Abstract

Programmable matter is well-researched from a theoretical perspective, but existing prototypes are yet to demonstrate all possible functionalities. This project aimed to create a functional system of self-assembling robots by integrating ideas from various projects. Areas of interest were the robot shape, electronics, actuation/latching method, wireless communication, and assembly algorithm. Electropermanent magnets were thought to be viable for latching with minimal power requirements, but material availability issues and insufficient magnetic force output were obstacles to successful actuation. Standard solenoids were considered as an alternative but also unsuccessful due to low magnetic force output. Currently, no alternative for actuation at this prototype's scale has been found. An assembly algorithm was created that succeeds in small-scale situations required by the prototype, but would have issues being scaled; alternatives have been identified but not implemented. Overall, this prototype system was not functional but discovered important limitations on actuation previously considered theoretically viable.

PROTOTYPING A PROGRAMMABLE MATTER SYSTEM

TEAM SAND

LUCAS ARMYN, JOSHUA LEE, MATEO LIM, ERIC O'LEARY, WASIF PERVEZ, HANOCK TSEGAYE, SIERRA RASPA

DR. PO-YEN CHEN

Thesis submitted in partial fulfillment of the requirements of the Gemstone Honors Program, University of Maryland, 2024

Committee: Assistant Professor Po-Yen Chen, Professor Wesley Lawson, PhD Candidate Joshua Little, Assistant Professor Michael Otte, Assistant Professor Sahil Shah

Acknowledgements

We would like to sincerely thank Dr. Po-Yen Chen, our mentor, for graciously offering us resources, feedback, and guidance throughout our project. We would also like to give many thanks to Mr. Joshua Little, whose input and advice helped improve many aspects of our research. Additionally, we are grateful to the Gemstone staff for giving us the opportunity to pursue our research interests and a funding platform through Launch UMD. We would also like to thank our discussants for their time and feedback. We also greatly appreciate our generous donors for funding our project. Furthermore, we would like to recognize our librarian, Ms. Sarah Rosa Weiss, for providing advice on our thesis. Lastly, we wish the best to our Gemstone cohort and extend our thanks for being so supportive throughout our research journey.

Table of Contents

List of Figures

List of Tables

I. Introduction

A. Background and History

Programmable matter has been a topic of interest to the scientific community since 1991, when Toffoli and Margolus were investigating its meanings and applications within chemistry and gasses [1]. While their original vision was in the field of chemistry, around the same time period, similar concepts to programmable matter also appeared in robotics in the form of reconfigurable robots [2].

Programmable matter is a vast pool of different fields, manufacturing methods, and applications. In chemistry, programmable matter is the intentional manipulation of materials, usually through self-induced folding, in order to create certain desired characteristics such as geometry, tensile strength, hardness, and more [3]. This relates to the field of soft robotics where the actuation and sensing of the soft robots are dependent on the material properties and chemical reactions of the robots' material. In biology, this can appear as programmable medicine or small scale bio-robots that inspect organs or deploy medicine *in vitro* [4]. In traditional robotics, this can manifest as swarm robots that self-assemble to create various geometries. This final definition of programmable matter will be the focus of this paper. In particular, this project focuses on self-assembly, which is a common problem within robotics and programmable matter. Specifically, this refers to a swarm of robots moving independently into one mass.

The influence of Moore's Law - the statement made by Gordon Moore that every two years, the number of transistors in an integrated circuit will double [5] - on making electronic components smaller made manufacturing of smaller robots more feasible, which increased interest in the small-scale robotics field. Further advancements were made from several notable groups until the present day, such as the Catoms, the Robot Pebbles, the Polybot, and the ATRON modules [6], [7], [8], [9].

Programmable matter, in the field of robotics, inherently requires programming. One of the most common applications of programmable matter is self-assembly, where a system of bots can actuate with the goal of forming a cohesive structure. The problem of moving a swarm of robots from one set of positions to another subject to certain constraints is an application of a modern field of algorithms research, Multi-Agent Pathfinding (MAPF). There exists a sufficient amount of theoretical research about this problem and its applications to swarm robotics. Of these papers, the majority focus on significantly more difficult tasks, such as communication in disruptive conditions, using sensors or computing on the bots to accomplish large scale tasks. By contrast, this project seeks to develop a simpler algorithm that allows a collection of specifically-shaped bots to travel from one two-dimensional configuration to another in order to create a working prototype of the entire system. Similarly, the communication methods and sensors used serve to showcase a complete, functional system.

This paper introduces a new programmable matter robot that combines aspects from prior research together into one cohesive device. It incorporates electropermanent magnets largely researched and used by the Robot Pebbles from MIT, the quasi-sphere shell from Carnegie Mellon's catoms, and basic commercially available sensors and electronics for the circuitry.

B. Motivation

Previous efforts in the field of self-assembly have made strides in developing robots that can attach to and climb over each other to form desired shapes, such as the M-blocks engineered by MIT researchers in 2018 [10]. However, there exist limitations that this project seeks to address which currently hinder the execution of self-assembly in robotics. Currently, many experiments have used cube-like or cylindrical shapes. A cubic shape requires a significant amount of energy to rotate and creates limitations on rotation direction. By contrast, a cylindrical shape is more optimal for rotation, but it has very small flat surfaces and causes latching between robots to become more difficult as a result. Therefore, the hybrid approach of implementing additional faces to create a quasi-spherical shape allows for rolling in the four cardinal directions with less energy consumption while also maintaining the increased surface area for latching afforded by the cubic elements of the design.

Researchers from the Claytronics Project explored the optimal shape of small-scale, programmable matter robots and concluded that the quasi-sphere is the most effective. However, their research on the quasi-sphere was mostly theoretical and simulation-based. The team at MIT who created the Robot Pebbles created a physical prototype while conducting their research into the efficacy of electropermanent magnets for small-scale applications in comparison to traditional electromagnets. However, the Robot Pebbles did not utilize the optimal shape of the quasi-sphere from the Claytronics Project in their research. Thus, this paper is unique in that it combines the aspects of both research papers into one robot by using electropermanent magnets housed in a quasi-spherical 3D printed shell and tests its performance.

Self-assembling small scale robots have several applications such as rapid prototyping, search and rescue missions, and possibly aiding in surgeries. Rapid prototyping is likely the most feasible application that can be derived from the robots presented in this paper, but with more development, the robots could reasonably be used in the other applications listed above.

C. Research Question

In the proceeding work, the various components of a complete programmable matter system are examined, revealing both the requirements and possible solutions for each aspect of the robotic system. The question this project is attempting to answer is if it is possible to create a swarm of small-scale robots that are capable of on command assembly into three-dimensional architectures. The necessary components this project examined are the shape of the robot, the method by which the robots latch to each other and maneuver themselves through space, and the pathfinding algorithms which dictate the robot's movement.

II. Literature Review

A. Background/History

1. Self-assembly

Since self-assembly is a key aspect of this project, it is important to precisely understand what characteristics the designed system needs to embody and what potential methods exist to achieve self-assembly. The definition of self-assembly is important to clarify, as it varies based on the field. In chemistry, the term refers to multiple objects which join together without external motivation. In robotics, however, self-assembly refers to multiple robots creating one singular entity that can then be split back into its original parts on command. This is the definition that this project will be using.

Self-assembly is composed of five characteristics: (1) the components of the system which self-assemble, (2) the interactions between each component, (3) the reversibility of the process or adjustability of components' positions within the assembled entity, (4) the environment in which assembly takes place, (5) and the mobility of each component [11]. These concepts will be discussed in greater detail throughout the following sections through their applications to this project.

The creation of specific shapes using self-assembling robots requires control over the way that the robots actuate, which will change based on the size of the individual robots. In the future, downscaling the robots to the micro- and nanoscale can make use of electrostatic or friction forces for actuation or latching to each other [12], but for the scope of this project, larger units on the orders of millimeters or centimeters can use onboard electromagnets to move [13].

2. Self-reconfigurable Robots

In recent years, the idea of self-reconfigurable (SR) robots has been popular in both thought and experiment. Scientists have been moving towards constructing robots that are able to alter themselves to adapt to new environments and tasks. SR robots are a relatively recent concept that provides a strategy to solve many problems on the macro-scale, including using some forms of self-assembly. The concept of an SR robot comes from biology, where each SR robot functions like a cell in a larger organism, capable of a number of different tasks [14]. These types of robots have incredible potential to utilize a number of characteristics, self-assembly included. The field of SR robotics offers a number of helpful ideas that can be applied to systems that purely focus on self-assembly. For one, different structures, such as lattice, chain, or hybrids of the two, and different algorithms for control have been theoretically tested and scaled to millions in the second case [14], [15], [16].

However, there is a distinction between self-reconfigurable robots and the self-assembling robots that this project focuses on. One self-reconfigurable robot is made of a number of modules that could be considered smaller robots, intended to act as a single, larger robot. By contrast, these self-assembling robots do not serve as components of a larger robot, but rather as individual robots that are part of a larger piece of non-robotic matter.

An SR robot's purpose is to exceed the limited adaptability of current robots that must be outfitted with specific tools for each task. In contrast, these self-assembling robots are created with the intended purpose of rapid, major changes to the shape and structure of the swarm as a whole [14].

3. Claytronics Project

In 2004, the Claytronics Project, funded by Intel, Carnegie Mellon, and the FEMTO-ST Institute, set out with the goal of developing sub-millimeter robots called catoms that can self-assemble into larger shapes [6]. The name "catom" is a portmanteau of "Claytronic atom."

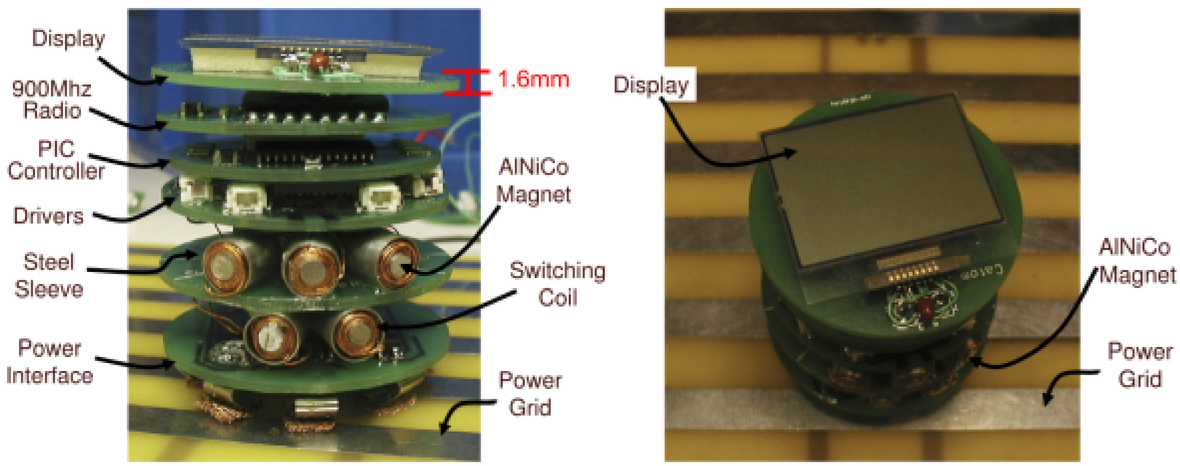


Figure 1. A component breakdown of a catom. The red text has been edited to show the size scale. Adapted from [6].

The catoms used solenoids as their means of actuation. This resulted in size limitations to their catoms because a large number of wire coils were required in order to get the necessary strength for movement, creating a constraint on minimum magnet size.

Each catom is essentially useless on its own; it is only when the catoms assemble together to form a larger shape that they become immensely useful and versatile. Eventually, the Claytronics Project intends to create a level of resolution that allows any object formed by the catoms to almost perfectly mimic a real object [6]. It is for this reason that the Claytronics Project is pushing for robots that are sub-millimeter because the smaller the bot, the higher the resolution of the object that is formed by the catoms. Using the Claytronics Project as inspiration, each robot in this project was originally intended to be self-contained in design, meaning that every electrical and mechanical component needed for operation would be incorporated into each bot; however, due to constraints of the sub-millimeter scale, the robots slightly diverge from this original intention of self-containment [6]. It is important to understand the goals and concepts of the Claytronics Project, as they lay the foundation for this project and offer a strong, guiding direction.

4. MIT Robot Pebbles

A major literature source referenced for the electronics and actuation of this project was the MIT Robot Pebbles doctoral thesis published in 2010 by Ara Knaian [7]. These small cubic robots introduced the idea of using electropermanent magnets (EPMs) rather than electromagnets as a form of actuation for each robot. EPMs are magnets formed using NIB and Alnico magnetic rods wrapped in coil with two iron caps at the top and bottom of the rods. The magnet can be turned on and off with a pulse of current in opposite directions, thus consuming less power than solenoids which need to have continuous current in order to provide a magnetic field.

The Robot Pebbles used 12 mm edge length, cube shaped robots with the PCB on the inside surface of the robots' shells, as shown in Figure 2.

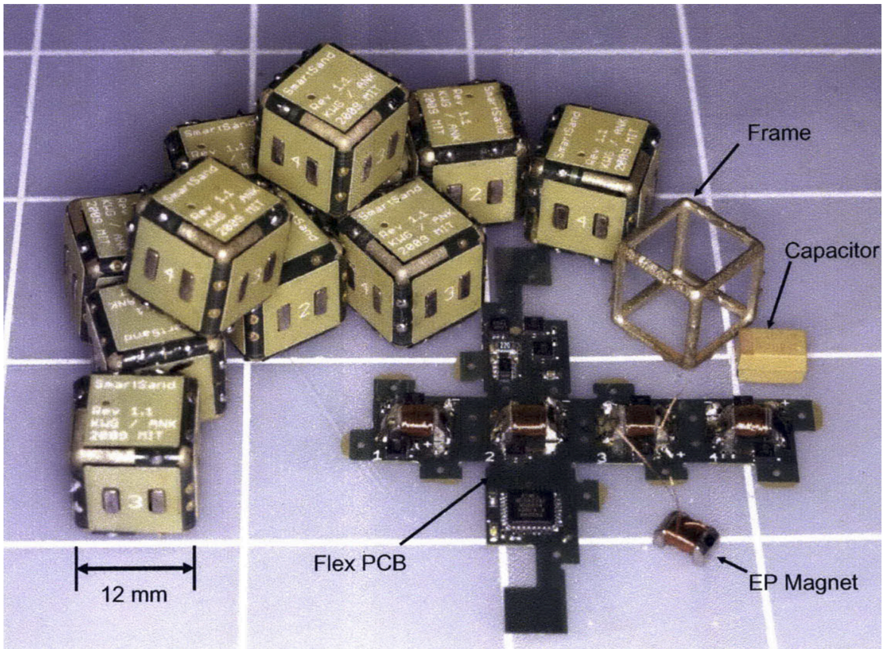
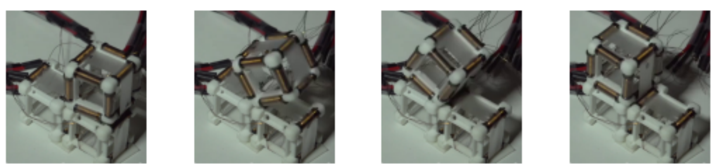


Figure 2. MIT Robot Pebbles with flex PCB and frame. Adapted from [7].

