THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF MECHANICAL AND NUCLEAR ENGINEERING

SEMI-AUTONOMOUS LINE FOLLOWING DOMINO DISPENSOR

JENNY LYNN SHIPLEY
SPRING 2013

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Mechanical Engineering
with honors in Mechanical Engineering

Reviewed and approved* by the following:

Dr. Sean Brennan
Associate Professor of Mechanical Engineering
Thesis Supervisor

Dr. H.J. Sommer III
Professor of Mechanical Engineering
Honors Adviser

Dr. Christopher Rahn
Professor of Mechanical Engineering
Faculty Reader

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

Ever since the 18th century, dominoes have been a prevalent pastime. Commonly, they were used in a similar manner as cards; games were played based on the value of the dots on the domino's face. In addition to these value-based games, domino toppling – standing dominoes on end in long lines so that they topple over sequentially when knocked over - has grown to become a popular way to use dominoes. Toppling has become so common that when one small event causes a long string of larger events, the phenomenon could be commonly called the domino effect. As prevalent as domino toppling is, there is a limitation put on the audience who can enjoy the hobby. Setting up dominoes in preparation of toppling takes a great deal of concentration and dexterity. If a device existed to automatically arrange the dominoes for a person to knock over later, domino toppling could become a more accessible amusement. Upon studying the current devices available, there are very few existing designs – and even less working prototypes – for an automated domino dispensing device. Thus, the following iterations of an Arduino based robotic domino dispensing robot came to be

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would first and foremost like to thank Dr. Brennan. Not only for helping me with this thesis, but also for being a great academic advisor the past four years. Additionally, I would like to thank Dr. Sommer for teaching me a great deal about the Arduino and mechatronics in general.

I would like to also thank Mike Robinson for his endless advice and help in making this project actually happen. Additionally, I must give many thanks to Matt Krott for his help in designing the first prototype.

Finally, I would like to thank my parents and siblings for their constant support.

Chapter 1

INTRODUCTION

1.1) Goal Statement

This design project aims to create a semi-automated domino dispensing robot based using an Arduino microprocessor. The robot should follow a line while dispensing dominoes along the line as it progresses. The final product should be mass marketable and producible, appealing to the domino's target age group functionally and aesthetically. Moreover, the robot shall be able to dispense a decent quantity of dominoes at a steady pace. The robot should be easily manipulated by a younger child, not involving any complex instructions or intricate fine motor skills.

During the process of designing the aforementioned robot, the standard cycle of product development procedures will be followed; customer analysis, researching of existing ideas, design planning, and prototype iteration. Additionally, the robot will be run on an Arduino microprocessor, utilizing its programing capabilities to electro-mechanically interface Lego motors, printed circuit boards, microchips, and other devices. The final product will be representative of a product that could successfully be sold on the commercial toy market.

1.2) Motivation

Domino toppling (the act of lining up and knocking over dominoes) is a tedious and difficult hobby. It takes patience and dexterity to be able to line up large numbers of dominoes without a slip of the hand or unsteady domino knocking over previously set-up dominoes. As it stands there is not a product that makes the task simple while still allowing custom path creation. Part of the draw of domino toppling is to create your own designs and paths; a toy that lays out dominoes only in a straight line would not appeal to as many potential customers as one that

would follow a user designed path.

Arudinos have become an increasingly popular microcontroller amongst the hobbyist community. They are incredibly versatile, reasonably durable, and the Arduino programing language is fairly straight-forward to learn. A person could easily see how such a microprocessor could be used to be the “brains” behind a domino set-up robot. It could be used quickly and effectively to interface the several motors, servos, and light sensors necessary to build the robot. Due to the Arduino’s abilities, availability, and accessibility; it will be chosen as the main microcontroller for the domino laying, line following robot.

1.3) Overview of End Design

After many months of research, design, and testing a robot prototype was created which met most of the aforementioned design criteria. The robot uses an array of five infrared light sensors to follow a dark line on a light background. As it follows the line, dominoes are dispensed along the line. The dominoes are stacked in a vertical hopper until they are pushed, one at a time, down a dispensing chute. The chute is arched so that the domino is righted as it falls from the hopper to the ground. A timing mechanism in the Arduino allows the robot to alternate between dispensing a domino and moving. The robot is not complex or hard to operate; one must only load dominoes into the hopper then flip the on/off switch. The final design will be described in further detail in Chapter Four. Overall, it functions successfully as a prototype of the desired domino dispensing robot.

Chapter 2

LITERATURE REVIEW

2.1) Learning Qualities

2.1.1) Playing as a Means of Learning

Via play, children interact with the world around them. In the process, a child's play allows them to explore, identify, negotiate, take risks, and create meaning. Play can be defined as something which is pleasurable, voluntary, process-orientated, self-motiving, active, and often symbolic (Shipley, 2008). Playing and learning are not synonymous, and there are types of play which stimulate more learning than others. “Good” play can be described as play which leads to positive learning outcomes; specifically in the areas of cognitive, emotional, social, and psycho-motor domains (Wood, 2004). Moreover, play can be education when developing the six areas of learning, as stated by Grimes (2006):

Table 2.1: Grimes’ six areas of learning

Six Areas of Learning
Personal, social, and emotional development
Communication, language, and literary development
Problem solving, reasoning, and numeracy
Knowledge and understanding of the world
Physical development
Creative development

Would using the domino laying robot be categorized as “good” play? The toy's creative aspects are obvious. A child can draw whatever shape they can imagine – or even draw a shape told to them by a parent/instructor. The child also gains motor skill in the loading of dominoes

into the robot, and gains the cognitive recognition of cause and effect by toppling the dominoes over. In addition, problem solving skills are developed when the child learns that when the robot stops dispensing dominoes, more dominoes must be loaded into the hopper. Given the positive effect on the aforementioned attributes, one can say that a child will have positive cognitive benefits from the domino robot.

2.1.2) Hands-on Learning

When communicating an educational idea or concept to a child, a popular and effective method of education is through hands-on learning. As stated in their book on teachers' perspectives on hands-on science teaching, Haury & Rillero (1994) strongly emphasize the value teachers see in hands-on education. One teacher talks about two theoretical ways to teach children about a puddle – the water, its properties, and the ecosystem within it:

> Option 1: Find a puddle and photograph it. Show the photograph to a seven-year-old child. Have her read about puddles. Later, ask her to talk about the puddle.
>
> Option 2: Find a puddle. Add one seven-year-old child. Mix thoroughly. Stomp, splash, and swish. Float leaves on it. Drop pebbles into it and count the ripples. Measure the depth, width, and length of it. Test the pH. Look at a drop under a microscope. Measure 250 mL of puddle water and boil it until the water is gone. Examine what is left in the container. Estimate how long it will take for 250 mL of puddle water to evaporate. Time it. Chart it. Now ask the child to talk about the puddle. If you were a seven-year-old child, what option would stimulate you to talk about the puddle?

Though his situation is theoretical, his point still remains valid. For a child to fully experience and learn about certain real-world concepts, hands-on experience is invaluable.

More than just a theory, hands-on learning has been shown to help both children and adults learn about a variety of concepts. Waliczek (1999) studied whether the outdoor activity of gardening can influence the environmental attitudes of children. Specifically, through an activity book of various gardening objectives the students would accomplish over the course of a school term. Waliczek showed that the hands-on activity of working directly with the environment significantly affected students' knowledge of and care for the environment. Another study (Parkinson, 2003) was performed on adult learners to see if hands-on activities could help teach them about woodland fires. As seen in the results in Figure 2.1, Parkinson found that the method of hands-on learning is an effective method for teaching; and, in her case, reaching adult audiences.

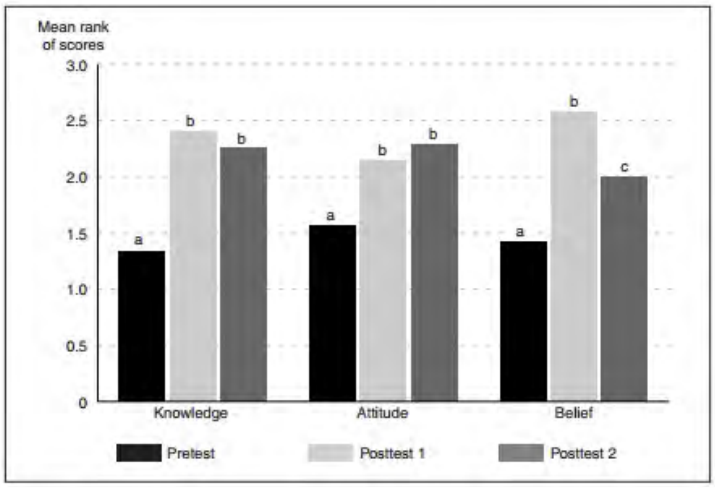


Figure 2.1 – Mean rank scores from three survey times (before, immediately after, and one month after workshop) from Parkinson's 2003 study on hands-on learning teaching adults

Why then is it important for children to tie-in hands on activities with their playtime routine? As stated by Therrell (2002), "Children's cognitive development, which includes creativity, discovery, language skills, verbal judgment and reasoning, symbolic thought, problem-solving skills, and the ability to focus and control behavior, are all heavily influenced by their play experiences". Therefore to further enhance this development, children should incorporate generous amounts of hands-on activities into their play. Examples of the aforementioned hands-on activities include dominoes, blocks, arts & crafts, and ride-on toys. All of the above have the capability to aid in the development of a child.

2.2) Audience Assessment

2.2.1) Dexterity

The act of setting up a lengthy line of dominoes can be challenging, even for an adult. It requires both the ability to grasp and position each block relative to each other and the ability to balance each domino vertically on its end. These skills of dexterity take time to develop, taking as long as several years to develop proficiently enough to set up dominoes. Through the use of a timed pegboard test (Figure 2.2.1), Gogola (2010) found that dexterity increased linearly with age among young children- ages 3 to 17. The pegboard test is a type of Functional Dexterity Test (FDT) which times how long it takes the subject to turn over all pegs in a predetermined order by manipulating the peg in their hand. The same FDT was used by Paula Lee-Valkov, doctor of plastic surgery at Bayor College of Medicine, in her study of hand dexterity norms of three, four, and five year olds (Lee, 2003). Lee-Valkov also found that as age increased, so did dexterity.

