Final Report: OnShape

Brenna Garcia
blg75@cornell.edu

Vivian Li
vml39@cornell.edu

Clive Duncan
cad324@cornell.edu

INTRODUCTION

Our goal is to create an intuitive interface that allows the user to make gestures to manipulate their physical environment. More specifically, the user controls the display through hand gestures under an XBox Kinect sensor. The image of the Kinect is read in real-time and translated into x-y and depth values, which are then sent to the Arduino and translated into a form that indicates which motors should run, which direction they should rotate in, and the number of steps they should run for. The motors allow the user to observe the linear motion of the objects on the actuated surface.

While this project was intended to be just an initial prototype, the final product we created was able to achieve the design goals we had envisioned. In addition, this interface invokes further research and design in two domains: first, it allows users to manipulate a 3D surface remotely, encouraging the development of devices to precisely mirror gestures in applications such as remote teaching or accessibility services. Next, it raises the possibility of interacting with our world in a 3D manner rather than the 2D manner that prevails in our phone and computer screens.

The two main parts of this design are the user interface and the block display. The user interface is composed of the XBox Kinect and the Processing code used to read the image from the Kinect. The image received from the Kinect is split into a grid of 5 x 6 squares from the area that within the range of detection, and each square is assigned a depth value based on the distance of the object in the square from the Kinect. The block display is composed of laser cut and 3D printed parts that are assembled into a box with racks inside the box to hold the motors in place. Each of the motors is connected to a pinion and aligned with a rack. The rack is then attached to a block, which is what the user sees on the top of the box. The rack and pinion system is used to translate the rotational movement of the motors into a linear movement so that the blocks can move up and down based on the number of steps specified by the stepper motor. A library called AccelStepper was used to control the simultaneous movement of each motor and to give the user a more realistic experience with the display. Finally, each motor is wired into one of three Arduino boards, and two power supplies were used to power all 30 motors.

The design goals we set for this prototype were to create a real-time display of an actuated surface that could be manipulated through user interaction in some way. Originally, we had intended on creating the user-interface display with 30 infrared sensors that would accomplish a similar goal as the Kinect. However, with the recent technological advances in motion detection, especially with the accuracy and detail of the Kinect image, we thought that using a Kinect would lead to a more realistic prototype that is suited for more modern technologies. In addition, while our project hardly touched on the image features of the Kinect, OnShape gives the audience an idea of how 3D remote sensing may work in future projects. Another design goal we had was to introduce the capabilities of 3D object manipulation. While our prototype turned out to be a display rather than serve a practical use, the idea that we could manipulate the physical world remotely was highly emphasized in the display.

RELATED WORK

The MIT Tangible Media Group has been working on several projects with similar mechanics for the past few years, including Relief, Recompose, inForm, and AnimaStage. Each of these projects uses gesture sensing to manipulate a series of independent parts.

As a precursor, Leithinger and Ishii's work on project "Relief: A Scalable Actuated Shape Display" allows users to render 3D models, such as geographical terrain, afforded through a malleable surface. That is, users can push and pull the pins to animate their displays. Unlike project "Relief", our design separates the user control from the display while maintaining a natural mapping interface. Nonetheless, OnShape does afford the possibility of representing interesting 3D forms [5].

Recompose was the first time they attempted this type of project, in 2011. The completed project recorded the user's gestures over the actuated surface as input, then translated the gestures into functional manipulation to move each of the blocks. The system was implemented with a series of machinery stacked in space, from top to bottom: the projector and depth camera, the gestural input range, the direct manipulation range, the actuated surface, and the computer. The camera was able to recognize the direction and depth of the user's gestures and reflect the details in the moving parts [6].

While Recompose focused on the ability to physically manipulate an actuated surface, its successor project inForm in 2013 built upon this ability by rendering 3D content along the surface and allowing users to physically interact with that content [4]. For example, terrain and architectural models could be replicated by the actuated surface and then manipulated by the hands of urban planners and architects. Another application involves using the shape display to create dynamic UIs in the form of buttons, sliders and knobs that the user can touch, push and interact with to change displays.

Finally, AnimaStage was the lab's most recent project in 2017 and allowed the user to craft objects directly on the surface, then "animate" the crafts and the surface through direct pin control, indirect pin control, and invisible string control [9]. The intended work presented in this paper is functionally similar to the projects from the Media Lab, but due to financial and hardware constraints, the ability to control specific elements such as pressure were omitted from this project, and the number of parts was reduced to accommodate the limitations on the amount of power that is available for running the motors.

