supposed to do. What should I do?

1. Take notes about the scenario! What is showing on the scoreboard? What was the robot supposed to do? What did it actually do? Record the information and send to e-mail contact so that we can schedule a visit and make alterations accordingly.

APPENDIX L: SPECIFICATIONS

Performance Based Specifications

Specification	**Promised (specified)**	**Delivered (actual performance from average of 10 trials)**	**Evaluation Method**	**Justification**
Accuracy in Stopping Distance	0.53 cm	0.43 cm +/- 0.11	Measure the difference between the robot's center to its target position after completing test move. Compared to the worst performing material (Vinyl photo material)	To evaluate reliability of robot movements responsible for taking the robots to their correct positions, specifically distance (Slave code testing)
Accuracy in Stopping Angle	228 degrees	4.3 degrees	Measure the difference between the robot's center line to its ideal direction of 0 degrees after completing test move	To evaluate reliability of robot movements responsible for aligning the robot's direction to the correct direction (Slave code testing)
Function Reliability	100% success	100% success	Measure the success rate of the robot's movements for each programmed movement and average success rate to arrive at a composite reliability metric	To evaluate overall reliability of robot's performance in executing the correct move e.g. when the robot is supposed to turn a corner, a move corner function sent (Master code testing)
Processing Time	1.60 seconds	< 10 milliseconds (no standard deviation or	Measure the delay between a button press and the initiation of the programmed robot movement	Processing time needs to be lower than 1.6 seconds, which is the maximum time to ensure

		average due to reaction time being larger than processing time)		correlative relationship between button press and initiation of the programmed robot movement
Speed of Robot	2.22 cm/sec	2.96+/-.071 cm/sec	Measure the time it takes for robot to complete the most basic maneuver based on various speeds. Compared to the worst material performance (Vinyl photo paper)	Robot needs to move around the board such that the game can be completed within the duration of gameplay schedule at HMS. This specifically tested the move forward three function.

Clinical Need Specifications

*promised specifications are based off of currently available game results obtained via a survey, where teachers were asked to quantify these qualities with assessments from 1-5, 1 being the best number achievable

Specification	**Promised (specified)**	**Delivered (actual performance)**	**Evaluation Method**	**Justification**
Social Interaction	1.83 +/- 1.04	1.36 +/- 0.63	Based on survey results that examine effectiveness of encouraging social development in the game when compared to previous games played	Play is an important element of child development. Key areas that are enriched for children who play games include teamwork, sharing, negotiation, and conflict resolution
Staff Involvement	1 +/- 0.00	1 +/- 0.00	Measure amount of time staff spends setting up, cleaning up, and helping with game. Combined with results from survey questions regarding staff involvement	Current games at HMS require extensive staff involvement. Reduction in this category would give the children the opportunity to take control of what they are doing without the help that they require in so many aspects of their daily

				lives
Student Engagement	2.32 +/- 1.39	1.45 +/- 0.60	Based on survey results that examine the level of engagement of children in the game when compared to previous games played	Any potential benefits the game may have will be irrelevant if the children are not motivated to play it; we want to create a game that the children feel invested in