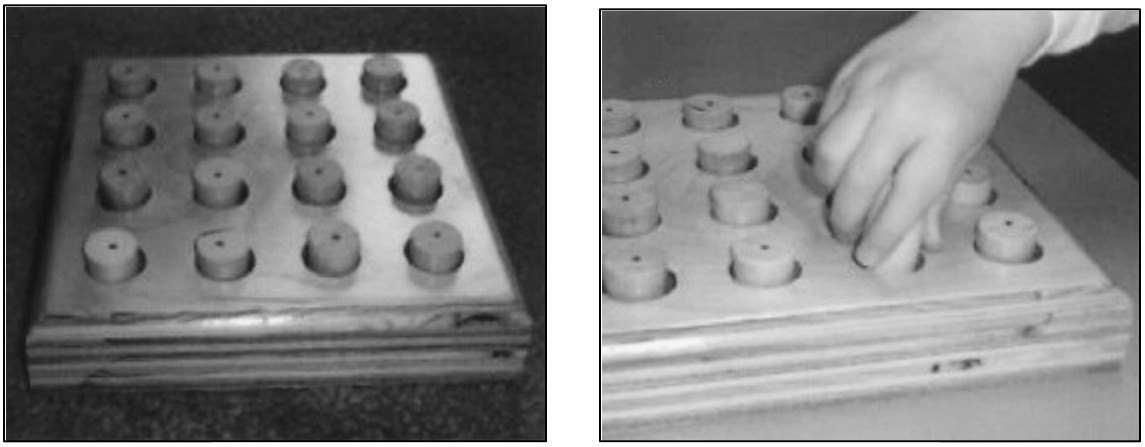


Figure 2.2.1 – Pegboard used in Lee's 2003 dexterity study

2.2.2) Target Domino Age Group

It is common practice for a toy manufacturer to list a recommended age group as the targeted user for its product. Additionally it is to the manufacturers' advantage to accurately determine the appropriate age category for their toys, and to label, promote, and market those

toys to that age group (Therrell, 2002). Age recommendations come from the Consumer Product Safety Commission (CPSC) and are based on their document *Age Determination Guidelines: Relating Children's Ages to Toy Characteristics and Play Behavior*. As stated within the guidelines themselves:

> This document...is based on a four-phase research endeavor...Phase I included a review of more than 200 articles...on the topics of play, toys, materials, and the developmental behaviors of children. Phase II was a research study into the toy purchasing decisions of adults, and Phase III was a research study involving the observations of children interacting with carefully selected toys. Phase IV involved the writing of a research document and new, replacement guidelines.

Based on these carefully formulated guidelines, companies have derived an appropriate age group for dominoes. In the case of many domino toppling toys, the recommended age is normally listed starting at either six or eight years of age. Domino Express and Domino Rally, two popular domino toppling games, list their recommended age groups as six to twelve years of age and six to eight years, respectively. It is therefore safe to say that the manufacturers of dominoes have determined that their target audience is in approximately the six to ten year-old age group. These children have the dexterity and patience to balance the dominoes, as well as the forethought necessary to plan out a path for the dominoes to follow.

2.2.3) Bridging the Gap

Ideally, the proposed domino robot would allow the domino industry to be able to expand its target audience. The robot could be used to set up the dominoes for the user, eliminating the high demand for dexterity, patience, and balance. The child would only have to load the dominoes into the robot – as simple as loading blocks back into the bins they store them in.

Additionally, the child would only have to be able to draw its path with marker on paper or lay down tape in the shape of the path it wants the robot to follow. All these tasks are achievable by children of ages three or higher (Therrell, 2002). As long as the robot does not contain open moving parts or small parts, there would be little to nothing hindering the lowering of the age recommendation for domino toppling.

2.3) Commercial Study

2.3.1) The Domino

When designing a domino laying machine, it would be wise to also research aspects of the domino itself. Thought to have originated in China during the 12th century, the domino has grown into a vastly popular game throughout the globe. It has become more regimented in size and shape, yet common commercial variations do occur. The most immediately noticeable difference is in the amount of dots, or "pips" on each half of the domino. Domino sets are named by the highest number of pips on each half of the domino. For example, a domino set containing blocks ranging from 0 to 6 pips on each half would be called a [6 – 6] or "double six" domino set (Figure 2.3.1a). Double six domino sets are the most common variation of the domino (Celko, 2005) and normally come in sets of 28 dominoes. However double nine, twelve, fifteen, and eighteen domino sets are commercially available and are used in various domino games.

Figure 2.3.1a – A line of double six dominoes

The amount of dominoes that come in a set is not arbitrary, but instead comes from the least amount of tiles necessary to have one of each numerical combination of pips. For instance, if you were to have a complete double 1 domino set it would contain three tiles: [0 – 0], [0 – 1],

and [1 – 1]. Numerically, the number of tiles needed for a complete set can be calculated by the formula:

$$N_{tiles} = \frac{n^2 + 3n + 2}{2}$$

Where *n* is the highest number of pips on the domino set. Once more, this proves why the most common domino set, the double six, contains 28 dominoes.

While manufactures make dominoes in countless sizes, several common sizes can be seen in the domino marketplace. As seen in Figure 2.3.1b, standard dominoes are usually 2” x 1” x 3/8”. Even larger yet are tournament dominoes, which are 2-1/8” x 1-1/8” x 7/16”. The possible variation in domino size might have to be a design consideration when designing the mechanism that will hold the robot's dominoes.

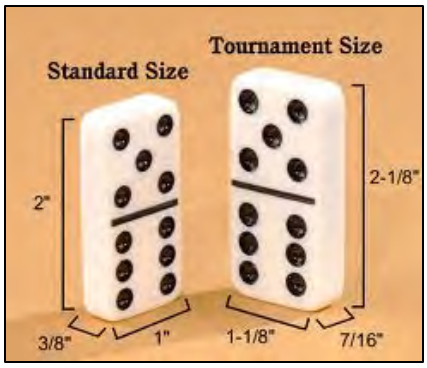


Figure 2.3.1b – Common domino sizes

2.3.2) Existing Set Up Techniques

There currently exist several techniques for setting up a series of dominoes. First, while manually setting up dominoes, one can use a template. A template consists of a slotted device which the user can slide the dominoes in for easy alignment and balancing. The tool can be made from a variety of materials: cardboard, metal, or even Legos (Figure 2.3.2a). The benefits of this tool include being able to manually set up a decent number of dominoes in a minimal amount of time as well as being easily fabricated. However, there exists the physical limitation of only being able to load as many dominoes as your hand can hold at one time. Also, the tool can only be used to make straight lines, and the dominoes can still be knocked over by the user during the

set up process. In other words, the template is still subject to user error and limited by the capabilities of the user. Additionally, these templates are often roughly made, used primarily as a quick fix for domino hobbyists.

Figure 2.3.2a – Homemade Lego domino template

In the world of automated domino set-up devices, there currently exists a Lego-built robot (Figure 2.3.2b) which is able to lay down dominoes in a straight path (Wandel). The robot uses a gear driven rotary arm to push dominoes out, one at a time, as the robot moves forward. The dominoes rest on the ground while in the robot's “hopper”. Then, they are pushed horizontally through a slot just larger than the domino to ensure they maintain alignment. The

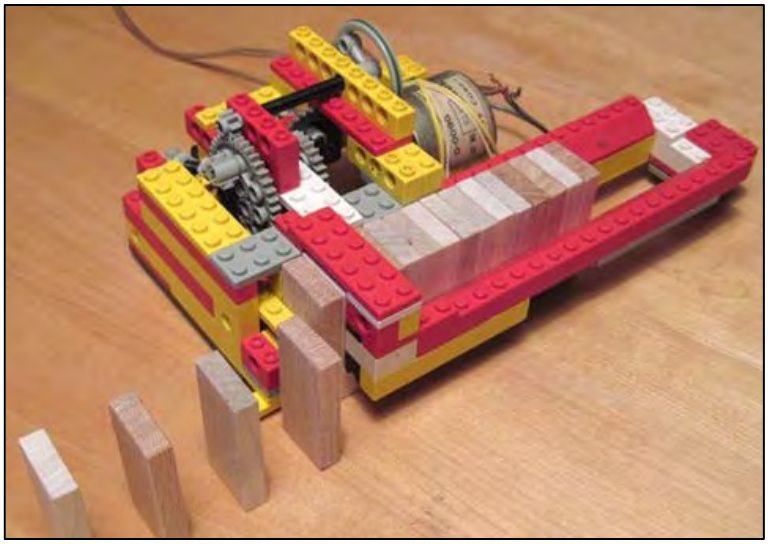


Figure 2.3.2b – Wandel’s robot using horizontal domino dispensing

steady rotation and constant speed of the robot ensure that the dominoes are evenly spaced. Limitations of this robot's design include its ability to only move in a straight line as well as its inability to perform on all material surfaces. The dominoes could get caught or be unable to slide on carpet, or possibly topple over in a nearly empty hopper. In addition, dominoes must be repeatedly added as the robot can only hold a very limited supply.

A patent search yields a different design for an automated domino set-up machine (Fassman, 1998). This machine utilizes a vertical hopper/domino storage area. As the machine moves it rights a single domino then uses a rotary arm to press the domino forward and out. The device is motor-driven and moves in a straight path. Once again, there is little storage space for dominoes, and the device is unable to lay dominoes in a non-linear path. However, this robot holds the dominoes off the ground, allowing it to fully control the motion of the domino. In fully controlling the domino's motion the robot limits the amount of possible occurrences which would disrupt the laying of dominoes. This design was successfully marketed by the Domino Rally chain disguised as a car, and the Domino Express brand as a “power dealer”. Both operate on the same principle design, the one described in Fassman's patent, which can be seen in Figure 2.3.2c.