The electropermanent magnets are soldered onto a flexible PCB on four sides of the cube, allowing for rotation, and powered by a 100 uF capacitor for power storage. The goal of the Robot Pebbles is similar to this project but differs in a few key ways, such as being designed to mimic shapes in only two-dimensions. However, this provides the precedent of using the electropermanent magnets that this project used for some stages of its development.

5. Kubits and ElectroVoxel

Another literature source that demonstrates working prototypes of self-reconfiguring modular robots (SRMRs) are MIT's Kubits and ElectroVoxels [17], [18]. Both of these projects used a novel idea at the time through the development of a "programmable magnet" (PRM) [17]. This magnet takes the idea of the EPM from the Robot Pebbles [7] and adds an additional state. Instead of just having an on and off mode, these programmable magnets also have the capability to reverse their polarity, allowing for positive polarization, negative polarization, and a neutral state [17]. These three states allow for attraction, repulsion, and neutrality, which act as forms of connection, actuation, and disconnection. The ability to have different polarization states of the magnet allow for a stronger attractive force between two faces of opposite polarity. Similar to EPMs, these magnets are also controlled by electrical pulses allowing a similarly reduced power consumption when compared to typical electromagnets. The different states are achieved by sending the electrical pulses at different amperage. Lower amperage results in the neutral state, while high positive and negative amperage results in positive and negative polarization respectively. These PRMs were implemented into a cubic design to form the modular robots named "Kubits". The electrical systems of these robots were externally connected to an Arduino Nano in order to control the switching and perform the flipping algorithm demonstrated in Figure 3.



(a) t=16.6ms (b) t=50ms (c) t=83.3ms (d) t=116.6ms

Figure 3 (a-d). Kubit modular robots performing flipping algorithm. Adapted from [17].

This demonstration gave promising results of the actuation method with the PRMs. However, the modular robots were still wired up to an external Arduino Nano which doesn't lend itself towards assembly with a large number of robots. The ElectroVoxels improved upon this research by designing a modular robot that utilized the PRMs while having an onboard Arduino Nano design to control the switching, without having external wires connected to the robots [18]. For this design, a custom PCB was developed which used H-Bridges to send pulses of current in different directions. An experiment with four ElectroVoxel modules on an air table and an experiment in a microgravity environment were conducted to test the actuation capabilities of these robots. The results of these experiments were very promising inspiring further research with modular robots with similar actuation methods.

B. Potential Applications

1. 3D Printing

Each individual robot acts as a “particle of matter” that joins together to create a larger shape or object. Sensors on the robots identify the robot’s position and orientation, a pathfinding algorithm selects the optimal path for a robot to travel to in order to create the desired shape, and magnetic actuators attach and repel the robots for movement. The smaller the robots, the more optimal and usable this application becomes. The end goal would be to design the robots such that each one is no larger than the size of a grain of sand in order to achieve a higher resolution and more advanced geometries. The robots would act functionally the same as 3D printing, but achieved in a different way. Furthermore, this method of 3D printing brings new benefits that traditional 3D printing cannot provide. The first is extended reusability of the prototyping material. Typical 3D printing materials such as PLA and ABS can be reused, but only a few times before the material begins to degrade [19]. Additionally, most PLA and ABS material is classified under Type 7 plastics, which have a limited number of municipal waste centers that process it in comparison to how much is produced [20]. The small robots would serve as a replacement for these materials, as they would be able to be reused for a much longer period of time. Prints could be reused after the task is completed by commanding the robots to dissolve back into a swarm, waiting to be reprinted into the next desired form [21], unlike PLA which needs to be repeatedly melted and resolidified after each print and eventually degrades after several uses.

2. Education and Entertainment

Many research groups have proposed the idea of using programmable matter in education and entertainment. In education, the robots could assemble into shapes that visually convey complex topics to students. The research group that built the I-Blocks - a set of modular electrical building blocks that are embedded within LEGO DUPLO pieces - conducted multiple studies on how these robots could be used for educational purposes [22]. These devices were tested by children in schools and hospitals where it was demonstrated that they encouraged student learning in construction and hardware and fostered their creativity.

In entertainment, the robots could be used as toys or games. For example, a Swedish research group proposed the Cubimorph, which is a modular device that can self-reconfigure to and from multiple types of daily use electronics, such as from a mobile phone to a video game console, thus acting as a form of entertainment [23]. Another research team proposed modular robotic tiles designed to incite mental and physical play for children [24]. Although this design of programmable matter is different from the one proposed in this thesis, the application is still relevant and has been cited by robot projects more similar to this project such as ATRON [25]. Programmable matter could also be used for three-dimensional movies and television [13].

C. Design

1. Shape

Constructing robots to be able to move and latch on to each other requires a specifically designed shape for each unit to operate effectively. Finding that optimal shape is difficult because each robot needs to be able to move quickly with low energy cost while also needing to have enough surface area for latching. In this sense, the ideal shape designs for each restriction seem to oppose each other; a sphere is the optimal shape for locomotion as its smooth curvature allows objects to roll with ease but only provides one singular point of contact between two catoms, whereas a cube provides the greatest uniform surface area for latching between catoms but requires significantly more energy to rotate and move in a practical way. Thus, a far more efficient shape for these catoms is an amalgamation of the sphere and cube to bring in the vital components from each one.

Various shapes have been observed in different research teams, a bulk of them stemming from the Claytronics Project. One of the first realizations of note was a small cylinder that used electrostatic forces to roll along a strip of material [12]. Here, the cylinder shape is excellent for movement, but the two cylinders only have one horizontal line as the point of contact. This surface area is immensely small and does not provide a sufficient sticking force between two catoms, which is similar to the problems created by the sphere. Another issue is that turning and multidirectionality are very difficult with this design. Other shapes used in past groups have been a stacked cylinder from another iteration of the catoms from the Claytronics Project, a cube from the MIT Robot Pebbles, and a stacked hemi-sphere from the ATRON project [6], [7], [25].

However, a research team for the Claytronics Project performed an in-depth study on the superlative shape for a catom that maximizes both actuation and latching and concluded that a quasi-sphere was the best choice [26]. The quasi-sphere is a combination of a sphere and a cube, resulting in a spherical shape with many flat surfaces.

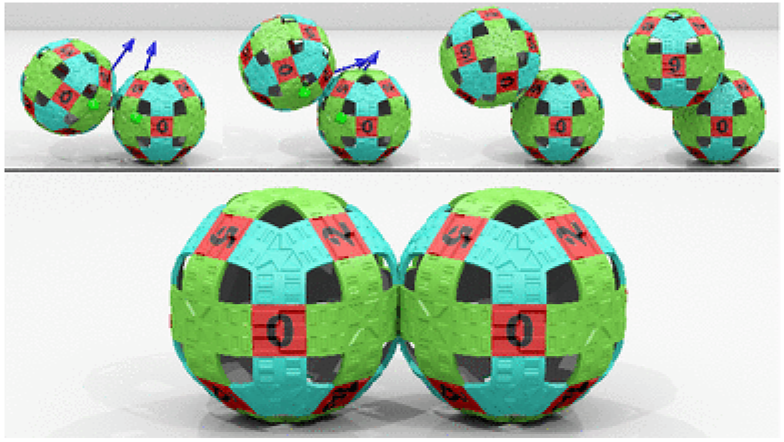


Figure 4. Three-dimensional simulation of the quasi-spherical catoms and the manner in which they connect to each other. Adapted from [26].

As seen in Figure 4, the quasi-sphere utilizes the roundness of a sphere for easy rotation but also increases surface area for contact points. Previous work demonstrates how the quasi-sphere allows for more precise movements and interactions between catoms, as the path that a singular quasi-sphere catom traverses is well-defined [27]. The robots designed in this thesis are most similar to a quasi-sphere, with some modifications for simplicity and to reduce the number of magnets needed for actuation.

2. Actuation

Actuation is the process in which components of robots move and contribute to the overall operation of the robot. The Polybot and M-TRAN use physical mechanical latches for actuation; however, downscaling this is difficult [8] [28]. Thus, one of the most common methods of actuation and latching that has been observed in similar robots is through the use of electrostatic forces which can be easily downscaled. The catoms of the Claytronics Project's prototypes contained a chip that generates electrostatic forces, utilizing electrodes on the left and right sides of the cylinder that have an attractive force to other electrodes, test substrates or charged rails the cylinders roll upon, as portrayed in Figure 5 [12]. The electrostatic forces used for actuation are created through the use of a capacitor that induces an electric field between the plates.

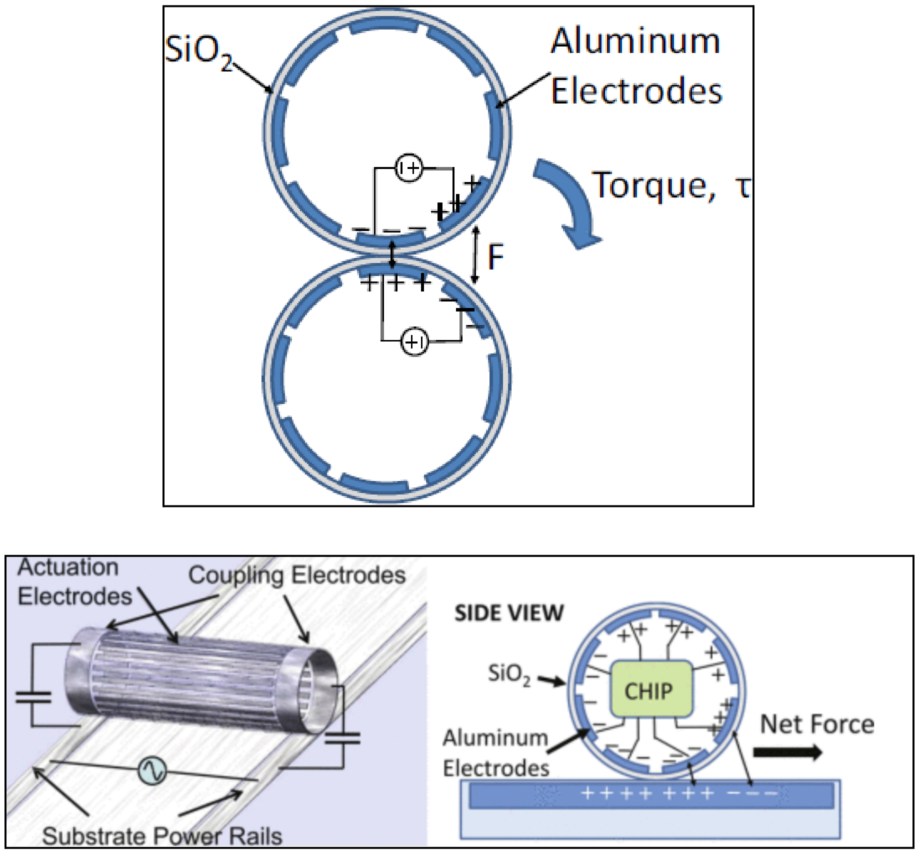


Figure 5. Cylindrical catom rolling along a strip via electrostatic actuation. The ends of the catom have electrodes that induce the necessary sticking force, which is created with the help of the chip. Source: Adapted from [12].

Besides electrostatic forces, another method of latching is through magnetic fields generated by electropermanent magnets [7]. The key functional distinction between the use of electropermanent magnets and electrostatic forces is the power required to induce these electromagnetic fields. Electropermanent magnets are an assembly of permanent magnets in such a way that they can be effectively demagnetized (turned “off”) or remagnetized (turned “on”) with a strong pulse of current, rather than the constant current that is required by standard electromagnets or electrodes.

This is achieved through the use of two magnets of different materials, with one having a significantly higher magnetic coercivity than the other. A magnet with a high coercivity is able to withstand a stronger magnetic field without becoming demagnetized, so when a strong current is pulsed around both magnets, the magnet with lower coercivity switches polarity while the other does not. By using this phenomenon to control the relative orientation of one of the magnet's poles to the other, the entire assembly can be effectively rendered demagnetized until current is pulsed in the opposite direction, turning the assembly back on. This is shown in Figure 8 in the Methodology section.

In electrostatic systems, the forces used for actuation are created through the use of a capacitor that induces an electric field between the plates. Some advantages of this method of actuation involve an easier fabrication process, smaller volumes, and a simpler overall system design [7].

On the other hand, electropermanent magnets are capable of generating greater holding pressure than the charged plates in air used in electrostatic actuation; they also require lower driving voltage, and they are able to be scaled down to the microscopic scale [7]. Electropermanent magnets are used in these systems for many of the same reasons that this project chose to use them over electromagnets, namely for power efficiency, but also to allow for fine control over the strength of the generated field.

D. Programming/Coding

1. Wireless Communication

Wireless communication is a necessity for the majority of modern robotics, especially when the robots are small and there are many of them. There are two main solutions to this problem - WiFi and Bluetooth - as well as many in-development technologies such as bots communicating through contact via sensors on touching faces [30]. When creating a general prototype, WiFi and Bluetooth are inexpensive and easy to use, and any other options are designed to handle niche cases that this prototype does not need to worry about.

Between WiFi and Bluetooth, the choice somewhat depends on application. WiFi has a long range, both in distance from the programmer to the transmitter and from the transmitter to the robot. However, it can be unintuitive to set up and require many additional add-ons. By contrast, Bluetooth functions at a significantly smaller range, but is by design significantly easier to set up. Also, as of Bluetooth 4.0 in 2009, a similar but independent communication method called Bluetooth Low Energy, or BLE, was available alongside traditional Bluetooth. BLE inherits the easy set-up of Bluetooth while also optimizing for smaller hardware size and reduced power consumption without compromising on transmission range.

2. Programming Style

There are two primary programming styles for a project of this nature - global and local - which refer to where the decisions for the swarm are made. A global style has a computer that is separate from the swarm to handle a majority of the decision-making, such as complicated calculations, running the assembly algorithm, or parsing sensor data. A local style lacks this outside computer, so it requires the robots of the swarm to do all of their computations. The choice between the two depends primarily on application. For a self-assembling swarm that wants to maintain a given shape regardless of disruptions to the structure or bots being destroyed, a local style makes the most sense, whereas for a swarm that wants to change from its given configuration to a new configuration on command, a global style is required to even give that new command. Of course, a project does not need to strictly choose between these two, and can model one while adopting characteristics of the other to better suit the project's needs. However, the required hardware and algorithms change drastically depending on the style.

Existing frameworks that utilize a global style show promising results in algorithmic efficiency, such as the method suggested by a Bar Ilan University research paper in 2016 in which a new programming language and compiler is developed to execute the necessary high-level commands [21]. For example, the high-level command to form a line would be given as individual orders for the closest robots to move to specific positions in that line, where a developed algorithm decides which bots move where and when for efficient construction and to avoid collisions. The researchers suggested that such a system could be developed for specific nanobots designed for targeted drug delivery with promising results.

Another good example of such high-level commands is the Meld programming language, formulated in 2007 by Carnegie Mellon. The university researchers described Meld as a declarative programming language that takes a global perspective of coding, based on an overlay network language called P2 [31].