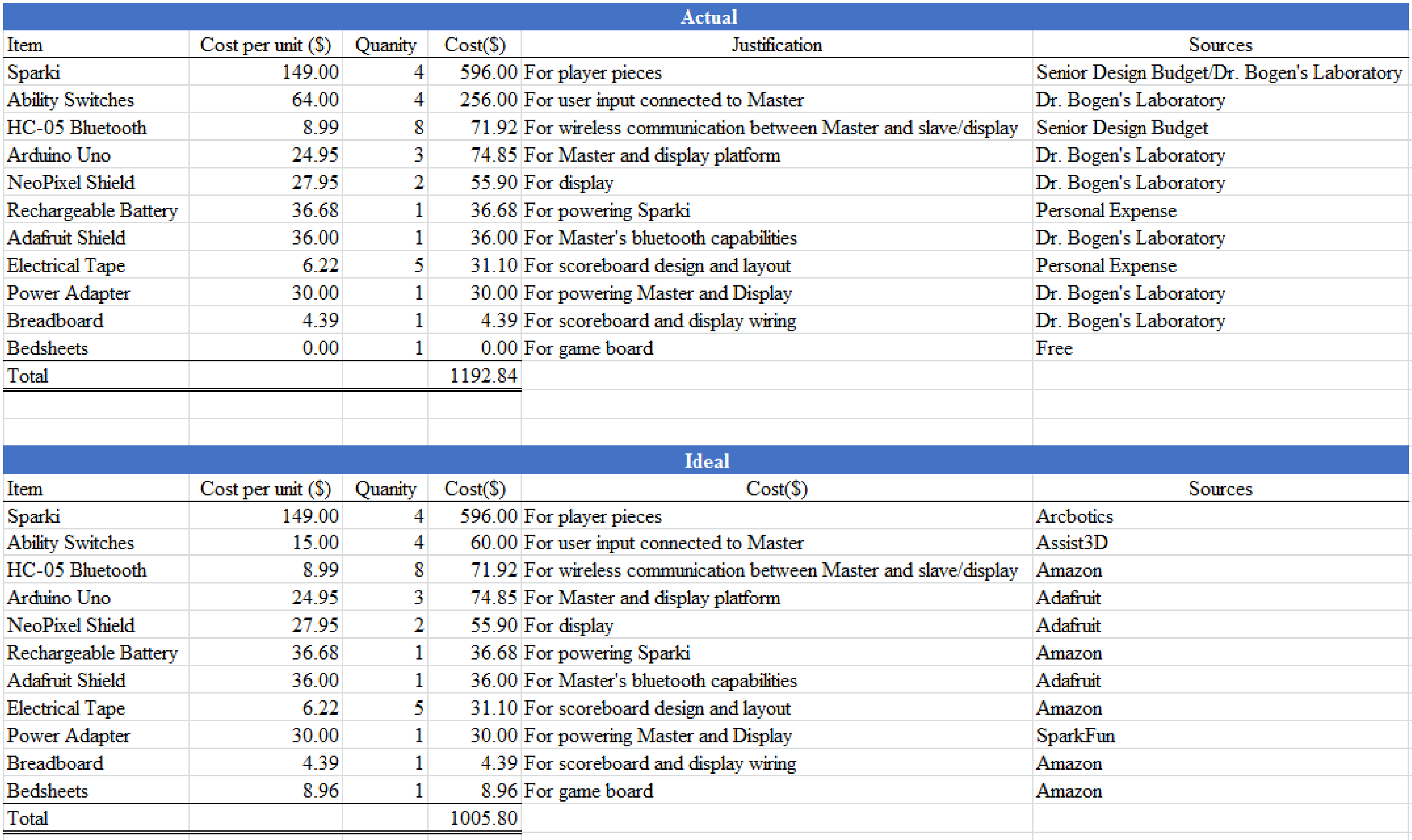
Design Specifications

Specification	**Promised (specified)**	**Delivered (actual performance)**	**Evaluation Method**	**Justification**
Robot: Sensors	5	5	Observation of number of sensors on robot	Line tracking requires at least 3 sensors. An additional sensor is necessary to pick up on other lines for cues on the game board. It was determined that 4 sensors were required for a robot game piece to navigate the game board. The 5th sensor was added to account for limitations of coding experience within the team
Robot: Method of Communication	10 meters	10.67 +/- 0.3 meters	Measure distance at which data is no longer transmitted via Bluetooth	Reliability of communication choice, wireless technology, ease of communication. The size of the room dictates a minimum distance at which the Master robot and Slave robots must communicate
Robot: Wheels	2	2	Observation of number of	Robot's mean of moving about the

			wheels on robot	board will contribute to the accuracy and consistency. Two robots are required for the robot to remain upright. Additional wheels would require additional servos to program.
Robot: Platform	Arduino	Arduino	QFD table with minimum of three platforms	Platform can be a serious limitation. Resources available in each platform differ, including the amount of technical support that can be found
User: Age Range of Players	1st-8th grade	1st-8th grade	Ask staff for age range of players in game	The game must address as large a population as feasible in order to achieve our clinical needs. This means that for our audience, the academic age range of the target children must be the limit
Game Logistics: Cost	$1070	Total: 1192.84 (See Budget for more details)	Add costs for all components for game-- Sparkis, utility buttons, game board, Bluetooth modules, LED screens, Master robot board, batteries, board for LED screen, electrical tape, theme elements	In order for the game to be replicable and marketable to other schools, it must be relatively affordable compared to other robotic games
Game Logistics: Number of Players	2	4	Observation of students playing game	In order to fulfill the social element of the game, the game must be designed such that at least two players can play the game