Another interesting application of actuated surfaces can be found in the project, "Kinetic Blocks – Actuated Constructive Assembly for Interaction and Design". Their project uses a pin-based shape display to accurately move and manipulate objects on the surface to assemble and disassemble structures. Likewise, the design of OnShape allows for manipulation of objects on the surface, however this is controlled by realtime user input and does not attempt to assemble structures [13].

Festo's project "WaveHandling" also sheds an interesting light on the utility of actuated surfaces. Their technology consists of many bellow modules that deform the surface, creating a wavelike motion that transports an object about the surface in a targeted motion. Due to limitations in budget, OnShape's design does not allow for a comparable level of precision in movement, but the user will still be able to manipulate the surface to move objects around [12].

Similarly, many projects involving gesture sensing have been implemented and simplified for commercial purposes as opposed to the research emphasis in the Media Lab. For instance, "3D Motion Tracking" is a project published on Instructables, a site where people can upload how to tutorials for their different creations, including Arduino projects. The article teaches the user how to track 3D motion using an Arduino by constraining the user's actions in a cardboard cube and connecting wires from the Arudino to the sides of the cube [7].

Our project also relies on controlling the precise motion of multiple stepper motors at once. One project that demonstrates a similar feature is "Arduino Marble Maze Labyrinth" by Ahmed Azouz on Arduino's Project Hub. This project uses three servo motors operating simultaneously with very slight adjustments in rotation to tilt and rotate a physical maze in 3D space, allowing the user to roll a marble from one end of the maze to the other. This project's physical design and code serve as excellent examples of how to get the motors operating precisely as a unit [2].

A similar challenge on a much greater scale is found in a project entitled "In Servo We Trust!" by Moushira on Arduino's Project Hub. This project was a personal challenge and experiment for the creator to control 135 servos with a single Arduino. Although the scale of this project is far beyond what we intend to accomplish, the designer's detailed discussion of the unique challenges and considerations of operating and powering large numbers of servos is important to our own work [8].

While our final design turned out to be fundamentally different than the examples we researched for the initial design, we were still able to draw on a lot of the previous work to implement our project. Specifically, we used the previous large-scale projects from the MIT Media Lab to inspire how we wanted our user to interact with our project, and we learned from the examples of operating multiple servo motors simultaneously in order to set up our own grid of 30 stepper motors.

FINAL PROTOTYPE

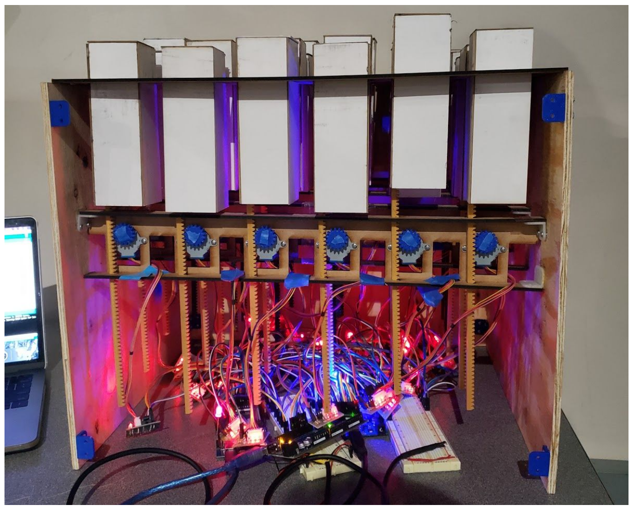

Figure 1. The final prototype in action.

A Drive folder containing our video demonstrations from our demo:

https://drive.google.com/drive/folders/1T5nqeyoQw1s43opsE2CNFIJJmqWGMYKs?usp=sharing

Github repository with all of our code: https://github.coecis.cornell.edu/vml39/onshape

Laser cutting and 3D printing files:

https://drive.google.com/drive/folders/1BrC6S_vEtUnvSRoOc0AjXmQiFpJZlp64?usp=sharing

FUNCTIONAL UNITS

There are two main functional units to the project: the block display and the user control area. The user interaction aspect is executed with a Kinect, where the Kinect is able to detect the user's hand motions in a 3D plane. The boundaries of the Kinect's sensor determine how the area in between is split into thirty squares in the x-y coordinate plane. Each of these squares corresponds to a block in the block display. The depth of the user's hand in relation to the Kinect determines the height of the block when it rises in the block display. The functional unit of the box display is comprised of stepper-powered pin movement, where the timing and amount of stepper rotation is controlled from the translated data being received from the Kinect.