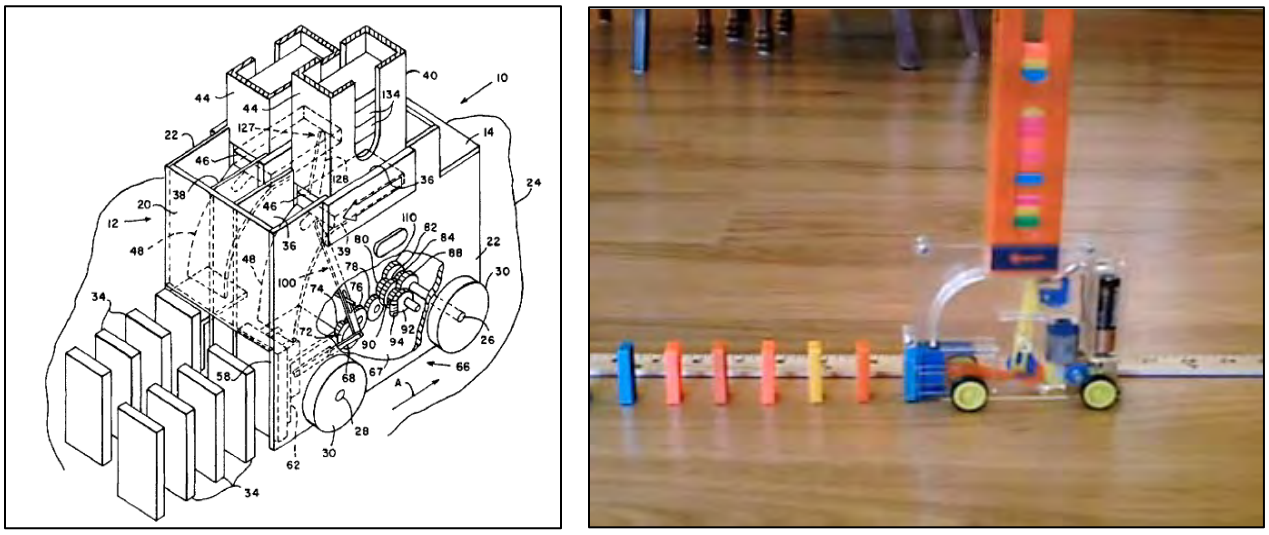


Figure 2.3.2c– Fassman’s patent and the Domino Express brand toy utilizing a similar method of domino dispensing

2.3.3) Line Following

The concept of a line follower is simple: a robot which can, without human input, follow a given path. A more detailed description would be a robot which, using sensors as a feedback mechanism, forms a self-correcting closed loop to follow a path. Most line followers follow visible lines on a contrasting background, yet it is also possible to build a line follower to follow an invisible path, like a magnetic field or infrared path.

Some systems, such as the Lego NXT system (Figure 2.3.3a), offer ways to create a line follower using a single light sensor. Other systems utilize an array of multiple sensors, consequently becoming faster and more accurate. While the preferable number of sensors in each array varies from robot to robot, creator to creator; often similar, simple algorithms are applied to read from and react to the sensors. Eddy Wright, of Wright Hobbies, uses an array of five sensors (Wright, 2003) while Priyank Patil of the K. J. Somaiya College of Engineering used an array of eight sensors (Patil, 2006). In both cases the sensors read 0 or 1; off or on the line, respectively. The values for each sensor were then stored in an array. The robot would then respond to the array, turning left or right accordingly. The difference between these two line following robots is how they chose to read and react to the arrays. Wright's robot would read the location of the ones within the array to determine how sharply to turn. Ones further from the center required sharper turns. Patil's robot, on the other hand would compare the amount of ones on the left half of the vector to the right half to decide how to turn. Both cases allow for the robot to read its location relative to the line it should be following and correct its movement correspondingly.

Figure 2.3.3a – Lego NXT line follower

The question that arises from both Wright's and Patil's robots is: What is the optimum number of sensors to adequately follow the line without over-saturating the system with information? The answer lies in the consideration of a single factor; speed. Wright acknowledges that three sensors are enough to follow a line efficiently, yet adds the extra two in order to allow his robot to compensate for an overshoot condition, shown in Figure 2.3.3b. At line following competitions, a robot can be going too fast to be able to respond to sharp turns and can overshoot the line. For the case of laying out dominoes, the robot will not be going fast enough for this condition to occur.

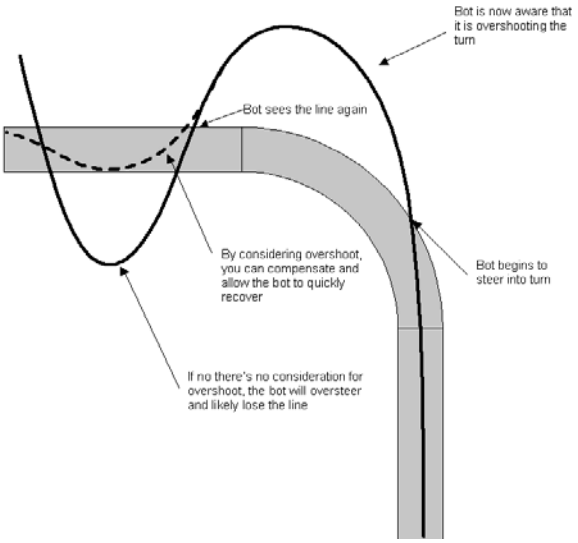


Figure 2.3.3b – More sensors allow for overshoot recognition

When deciding what hardware a line follower uses, one concern is that “many line-following robots have difficulty tracking masking tape when the robot employs infrared LEDs and sensors...masking tape is to some extent transparent to infrared, thus the background color bleeds through and the robot can’t see the line” (Cook, 2002). To circumvent the bleed through problem, several robots use several cadmium-sulfide photoresistors. However, “Cadmium-sulfide photoresistors react fairly slowly...phototransistors react orders-of-magnitude faster” (Cook, 2002). Thus the designer of a line following robot must decide whether they will sacrifice versatility for speed when choosing which light sensors to use on their line follower.

2.3.4) Assessment of Cost

Given that the domino robot will be a toy for young children, how much should it cost? While no product exists on the market which exactly matches the domino robot's functionality, the price of

similar products can be compared. Complex, multifunctional robots sell for upwards of $60, with interactive humanoid robots selling for upwards of $100. Given the domino robot's limited functionality and interactive capabilities, these estimates should be seen as a ceiling for its retail price. A more comparable robot would be a single or very basic multifunctional RC children's toy. These toys cost in the area of $20 - $40; a more appropriate estimate of the domino robot. Given the robot's lack of designated parts and reasonably inexpensive estimated cost, it can be anticipated that the final product could be sold on the mass market at the lower end of the aforementioned range.

Chapter 3

FIRST DESIGN ITERATION

3.1) Executive Summary

The following chapter will focus on the first iteration of the domino robot's design. In this design, an Arduino Uno reads an array of five infrared sensors and outputs appropriate voltages to the left and right motors in order to keep the vehicle following the line, while a separate electromechanical system is responsible for positioning the dominoes on top of the line. The dominoes are pushed horizontally out of a "sidecar" which the robot pulls beside it. Based on the observations of this prototype, several improvements were found that would improve the domino-dispensing system in future design iterations.

3.2) Bill of Materials

Part	**Quantity**	**Cost**
Lego Mindstorms Education Resource Kit	1	$ 99.95
SN754410 Quadruple Half-H bridge driver	1	$ 2.35
Parallax QTI (Infrared) Sensor	5	$ 9.99
Lego Power Functions XL Motor	2	$ 9.99
Hi-tec HS-325 HB servo	1	$ 12.99
Arduino Uno board	1	$ 23.00
Express PCB printed circuit board	1	$ 51.00
4 AA batteries	1	$ 6.49
Set of 28 dominoes	1	$ 3.99
	Total Cost	$267.70

Additional wires and a small proto board were also used. In addition, black electrical tape was used when outlining the path to be followed.

3.3) System Design

In essence, the domino bot is a self-correcting line follower which dispenses dominoes along the path which it is following. It does so by pushing the dominoes horizontally from a "sidecar" where the dominoes are stored. Within the sidecar, the dominoes are held vertical using an elastic band. The arm which pushes the domino outward is controlled by a RC Servo, which in turn is operated by the Arduino. The Arduino also reads the input from five infrared light sensors to determine its position relative to the line, and adjusts power to the motor driver chip accordingly. The motor driver then draws upon its own power supply to provide adequate power to each wheel. The wheels are turned using a differential steering algorithm, described further in section 3.5.1. The robot alternates between moving and dispensing a domino, allowing for the robot to be stationary as the domino is released. The design of storing the dominoes in a side car and dispensing them horizontally was inspired by a concept found on Youtube.[1] The design was improved upon and turned into the first design iteration prototype, which can be seen in Figure 3.3 below.