Overlay networks are essentially mother-programs that control how other networks will operate. A real-world instance of an overlay network would be the internet, which performs routing functions that dictate what happens to messages from one computer network to another. The internet acts as the mother-network, while other devices, such as smartphones and laptops, represent the child-networks. The mother-network contains programs that choose what paths messages take between devices. Messages can be anything ranging from emails to text messages to multi-factor authentication, in which the user's access request is the message. This path selection process utilizes routing metrics that take into consideration information such as bandwidth, network delay, hop count, path cost, load, maximum transmission unit, reliability, communication cost, and geographical distances to find the best and most efficient path for messages to travel [32]. The individual computer networks just send information, but the overlay network is the program that globally controls how all the messages are sent between computer networks. An illustration of this functionality by Rainer Baumann et al. [32] can be seen on Figure 6. Meld, or a similar programming language, can be utilized as a means of taking an overlay network approach to self-assembly [31].

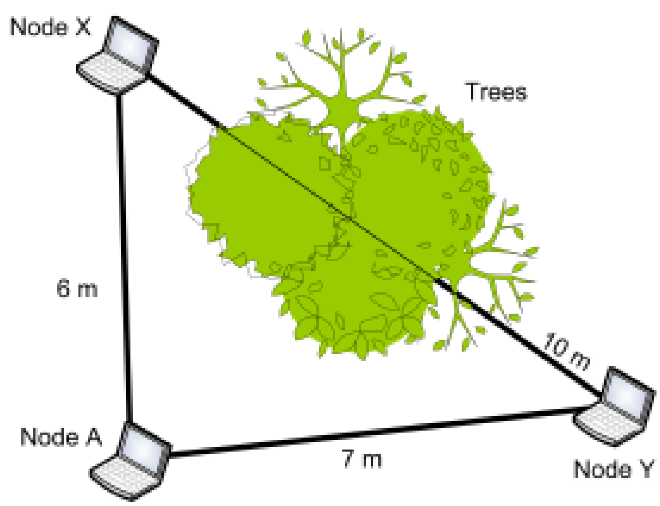


Figure 6. Node A (The Overlay Network) recognizes the physical limitations (Trees) of sending information directly between Node X and Y, so Node A re-routes the transfer of information for efficiency's sake, even though it is a longer path. Source: Adapted from [32].

Unlike the global approach to self-assembly, the local style takes a collaborative approach to how robots navigate and attach to each other, through communicating with each other and the environment to accomplish a task. Algorithms to guide a network of bots in their pathing and assembly could take the form of local interaction between singular robot units to guide the larger swarm into the correct positioning, with each bot's unique coordinates being measured relative to other nearby units. It has already been developed for swarm systems of 1000 individual units [33]. While the size of units heavily affects locomotion and internal systems, the algorithm for relative positioning is applicable regardless of robot scale, with related changes being made to account for the unique environmental challenges associated with each size.

There are several papers that illustrate the feasibility of using interactions between members of the swarm to facilitate robot movement into desired paths and positions. The first is the slime mold algorithm, as proposed by Schmickl and Crailsheim [34]. This algorithm is based on how slime mold amoebas secrete a certain chemical known as cAMP at certain frequencies, giving information to other slime amoebas about how to navigate away from areas of low food density. This process is supplemented by each amoeba's ability to become immune to the sensing of cAMP concentration for a set period of time after a certain threshold is reached, allowing for soft resets of their navigation and continuous updates to fit any changes in the environment. Two Austrian researchers built a swarm navigation system based on the slime mold method and simulated it by assigning roles (empty bot, loaded bot, and random walker) to different bots in pursuit of the overall end goal: to collect all the pieces of dust in the environment and place them in a designated disposal area. As each bot calculated where to move by analyzing the signals sent by bots around it and understanding its own current state, the bots were able to efficiently move back and forth between the collection and disposal sites, with the random walkers mapping out various spaces throughout the navigable region. They also found that their simulated robots could find the optimal path for the situation after adjusting the number of options and spaces that the robots had to move through [34].

Another algorithm to consider is the bat algorithm which relies on the loudness of echolocation signals in a certain area as a stimulus that indicates what directions different members of the swarm should go in. Random paths are also employed to ensure that several parts of the navigable region are covered and that the change in position of the bots is based on continuously updating stimuli. This algorithm is influenced by external stimuli (loudness) as opposed to the slime mold algorithm which primarily considers internal state metrics such as dust concentration [35].

As an algorithm for navigation and movement is developed, it is also necessary that the performance and efficiency can be quantified. To that end, there is a model designed by two German researchers which can be used to evaluate the performance of a control/navigation algorithm and give a reasonable indication of how the swarm will react to the instructions that it is given [36]. Their work proposes a series of processes outlined by equations, state diagrams, and other displays that take into account the displacement of the robots based on collisions, probabilities of encounters, reactions to communications and stimuli, and where the robots congregate.

For the scope of this project, wireless communication between the bots and an external resource like a computer for the purpose of self-assembly would lessen complexity of communication. As mentioned previously, a bluetooth connection could accomplish this, BLE specifically for optimizing this global style.

3. Memory Storage

In order for the robots and algorithms to function properly, they have to receive input data, such as instructions for which magnets to provide voltage for and the sequence of operations to get from one point to another. Determining the means of communication between bots, whether external or direct, impacts how memory storage would work with self-assembling robots. In order for such calculations to be made, there needs to be a means of storing the information that is received before a program acts on that data. This process is called memory storage. There are two types of memory: volatile and non-volatile. Volatile memory is memory that loses its data when the device is turned off. Non-volatile memory is memory that does not lose its data even after the device has been turned off. Flash memory would be an example of non-volatile memory, and RAM would be considered volatile memory [37]. Due to the repurposability of self-assembling robots, the information it needs to hold the configuration is no longer necessary once disassembled. Volatile memory would allow for input data to be removed after the bots have turned off and disassembled, preventing any issues of bots latching indefinitely, or inability to process new input data due to existing previous data.

Another aspect of memory storage to consider is the location of memory (i.e. whether the individual bots contain this memory or there would be an external device that holds this information). Multiple researchers from a plethora of universities, including Tel Aviv University and MIT, collaborated to answer this very query, in which they describe two distinct memory models: external and internal. They found when analyzing the performance of algorithms, an external memory model was effective at handling computations that do not fit in the main memory unit through outsourcing storage space from an external source such as a hard disk. They noted the high speed of the internal memory model, which only utilizes its main memory unit, but limited space for computations [38]. As to how this ties to self-assembling robots, each bot must be able to store input data, such as electrostatic forces. Then at some point, calculations have to be made as to how it will interact with other bots. If these bots tried to perform the computations on their own, using an internal memory model on a micro or nano scale, there would exist the issue of space. For the purposes of this project, prototyping on the macroscale lessens the negative externalities of memory management. Despite this, a global programming style would be able to maximally take advantage of these conditions. With the physical components and the programming framework developed to accommodate pathfinding algorithms, the final piece to this puzzle would be the actual algorithm itself.

4. Pathfinding

As discussed previously, the method of communication between robots is essential to understand how the robots configure into the desired shapes. Researchers from Dongguk University in 2021 explored local communication for real-time decision-making of each robot allowing for implementation of roll direction determination, and collision avoidance measures [39]. Their pathfinding algorithm stores possible paths in a tree that grows as more points in space are appended as nodes. A local piecewise linear path is created between the node, the parent node, and the colliding grandparent node to handle collisions. The researchers utilized interpolation by calculating the distance of obstacles to the path to handle sharp corners and accounts for kinetic energy of robots. The Figure 7 illustrates a demonstration of this algorithm with q_2 representing the grandparent, q_1 the parent, and q_0 the goal.

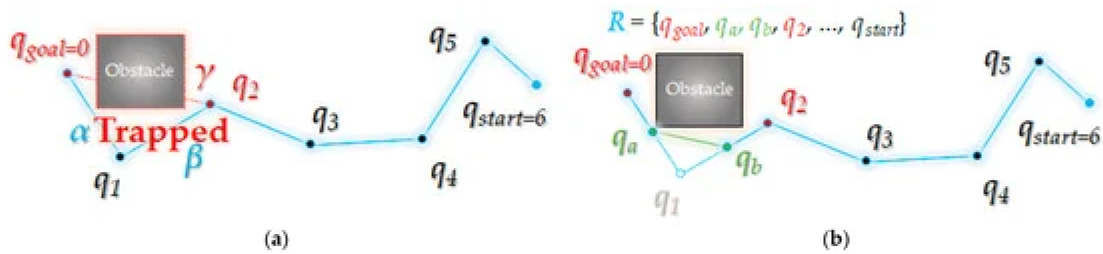


Figure 7. Node = q0, parent = q1, grandparent = q2. (a) the tree is not free from collision. (b) the parent node is replaced by qa and qb reducing the angle of turn required. Adapted from [39].

Despite the major utility of considering intricate environmental and internal variables (i.e. collisions and kinetic energy), for the scope of this project in which the restrictive parameters control for these possible issues, a more useful and wholesale approach to pathfinding would be an optimized shortest path algorithm for a set of robots.

The most applicable pathfinding problem is known as Multi-Agent Pathfinding (MAPF). Informally, the MAPF problem is to take a set of agents, represented as vertices of a graph, and calculate paths to ending locations, also vertices, without collisions between agents according to given constraints, such as finding the shortest path. Formal definitions of this problem and its constraints can be found across the literature, and one attempt to compile these was made at a conference in 2019 by Stern et al. [40]. There are multiple types of conflicts that may be considered a collision, but the most common collisions of concern are vertex conflicts and swapping conflicts. Informally, the former occurs when two agents occupy the same vertex at the same time stamp, and the latter occurs when two agents switch positions in the space of one unit of time, implying, in many applications, that a collision occurred during the movement process.

This area of problems has had many different solutions of varying levels of optimization. Many of the most effective modern solutions are based on an A* algorithm, which is a well-studied weighted-graph shortest path algorithm created in 1968 [41]. This algorithm is used as the base to more complicated systems, such as allowing multiple stops for one agent [42] or creating a tree of collisions to reduce the number of states the algorithm sees [43]. There has yet to be a single algorithm that is obviously the best overall [44], and many algorithms are tailored to a more specific type of MAPF problem. Utilizing the existing research on pathfinding algorithms as a foundation, this paper seeks to build a system capable of performing such calculations for self-assembly.

III. Methodology

A. Manufacturing

1. Structure

The outer shell of the robot was designed and created using the CAD software, SolidWorks. The many different iterations of the shell were 3D printed using Bambu and Prusa printers. The decision to use 3D printing for the manufacturing of the structure was mainly based on the accessibility and ease of use of this method. The design was modeled after the quasi-spherical shape used in the Claytronics Project [6]. However, due to the need to include internal components like the magnets and electronics, the shell had to be printed in two parts and assembled after the internals were inserted.

Several iterations of this design were created as new constraints were introduced, including the size of the electronics and the need for holes in the faces for the magnets to better latch onto another module. Initial prints were more cubic than originally intended, so the design was adjusted accordingly. Eventually, the team decided that the separation of T-tracks and Y-tracks would complicate the path finding algorithm too much, so the design was simplified so that the robot was constrained to only orthogonal axes of rotation. In addition, seeing as the quasi-sphere has a symmetrical structure, the team formulated a design that would only require one file to print, which improved the efficiency of the CAD and printing process. Once the specifications were finalized, the shell had reached a point where only minor modifications were required. Refer to the Results section for a detailed examination of the design iterations.

2. Electronics

An ESP32-C3-MINI is used to connect to the bots over Bluetooth and control the electropermanent magnets. A constraint with small microcontrollers and the ESP32-C3-MINI is the lack of General Purpose Input/Output (GPIO) pins, which are used to drive the EPMs. The solution to this was to use 4:16 demultiplexers to essentially create 16 GPIO with 5 GPIO pins (5 instead of 4 since a Latch Enable pin is necessary for zero output). The issue with this solution is that they only can have one output HIGH at a time. However, since electropermanent magnets are controlled with pulses of current, this is not a severe issue. Since the required current cannot be achieved through the GPIO output, a 20V charged capacitor is discharged to achieve the current level needed to turn on the electropermanent magnet. This capacitor is controlled by a half bridge. The GPIO pins are used to turn on and off MOSFETs which act as switches. An Adafruit BNO08x internal measurement unit (IMU) is also connected to the microcontroller. This connection is made through a standard UART connection which is supported by the microcontroller. The IMU outputs rotation vector data that a program can use to determine orientation of the robot.

Initial testing was done with commercial microcontroller boards, using an Arduino Uno to initially test logic, then a ESP32-C3-DevKitM-1 to test Bluetooth functionality. Additionally, this testing also characterizes the required current and pulse time to turn on the electropermanent magnet. In order to miniaturize the robots, the microcontroller and demux setup was custom designed onto a smaller board to save space while also providing all the needed functionality for Bluetooth, programming, and GPIO. The custom design was done through KiCad 7.0 and distributed and assembled through PCBWay. The custom PCB allows for a smaller design of the robots to be tested as a complete prototype.

3. Electropermanent Magnets

To reach this project's final goal, the method of actuation needed to be carefully chosen. Choosing electropermanent magnets was chiefly on the basis of power conservation within the unit, as the power supply for each robot was very limited due to their size. Electropermanent magnets work on the principle of electromagnetic induction and the use of magnets with very different coercivities. When power is pulsed through a wire which has been coiled around a permanent magnet, a magnetic field is generated which interacts with the preexisting magnetic field. If this generated magnetic field is strong enough to overcome the coercivity of the permanent magnet, then the generated field can override the permanent field to effectively switch the polarity of the magnet. The apparatus of an electropermanent magnet is designed in such a way that there are two magnets side by side, one of which has a much higher coercivity than the other. This way, when the current is pulsed, only one of the magnets will change polarity relative to the other. With this, the magnetic poles of each magnet can either be aligned or the reverse of each other and result in either a net magnetic flux through the end caps, turning it on, or a nearly net zero flux, turning it off as shown in Figure 8. The hysteresis diagram of this process in Figure 8 shows this cycle, where in the first step the magnets are oppositely polarized is no net magnetic flux. In the following steps, power is pulsed through the coil, generating a magnetic field strong enough to flip the polarity of the lower coercivity magnet and align the two magnets.

The magnets that were used in the assemblies were NIB magnets with a coercivity of 1E6 A/m and Alnico Grade 5 magnets with a coercivity of 4.8E4 A/m [12], allowing the Alnico magnet to modulate while the NIB field remains constant.