Box and grid design

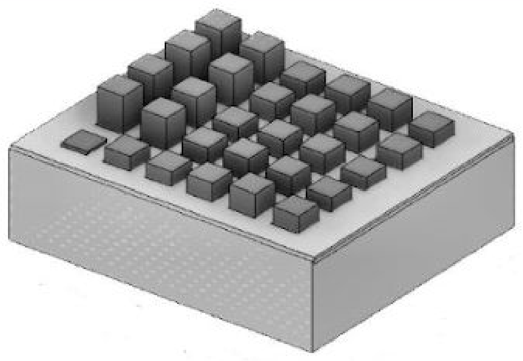

Figure 2. Block display (external)

Our use of materials and range of motion in our design for the box and grid display has changed since our initial design. Instead of acrylic, the outer box and grid components will now be made out of chipboard and the thirty blocks themselves will be made out of a lighter, thick, paper-like material used in architectural modeling that is sturdy but still much lighter than chipboard. We have also added an additional 5 cm of vertical motion to the blocks by extending their length and the length of the rods that support them to create a grander effect and increased control over the depth of the block based on user feedback.

The block display integrates several essential components. Visible from the outside are the 30 blocks distributed across a chipboard grid with slots measured to fit each block precisely. Each block is spaced 2 cm apart from each other on every side. At rest, the boxes stick just 1 cm above the surface of the grid, with the other 14 cm of the block length housed inside the box for a total height of 15 cm. As shown in Figure 2, each of these blocks is hollow to accommodate its internal components.

The exterior frame of the box will be 48 x 40 x 23 cm. The sides of the box will be slotted to allow for the mounting of interior cooling fans, accounting for the heat that the running of 30 motors will produce.

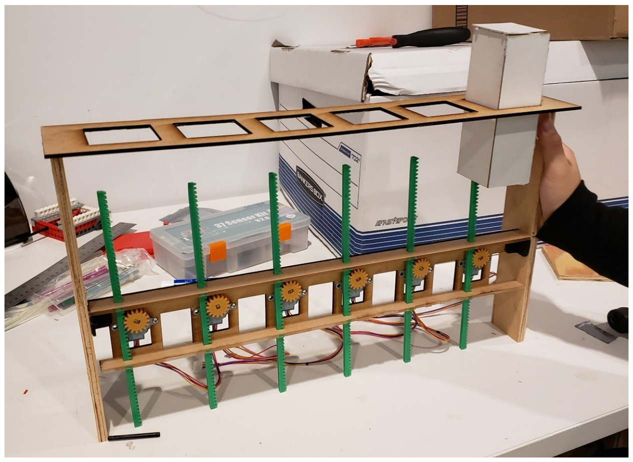

Figure 3. Demonstration of the fit of interior box and grid components

From our initial design we made several revisions to the box and grid assembly. The first of which was that we created L-angle brackets as a mounting system for securing the racks to the interior of the box. This will be discussed in more detail in the "Stepper motor rack" section. We also decided to secure the four sides of the box using stronger, thicker 3d printed L-angle brackets and wood glue on the inner edges. We realized quickly that the chipboard sheets we had been supplied were not tall enough to account for the needed height of the box, so we amended our outer box design to be made of plywood. This made it necessary to use a drill and longer screws to secure the box together. We also decided after assembling the final prototype to leave one side of the box off entirely, eliminating the need for the cooling fans and slotted sides we had planned for in our initial design. This kept the inside of the box cool while allowing for easier monitoring of the interior mechanism and creating a cool view for the demo. The final box and grid design performed well and remained secure throughout the demonstration.

Stepper-powered pin movement

Pin movements are powered by 30 ELEGOO 28BYJ-48 ULN2003 5V small stepper motors. These motors connect to their own individual driver boards and use four further wires to connect to the digital pins of the Arduino Mega 2560. The 30 motors were connected to 3 Arduino Mega 2560s with 10 motors connected to each. The AccelStepper library is used to regulate the simultaneous stepping of each motor. I.e., while one motor is stepping *up*, another can be stepping *down*.