[1] This video can be viewed at https://www.youtube.com/watch?v=QgGZ2YGDIGM

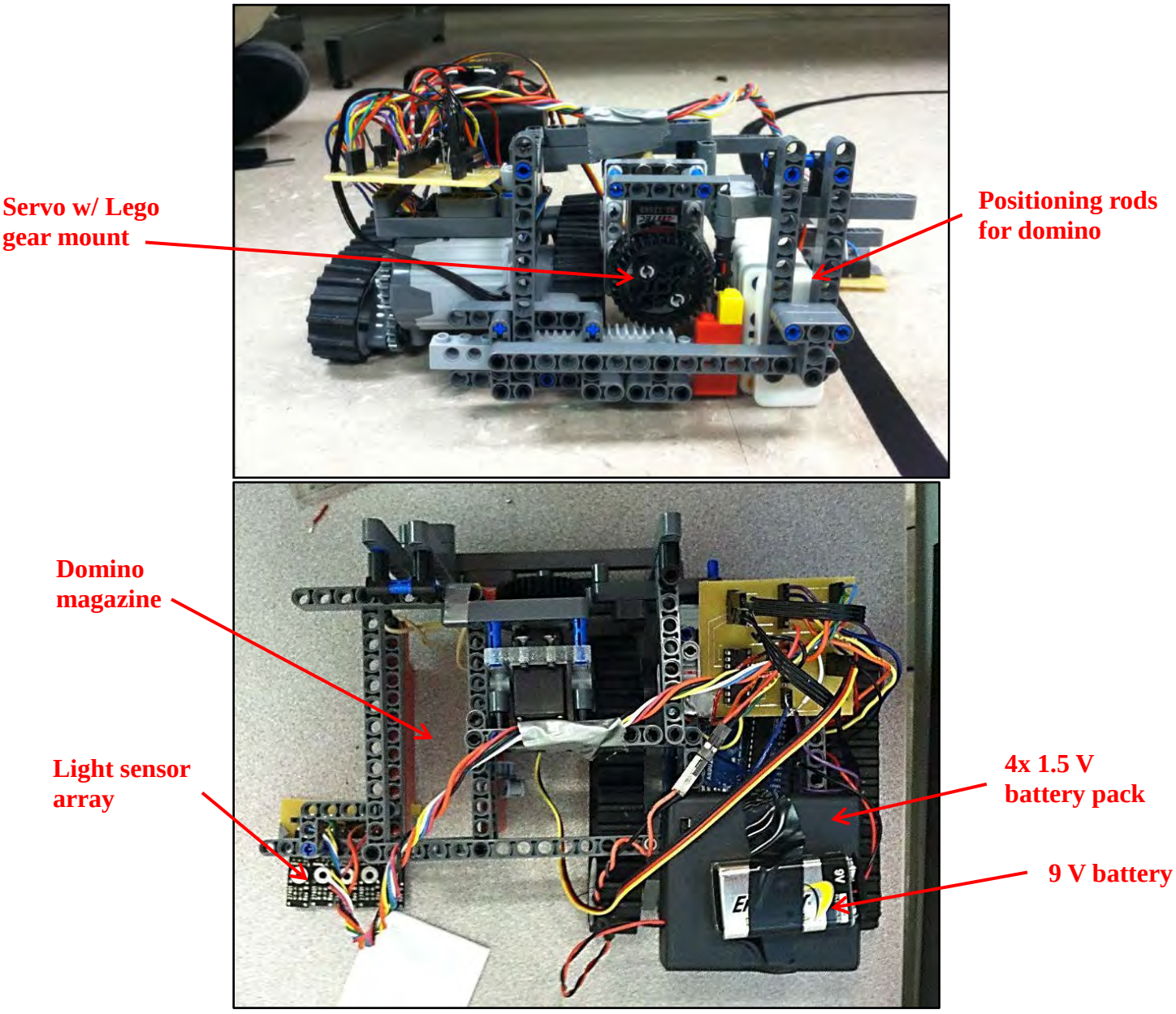


Figure 3.3: Front and top views of the vehicle.

During testing, the dominoes that were placed in the magazine did not turn with the car as anticipated. Small fixes such as modifying the dominoes' point of contact with the positioning rods and cutting perfectly flat dominoes were attempted, but none solved the issue. Eventually the issue was addressed by using rubber bands to pull the dominoes forward one by one so that they could be pushed out.

3.4) Description of Hardware

Motors

The robot is driven by two Lego Power Functions XL-Motors, with each motor controlling the treads on one side of the robot. The motors were compatible with the Lego Mindstorms gears and body parts. The ease at which the motors assembled with the rest of the robot's parts was the main reason they were selected for the design. The XL-Motor was chosen over the Power Functions M-Motor because it was able to output a higher amount of torque.

Motor drivers

The SN754410 is a 16-pin chip that was used to provide enough current to drive both motors. It was necessary due to the Arduino only being able to source about 200 mA of current at any given time. The supply voltage for the motors is provided by four 1.5 V AA batteries and the supply voltage for the logic is provided by the Arduino +5 V output pin. This controller uses TTL logic to determine the direction of rotation for the motor, and it can also provide an analog voltage between 0 and $+V_{IN}$ that drives the motor. Two pins, one for each motor, take a pulse-width-modulated signal from the Arduino to determine the voltage that is put out to that motor.

Arduino

For this project, an Arduino Uno was used as the main microprocessor. The board was able to put out voltages ranging from 0 V to +5 V. To make the vehicle electrically self-sustaining, a 9V battery was used to power the Arduino.

Light Sensors

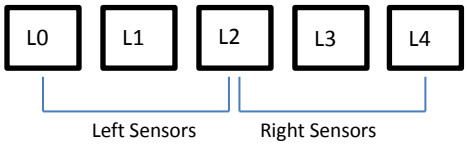
Each Parallax QTI sensor requires an input of +5 V and generates an analog voltage from 0 V to +5 V, where +5 V is a perfectly absorbing surface (black) and 0 V is a perfectly reflective surface (white). The output from the sensors were read by the Arduino and used to determine the relative location of the line.

RC Servo

The servo used in the robot was a standard RC servo that was able to be programed using the Arduino servo library. The supply voltage of +5 V came from the Arduino.

3.5) Description of Software

3.5.1) Discussion of Differential Steering Algorithm



Steering of the robot was accomplished by feeding the analog input voltages from the array of five light sensors into a steering algorithm. Once the light sensor values had been read, a system of weights was then applied to the five readings in order to come up with a ratio of light on both the left and right side. The algorithm was designed such that more weight was applied to sensors 0 and 4, since stronger steering would be required if the vehicle managed to get far off the line. Hence, light sensors 0 and 4 have a weight of 2, sensors 1 and 3 have a weight of 1.6, and sensor 2 has a weight of 1.33. The weighted input vales are then summed to determine how much light is being read on either side of the line sensor:

$$Left\ Light = \sum_{i=0}^{2} \frac{L_i / L_{total}}{W_i}$$

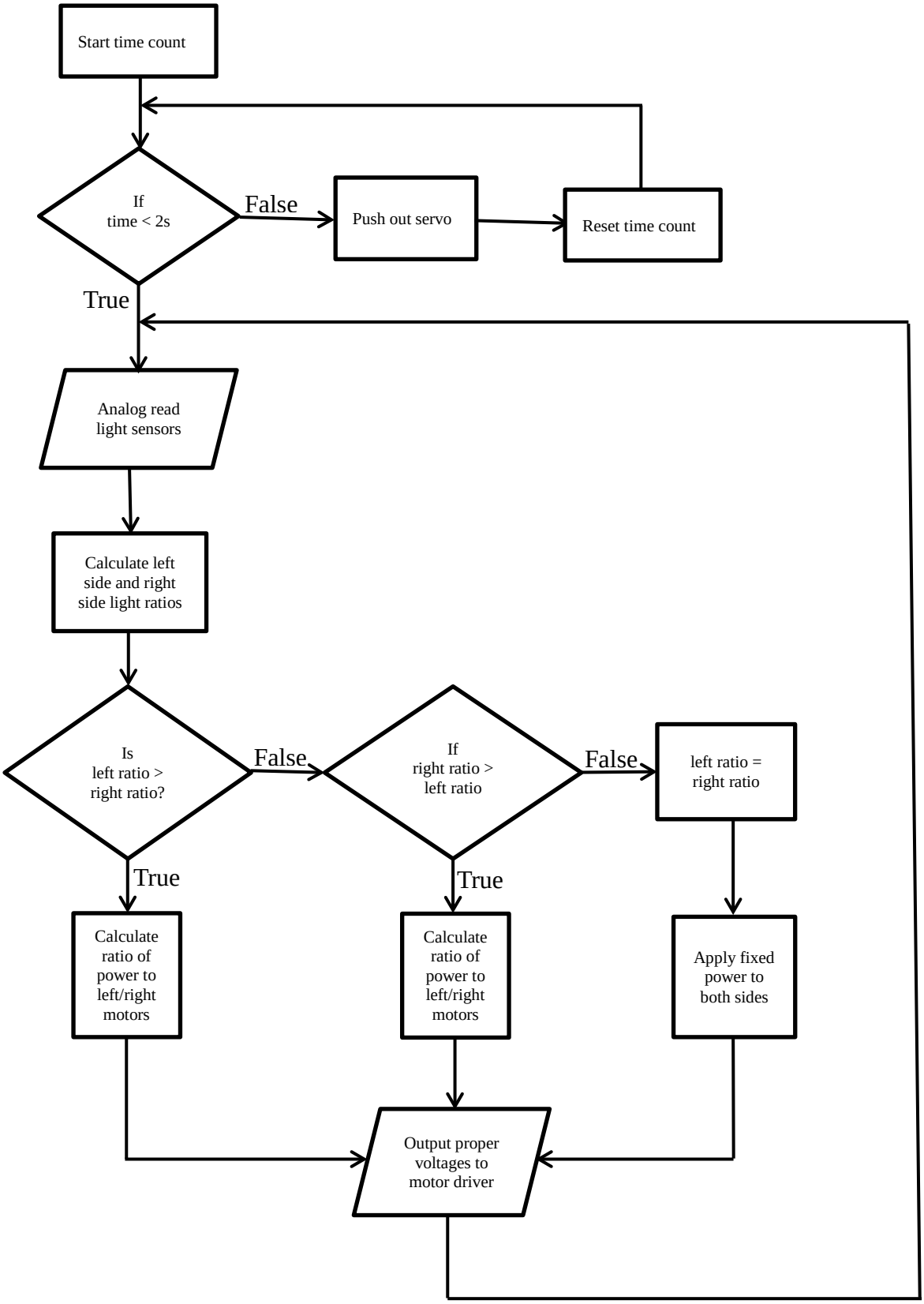
$$Right\ Light = \sum_{i=2}^{4} \frac{L_i / L_{total}}{W_i}$$

Next, the robot then decided what action it needed to take in response to these ratios. If the left ratio was larger than the right ratio, the robot would need to turn left. On the other hand, if the right ratio was larger than the left ratio, the robot would need to turn right. If the two ratios were equal, the line is centered and the robot should continue going straight.