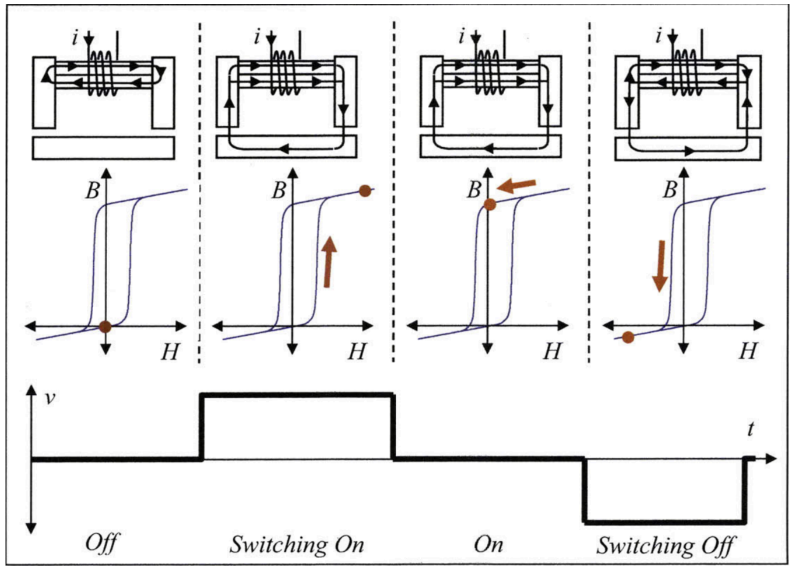


Figure 8. Modulation of magnetic circuit.

Figure 9. Initial version of EPMs.

After testing with these hand made assemblies, it became apparent that the results were inconsistent with themselves, most likely due to imperfections in the construction and the grade of the material that were used. The magnet rods were slightly different lengths, the end piece cuts were not flush with their end pieces, and the epoxy holding the rods in place was uneven, causing the whole construction to tilt slightly and have uneven contact with both the magnets and the surface, as shown in Figure 9. To combat this, new pieces were machined with dremels from larger rods and pieces of metal, which would allow us to more accurately control the alignment of the measurements.

These newly machined designs were tested with multiple different diameters and lengths, ranging from a half inch to one inch in length and from a quarter inch to three quarters inch in diameter. Different lengths and diameters change the amount of coil turns that can be wrapped around the magnets, as well as affecting the weight of the assembly and its physical dimensions. The amount of coils would affect the strength of the generated magnetic field, while the size and weight would affect the internal layout of the robot and most importantly the total mass.

B. Programming

1. Assembly Algorithm

Firstly, a decision between local and global programming style needs to be made. The intended applications of this project are to assemble into a specific structure on command, which would require the user to give that structure as input to the system. A global programming style makes the most sense for this, as it is intuitive and simple compared to a local style.

Next, it is important to understand the limitations of the system. As mentioned in the section on structure, the shell of the bots is constructed so that the bot can change orientations like a cube, but with a transitional track between faces. Using this construction limits movement to be purely along cardinal directions with the potential to stop or pause halfway through a movement. For simplicity, and since it does not match neatly with other bot configurations, standing on transitional tracks is not considered to be a valid position. So, the algorithm is left to consider agents that move a constant distance in cardinal directions.

This allows for a simplified viewing of the bots and their movement as a two-dimensional grid, in which bots occupy a single space and move in cardinal directions. Similarly, this allows for a simple way to write the code of the algorithm and representation of the bots, as two-dimensional arrays are very easy to use. While other data structures may be faster, budget and time constraints imply that any practical use of this code will be on small - less than five - numbers of bots within small areas - less than five bots of length in all dimensions.

Given this representation of the bots, an algorithm then needs to decide which start and end points should be paired and determine paths between those points that satisfy both requirements of shortest path and no collisions. One way to approach this solution is to first pair start and end positions by calculating the distance between each pair of starting and ending points, and then to choose pairs such that each start and end point is present in exactly one pair but the sum of distances related to those pairs is minimal. This also prevents swapping conflicts, as a situation resulting in two bots trying to switch positions implies that they are each closer to the other's end position than where their current path leads.

Given start and end positions, many shortest path algorithms would suffice, although collision-free algorithms are harder to find. As this application is for very few bots, sparsely placed in a small area, brute force of calculating all paths and checking each combination of paths per bot until a collision-free set is found will suffice in the majority of practical cases. An example of the algorithm's resulting paths can be seen below.

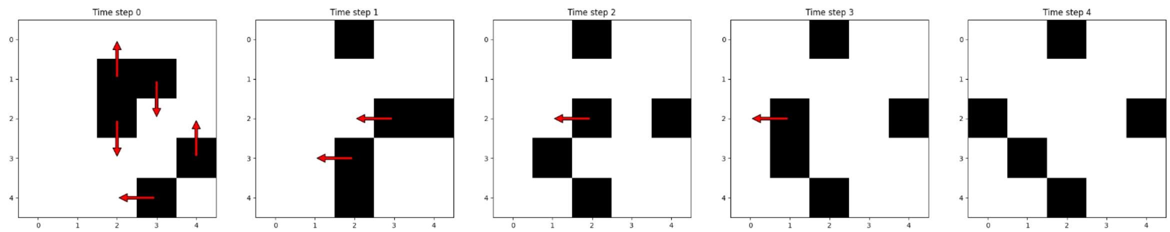


Figure 10. An example of the algorithm moving bots from starting positions to ending positions through four time steps of movement. Black squares indicate bots, and red arrows indicate the next move for a given bot.

2. Networking

For the intended application of this project, a global style of programming is most logical. This allows for the swarm of bots to change its structure at the user’s command, to the user’s specification. Elements of local programming may offer advantages to have the swarm work independently and in extending this work to specific applications, but does not accomplish the primary goal of this project.

Regardless of programming style, this project requires wireless communication in order for the bots to have much use, as wires would severely limit range and become tangled. Using a global style means that this communication will be between individual bots and a master computer. This allows for the use of the computer as a powerful, outside observer which has knowledge of the position of every bot and the processing power to run the algorithm.

The potential methods for wireless communication are WiFi, Bluetooth, and Bluetooth Low Energy (BLE). The long range of WiFi is simply unneeded for this application, and the less intuitive setup process makes it less appealing. By comparison, BLE has the same range as Bluetooth while also having the lowest power requirements, which is invaluable when battery capacity is constrained by space and movement is expensive.

Another advantage of using BLE with an ESP32 chip is that not only is the setup relatively simple, but there are code files and libraries that handle almost everything concerning establishing a connection and communicating with an ESP32, which are publicly available as Arduino libraries for the chip. However, since the server is established on a computer instead of a chip, another method is needed to create that server. This is easily handled with the Bless Python package, which also includes example code for creating a server on the package's Github.

Now that a connection has been established between the bots and the server, it is now necessary to consider how the bots will get information from the server. Because of the reduced number of bots in this prototype system, a simpler solution will suffice. One such solution is to construct a single message for each instance of time, broadcast it to all the bots, and have them parse a specific portion of the message. Since the system handler already needs to arrange the bots in specific starting locations, it is not a significant increase in effort to assign each bot a unique ID number, alter the algorithm to handle those IDs, and compose the message from server to bot such that every byte of the message corresponds to a specific ID.

3. Movement Code

With this model of splitting code between computer and bots, the computer's responsibility is to determine, based on start and end positions, which directions a given start position should rotate to reach the ideal end position. However, the computer is not responsible for determining the correspondence between start position and specific bot or determining which magnets need to be activated for a specific rotation given a current orientation. Because this system does not feature absolute positioning capabilities on the bots, it is up to the system operators to position the robots in accordance with their start positions in the algorithm. Determining which magnets to activate also requires some human input, but only as the initial state of a recursive system.

To do so, each bot keeps track of its current orientation. Using the shell's likeness to a cube, that orientation can be tracked just as easily by tracking the position of two faces, offset by ninety degrees. Each possible orientation of the cube can be uniquely determined by those two faces, which will be named as reference faces. And, those faces can also be viewed as points on the unit sphere, where the center of the bot is the center of the sphere. The transformation of points along the unit sphere is very well understood, as it is modeled by three-dimensional rotation matrices, and since every complete movement is a ninety degree rotation, those matrices become very simple as theta is always $\frac{\pi}{2}$. From here, it is easy to write a simple but long conditional statement that converts current orientation and a requested movement to a pair of magnets to activate sequentially to complete the rotation. However, this inductive method of navigation requires a base case, which is to know the starting rotation. The current model accomplishes this by human input, requiring that when the bots are manually placed into their starting positions, they are also set in a specific orientation. In order to do this, the shell is physically marked so that the two reference faces are clearly distinguishable.

Utilizing the Adafruit BNO08x internal measurement unit (IMU) for a more precise orientation tracking method, the bot can send quaternion data, a four-dimensional complex number written as $q = a + bi + cj + dk$ where *a, b, c,* and *d* are real numbers and *i, j,* and *k* satisfy the condition $i^2 = j^2 = k^2 = ijk = -1$ [45]. These values can be useful for representing orientation or rotation of three-dimensional objects more precisely as they avoid the problem of gimbal lock which is the phenomenon of losing the ability to interpret rotation along one axis due to a previous rotation on another [46]. This becomes a huge problem for a system capable of a sequence of multiple rotations that at any point can cause gimbal lock, which simple rotation matrices can't account for. After obtaining quaternion outputs from the IMU, the initial step is to establish a reference frame by defining the three axes based on the IMU's definition. These axes are then initially aligned with the corresponding faces of the bot, providing a consistent and static frame of reference for orientation tracking. This can be done with a set of 3 points represented by rotation vectors: $x = [1, 0, 0]$, $y = [0, 1, 0]$, and $z = [0, 0, 1]$. With the axes defined, the quaternion outputs are adjusted to zero out the *i, j,* and *k* components by orienting the bot until these values align with three faces on the bot as they would on the unit sphere. Subsequently, two vectors are selected to serve as the basis for all movements, corresponding to the two faces mentioned earlier, offset by ninety degrees.

Given a point representation p of one of the faces, a rotation can be modeled in a given direction by multiplying by the quaternion corresponding to that direction to get a new point representation of the face's expected position, $q * p * q^{-1}$. The actual points obtained from the IMU after a rotation are compared with the expected points to assess the accuracy of the rotation. If the points are relatively close, the rotation is deemed successful; otherwise, discrepancies indicate issues that may need to be addressed, such as calibration or hardware adjustments. The resulting points after rotation are stored for future reference, ensuring continuity in the navigation process. By integrating orientation comprehension capabilities, this systematic approach enables precise control of movement, facilitating efficient and accurate operation in various situations.

C. System Integration

The robot was built and integrated on a much larger scale in order to test proof of concept with easily attainable off the shelf components. The off the shelf solenoids were hot glued into the holes of the 3D printed shell with a diameter of around 4.3 inches. The tip of the hot glue gun was used to slightly expand the holes in the PLA shell in order to press the solenoids into the shell.

An Arduino Uno was used as the microcontroller. The Arduino was wired to the NTE4514B demultiplexer pressed in a 3.25” x 3.25” breadboard using jumper wires. A series of bipolar junction transistors (BJTs) were connected to the outputs of the demultiplexer which were switched on at specific times. One end of each solenoid was connected to the emitter of the transistor while the other side was grounded. A 5V source from the Arduino was wired to the collector end of the transistor and each output of the demultiplexer was connected to a base terminal of a BJT. When the demultiplexer sent the output signal, the 5V at the connector would power the solenoid. Only five solenoids were integrated because those were the ones available at the time, and it was unnecessary to add more for the purpose of this test. The breadboard circuit was connected to a laptop for power and serial communication, inserted into the center and enclosed in the robot shell. The final integrated robot is shown below.

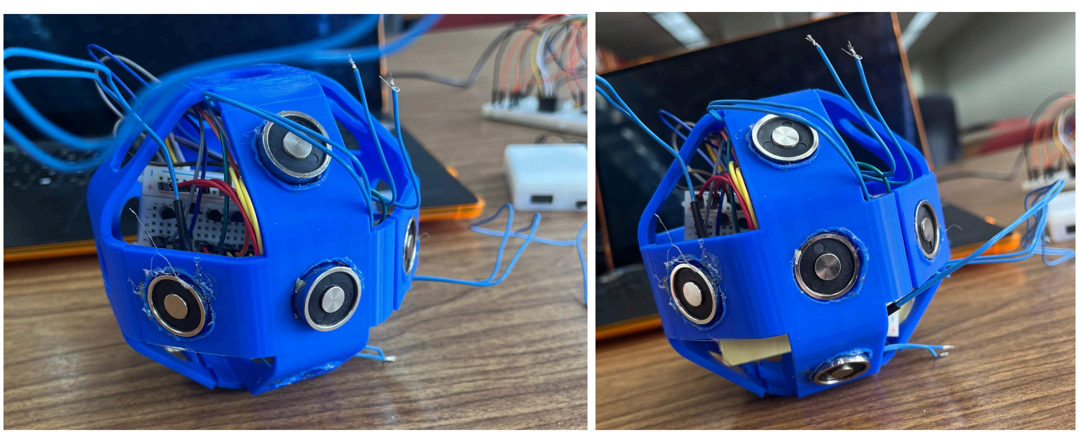


Figure 11. Final assembly of integrated robot.

IV. Results

A. Structure

The progression of the 3D printed structure can be followed in the figures below.

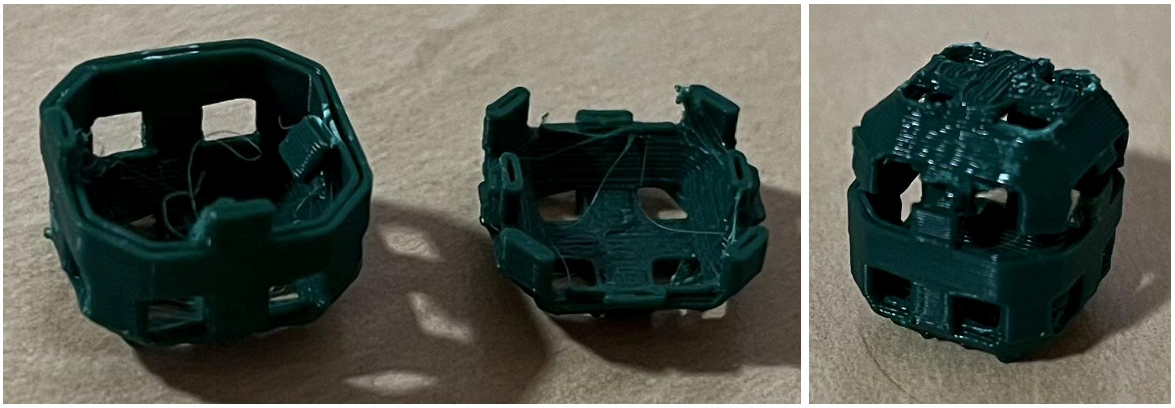


Figure 12. First iteration of the 3D printed structure, separated into its top and bottom halves (left) and assembled (right).