				independently
Game Logistics: Number of Robots per Player	1	1	Observation of number of robots each player	Each utility button can be linked with only one robot, and in order to foster the idea of spatial cognition, it is important to establish a specific cause and effect mechanism
Game Logistics: Game Duration	60 minutes	47 minutes to 2 hours and 37 minutes	Time length of game from initial setup to cleanup	This needs to be a game that can fit within one class period at the HMS such that it can be played during the day with some staff supervision
Game Logistics: Number of Utility Buttons per Player	1	1	Observation of number of utility buttons that each player controls	The children at HMS have different skill levels, but 90% of children can press one utility button
Game Logistics: Battery Life	60 minutes	273 +/- 49 minutes	Time battery usage to determine how long the battery will last	Batteries need to last at least one game duration such that a game can be completed
Game Board: Width of Line	< 2.2 cm	1.27 cm (this is ideally exact as designated by the manufacturer)	Measure width of lines that are taped on game board	The width of the game board lines are limited by the distance between the three sensors on the robot and the width of these sensors
Game Board: Board Size	305 cm x 241 cm	203.8 cm x 203.8 cm	Measure size of board	Limited by the size of available tables at HMS
Game Board: Theme	Princesses and Princes	Outer space / primary colors theme	Poll the students to ensure that the game theme is successful	Game should be interactive and related in order to increase game engagement

APPENDIX M: BUDGET



Actual

Item	Cost per unit ($)	Quanity	Cost($)	Justification	Sources
Sparki	149.00	4	596.00	For player pieces	Senior Design Budget/Dr. Bogen's Laboratory
Ability Switches	64.00	4	256.00	For user input connected to Master	Dr. Bogen's Laboratory
HC-05 Bluetooth	8.99	8	71.92	For wireless communication between Master and slave/display	Senior Design Budget
Arduino Uno	24.95	3	74.85	For Master and display platform	Dr. Bogen's Laboratory
NeoPixel Shield	27.95	2	55.90	For display	Dr. Bogen's Laboratory
Rechargeable Battery	36.68	1	36.68	For powering Sparki	Personal Expense
Adafruit Shield	36.00	1	36.00	For Master's bluetooth capabilities	Dr. Bogen's Laboratory
Electrical Tape	6.22	5	31.10	For scoreboard design and layout	Personal Expense
Power Adapter	30.00	1	30.00	For powering Master and Display	Dr. Bogen's Laboratory
Breadboard	4.39	1	4.39	For scoreboard and display wiring	Dr. Bogen's Laboratory
Bedsheets	0.00	1	0.00	For game board	Free
Total			1192.84		

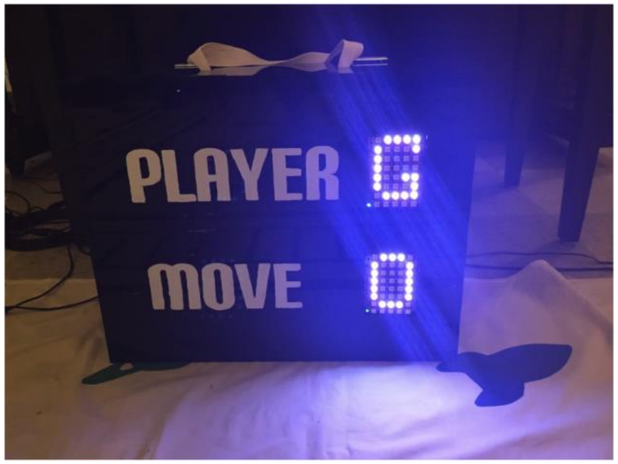
Ideal

Item	Cost per unit ($)	Quanity	Cost($)	Cost($)	Sources
Sparki	149.00	4	596.00	For player pieces	Arcbotics
Ability Switches	15.00	4	60.00	For user input connected to Master	Assist3D
HC-05 Bluetooth	8.99	8	71.92	For wireless communication between Master and slave/display	Amazon
Arduino Uno	24.95	3	74.85	For Master and display platform	Adafruit
NeoPixel Shield	27.95	2	55.90	For display	Adafruit
Rechargeable Battery	36.68	1	36.68	For powering Sparki	Amazon
Adafruit Shield	36.00	1	36.00	For Master's bluetooth capabilities	Adafruit
Electrical Tape	6.22	5	31.10	For scoreboard design and layout	Amazon
Power Adapter	30.00	1	30.00	For powering Master and Display	SparkFun
Breadboard	4.39	1	4.39	For scoreboard and display wiring	Amazon
Bedsheets	8.96	1	8.96	For game board	Amazon
Total			1005.80		