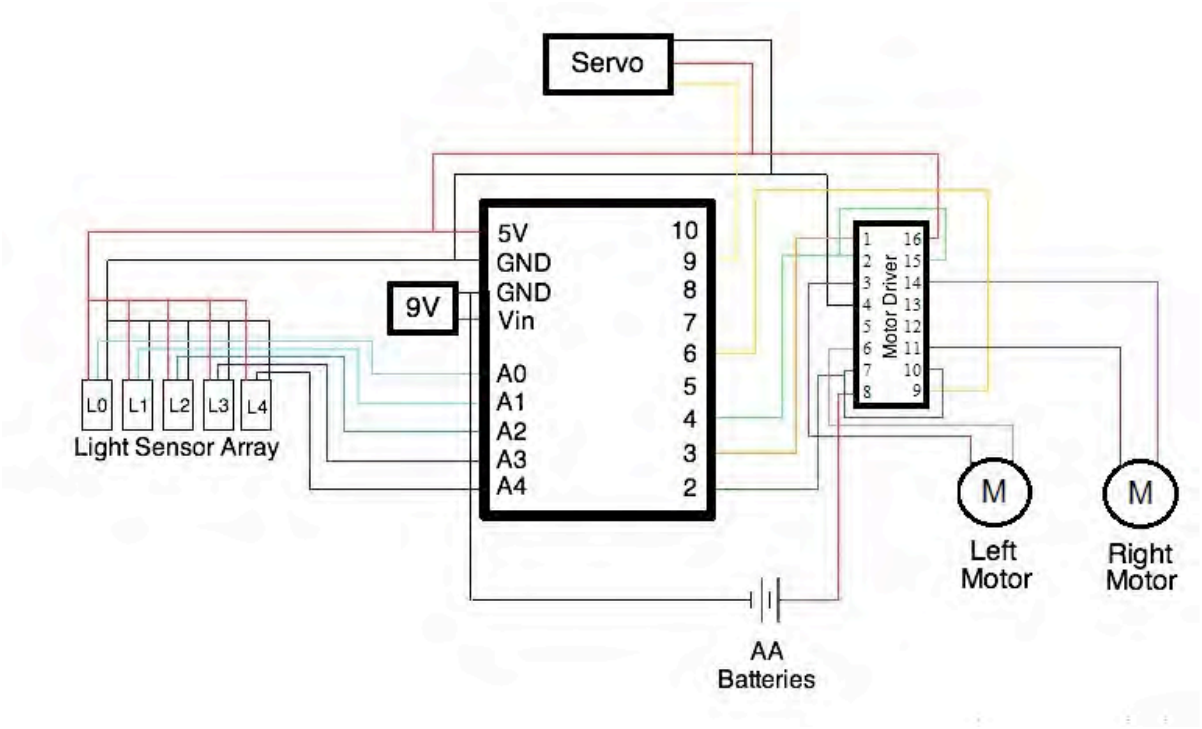
The power applied to each motor was then calculated on a case by case basis depending on which turning action was required. If the robot had to turn, the wheel on the inside of the turn would need to be slowed down and the outside wheel would need to speed up. The power to both the left and right wheels was calculated as a value between 0.5 and 1, where 1 was full power to the wheel. Any value below 0.5 would not produce enough current from the AA batteries to run the motors. To utilize the full possible output range, the sharpest turn should be a 0.5/1.0 ratio. Additionally, going straight would be a 0.75/0.75 power ratio to each motor. Therefore, the power to the inside wheel was the calculated left/right ratio and the power to the outside wheel was 1.5 minus this number.

As an example of these calculations, look at the case when the line was picked up on the L4 (rightmost sensor). The ratio of light at that sensor (L_i/L_{total}) would be 1, divided by the weight of 2 would give us a right ratio of 0.5 and left ratio of 0. The robot would recognize it needed to turn right and write the power of the right wheel (the inner wheel) as 0.5 and the right wheel (the outer wheel) as 1. Another example would be if the line was halfway between sensors L1 and L2. The ratio of light at each sensor (Li/Lt) would be 0.5, and summing the each ratio divided by each weight would give us a left ratio of 0.688 and a right ratio of 0. The left power (inner wheel) would be 0.688 and the right power would be 1.5 minus that number, or 0.812. If the line is perfectly centered, the output to each motor is 0.75.

3.5.2) Arduino Code Flow Chart



3.6) Circuit Diagram



3.7) Results & Conclusions

The first design iteration met many of the initial design goals, especially from a mechatronics standpoint. The line-following algorithm was particularly challenging since the vehicle's weight was imbalanced and required much more of a differential to successfully turn left. Despite the challenge, the line-following capability turned out to be the most successful part of the prototype after it was fine-tuned and the appropriate power ratios were found.

A large fault of the first design iteration was that the domino-dispensing mechanism itself did not function properly. The assumption that the dominoes would stand unaided set back the design greatly. After introducing rubber bands to hold the dominoes in place, the vehicle was able to lay out 4-5 dominoes in succession. Given this small success, it seems feasible that a passive component could push the dominoes into position. A well-calibrated compression spring or constant force spring would have been a much more robust component for the system and could have provided the steady force needed to hold the dominoes in place.

Additionally, a gravity-fed mechanism may be feasible and preferable to a spring fed dispensing system. A gravity-fed domino dispenser could eliminate the need for a side car and allow the light sensors to be placed directly on the vehicle. In eliminating the side car, the vehicle would have a better steering differential, and would be able to handle sharper turns. The second design iteration should be based around a gravity-fed, vertical dispensing mechanism in order to fix several of the issues the first design iteration faced.

Chapter 4

SECOND DESIGN ITERATION

4.1) Executive Summary

The following chapter will focus on the second iteration of the domino robot's design. In this design, as in the first, an Arduino Uno reads an array of five infrared sensors and outputs appropriate voltages to the left and right motors in order to keep the vehicle following the line, while a separate electromechanical system is responsible for positioning the dominoes on top of the line. However, in this iteration, the dominoes are stacked in a vertical hopper atop the robot's main body. Once pushed from the stack, dominoes fall down an arched dispenser to be turned upright. As with the first design iteration, a single domino is dispensed at a time, with the machine stopping to release each domino before continuing on its path.

4.2) Bill of Materials

Part	**Quantity**	**Cost**
Lego Mindstorms Education Resource Kit	1	$ 99.95
SN754410 Quadruple Half-H bridge driver	1	$ 2.35
Parallax QTI (Infrared) Sensor	5	$ 9.99
Lego Power Functions XL Motor	2	$ 9.99
Hi-tec HS-325 HB servo	1	$ 12.99
Arduino Uno board	1	$ 23.00
Express PCB Printed Circuit Board	1	$ 81.46
6 AA Batteries	1	$ 8.99
Battery Holder	1	$ 1.81
On/Off Rocker Switch	1	$ 1.15
Set of 24 dominoes	1	$ 3.99
	Total Cost	$305.62

Additional wires and a small proto board were also used. In addition, black electircal tape was used when outlining the path to be followed.

4.3) System Design and Improvements

Once more, the domino robot is a self-correcting line follower which dispenses dominoes along the path which it is following. However in its second design iteration, the robot dispenses the dominoes by pushing them from a vertical hopper. The robot once more alternates between moving and dispensing a domino, again allowing for the robot to be stationary as the domino is released. When it comes time for the robot to dispense a domino, an arm controlled by a RC servo pushes a domino lying horizontally from the hopper. The domino then slides down an arched dispensing chute until it is standing vertically. Once the domino is standing vertically on its own, the Arduino begins to move again. The domino is left behind in its place along the designated path. The RC servo and timer are all operated within the Arduino. The Arduino also reads the input from five infrared light sensors to determine its position relative to the path, and adjusts power to the motor driver chip accordingly.

Unlike the previous design, in this design iteration the motor driver, motors, Arduino, and RC servo all draw upon a common power source. In order to be able to handle the increase in power demand, the second prototype uses six AA batteries instead of the four AA batteries and one 9V battery. The voltage from the six AA batteries is fed directly to both V_{in} on the Arduino board and V_{in} for the motors on the motor driver. Both the logic for the motor driver and the RC servo require +5V, provided by the +5V output pin on the Arduino. By centralizing the input voltage on the robot, the second design iteration was able to have an on/off switch – a common and helpful feature on electronic devices.

An additional difference from the previous design is the transition from treads to wheels for the robot's propulsion. As the robot became heavier, the gears connecting the treads to the motors began to sporadically slip. Additionally, the treads themselves would gather dust and

dirt and would slip on the floor. Slipping affected how the robot would steer, making the steering difficult to predict. By placing wheels directly on the motors' axels the problem of slippage was fixed. Luckily, the steering algorithm itself only had to be edited very slightly from what it was for the first design iteration. Changing from four AA batteries to six AA batteries powering the motors meant that the motors could be supplied with a power ratio ranging from 0.3 to 1, where 1 is full power to the motor. Previously, no ratio bellow 0.5 was possible due to the amount of current being provided by the four AA batteries. This meant that the robot's strongest turn could now be a 0.3 to 1 ratio, whereas before it was 0.5 to 1. By having a stronger possible turn ratio, as well as no heavy sidecar, the second robot prototype was able to make sharper turns.