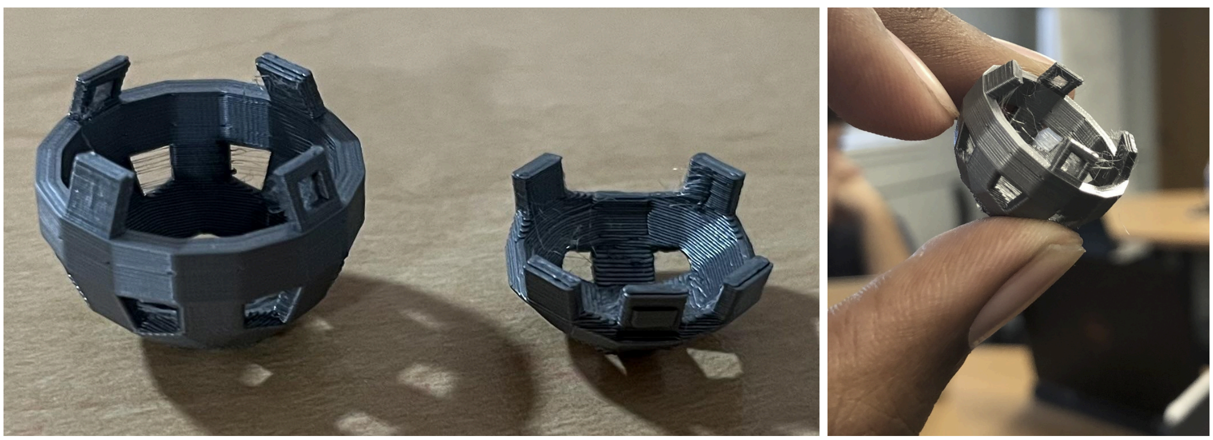


Figure 13. Second iteration of the 3D printed structure, characterized by its more spherical shape. Size comparison with human fingers shown on the right.

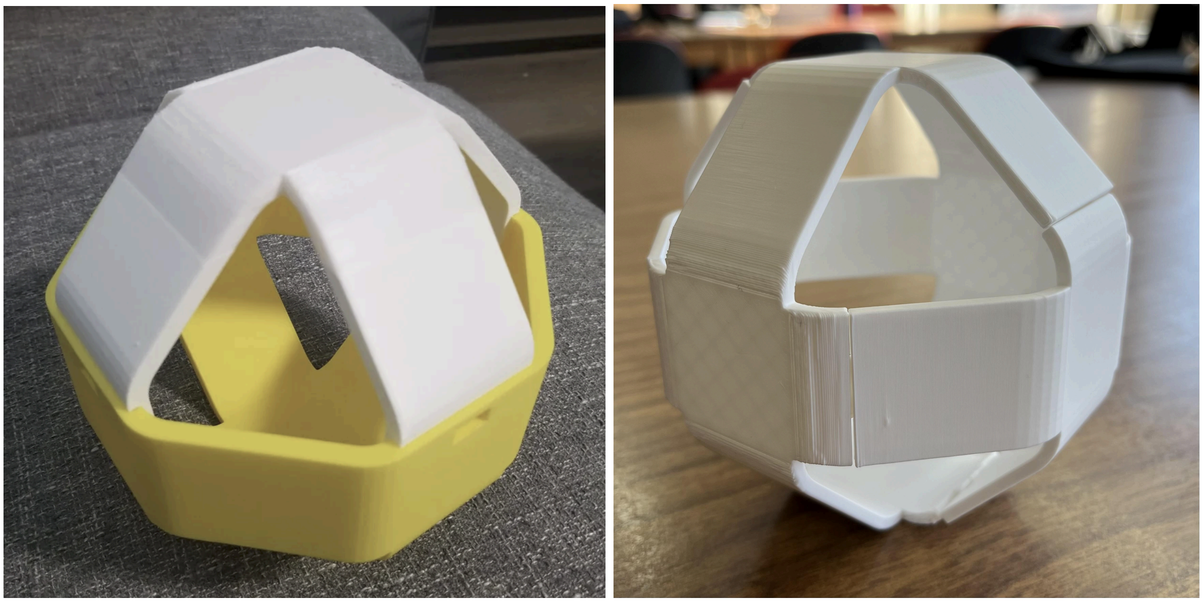


Figure 14. Larger version of the structure, excluding the Y-tracks for movement simplicity.

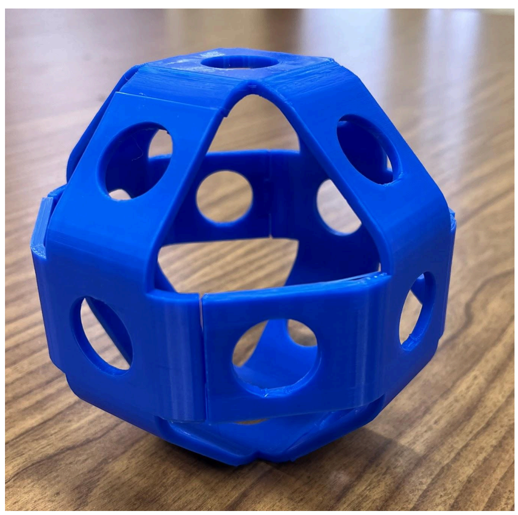


Figure 15. Structure including holes for the magnets.

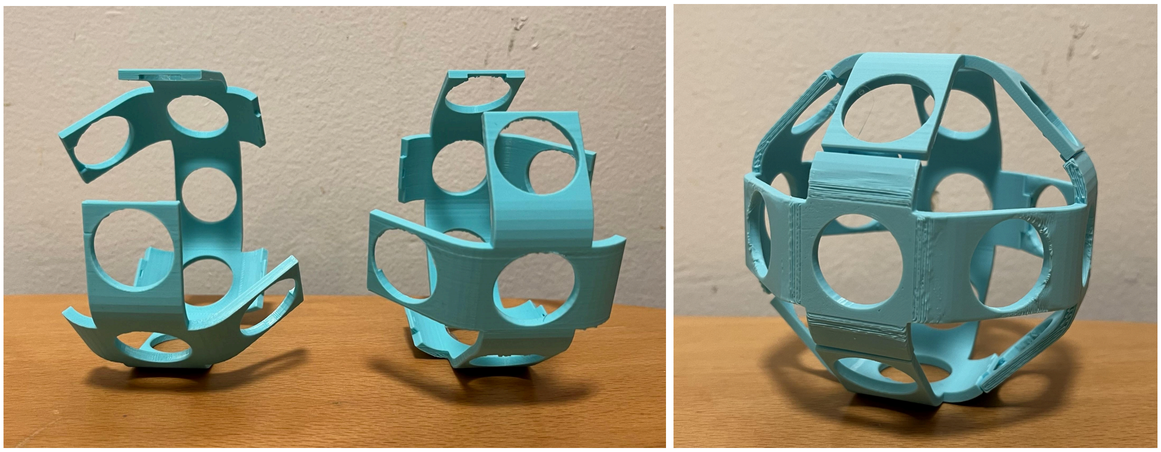


Figure 16. Structure with larger holes for better latching force, now with a different design that would only require one file to be sent to the 3D printer for better printing efficiency.

B. Electronics

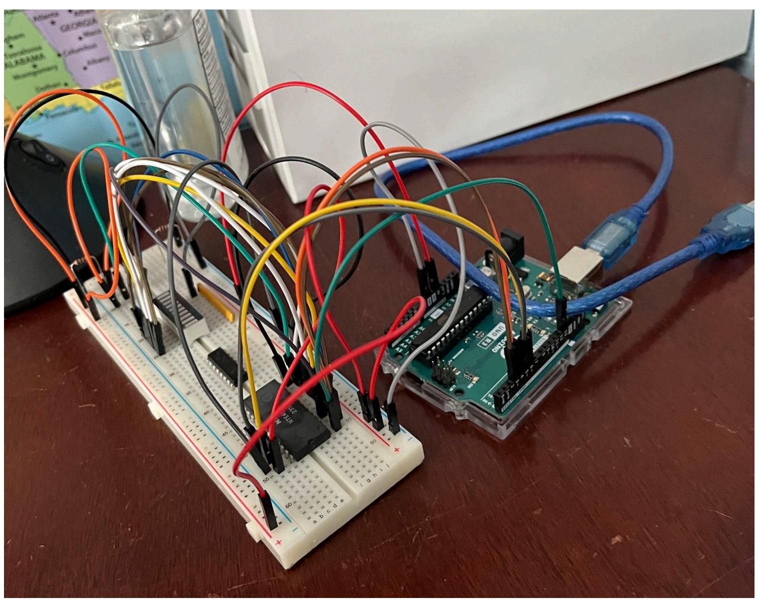


Figure 17. Arduino setup for testing output extension system with NTE4514B demultiplexer.

Initial component tests with a 4:16 demultiplexer and Arduino Uno confirmed the functionality of the system to extend the number of output pins for a microcontroller. Using a demux, the number of output pins could be increased to 16 using 5 GPIO pins from the microcontroller. Even though there are only 4 select pins for the demultiplexer, another GPIO pin must be used since there is a Latch Enable (LE) pin. In order to have no outputs and control pulsing, LE must be set to LOW.

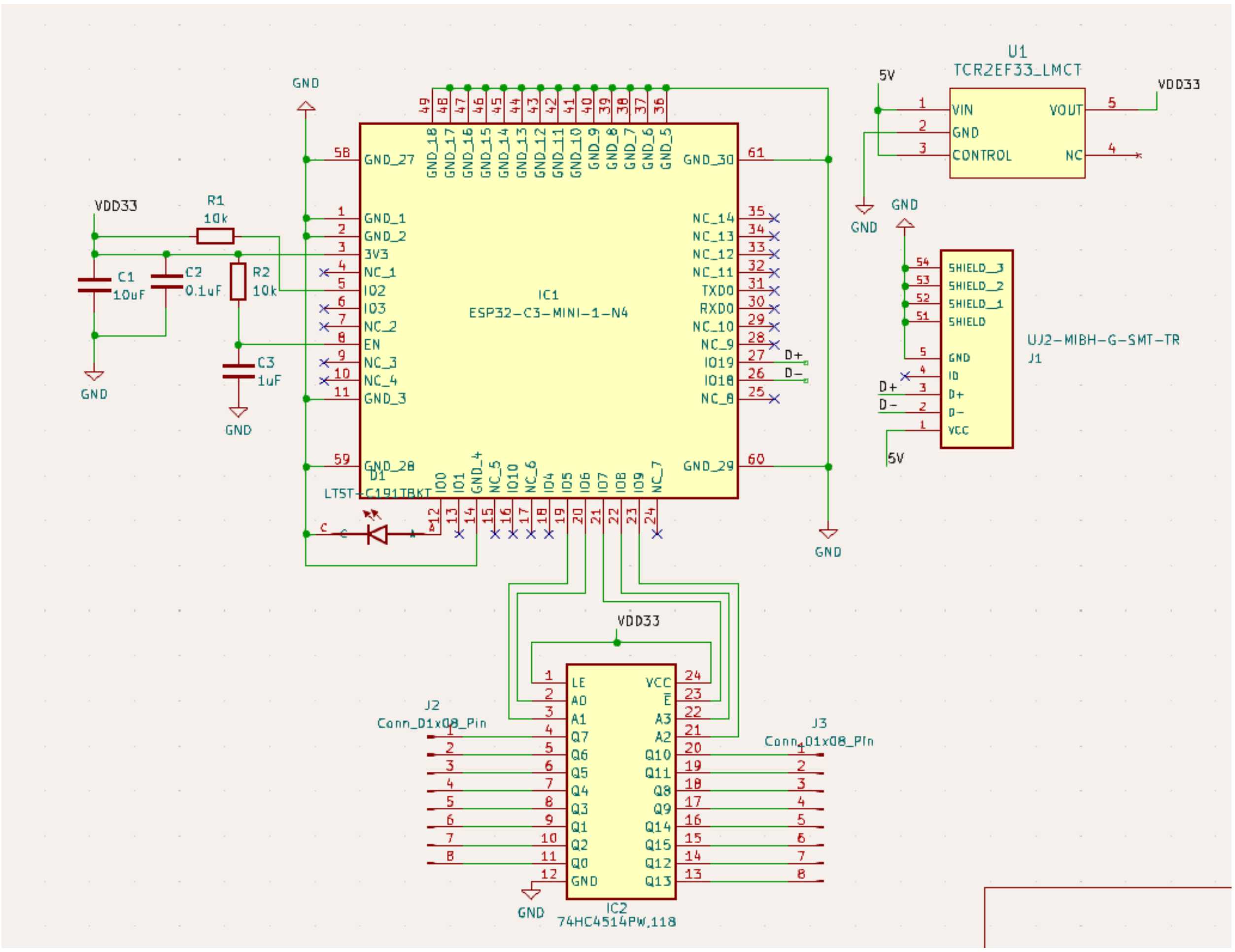


Figure 18. KiCad Schematic of custom PCB.

The design of the custom PCB uses the minimum component requirements to have the capabilities of programming, Bluetooth, and GPIO control. The design process starts with creating the schematic where a system of resistors and capacitors (all 603 footprint) is connected on the left side of Figure 18 to 3V3 and EN to allow for proper boot up process of the microcontroller when powered on. The components on the bottom side of the schematic are a

Blue LED (603) for programming testing and a 74HCA514PW, 188 Demultiplexer (15.5mm x 7.5mm) connected to 2 1x8 2.54mm spaced male connectors for increasing GPIO pins. This is powered and programmed through a USB micro B. Since USB operates a 5V a 3.3V linear regulator (TCR2EF33LM) is used to step down the voltage for operable conditions for the microcontroller and demux.

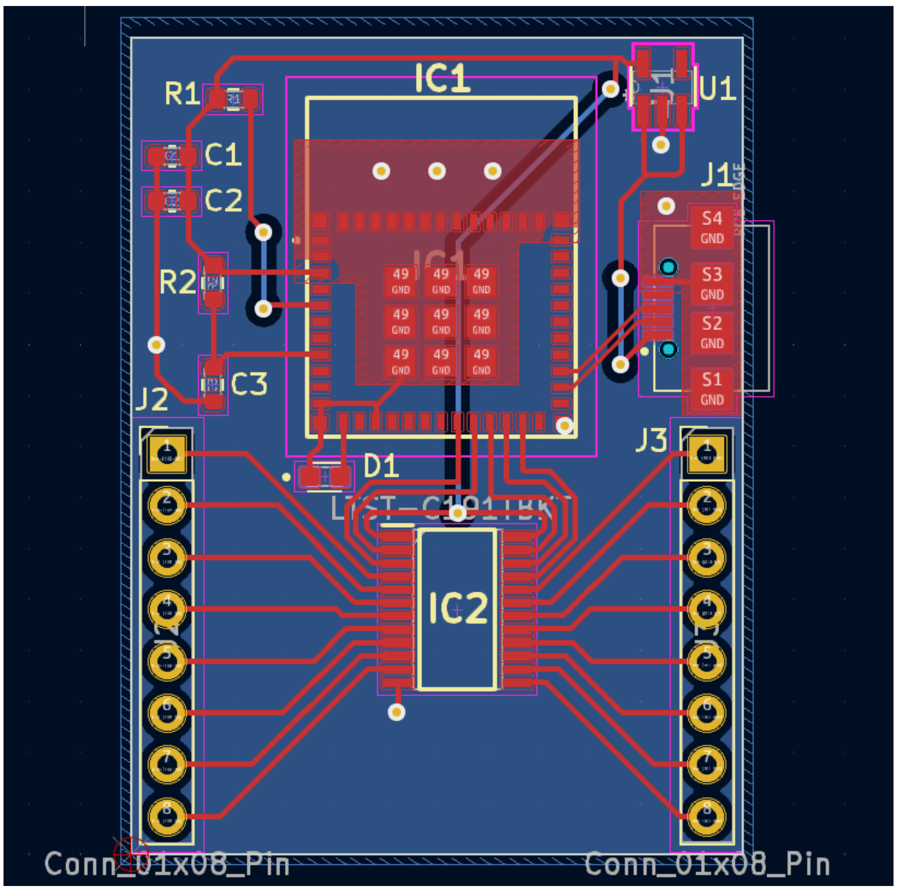


Figure 19. KiCad PCB Layout of custom PCB.