Each central cylinder of the stepper motors is fitted with a 3D printed pinion 2 cm in diameter, designed in conjunction with a 3D printed rod. The pinion is a small 24-teeth gear modelled after the Mc-Master Carr component with a width of ¼ inch. Our initial Fusion 360 model of the rack and pinion had small teeth which an earlier iteration revealed would cause slippage. In the final prototype we made the teeth bigger which was much more effective in guaranteeing consistent locking between rack and pinion. Additionally, the motors are powered by two 5V power supplies. After experiencing issues with motors drawing too much current, we eventually determined, with the guidance of Professor Guimbretière, that having two 5V 2A power supply to provide voltage to 15 motors each would be sufficient as opposed to our plan of using 5V voltage regulators. The power supplies were able to move the motors as expected.

One aspect of the design of pin movement that we did not consider was how we would zero the system in between demos. It is important that when starting the system that all blocks are on the same initial level. We were able to overcome the issue by pushing down on the blocks to level them when the system is powered off.

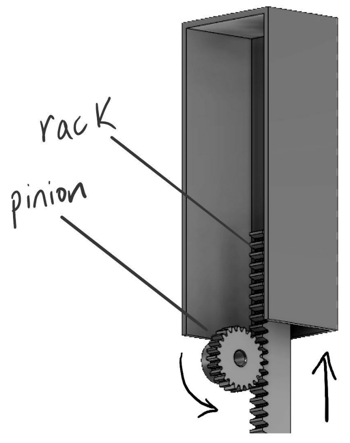

Figure 4. Stepper Rack and Pinion

Stepper motor rack

The stepper motor rack has been redesigned significantly since our initial design. It became clear early on that our plan to use L-shaped hooks to keep the rods in place would require attaching these pieces by hand since the laser cutter cannot render 3D pieces, and this would create issues of human error with lining the components up precisely. We then moved on to the idea of using a set of shelves, one above and one below the motor rack, with perfectly sized slots in them to keep the rod in place and pin it carefully between these shelves and its pinion. However, we again struggled with the idea of carefully measuring the location of all three of these layers of shelves and racks in the box to make for a perfect alignment. With all this in mind, we created a new motor rack design that will create a very precise and snug fit while still being easily modeled in Fusion 360 and laser cut for precision.

There will be five motor rack units total, with each unit housing six motors. The motor racks will feature a vertical piece with keyhole shaped windows for each motor to mount into and holes for mounting the motor with screws. Each motor rack unit will also have two stabilizing racks built in at the top and bottom and fixed horizontally to the rack at 90 degree angles. Additional screw holes on either end of the motor rack unit will allow for a 3d printed bracket to screw securely into the sides of the main frame at either end to hold the apparatus securely in place.

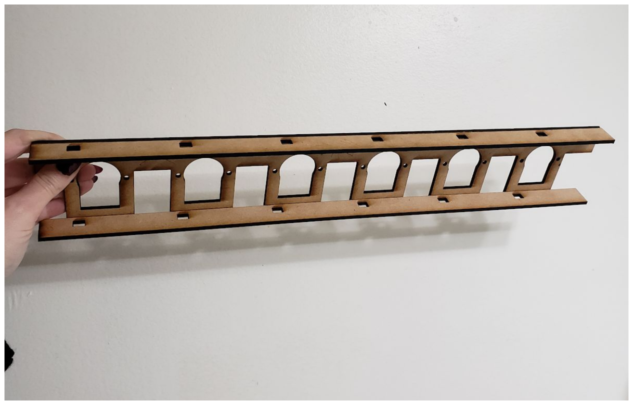

Figure 5. Motor rack with keyhole design. Stabilizing racks are mounted along the top and bottom edge.

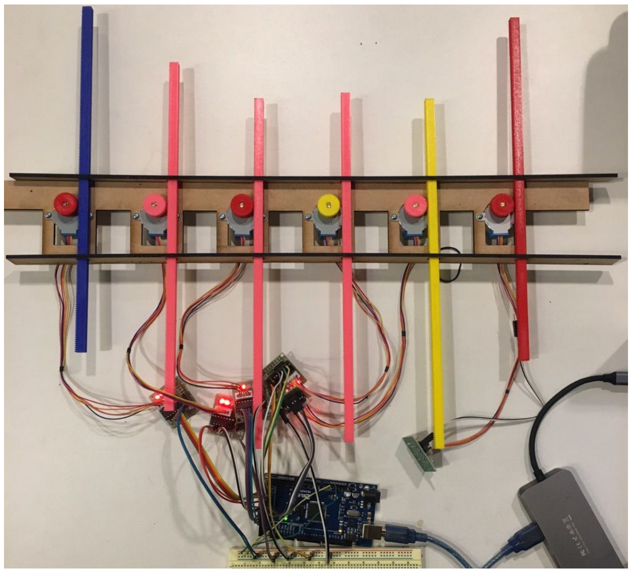

Figure 6. An example of one row of the grid, with six motors in the row. This demonstrates the basic layout and spatial orientation of our features.