All physical changes to the robots designed are labeled below in Figure 4.3 (a) and (b):

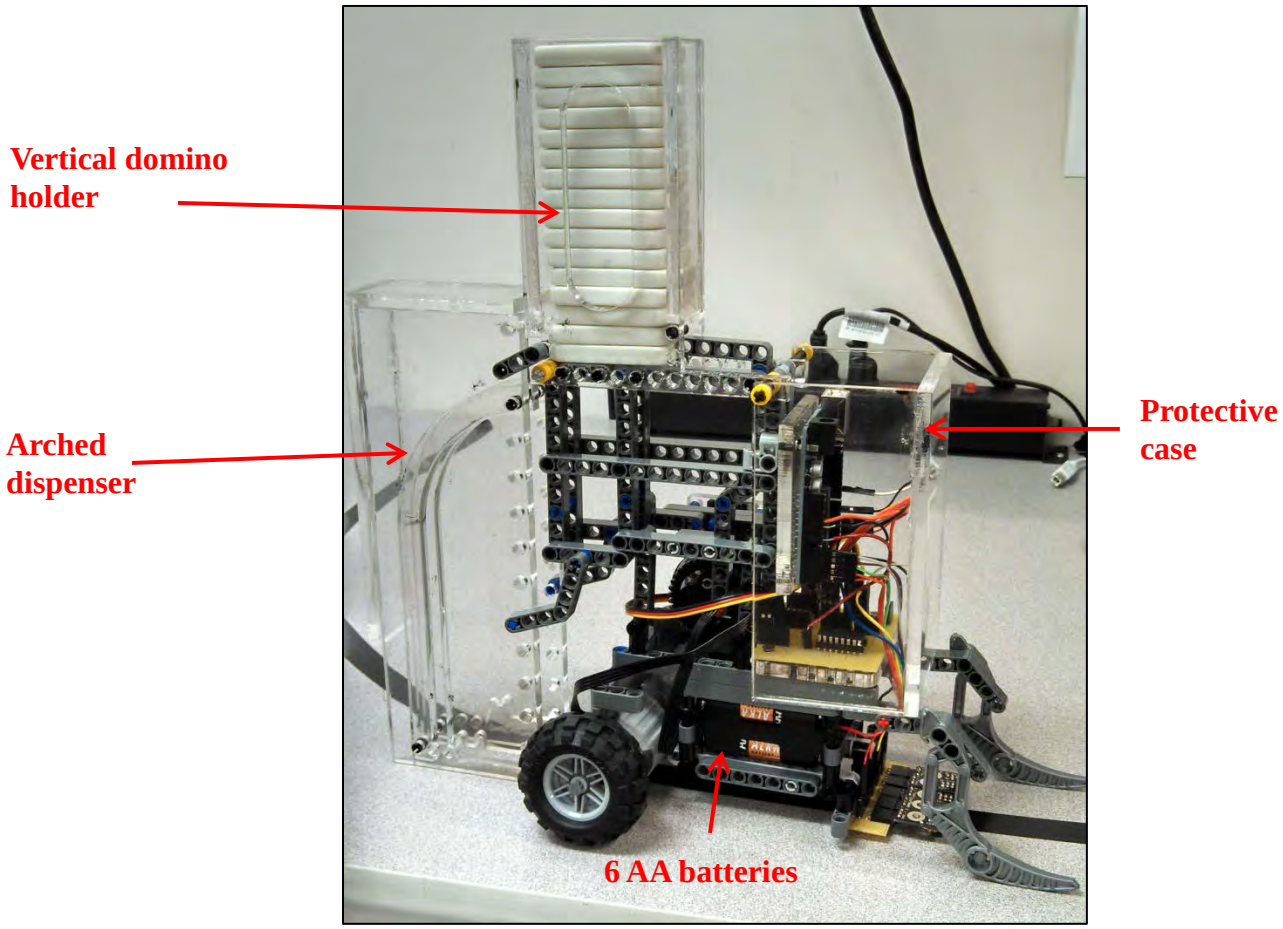


Figure 4.3a – Overview of system design

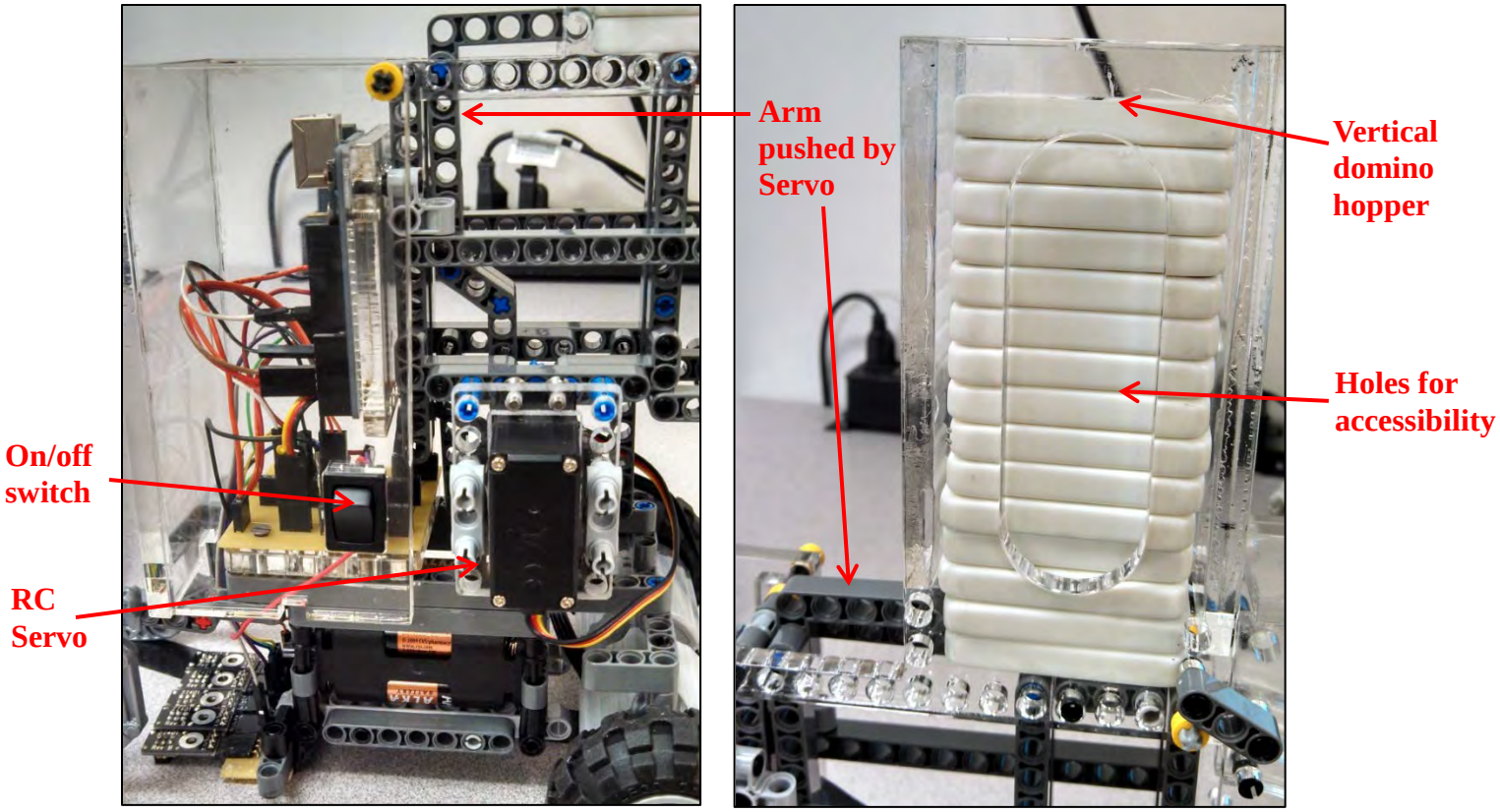
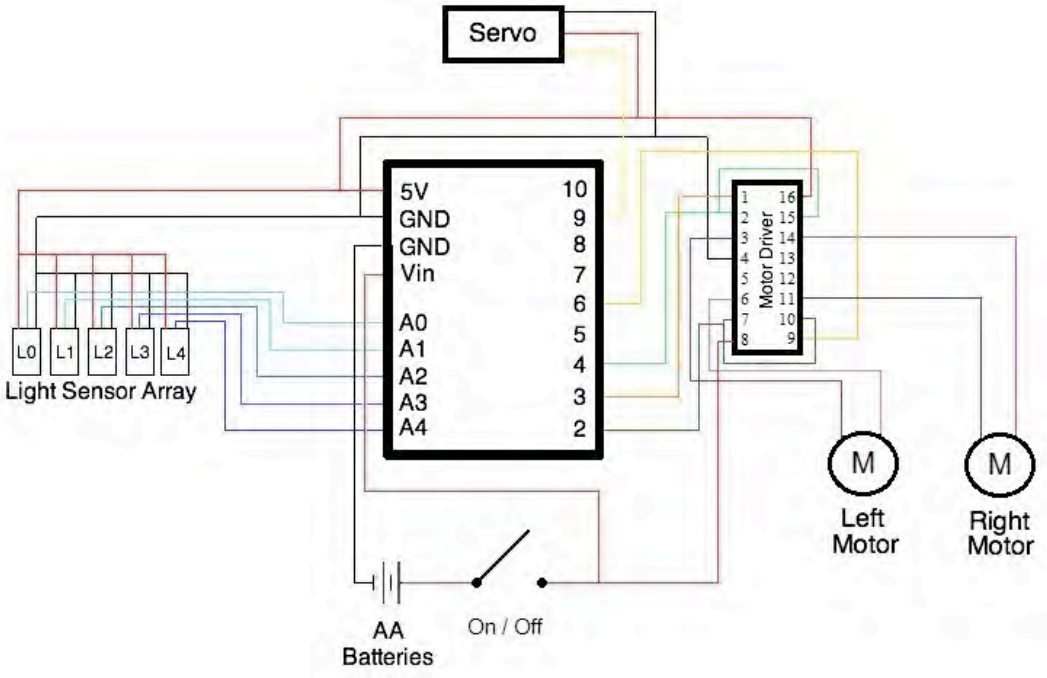


Figure 4.3b – Close ups of side and top

4.4) Circuit Diagram



4.5) Conclusions and Final Recommendations

With a few additional changes, the domino robot designed in this project could be a successful, marketable product. Areas of possible improvement include the consistency of the domino remaining upright upon dispensing, the number of dominoes being held, and domino stability and spacing on turning. All of these issues are non-critical, yet each would improve the performance of the robot. Moreover, the design issue of domino interchangeability has yet to be addressed, and the robot's casing has been intentionally left incomplete. If one were to fix all these small issues, the final product produced would be nearly flawless.