After the schematic is set and libraries are uploaded with footprints of all of the components, the PCB design process starts. The final custom PCB design is a 2 sided 40mm x 30mm PCB. Components are placed on one side due to manufacturing costs and lead times of PCBs with components on both sides.

C. EPMS

Figure 20. Magnetic rod while being machined.

The first step conducted was theoretical calculations derived from the MIT Robot Pebbles thesis [7]. One of the first values that needed to be calculated was the necessary current to switch the polarity of the magnet assembly. Equations 1 and 2 were used to calculate the maximum current within the wire and the switching voltage, where B_{mag} and H_{mag} denote the magnetic flux density and magnetic field strength at saturation respectively, R is the resistance of the wire, and P_{leak} is the leakage permeance of the magnet assembly. H_{mag} is a property of the magnet's material composition and follows a hysteresis diagram, with a value of approximately 200 kA/m for AlNiCo magnets. N is the number of wire turns in the coil, a and b are the face dimensions of the end cap where it makes contact with the latching target, μ_0 is the permeability of free space constant, and B_r is the remanence of the magnets.

$$I_{max} = \frac{H_{mag}L}{N} + \frac{\pi d^2 N_{rods}\left(B_{mag} + B_r + \mu_0 H_{mag}\right)}{8N\left(\frac{\mu_0 ab}{2g} + \mathcal{P}_{leak}\right)}$$

Equation 1 [12]

$$V(t) = N\left(\frac{dB_{alnico}}{dt} + \frac{dB_{NIB}}{dt}\right)\frac{\pi d^2 N_{rods}}{8} + I(t)R$$

Equation 2 [12]

However, the equation for switching voltage does not need to be numerically solved. It can be seen visually from the equation that it is a combination of voltages from Faraday's Law and Ohm's Law [12], and additionally, it can be surmised that the minimum voltage necessary to switch the magnet can be obtained from an analysis of the steady state value for switching voltage obtained by setting the transient terms to 0. This gives us a simple analysis of Ohm's law using the value of I_{max} and R, the resistance of the wire, to give us the minimum necessary voltage to switch the magnet polarity after an arbitrary amount of time for the transient to dissipate [12]. With only 50 turns of the coil I_{max} is about 101.6 A for an AlNiCo magnet. With an estimated resistance over the entire coil of about 21 mOhms, this leaves us with a numerical value of 2V. This I_{max} value is not something that could be produced, but if a magnet with a smaller length and many more turns could be produced, the I_{max} value decreases according to the first term of the equation only, assuming the air gap distance is 0.

One of the other important values to calculate for this project was the necessary force to tip the robot given a certain air gap distance between two EPMs on two different robots. To compute this, Equation 3 was used where L is the length of the magnetic rods,, $H_m(t)$ is the axial magnetic field intensity within the magnets, $I(t)$ is the current in the wire, and g is the air gap distance.

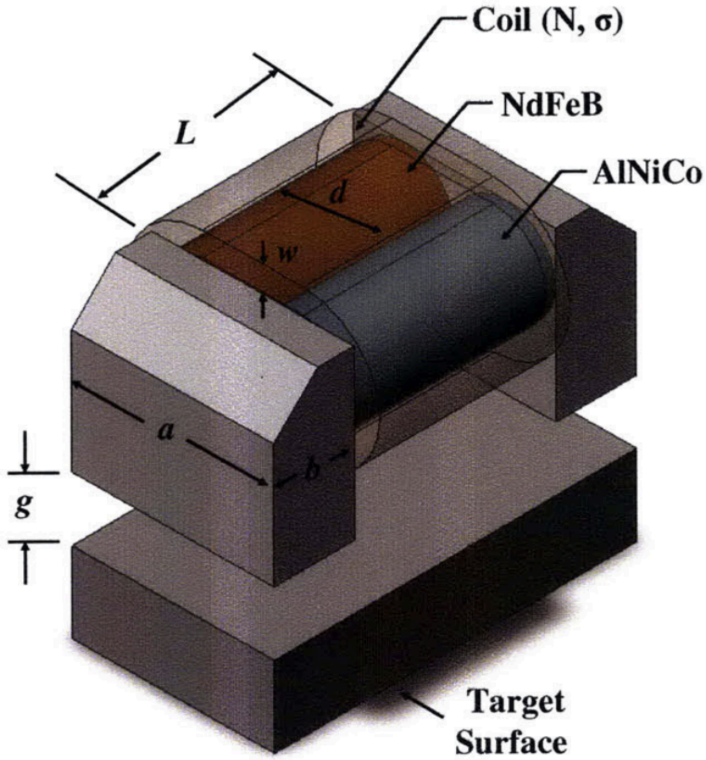


Figure 21. Overview of EPM assembly and physical variables.

$$F = \mu_0 \frac{ab}{4}\left(\frac{NI(t) - H_m(t)L}{g}\right)^2$$

Equation 3 [12]

The axial magnetic field intensity H_m will be defined as H_{mag} here since the magnet will be at saturation point. Using this equation with a 2 A current, 50 turns, an *ab* value of 0.125 cm^2, and length of 2.54 cm and an air gap distance of about 1.6 cm found through a trigonometric solution of the furthest the magnet would be from the ground, the calculated force exerted by the assembly was found to be about 3.8 N.

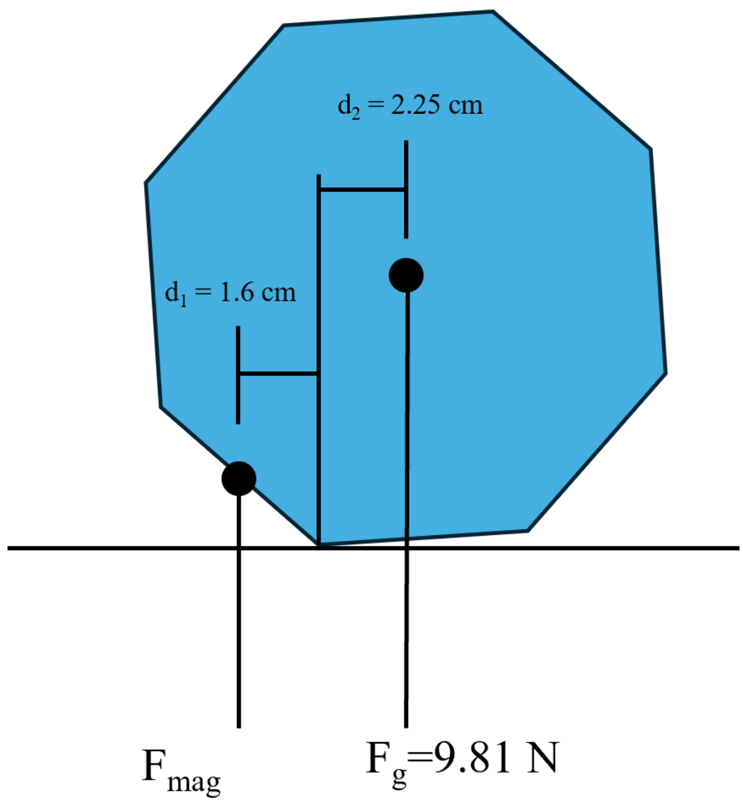


Figure 22. Single robot free body diagram for tipping.

The necessary force to cause the assembly to tip can be found by a simple free body diagram analysis of the moments caused by gravity and magnets about the center of the robot using Equation 4.

$$F_{mag} d_1 = F_g d_2 \quad \textbf{Equation 4}$$

Given the robot's mass of about 1 kg centered 2.25 cm from the tipping point and a magnet force centered about 1.6 cm from the tipping point, the minimum force required to tip the robot is 13.79 N. This is larger than what the magnets are currently capable of producing, so the current magnet assemblies are incapable of causing the system to roll in its current, larger state.

The most reasonable way to increase the force output is to increase the current and/or the number of wire turns in the magnets. However, this does not solve all of the issues. After testing with these hand-made assemblies, the results appeared to be inconsistent, most likely due to imperfections in the construction, methods, and material quality undertaken throughout the process. Due to the physical alignment of domains within magnetic materials, attempts to physically cut these materials causes them to chip and flake, making it very difficult to achieve a clean cut. Additionally, heat effects from friction while cutting can cause imperfections in the material, which can negatively affect the strength of the magnetic field being generated by causing misalignment of the domains at the edges of the cut. Another issue is that although the holding force of the assembly can be increased or decreased, the EPM was unable to be fully demagnetized and re-magnetized. From this testing, it is concluded that aside from construction defects in the assembly, another reason for subpar EPM performance is due to the current in the coil being insufficient to generate a powerful enough magnetic field.

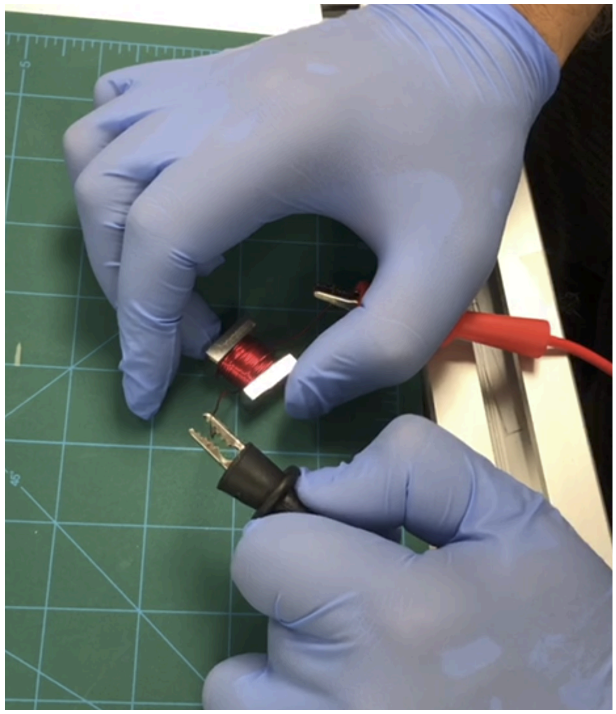


Figure 23. Testing of EPM magnetic field output at different currents and pulse lengths.

There are numerous ways this issue could be addressed. In Figure 23, the first iteration of the assembly is shown. In this assembly, the coil is wrapped around both magnets. While this was initially done to meet construction constraints, it also increases the total length of wire used, which increases the power loss over the wire and thus decreases the strength of the generated magnetic field. Additionally, this means that if the higher coercivity magnet has a lower coercivity than intended due to material impurities, the magnetic field could influence both magnets and thus not have the desired effect on the assembly. Wrapping the coil exclusively around the lower coercivity magnet would remove the possibility of interference with the wrong magnet.

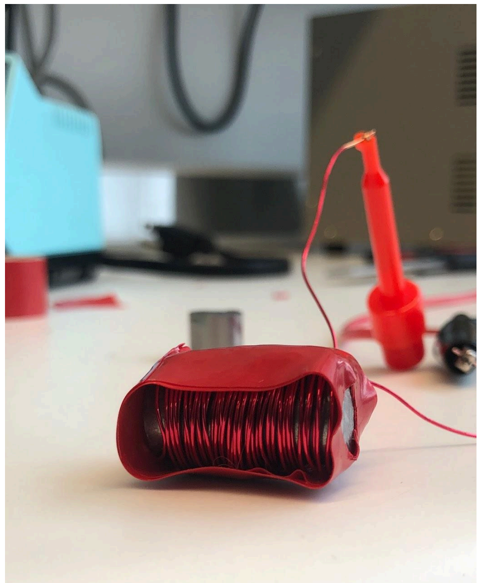


Figure 24. Construction and testing of solenoid (electromagnetic actuator).

Solenoid prototypes were constructed as an alternative to electropermanent magnets due to the EPM assemblies being unable to generate the required rolling force. The prototype shown in Figure 24 consisted of 45 coils of copper wire around an iron rod with a diameter of 0.5” and a length of 1”. However, a similar lack of magnetic force when separated was observed, due to the small number of coils that could be achieved by hand. These kinds of magnets were able to generate enough force for latching when two were touched together; however, they were not strong enough to pull the bots together from a distance.

D. Programming

The code used is publicly available at https://github.com/team-sand/sand.

The IMU was successful in outputting orientation information. The figures below demonstrate attempts at 90 degree rotations in the negative direction around the x-axis for the points initialized as [0 1 0] and [0 0 1]. Important to note that the imu is not integrated with actual bot units. This is purely testing the imu and analyzing the outputs.

```
Received: r: 0.71 i: 0.71 j: 0.02 k: -0.01
Actual vectors:
av1:  [4.22325766e-02 2.97412511e-04 9.99107762e-01]
av2:  [ 1.40775255e-02 -9.99900862e-01 -2.97412511e-04]
Expected vectors:
Point 1:  [0, 0, 1]
Point 2:  [0, -1, 0]
```

Figure 25. -90 degree rotated state of IMU. Actual and expected vectors closely match.

The actual vectors obtained from the quaternion *0.71 + 0.71i + 0.02j - 0.01k* are relatively close to the expected points [0 0 1] and [0 -1 0] within a margin of error of about .01 radians, or about half a degree, indicating a good 90 degree rotation about the x-axis.

```
Received: r: 0.99 i: 0.14 j: 0.00 k: 0.09
BAD! rotation difference:
 53.18  degrees along the  x gimbal
Actual vectors:
Point 1:  [-0.18, 0.95, 0.28]
Point 2:  [0.03, -0.28, 0.96]
```

Figure 26. Rotated about 53 degrees, is not 90 degrees, thus failure.

Here the actual vectors obtained from the quaternion 0.99 + 0.14i + 0j + 0.09k are 53.18 degrees from the required 90 degrees rotation around the x-axis. This may indicate some hardware or environmental problems impacting rotation functionality of the bot.

E. Integration and Testing

The Arduino was uploaded with code that instructed the demultiplexer to turn on the first two solenoids in the robot to create movement. The integrated robot was placed on a metallic surface to allow the solenoids to magnetically connect to the ground. Note that although Figure 11 displays the robot resting on a wooden surface, the tests were conducted on a separate metallic surface. The goal of the test is visually depicted in Figure 27.