After producing these new rack designs on the laser cutter we addressed the issue of attaching all three pieces per rack to each other and mounting them to the outer frame of the box. We used superglue to adhere the rack pieces to each other and then 3d printed L-angle brackets that would mount either end of the rack to the wooden box. We had originally planned on drilling holes in the box and the rack to mount these brackets using small screws and nuts. However, we were concerned with the precision that would be required to mark where to drill the holes in exactly the right place for each rack to be perfectly spaced apart and perfect in height from the bottom of the box, and with the idea that if we put a hole in the wrong spot our mistake couldn't be easily fixed or adjusted. We were also concerned that the chipboard might split if we tried to drill two holes in such a small area. Therefore, we modified our design to use superglue and wood glue to attach the racks to the box so that we could make last minute adjustments for precision before the glue set. We also attached an additional L-angle bracket from the bottom horizontal piece to the box to prevent rotation of the racks, as per Professor Guimbretière's advice. During their final performance the racks were quite sturdy, and were not brought off the box by the weight of the stepper motors nor did they twist or become unstable when the rods moved up and down within the slots.

User interaction space

The goal of this functional unit is to allow the user to remotely interact with the block display. The execution of the user interface has been modified since the initial prototype from 30 IR sensors in a grid pattern to using an XBox Kinect. The Kinect is able to detect motion in a 3D space, and for the purpose of this project, will return the x, y, z coordinates of the user's hand in this space. It does this using an RGB color VGA video camera and a depth sensor which work together to reconstruct the user's motion and image. The Kinect is mounted from above, and a visual representation of the Kinect's image is displayed on the computer to give the user feedback on their gestures. When the user moves their hand within this 3D space, the Kinect sends data about the user's hand's image and its movements – specifically, the Kinect sends a byte to indicate which square is being activated and another byte with the depth data, constrained to a number between 0 and 9 (0 indicating the user's gesture as closer to the Kinect's camera). This data is then translated to fit a 30-block grid, where each block on the grid receives the information of whether there is an object in this space in the grid and if so, the object's depth. This information is sent to the Arduino through the Serial port. Finally, the information is used to mark which motors need to be run and in which direction. This process allows near-simultaneous feedback in the block display based on the user's movement under the Kinect.

While the Kinect was able to create an accurate image of the user's gestures, there were still a few issues that arose in its integration with the rest of the design. A small issue was that the Kinect was reconstructing the image with the x-y coordinates mapped inversely, i.e. the image was displaying the opposite of the user's movements. It was a challenge to figure out a solution given the way we were sending data to the three different Arduino boards. The solution we found was to reconstruct the Kinect's image from the bottom corner to the top when the image was being drawn on the screen. Another issue we faced was how to communicate with the three different Arduino boards and distinguish between which motor we were trying to communicate with. First, we had to set up the different boards to listen to different Serial ports, then determine the mapping of each board and its motors. Originally, 30 bytes were getting sent through a singular Serial port, and when put into a for-loop, would send the depth of each motor sequentially, from 1 to 30. However, the addition of two more boards meant the system had to first send a byte indicating which of the 30 motors it was referencing, and the boards had to listen to and process all of the information being sent. Finally, a challenge we were unable to overcome was the speed of the motors in comparison to the real-time feedback being received from the Kinect. The Kinect was able to quickly reconstruct the image each time it changed, but the motors had difficulty catching up to the speed of change and would lag behind. This was slightly compensated for on the Arduino side by constantly calculating the current position of the motor and the distance it needed to move to reach its target height before the motors were moved, but the computation and the way the board had to listen to all of the bytes being sent on the Serial Port before all of the motors could run simultaneously still caused a delay in the display.