The main issue facing this prototype is that the dominoes dispensed did not always remain upright. About twenty percent of the time, the domino fell over upon hitting the ground. This behavior was wildly inconsistent, and several tests were done where no dominoes fell over upon hitting the ground. An inexpensive domino set was used in testing, thus the dominoes had rounded corners and edges. It could be this factor which caused the dominoes to sporadically fall over. If dominoes with square edges were used, the problem might fix itself. If not, a secondary arm could be used to push out the domino, stabilizing it in the process. This is the design utilized by the Domino Express brand domino dealer, seen below in Figure 4.4.

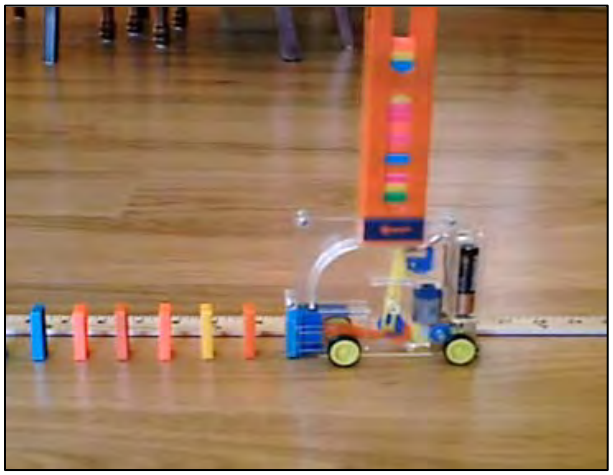


Figure 4.4 – The Domino Express brand domino dealer, utilizing a secondary arm

While the current robot's design allowed space for the robot to hold sixteen dominoes, it was unable to functionally hold more than seven at once. If more than seven dominoes were in the holder, the RC servo was unable to provide enough torque to push the bottom-most domino out and down the dispenser due to the weight of the dominoes on top of it. The dominoes used in testing could be estimated to be “average” dominoes, thus it is safe to say the problem would persist even using various different domino brands. However, if a servo with higher torque output was used the problem would be fixed.

When turning, the robot would at times hit the domino it just dispensed with the edge of the dispenser itself. The struck domino would then fall, and in true domino nature knock down its neighboring dominoes. While this issue was frustrating it could be fixed, but with a cost. If the robot were to go straight for a short bit of time after dispensing each domino, it would be sure not to hit it. However, this would greatly reduce the turning radius of the robot.

Domino spacing is a factor to be considered when domino stacking. Dominoes along a straight line can have an even, consistent spacing. However, around a turn dominoes must be placed closer and can even be angled into the turn. This change is made to ensure that the domino falls “into” the turn and still hits its neighbor. The robot prototype built lays all dominoes at a consistent spacing, regardless of whether it is turning or not. In testing, the robot did not ever encounter a situation where the even spacing around a turn caused the domino line to break. Thus it is not a high priority change or alteration, but it would add robustness to the robot's design.

Regarding physical design changes, the robot could be made to hold several different types of dominoes. If the robot was able to hold many types of dominoes, it would ensure that the product would be able to work with any domino that a child has lying in their toy box. This would contribute to the product’s value – moms and dads wouldn't have to buy additional

dominoes to use it. Another physical improvement could be to entirely cover the robot's moving parts. The domino robot started to get unfeasibly heavy as the acrylic casing was built and added to it. For this reason, the robot was left open on its midsection, with only the electrical components covered. In theory, a small child could still get their finger caught on the servo gear or the arm it is pushing. For safety reasons it would be necessary for the casing to be completed; for performance reasons, the casing should be fabricated with a lighter weight material.

All in all, the second design iteration of the domino dispensing robot came very close to meeting its design goals. It was able to successfully use a light sensor array to accurately follow a user-created path. It was designed to be able to hold many dominoes at once; however it did not functionally meet that design goal. Additionally, the robot could successfully be a toy usable by a larger audience than just the dominoes themselves – it requires no intricate maneuvers or dexterous tasks. While the robot still visually looks as though it is made of Lego parts, it incorporates many more acrylic parts than the first prototype; more closely resembling a product which would be sold on the mass market. Finally, it withstood a significant amount of testing without any large functional failures. Hence, it is safe to say that the toy would survive a life out in the “real world”, with only minor breaks or failures during its career as a child’s pastime. Despite several areas of possible improvement the second design iteration of the domino robot completed many of its design goals, making it a successful working prototype of the ideal domino dispensing robot.

REFERENCES

Celko, J. (2005, December 27). *The mathematics of dominoes*. Retrieved from http://www.pagat.com/tile/wdom/math.html

Cook, D. (2002). *Sandwich, an easy to build robot that follows lines*. Retrieved from http://www.robotroom.com/Sandwich.html

Fassman, A. (1998). *U.S. Patent No. 5782377*. Washington, DC: U.S. Patent and Trademark Office.

Gogola, G. R., Lacy, B., Morse, A. M., Aaron, D., & Velleman, P. (2010). Hand dexterity values for 3- to 17-year-old typically developing children. *Journal of Hand Therapy, 23*(4) doi: http://dx.doi.org/10.1016/j.jht.2010.09.051

Grimes, R. (2006). *Six early learning areas*. Retrieved from http://www.silkysteps.com/the-six-areas-of-learning-ELGs-early-years-statements-to-guide-progressive-provision.html

Haury, D., & Rillero, P. (1994). *Perspectives of hands-on science teaching*. Columbus, Ohio: ERIC Clearinghouse for Science, Mathematics and Environmental Education. Retrieved from http://www.eric.ed.gov/PDFS/ED372926.pdf

Lee, P. M., Aaron, D. H., Eladoumikdachi, F., Thornby, J., & Netscher, D. T. (2003). Measuring normal hand dexterity values in normal 3-, 4-, and 5-year-old children and their relationship with grip and pinch strength. *Journal of Hand Therapy, 16*(1), 22-28. doi: http://dx.doi.org/10.1016/S0894-1130(03)80020-0

Parkinson, TM (2003). "Hands-on learning: Its effectiveness in teaching the public about wildland fire". *Journal of forestry (0022-1201)*, 101 (7), p. 21.

Patil, P. (2006). *Line following robot*. Department of Information Technology, K.J. Somaiya College of Engineering, Mumbai, India. Retrieved from `= http://www.kmitl.ac.th/~kswichit/ROBOT/Follower.pdf

Shipley, D. (2008). Empowering children. Play based curriculum for lifelong learning. (Fourth edn). USA: Nelson Education.

Therrell, J. A. U.S. Consumer Product Safety Commission, (2002). *Age determination guidelines: Relating children's ages to toy characteristics and play behavior*. Retrieved from website: http://www.cpsc.gov/PageFiles/112234/adg.pdf

Waliczek, T.M. (1999). "School gardening: Improving environmental attitudes of children through hands- on learning". *Journal of environmental horticulture*, 17 (4), p. 180.

Wandel, M. (n.d.). *Lego domino row building machine*. Retrieved from http://woodgears.ca/domino/index.html

Wood, E. (2004). Developing a pedagogy of play. *Early childhood education: Society and culture*, 19-30.

Wright, E. (2003). *Line following - a guide to using sensors*. Retrieved from http://www.wrighthobbies.net/guides/linefollower.htm