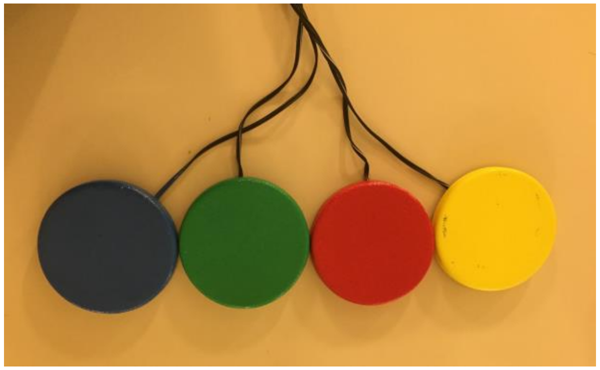
Ideal		
Item	Sources	Links
Sparki	Arcbotics	http://www.arcbotics.mybigcommerce.com/sparki-the-easy-robot-for-everyone/
Ability Switches	Assist3D	http://www.assist3d.org/
HC-05 Bluetooth	Amazon	http://www.amazon.com/JBtek-Wireless-Bluetooth-Transceiver-Arduino/dp/B00L083QAC/ref=sr_1_cc_1?s=aps&srs=2530190011&ie=UTF8&qid=1430175803&sr=8-1-catcorr&keywords=HC-05+Bluetooth
Arduino Uno	Adafruit	https://www.adafruit.com/products/50
NeoPixel Shield	Adafruit	http://www.adafruit.com/products/1430
Rechargeable Battery	Amazon	http://www.amazon.com/AmazonBasics-AA-Rechargeable-Batteries-16-Pack/dp/B007B9NV8Q/ref=sr_1_1?ie=UTF8&qid=1430175455&sr=8-1&keywords=rechargeable+aa+batteries
Adafruit Shield	Adafruit	http://www.adafruit.com/arduino?gclid=CNm3xe_ErsUCFdOPHwodCYcAzA
Electrical Tape	Amazon	http://www.amazon.com/gp/product/B000TPG3J8/ref=oh_aui_detailpage_o05_s00?ie=UTF8&psc=1
Power Adapter	SparkFun	https://www.sparkfun.com/products/298?gclid=CMzepoDFrsUCFQWRHwodS2UAwQ
Breadboard	Amazon	http://www.amazon.com/KKmoon-Solderless-Breadboard-BreadBoard-Arduino/dp/B00SWPJ6IW/ref=sr_1_10?s=electronics&ie=UTF8&qid=1430175337&sr=1-10&keywords=breadboard+electronics+kit
Bedsheets	Amazon	http://www.amazon.com/Sheet-White-Bedding-Thread-Percale/dp/B00HH9AT6W/ref=sr_1_31?s=home-garden&ie=UTF8&qid=1430175274&sr=1-31&keywords=bed+full+size+sheet+-pillowcase