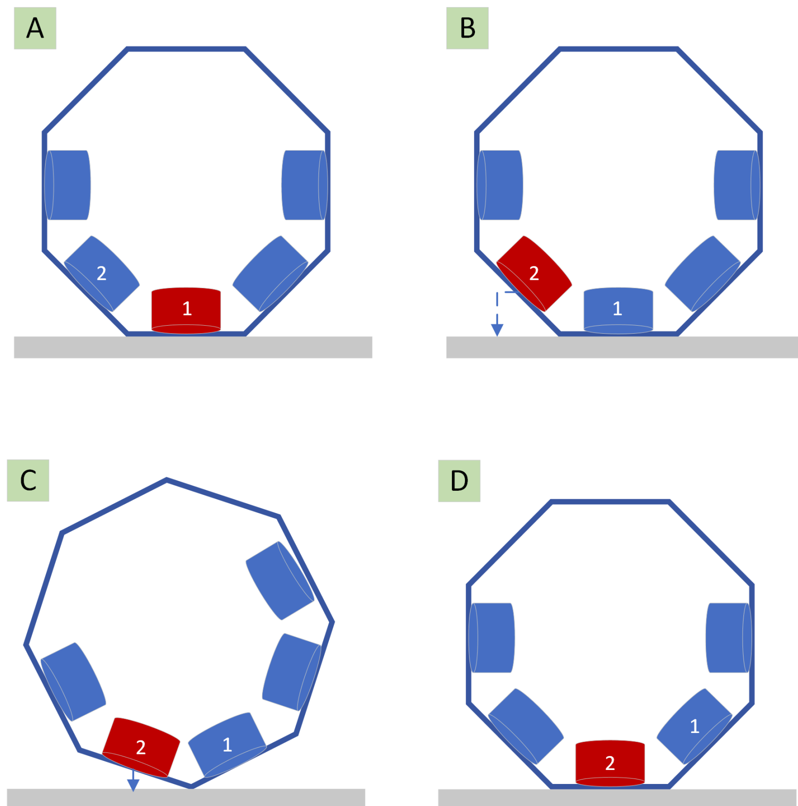


Figure 27. An example of two solenoids actuating the robot by causing tipping and thus rotation. Starts in the top left (A) and ends at the bottom right (D). Octagons represent a cross-section of the robot shell, and short cylinders represent solenoids. Red solenoid means turned on and magnetized while arrows represent where the solenoid tries to magnetically latch to.

The results showed that the circuit and demultiplexer succeeded in turning the solenoids on and off. When the demultiplexer turned on a solenoid, the robot was latched to the surface and could not be pulled away without significant force. However, when the demultiplexer turned the solenoid off, the robot could be easily pulled away from the surface. In the case where the solenoid transitioned from on to off, there was some residual latching, thus the robot needed to be gently pulled away from the surface, otherwise it remained lightly latched.

However, the results also showed that there was no movement from the robot. Although the circuit and solenoids were successful in latching, the force of the magnets was not large enough to cause the robot to tip, which is needed for actuation.

When it was realized that the magnets were not strong enough, a second small test was conducted. A separate solenoid not attached to the robot was held approximately 0.5 cm away from the solenoid attached to the robot to observe if the robot could tip with a significantly smaller gap between the magnets. A diagram of this experiment is shown below.

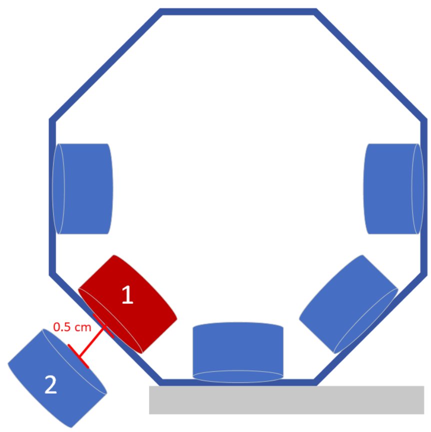


Figure 28. An example of holding a detached solenoid (2) close to an attached solenoid (1) approximately 0.5 cm away from each other. Not drawn to scale.

Despite the distance becoming significantly smaller, the solenoids still did not have enough force to cause the robot to tip and latch. It was only when the detached solenoid was slightly touching the attached solenoid that the robot was able to latch. Once the solenoids latched together, a slight tug on the detached solenoid showed that the magnets had enough force to remain latched even while tipping. This indicates that the solenoids have sufficient force to keep the robot latched to a surface but do not have sufficient force to cause tipping when there is a gap between the solenoid and latching surface.

V. Discussion

A. Structure

The structure of the robots is primarily a container for the internals, providing structure and shape in which the other aspects of this project can work. As such, the size and shape of the structure followed the changes and updates that arose with the group's expanding understanding of the needs of the project as time passed. Throughout these design changes, though, the general shape of the structure stayed fairly consistent. At first, the structure was small and complex, which reflects the team's lofty goals of making a very small-scale working prototype with nearly free movement in all directions. Then, as the team realized the complexity of the movement and the size of the electronics that would need to be housed within the structure, the design expanded and simplified accordingly. Finally, the structure included holes in the faces so that the magnets could lie flush with the outside of the shape, following the need for better latching force. In the future, the shape and structure of the robots will reflect the design requirements dictated by the other subsystems, whether that be increasing the number of faces for better movement capabilities or decreasing the overall size for better resolution in the self-assembled shapes.

B. Electronics

The final custom PCB demonstrates the miniaturization of a commercial microcontroller with Bluetooth capabilities on a smaller overall package while also increasing the number of output pins . The downside of the overall design is that it removes features like JTAG, but for the purpose of the project these features go unused and would only add to the component count and increase the size of the board. Further miniaturization is definitely possible with redesign of the board to a more efficient layout by using both sides of the board and smaller components, but the limiting factor for this was time constraints from manufacturers, their costs, and the programming of the specific microcontroller and pins.

While not considered to just focus on functionality, power usage is another important aspect that could be analyzed in the future for better designs. For small scale devices such as this project, power consumption is an important problem to tackle since batteries on this scale are restricted by voltage and amp hours. For this project choosing components that reduce current consumption would decrease the overall power usage and allow the robots to stay powered on longer. While no analysis was done within the constraints of this project, EPMs were chosen since the magnets can be controlled by pulses and don't require a constant current which would use less power in comparison to typical electromagnets.

For future development of the PCB design, the IMU can be integrated using the Tx and Rx pins on the microcontroller for UART communication. Additionally, creating individual boards for the electropermanent magnets with capacitors and transistor half bridges would allow for more compact components and better control over switching for the current pulses through the magnets. Rigid-flex PCBs would also make sense for the origami of the shape of the final robot for a more robust assembly process. This option was explored, but ultimately cost and time prevented further development on this option.

C. EPMs

The electropermanent magnets were intended to serve as the actuation mechanism for each robot. Although attempts to create powerful enough EPMs to generate the required force for rolling were ultimately unsuccessful, the lowered power requirements from this sort of assembly shows promise for future applications in similar technologies. A powerful enough latching force was achieved using solenoids; however, these were unable to generate enough force to roll the robots from a resting position. Additionally, useful insights into why the system did not work were discovered throughout the process. The required pulse current to switch the magnets in a short enough amount of time to be functional was too high with the current assembly design, and the size of both the individual assemblies and the robot itself was too large. If the mass and size of both the robots and the EPM assemblies could be reduced while still maintaining the amount of wire turns in the magnets, the robots could be actuated fully with EPMs.

In the future, it may be beneficial to use a wire threader for the manufacturing process of EPMs and solenoids. Both magnet types require hundreds of coils tightly wrapped around the magnetic or ferromagnetic rods in order to produce a sufficient amount of magnetic force for actuation. The coil wrapping process is cumbersome, and the number of coils is not maximized when wrapping is performed manually. Thus, having a device that automates the process will increase efficiency and quality of the EPMs and solenoids.

Collection of quantitative data was an important goal for this aspect of the project, but the many issues that were encountered encouraged the team to focus more on working to improve the prototypes than collecting data on the efficacy of an assembly whose lack of success could already be observed qualitatively. Moving forward, it would be beneficial to 1) perform more tests on the magnetic assemblies after finalizing prototypes and manufacturing processes and 2) perform statistical analysis to get a better idea of how strong of a magnetic field they can generate consistently and whether they match the expected output from the equations in previous sections to help diagnose any additional areas for improvement.

D. Programming

The BLE and movement code primarily serve to translate between the assembly algorithm and the bot's physical movements. As such, as long as they function as necessary, there is not much to optimize that falls within the scope of this project.

The final version of the assembly algorithm does not entirely work as intended. In particular, it fails a handful of edge cases where no paths without collision can be found. For the use cases of this project, this is not an issue, as budget and time constraints naturally limit the density of bots, which naturally causes a drastic reduction in collisions. Because of the availability of MAPF algorithms that solve this problem, and the lack of necessity to update for this project's needs, this specific algorithm does not need to be developed further.

Since any effort to replicate or improve on this work is likely to want to upgrade or replace the algorithm regardless, future works can also include choosing algorithms that solve more difficult problems and to adapt the rest of the system to overcome those respective challenges. For example, common MAPF challenges such as adding obstacles into the assembly area, increased density of bots, and adding a third dimension can be considered. In fact, adding a third dimension was an original goal of this project, and given more time, would be one of the next considerations. Switching to three dimensions introduces a massive amount of complexity, as the MAPF problem is typically considered in two dimensions, whereas this modification introduces a particularly challenging set of constraints for movement. Here, a bot can only adjust its z-axis position by climbing on top of stationary bots, and across any xy-plane other than the ground, movement is restricted to traversing across the bodies of other bots.

Another potential adaptation worth considering is changing the possible directions for a bot to move. Updating the structure of the bot to allow movement forty-five degrees between cardinal directions or allow half rotations of only forty-five degrees would greatly increase the capabilities of the swarm and allow for much more complex movement and structures. However, this would require very careful consideration as the bots move in discrete fractions of their circumference, and movement along forty-five degree angles would no longer align to the original grid.

A third task, which is more important to creating a complete system, would be to also specify a protocol for disassembly. However, it is very likely that this can be accomplished simply by running the assembly algorithm, starting from a more dense structure and ending with a sparse structure where bots are separate and can be collected by the system operator. This problem may become more complex when adding a third dimension and when the completed system is rotated in ways so that assembly may be difficult. For example, a structure with a wide base, flipped vertically to have a large overhang, may have a difficult time simply following the assembly algorithm backwards.

Something to note about the IMU is the limitation in accuracy of orientation data. The IMU is capable of producing more precise quaternion data but this requires an I2C connection that the corresponding library can interact with. The problem is this requires a large amount of memory, of which the prototype was incapable of handling. Thus, this project was limited to UART RVC that only produced yaw, pitch, roll, and accelerometer information, which was converted into quaternions and operated upon. Utilizing an external memory storage device like flash memory would allow for highly accurate information to be derived from the IMU. In addition, with more accurate information, a future project could utilize both the information of the bots intended path and orientation data could account for over rotations or rotations in the wrong direction by reorienting itself back on the intended path. This functionality could be integrated with the movement code.

E. Integration and Testing

In the EPM section, smaller, custom EPMs and solenoids were manufactured and tested, whereas with system integration, off the shelf solenoids rated for the proper holding force were tested. The experiments demonstrated that even with a solenoid with overly sufficient latching force, it still cannot overcome the tipping force needed for movement due to the air gap. This result was examined within the EPM section as well. Therefore, a priority in future studies should be to design a latching method that is strong enough to overcome the air gap barrier.

Optimization for this problem could also relate to the structure of the robot. Increasing the number of faces on the robot to create a more spherical shell would reduce the distance between a face with a solenoid and the surface which it must latch to, thus decreasing the air gap. This change in conjunction with a more effective latching method could solve the air gap problem.

Despite the solenoids having insufficient force to cause actuation, the results showed that they had sufficient force to maintain latching. This is a success because it means that the robot will be able to remain physically connected to a neighboring robot and therefore will allow the system of robots to maintain their desired shape/formation.

Additionally, the results demonstrated that the preliminary breadboard circuit prototype succeeded in its function to control the solenoids in a specified order and at a specified time. This means the logic of the electronics is effective and can be used moving forward.

One potential way to make actuation more feasible is to decrease the mass of the robots, thereby decreasing the force required to cause tipping. The total mass of the system is tabulated in Tables 1 and 2.

Item	Mass per Unit [g]	Qnty	Total Mass [g]
Half of PLA 3D Printed Shell (Dark Blue)	29.06	2	58.12
Solenoid	24.75	18	445.5
Battery Pack (4.5V)	100.32	1	100.32
BJT	0.17	18	3.06
Larger jumper wire M-M	0.87	18	15.66
Small jumper wire M-M	0.56	28	15.68
Devkit	6.89	1	6.89
Small Breadboard	36.82	2	73.64
Demux	3.88	1	3.88
Total			722.75

Table 1. Mass budget of the integrated system.

Subsystem	Total Mass [g]
Structure	58.12
Actuation	445.5
Electronics	219.13

Table 2. Mass budget of the system categorized by subsystems.

The total mass of the integrated assembly is 722.75 g. This is very large, especially since the goal is to downsize and minimize weight as much as possible. Clearly, the actuation subsystem has the highest mass at 445.5 g. This is because of the large individual mass and total quantity of the solenoids. Ideally, switching over to smaller, more compact EPMs will decrease this weight drastically. Additionally, the electronics subsystem has a mass of 219.13 g, but this can be lowered as well by using a PCB which is more compact and uses much smaller parts. Lastly, although structure contributed the least to the overall mass at 58.12 g, the weight can still be reduced by decreasing the wall thickness and shrinking the size of the faces to more closely fit the magnets.

In the eventual extension of the self-assembling robots to the third dimension, the robots need to be able to withstand the weight force of other robots resting on top. Therefore, a finite element analysis study was conducted to determine how much weight the structure could handle without failure due to yielding or excessive deflection (see Figure 29). This simulation is by no means perfect, as it assumes a 100% infill and ideal material properties. In addition, the analysis assumes that the size scale of the robots will remain unchanged, which goes against the eventual goal of the project to downscale the robots, but it provides a relative gauge for how strong the structure can be. Under 50 N of load, which corresponds to about 5 kg of weight resting on top of the shell the von Mises stress is well below the yield strength, with a safety factor of approximately 15. However, the displacement of the top face is a bit excessive, moving about 8.65 mm from its initial position. Considering the inner diameter of the shell is about 100 mm, this should be considered a failure. However, this should be more weight than the robots will individually encounter in normal applications, so these results are encouraging for future work and testing.

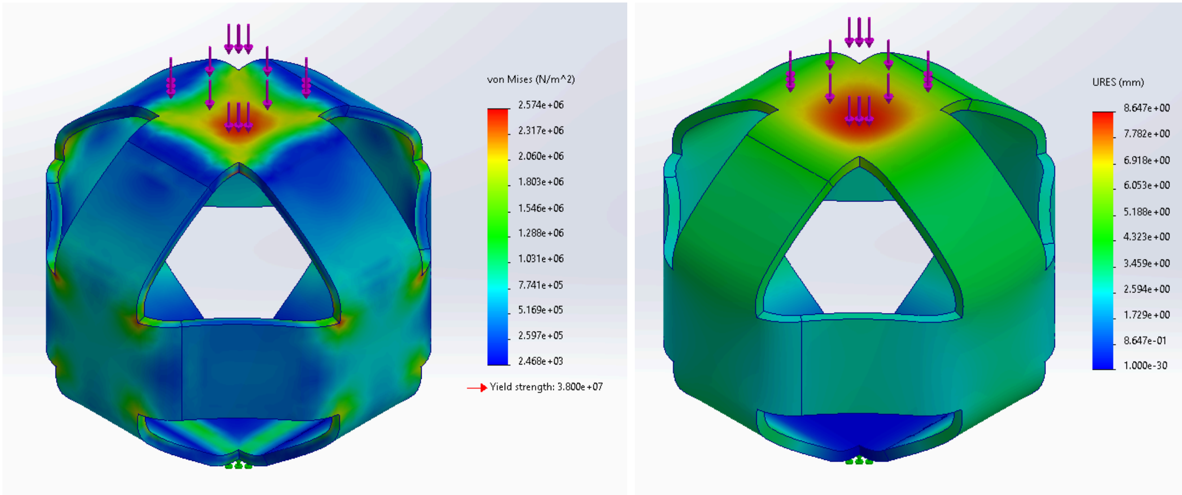


Figure 29. Von Mises stress (left) and displacement (right) FEAs of structure under 50 N loading conditions.