Combining the functional units to create gesture-controlled movement

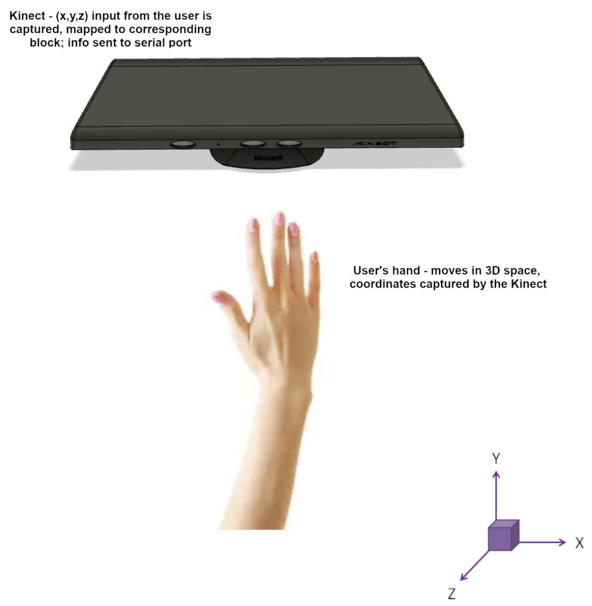
Kinect - (x,y,z) input from the user is captured, mapped to corresponding block; info sent to serial port
User's hand - moves in 3D space, coordinates captured by the Kinect
Y
X
Z

Figure 7. A visual of how the user will interact with the Kinect in a 3D space.

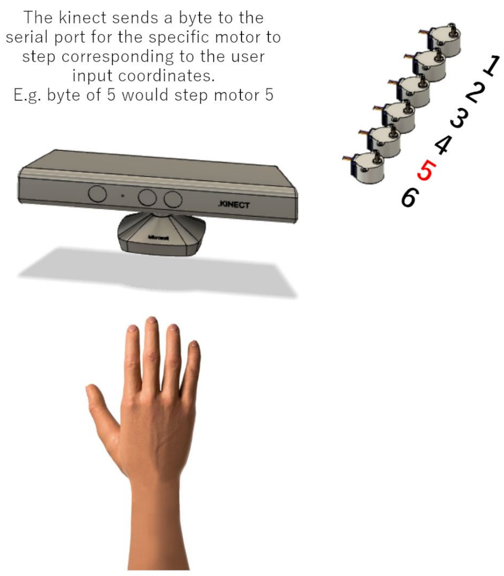
The kinect sends a byte to the serial port for the specific motor to step corresponding to the user input coordinates.
E.g. byte of 5 would step motor 5
1
2
3
4
5
6

Figure 8. How the user's movement corresponds to each motor in the block display.

The Kinect is mounted on a stand close to the block display. The user's gestures below the Kinect which registers the 3D coordinates. The reading from the Kinect is stored and manipulated to control the movement of the blocks in the block display. That is, the field in the Kinect camera is mapped to the 30 blocks, and the depth sensor determines the range of motion by the block. In an earlier iteration of the prototype we intended on using 30 IR sensors to map the gestures to block movements. Below is a simplified circuit model depicting how a particular IR sensor would have mapped to a stepper motor.

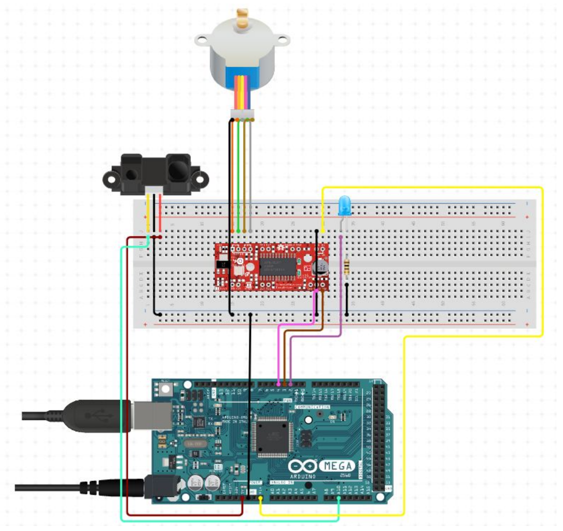

Figure 9. Simplified circuit diagram showing the relationship between input and output.

With the advice of Professor Guimbretière, we decided that using the Kinect for gesture detection would me more time efficient and easier to manage.