APPENDIX A: ARDUINO CODE

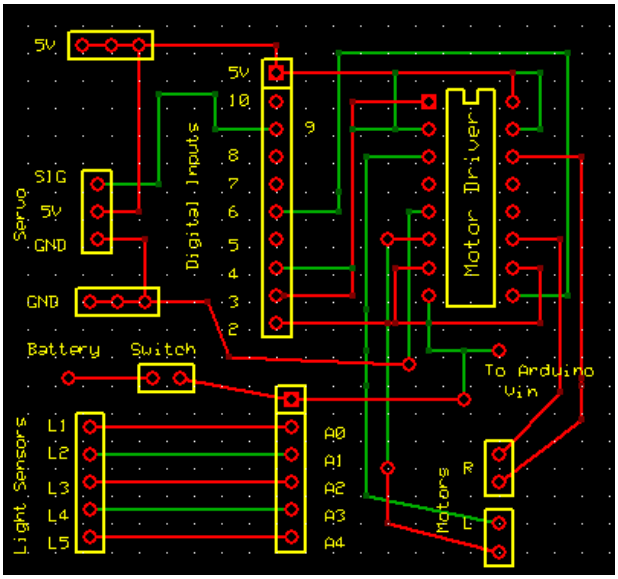
```
const int light0 = A0;  // Analog input pin light sensor
const int light1 = A1;  // Analog input pin light sensor
const int light2 = A2;  // Analog input pin light sensor
const int light3 = A3;  // Analog input pin light sensor
const int light4 = A4;  // Analog input pin light sensor

int lightvalue0 = 0;        // value read from the leftmost light sensor
int lightvalue1 = 0;        // value read from the leftish light sensor
int lightvalue2 = 0;        // value read from the middle light sensor
int lightvalue3 = 0;        // value read from the rightish light sensor
int lightvalue4 = 0;        // value read from the rightmost light sensor

int left_ratio = 0 ;        // ratio of light being read on the left side
 int right_ratio = 0;        // ratio of light being read on the right side
int total_light = 0;        // total light being picked up
float left_power = 0;       // power ratio going to the left wheel
float right_power = 0;      // power ratio going to the right wheel
float strength=0;           // Dummy, used in calculating power ratios
float strength2=0;          // Dummy, used in calculating power ratios

const int motor_direction1 = 2;  //The pin controling the left motor's direction
const int motor_direction2 = 4;  //The pin controling the right motor's direction
float left_motor = 3;            //The pin controling the left motor's power
float right_motor= 6;            //The pin controling the right motor's power

unsigned long start_time=0;     //Used by the timer
unsigned long current_time=0;   //Used by the timer
unsigned long end_time=0;       //Used by the timer

//servo setup
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position

void setup() {
   // initialize serial communications at 9600 bps:
  Serial.begin(9600);
  // set all light reading analog pins to inputs
  pinMode(light0, INPUT);
  pinMode(light1, INPUT);
  pinMode(light2, INPUT);
  pinMode(light3, INPUT);
   pinMode(light4, INPUT);
  //set all pins for motor direction and power as outputs
  pinMode(motor_direction1, OUTPUT);
  pinMode(motor_direction2, OUTPUT);
  pinMode(left_motor, OUTPUT);
  pinMode(right_motor, OUTPUT);
  // attach the servo on pin 9 to the servo object
  myservo.attach(9);
  //start the timer
  start_time=millis();
}
```

```
void loop() {
  //run the motor for 1.2 seconds
  if (end_time<1200){
    // read the analog input values:
    digitalWrite(light0, HIGH);           // Pin HIGH (discharge capacitor)
      delay(1);                           // Wait 1ms
     digitalWrite(light0, LOW);           //Turn off internal pullups
    lightvalue0 = analogRead(light0);  // read light value
    digitalWrite(light1, HIGH);          // Pin HIGH (discharge capacitor)
      delay(1);                          // Wait 1ms
     digitalWrite(light1, LOW);          //Turn off internal pullups
    lightvalue1 = analogRead(light1);  // read light value
    digitalWrite(light2, HIGH);          // Pin HIGH (discharge capacitor)
      delay(1);                          // Wait 1ms
     digitalWrite(light2, LOW);          //Turn off internal pullups
    lightvalue2 = analogRead(light2);  // read light value
    digitalWrite(light3, HIGH);          // Pin HIGH (discharge capacitor)
      delay(1);                         // Wait 1ms
     digitalWrite(light3, LOW);         //Turn off internal pullups
    lightvalue3 = analogRead(light3);  // read light value
    digitalWrite(light4, HIGH);          // Pin HIGH (discharge capacitor)
      delay(1);                          // Wait 1ms
     digitalWrite(light4, LOW);          //Turn off internal pullups
    lightvalue4 = analogRead(light4);  // read light value

    // filter out any values that might come from the light sensor reading the
     // paper or background, typically low values
    if (lightvalue0 < 100) lightvalue0=0;
    if (lightvalue1 < 100) lightvalue1=0;
    if (lightvalue2 < 100) lightvalue2=0;
    if (lightvalue3 < 100) lightvalue3=0;
    if (lightvalue4 < 100) lightvalue4=0;

    //calculate the steering power ratio
    left_ratio = (lightvalue0 + lightvalue1 + lightvalue2);
    right_ratio = (lightvalue2 + lightvalue3 + lightvalue4);
    total_light= (lightvalue0 + lightvalue1 +lightvalue2 +lightvalue3 + lightvalue4);

    //write the steering ratio and direction
    digitalWrite(motor_direction1, HIGH);
    digitalWrite(motor_direction2, LOW);
     if (left_ratio>right_ratio){        //If the robot is left of center
      strength=lightvalue0*.95 + lightvalue1*.85  +lightvalue2*.75;
      strength2=(float)strength/left_ratio;
      right_power = (float)left_ratio/total_light*strength2 ;
       strength=lightvalue0*.35 + lightvalue1*.45  +lightvalue2*.55;
      strength2=(float)strength/left_ratio;
      left_power = (float)left_ratio/total_light*strength2;
      analogWrite(left_motor,left_power*255);
       analogWrite(right_motor,right_power*255);
    }
    if (left_ratio<right_ratio){         //If the robot is right of center
      strength=lightvalue2*.75 + lightvalue3*.55  +lightvalue4*.4;
      strength2=(float)strength/right_ratio;
      right_power = (float)right_ratio/total_light*strength2 ;
```

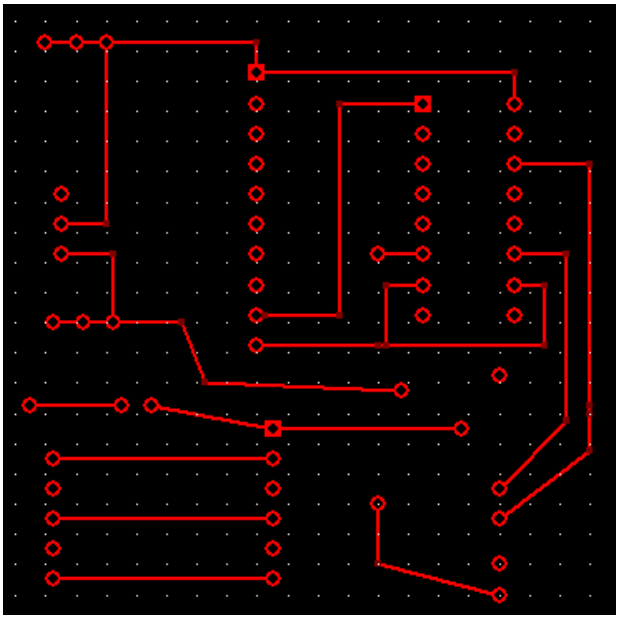
```
      strength=lightvalue2*.55 + lightvalue3*.65  +lightvalue4*.75;
     strength2=(float)strength/right_ratio;
     left_power = (float)right_ratio/total_light*strength2 ;
     analogWrite(left_motor,left_power*255);
      analogWrite(right_motor,right_power*255);
   }
   if (left_ratio==right_ratio){       //If the robot is centered
     right_power = 0.75;
     left_power = 0.6;
     analogWrite(left_motor,left_power*255);
     analogWrite(right_motor,right_power*255);
    }
   delay(10);
   //see how much time elapsed
   current_time=millis();
   end_time=current_time-start_time;
}
 if (end_time>=1200){       // If enough time has passed to dispense a domino
  //stop motors
  analogWrite(left_motor,0);
   analogWrite(right_motor,0);
  //push out servo
 for(pos = 10; pos < 130; pos += 2)  // goes from 10 degrees to 130 degrees
 {myservo.write(pos);
  delay(15);
 }
 end_time=0;
  start_time=current_time;
 }
 // print the results to the serial monitor if need be:

 Serial.print("S1 : " );
 Serial.print(lightvalue0);
 Serial.print(" S2 : ");
 Serial.print(lightvalue1);
 Serial.print(" S3 : ");
 Serial.print(lightvalue2);
 Serial.print(" S4 : ");
 Serial.print(lightvalue3);
 Serial.print(" S5 : ");
 Serial.print(lightvalue4);

 Serial.print("     L: ");
 Serial.print(left_power);
 Serial.print(" R: ");
 Serial.println(right_power);

 // wait 10 milliseconds before the next loop
  // for the analog-to-digital converter to settle
 // after the last reading:
 delay(10);
}
```

APPENDIX B: PRINTED CIRCUIT BOARD

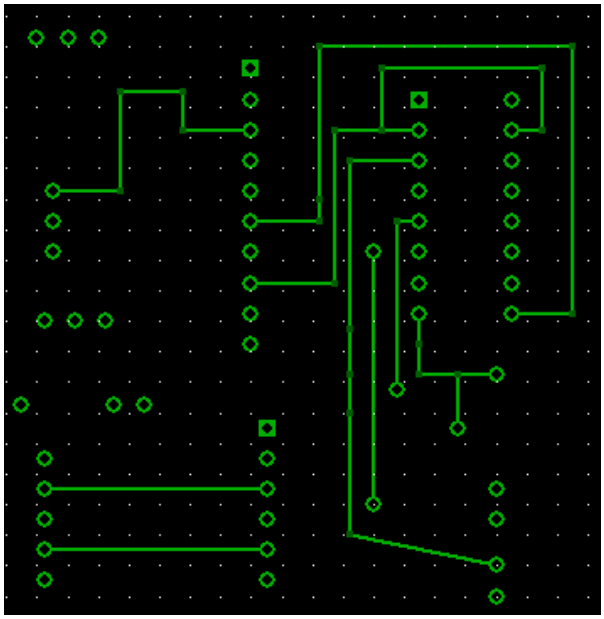
All Layers:



Top Copper:



Bottom Copper:



ACADEMIC VITA

Jenny Shipley
6620 Babak Drive
Frederick, MD 21702
Jship5916@gmail.com

Education

The Pennsylvania State University, University Park, PA
Bachelor of Science in Mechanical Engineering
Honors in Mechanical Engineering, Schreyer Honors College
Thesis Title: Semi-autonomous Line Following Domino Dispensor
Thesis Supervisor: Dr. Sean Brennan

Work Experience

Engineering Co-op
General Electric Aviation, Cincinnati, OH
Summer 2012
-Assessed hazards of plant operations and processes
-Collaborated with a team spanning all managerial and operational levels
-Gained an understanding of the manufacturing process

Honors Research Intern
Applied Research Lab, Penn State
May 2011 - May 2012; August 2012 - May 2013
-Create graphical user interfaces in MATLAB
-Explore the properties of fuzzy logic operators
-Determine applications for ordered weighted averaging

Relevant Skills

MATLAB, ANSYS, SolidWorks

Leadership, Service, and Awards

Recipient; Schreyer Honors College Academic Excellence Scholarship
Recipient; Dr. Susan Rankin Award for Leadership, Integrity, and Outstanding Contributions to Penn State University
President; Penn State's Rainbow Roundtable
Treasurer; Out in Science, Technology, Engineering and Mathematics
Volunteer; The Lesbian, Gay, Bisexual, and Transgender Student Resource Center