Conclusion

In the original iterations of the project goals, it was intended to create robots with sizes on the order of a few nanometers in any dimension. This ultimately proved infeasible due to the highly specialized equipment necessary for nanoscale manufacturing and testing and the limitations of the current technology. Photolithography equipment and other similar systems require specialized training, have large material and operation costs, and can sometimes require very long waiting periods to gain access to facilities or receive finished products. Time, budget, and niche materials all presented insurmountable barriers for the group, so the decision was made to pivot towards a larger scale system on the order of several centimeters in size that proved far more feasible to design, realize, and analyze.

This size scale led to equipment, materials and costs that were more realistic, such as 3D printers, electromagnets/EPMs, and preassembled microcontroller breakout boards, which ultimately led to more progress towards a working prototype than could have been achieved if nanoscale had still been pursued.

Overall, the project explored two main methods of actuation (EPMs and solenoids), determined advantages and disadvantages of the original quasi-sphere and modified quasi-sphere, verified circuit logic, and built upon prior pathfinding algorithms for swarm robots. Useful knowledge was gained on manufacturing of parts, selecting quality materials, calculations, pathfinding, and system integration, therefore contributing to the field of self-assembling programmable matter.

Glossary of Terms

Bluetooth Low Energy (BLE) - A Bluetooth-like wireless communication protocol, optimized for minimal power consumption without sacrificing transmission range.

Catom - The name of each individual robot in a swarm of robots specifically created by the Claytronics Project. The name is a portmanteau of "Claytronic" and "atom."

Demultiplexer - A circuit which selects one output based on a combination of selection inputs. Outputs correlate to the binary combination of the selection inputs.

Electropermanent Magnet (EPM) - The magnet created by the Robot Pebbles project. A magnetic circuit is formed using magnetic rods of very different coercivity wrapped in coil with two iron caps at the top and bottom of the rods. The magnet can be turned on and off with a pulse of current in opposite directions.

Global Programming Style - A style of programming a group of robots by sending instructions from one main computer to each robot.

H-Bridge - A circuit composed of four transistors that allow bidirectional flow of current over a load at different points in time. Typically used to drive motors clockwise and counterclockwise.

Inertial Measurement Unit (IMU) - An electronic device which measures one or a combination of either linear force, angular velocity, or magnetic strength. This is done through accelerometers, gyroscopes, and magnetometers respectively.

Local Programming Style - A style of programming a group of robots where all communication happens between robots of equal rank.

Quaternion - A mathematical representation of orientation or rotation in three dimensional space.

Gimbal Lock - A phenomenon where a system loses the ability to interpret rotation due to previous rotations.

Moore's Law - The statement made by Gordon Moore that every two years, the number of transistors in an integrated circuit will double [5].

Programmable Magnet (PRM) - The magnet created by the Electrovoxel project. An electropermanent magnet with added capabilities to reverse polarity.

Programmable Matter - In robotics, the concept of using small individual robots to make up a mass of swarm robots capable of assembling into various two-dimensional and three-dimensional geometries.

Quasi-sphere - The amalgamation of a cube and sphere into one shape. The Claytronics project deemed this to be the optimal shape of a robot within a programmable matter system.

Self-assembly - When individual robots within a swarm can still move independently, but work together to form two-dimensional and three-dimensional shapes.

Self-reconfigurable Robot - A robot with the capability to reshape itself.

Swarm Robotics - A branch of robotics focused on coordinating large numbers of simpler robots.

Universal Asynchronous Receiver/Transmitter (UART) - A communication protocol using two wires to receive and transmit data in both directions using an Rx and Tx pin, respectively.

References

[1] T. Toffoli and N. Margolus, “Programmable matter: concepts and realization,” in *Proceedings of the NATO Advanced Research Workshop on Lattice Gas Methods for PDE’s : Theory, Applications and Hardware: Theory, Applications and Hardware*, NLD: North-Holland Publishing Co., 1991, pp. 263–272.

[2] T. Fukuda and Y. Kawauchi, “Cellular robotic system (CEBOT) as one of the realization of self-organizing intelligent universal manipulator,” in *Proceedings., IEEE International Conference on Robotics and Automation*, 1990, pp. 662–667 vol.1. doi: 10.1109/ROBOT.1990.126059.

[3] E. Hawkes *et al.*, “Programmable matter by folding,” *Proc. Natl. Acad. Sci.*, vol. 107, no. 28, pp. 12441–12445, Jul. 2010, doi: 10.1073/pnas.0914069107.

[4] S. Martel, M. Mohammadi, O. Felfoul, Z. Lu, and P. Pouponneau, “Flagellated Magnetotactic Bacteria as Controlled MRI-trackable Propulsion and Steering Systems for Medical Nanorobots Operating in the Human Microvasculature,” *Int. J. Robot. Res.*, vol. 28, no. 4, pp. 571–582, Apr. 2009, doi: 10.1177/0278364908100924.

[5] “Moore’s law, n. meanings, etymology and more,” Oxford English Dictionary. Accessed: Apr. 15, 2024. [Online]. Available: https://www.oed.com/dictionary/moores-law_n

[6] S. C. Goldstein and T. Mowry, “Claytronics: A Scalable Basis For Future Robots,” p. 6.

[7] A. N. Knaian, “Electropermanent Magnetic Connectors and Actuators: Devices and Their Application in Programmable Matter,” p. 206.

[8] “Connecting and disconnecting for chain self-reconfiguration with PolyBot | IEEE Journals & Magazine | IEEE Xplore.” Accessed: Mar. 25, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/1159222

[9] E. H. Østergaard, K. Kassow, R. Beck, and H. H. Lund, “Design of the ATRON lattice-based self-reconfigurable robot,” *Auton. Robots*, vol. 21, no. 2, pp. 165–183, Sep. 2006, doi: 10.1007/s10514-006-8546-1.

[10] L. Hardesty, “Surprisingly simple scheme for self-assembling robots,” MIT News | Massachusetts Institute of Technology. Accessed: Mar. 17, 2024. [Online]. Available: https://news.mit.edu/2013/simple-scheme-for-self-assembling-robots-1004

[11] G. M. Whitesides and M. Boncheva, “Beyond molecules: Self-assembly of mesoscopic and macroscopic components,” *Proc. Natl. Acad. Sci.*, vol. 99, no. 8, p. 4769, Apr. 2002, doi: 10.1073/pnas.082065899.

[12] M. E. Karagozler, A. Thaker, S. C. Goldstein, and D. S. Ricketts, “Electrostatic actuation and control of micro robots using a post-processed high-voltage SOI CMOS chip,” in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, May 2011, pp. 2509–2512. doi: 10.1109/ISCAS.2011.5938114.

[13] A. Guin, “Programmable Matter - Claytronics,” 2012, p. 11.

[14] S. Murata and H. Kurokawa, “Self-reconfigurable robots,” *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 71–78, Mar. 2007, doi: 10.1109/MRA.2007.339607.

[15] M. Yim *et al.*, "Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics]," *Robot. Autom. Mag. IEEE*, vol. 14, pp. 43–52, Apr. 2007, doi: 10.1109/MRA.2007.339623.

[16] R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous self-assembly in swarm-bots," *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1115–1130, Dec. 2006, doi: 10.1109/TRO.2006.882919.

[17] S. Hauser, M. Mutlu, and A. J. Ijspeert, "Kubits: Solid-State Self-Reconfiguration With Programmable Magnets," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6443–6450, Oct. 2020, doi: 10.1109/LRA.2020.3013884.

[18] M. Nisser, L. Cheng, Y. Makaram, R. Suzuki, and S. Mueller, "ElectroVoxel: Electromagnetically Actuated Pivoting for Scalable Modular Self-Reconfigurable Robots," in *2022 International Conference on Robotics and Automation (ICRA)*, May 2022, pp. 4254–4260. doi: 10.1109/ICRA46639.2022.9811746.

[19] D. Hidalgo-Carvajal, Á. H. Muñoz, J. J. Garrido-González, R. Carrasco-Gallego, and V. Alcázar Montero, "Recycled PLA for 3D Printing: A Comparison of Recycled PLA Filaments from Waste of Different Origins after Repeated Cycles of Extrusion," *Polymers*, vol. 15, no. 17, Art. no. 17, Jan. 2023, doi: 10.3390/polym15173651.

[20] O. US EPA, "Recycling Infrastructure and Market Opportunities Map." Accessed: Feb. 26, 2024. [Online]. Available: https://www.epa.gov/circulareconomy/recycling-infrastructure-and-market-opportunities-map

[21] R. Catry, A. Mohand-Ousaid, M. Rakotondrabe, and P. Lutz, "Presentation, Modeling and Experiments of an Electrostatic Actuator Based Catom for Programmable Matter," *Actuators*, vol. 9, no. 2, Art. no. 2, Jun. 2020, doi: 10.3390/act9020043.

[22] J. Nielsen and H. H. Lund, "Modular robotics as a tool for education and entertainment," *Comput. Hum. Behav.*, vol. 24, no. 2, pp. 234–248, Mar. 2008, doi: 10.1016/j.chb.2007.01.011.

[23] A. Roudaut, D. Krusteva, M. McCoy, A. Karnik, K. Ramani, and S. Subramanian, "Cubimorph: Designing modular interactive devices," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3339–3345. doi: 10.1109/ICRA.2016.7487508.

[24] H. H. Lund, T. Klitbo, and C. Jessen, "Playware technology for physically activating play," *Artif. Life Robot.*, vol. 9, no. 4, pp. 165–174, Dec. 2005, doi: 10.1007/s10015-005-0350-z.

[25] D. Brandt, D. J. Christensen, and H. H. Lund, "ATRON Robots: Versatility from Self-Reconfigurable Modules," in *2007 International Conference on Mechatronics and Automation*, Aug. 2007, pp. 26–32. doi: 10.1109/ICMA.2007.4303511.

[26] B. Piranda and J. Bourgeois, "Designing a quasi-spherical module for a huge modular robot to create programmable matter," *Auton. Robots*, vol. 42, no. 8, pp. 1619–1633, Dec. 2018, doi: 10.1007/s10514-018-9710-0.

[27] *Motion capabilities of 3D Catoms*, (Feb. 15, 2021). Accessed: Oct. 07, 2021. [Online Video]. Available: https://www.youtube.com/watch?v=3R3o74Afx5Y

[28] K. Gilpin, K. Kotay, and D. Rus, "Miche: Modular Shape Formation by Self-Dissasembly," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 2241–2247. doi: 10.1109/ROBOT.2007.363653.

[29] M. E. Karagozler, S. C. Goldstein, and J. R. Reid, "Stress-driven MEMS assembly +

electrostatic forces = 1mm diameter robot," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009. doi: 10.1109/IROS.2009.5354049.
[30] K. Gilpin and D. Rus, "Modular Robot Systems," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 38–55, 2010, doi: 10.1109/MRA.2010.937859.
[31] M. P. Ashley-Rollman, S. C. Goldstein, P. Lee, T. C. Mowry, and P. Pillai, "Meld: A declarative approach to programming ensembles," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, Oct. 2007, pp. 2794–2800. doi: 10.1109/IROS.2007.4399480.
[32] R. Baumann, "A Survey on Routing Metrics," p. 53, Feb. 2007.
[33] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, Aug. 2014, doi: 10.1126/science.1254295.
[34] T. Schmickl and K. Crailsheim, "A Navigation Algorithm for Swarm Robotics Inspired by Slime Mold Aggregation," in *Swarm Robotics*, E. Şahin, W. M. Spears, and A. F. T. Winfield, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 1–13.
[35] P. Suárez and A. Iglesias, "Bat Algorithm for Coordinated Exploration in Swarm Robotics," in *Harmony Search Algorithm*, J. Del Ser, Ed., Singapore: Springer, 2017, pp. 134–144. doi: 10.1007/978-981-10-3728-3_14.
[36] H. Hamann and H. Wörn, "A framework of space–time continuous models for algorithm design in swarm robotics," *Swarm Intell.*, vol. 2, no. 2, pp. 209–239, Dec. 2008, doi: 10.1007/s11721-008-0015-3.
[37] "What is Memory?," Computer Hope. Accessed: Nov. 21, 2021. [Online]. Available: https://www.computerhope.com/jargon/m/memory.htm#volatile
[38] D. Ajwani, I. Malinger, U. Meyer, and S. Toledo, "Characterizing the Performance of Flash Memory Storage Devices and Its Impact on Algorithm Design," in *Experimental Algorithms*, vol. 5038, C. C. McGeoch, Ed., in Lecture Notes in Computer Science, vol. 5038. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 208–219. doi: 10.1007/978-3-540-68552-4_16.
[39] J.-G. Kang, Y.-S. Choi, and J.-W. Jung, "A Method of Enhancing Rapidly-Exploring Random Tree Robot Path Planning Using Midpoint Interpolation," *Appl. Sci.*, vol. 11, no. 18, Art. no. 18, Jan. 2021, doi: 10.3390/app11188483.
[40] R. Stern *et al.*, "Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks." 2019.
[41] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968, doi: 10.1109/TSSC.1968.300136.
[42] F. Grenouilleau, W.-J. van Hoeve, and J. N. Hooker, "A Multi-Label A* Algorithm for Multi-Agent Pathfinding," *Proc. Int. Conf. Autom. Plan. Sched.*, vol. 29, no. 1, pp. 181–185, May 2021, doi: 10.1609/icaps.v29i1.3474.
[43] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artif. Intell.*, vol. 219, pp. 40–66, 2015, doi: https://doi.org/10.1016/j.artint.2014.11.006.
[44] A. Felner *et al.*, "Search-Based Optimal Solvers for the Multi-Agent Pathfinding Problem: Summary and Challenges," *Proceedings of the International Symposium on Combinatorial Search*, vol. 8, no. 1. Association for the Advancement of Artificial Intelligence (AAAI), pp. 29–37, Sep. 2021. doi: 10.1609/socs.v8i1.18423.
[45] G. S. and B. Eater, "Visualizing quaternions, an explorable video series." Accessed: Mar.

17, 2024. [Online]. Available: https://eater.net/quaternions
[46] D. Hosier, “AVOIDING GIMBAL LOCK IN A TRAJECTORY SIMULATION,” Jul. 2018, [Online]. Available: https://apps.dtic.mil/sti/pdfs/AD1055301.pdf