Appendix N: Final Product Components

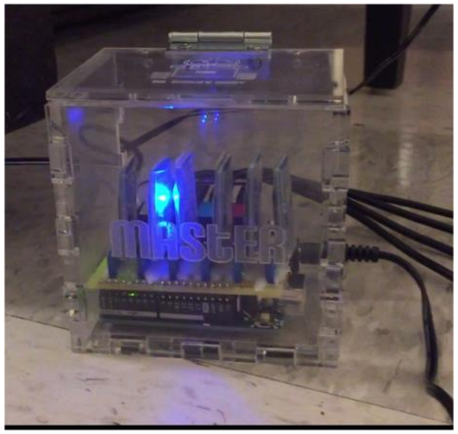
1. Scoreboard



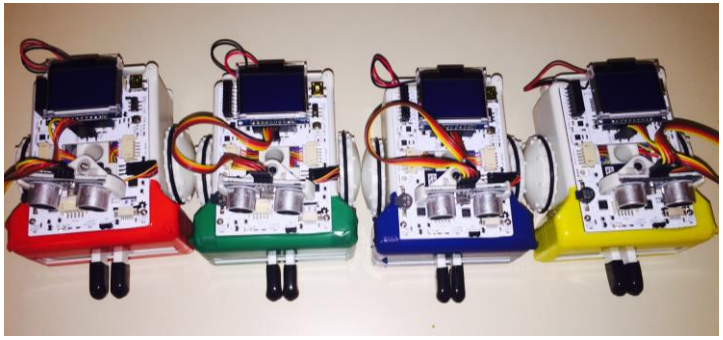
2. Ability switches x 4



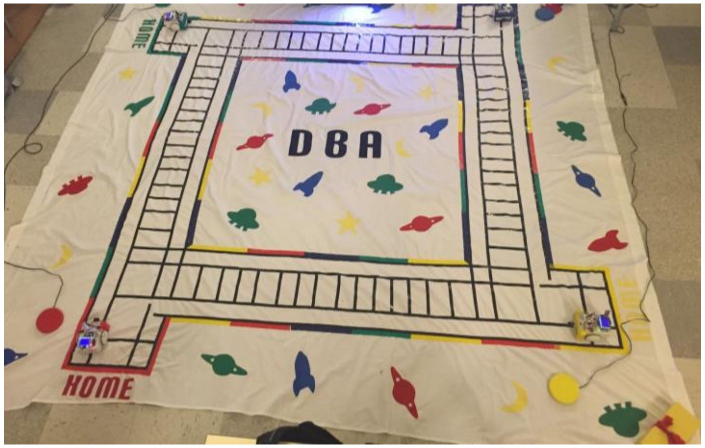
3. Master Robot



4. Sparkis x 4



5. Game Board



6. Bricks x 4

7. Instruction Manual (Appendix K)

REFERENCES

Boeree, D. C. (2003). *The Cerebrum*. Retrieved from General Psychology: http://webspace.ship.edu/cgboer/genpsycerebrum.html

Data & Statistics for Cerebral Palsy. (2013, 27 December). Retrieved from Centers for Disease Control and Prevention: http://www.cdc.gov/ncbddd/cp/data.html

Deutsch, J. (2008, August 8). Use of a Low-Cost, Commercially Available Gaming Console (Wii) for Rehabilitation of and Adolescent with Cerebral Palsy. *Physical Therapy, 88*(10), 1196-1207. Retrieved from http://ptjournal.apta.org/content/88/10/1196.full.pdf

Ginsburg, K. (2007, January 1). The Importance of Play in Promoting. *Pediatrics, 119*(1), 182-191. Retrieved from http://pediatrics.aappublications.org/content/119/1/182.full.pdf+html

Gordon, Ann Miles, and Kathryn Williams Browne. *Beginnings and Beyond: Foundations in Early Childhood Education.* 9th ed. N.p.: Skyline College, 2014. Print.

Hinchcliffe, A. (2007). *Children with Cerebral Palsy: A Manual for Therapists, Parents and Community Workers, Second Edition.* New Delhi: Sage Publications India Pvt Ltd. Retrieved from http://knowledge.sagepub.com/view/children-with-cerebral-palsy-2e/SAGE.xml

Ketelaar, M. e. (2001, September). Effects of Functional Therapy Program on motor Abilities of Children with Cerebral Palsy. *Physical Therapy, 81*(9), 1534-1545.

Li, Y. e. (2008). A Tangible Tabletop Game Supporting Therapy of Children with Cerebral Palsy. *In Proceedings Fun n' Games*, 182-193. Retrieved from http://www.idemployee.id.tue.nl/p.markopoulos/downloadablePapers/2009_Interact_Grounding_Romeroetal.pdf

Majnemer, A. e. (2008, September 17). Participation and enjoyment of leisure activities in school-aged children with cerebral palsy. *Developmental Medicine & Child Neurology, 50*(10), 751-758.

palsy. (2014). (Merriam-Webster, Incorporated) Retrieved from Merriam-Webster: http://www.merriam-webster.com/dictionary/palsy

Piaget, Jean. The origins of intelligence in children. Vol. 8. No. 5. New York: International Universities Press, 1952.

Play. (n.d.). Retrieved from Caring for Cerebral Palsy: http://www.caringforcerebralpalsy.com/play.html

Sinno, D., Charafeddine, L., & Mikati, M. (2013). *Enhancing Early Child Development.* Springer Science+Business Media, LLC. doi:10.1007/978-1-4614-4827-3_1

Special Perspectives: Cerebral Palsy. (2014). Retrieved from FatBrain Toys: https://www.fatbraintoys.com/special_needs/cerebral_palsy.cfm

TASH Ability Switches. (n.d.). Retrieved from EVAS: http://www.evas.com/abcpdf/showpdf.aspx?dsname=datasheet997

Trafton, A. (2009, May 9). *Robotic therapy holds promise for cerebral palsy*. Retrieved from MIT News: http://newsoffice.mit.edu/2009/robotherapy-0519

UNICEF. (2012). *A Summary of hte UN Convention on the Rights of the Child.*