Originally we planned on using 10 analog pins for the range sensor, 40 digital pins for the ten motors and 10 digital pins for the 10 blue LEDs for each of 3 Arduino Megas. Since we abandoned the IR sensor setup, we had no need for the LEDs. However, we still needed to use 3 Arduino Megas to connect the 30 motors to the digital pins. Using three Megas also helped us to provide room within the box for the racks and pinions to move free. In addition to that, it allowed us to run identical code through 3 serial ports, each of which controlled a specific Arduino board. When running the Kinect to communicate via serial port, it seemed to result in slow movement of the blocks. So while the Kinect eventually integrated well and the functionality was present, the movement of the system was slower than anticipated.

Additional functionality that we were able to implement was to allow the distance of the user's hand from the sensor surface to control for the height of the corresponding block. This mapping was achieved within our code by reading the size of the input value from the range sensor and incrementing the height of each block according to the relative value. This allows users to not only control which blocks move but by how much depending on whether they move their hand closer or farther from the surface.

FUTURE WORK

There were a few potential areas for improvement discovered after demonstrating our final prototype. Although our physical assembly functioned well enough to create the desired effect, we could have produced more impressive results had the pins moved faster in response to user input. In our final prototype the stepper movement was gradual and so users often had a hard time seeing how precisely the mapping of their movements was occurring since they would have to hold their hand in place long enough for the pins to move into place. Faster stepper motor movement could have been achieved by increasing the current limit or supplied voltage, but we hesitated to damage our motors in the process of experimentation with increasing the speed. Additionally, the assembly occasionally had some random blocks moving based on ghost inputs on the Kinect camera, but this problem is something we would likely not be able to solve with our assembly or integration.

REFERENCES

1. American Tech. 2018. Linear engine , How to make linear motor step by step , science school project 2018. Video. (1 March 2018). Retrieved September 19, 2019 from https://www.youtube.com/watch?v=Oeoj9ZCQbAk
2. Ahmed Azouz. 2019. Arduino Marble Maze Labyrinth. Retrieved September 21, 2019 from https://create.arduino.cc/projecthub/AhmedAzouz/arduino-marble-maze-labyrinth-bd9ea6?ref=tag&ref_id=servo&offset=14
3. Doug Domke. 2019. Servo Motor Artwork. Retrieved September 22, 2019 from https://create.arduino.cc/projecthub/doug-domke/servo-motor-artwork-79e2d3?ref=platform&ref_id=424_trending___&offset=7
4. Sean Follmer, Daniel Leithinger, Alex Olwal, Akimitsu Hogge, Hiroshi Ishii. 2013. inFORM: Dynamic Physical Affordances and Constraints through Shape and Object Actuation. UIST 2013.
5. Daniel Leithinger, Adam Kumpf, Hiroshi Ishii. 2010. Relief: A Scalable Actuated Shape Display. *TEI 2010.*
6. Daniel Leithinger, David Lakatos, Anthony DeVincenzi, Matthew Blackshaw, Hiroshi Ishii. 2011. Recompose: Direct and Gestural Interaction with an Actuated Surface. *CHI 2011*.
7. Maartjeeee26. 2017. Instructables. Retrieved September 21, 2019 from https://www.instructables.com/id/TfCD-3D-Motion-Tracking/
8. Moushira. 2017. In Servo We Trust! Retrieved September 21, 2019 from https://create.arduino.cc/projecthub/Maya/in-servo-we-trust-6725f1?ref=tag&ref_id=servo&offset=43
9. Ken Nakagaki Udayan Umapathi Daniel Leithinger Hiroshi Ishii. AnimaStage: Hands-on Animated Craft on Pin-based Shape Displays. *DIS 2017*.
10. Robot Brigade. 2011. Linear motion using a motor. Video. (7 June 2011). Retrieved September 19, 2019 from https://www.youtube.com/watch?v=ifVxd5DLE0g&feature=youtu.be
11. thatguyer. (n.d.) NeoPixel Reactive Table. Retrieved September 21, 2019 from https://www.instructables.com/id/NeoPixel-Reactive-Table/
12. WaveHandling: Transporting and sorting in one. Retrieved September 22, 2019 from https://www.festo.com/group/en/cms/10225.htm
13. Daniel Windham, Philipp Schoessler, Daniel Leithinger, Sean Follmer, Hiroshi Ishii. 2015. Kinetic Blocks - Actuated Constructive Assembly for Interaction and Display. *UIST 2015*