University of Colorado **Boulder**

Final Design Report - Spatial Visualization Blocks

ABSTRACT

Spatial visualization, or the ability to mentally manipulate complicated shapes in different orientations, is a primary predictor of success in engineering design courses and overall retention in engineering programs. Although it has been proven as a learnable skill, the current strategies for practicing spatial visualization (SV) are tedious and often require paper-and-pencil exams, resulting in learning barriers. To make this learning process more engaging, the ENGIcation team has designed a hands-on live-feedback learning system.

The project consists of electronically connected blocks, a baseplate that houses the key electronic circuit, and a phone app with various practice problems. Users first open the app interface and choose a practice problem from three categories of spatial visualization skills. Once they are prompted with a specific set of orthographic or isometric views, they can begin building the solution model on the baseplate. Throughout the building process, the bluetooth-connected circuit continuously checks for correctness of the model based on voltage drops across resistors in each block. If the model voltages match corresponding reference voltages, a green LED turns on, providing the user live feedback on the correctness of their model. At any point, they can switch to a new problem on the app or check the correct answer.

Figure 1. Final Design

University of Colorado **Boulder**

Final Design Report - Spatial Visualization Blocks

I. BACKGROUND

A 2015 CU Boulder study identified several factors that predict success in design courses and retention in engineering degree programs: previous manufacturing experience, ACT math scores, playing with legos or other construction tools as children, gender, and success on the Purdue Spatial Visualization Test (PVST – see Figure 1). Most of these factors are related to competency in spatial visualization, which has been proven to be a learnable skill. After an experimental group completed a 10-week course involving paper-and-pencil practice problems, their scores on the PSVT increased significantly as compared to the control group (See Figure 2). In this context, practice indeed makes perfect.

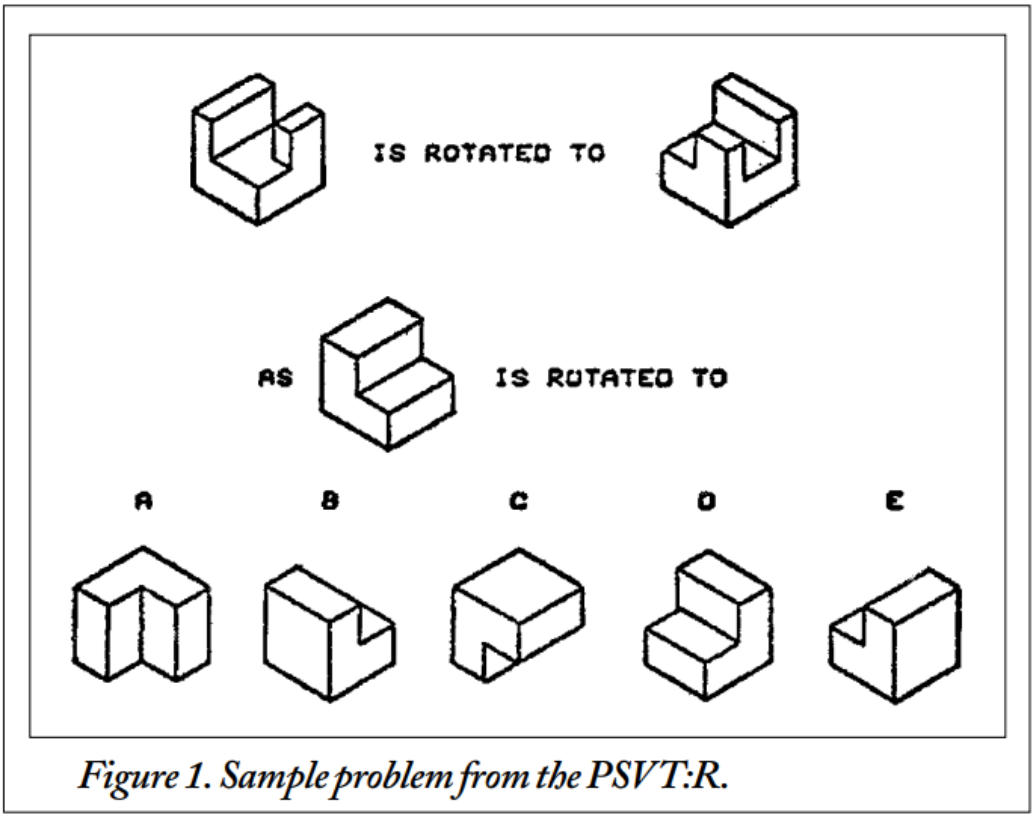


Figure 1. Sample problem from the PSVT:R.

Final Design Report - Spatial Visualization Blocks

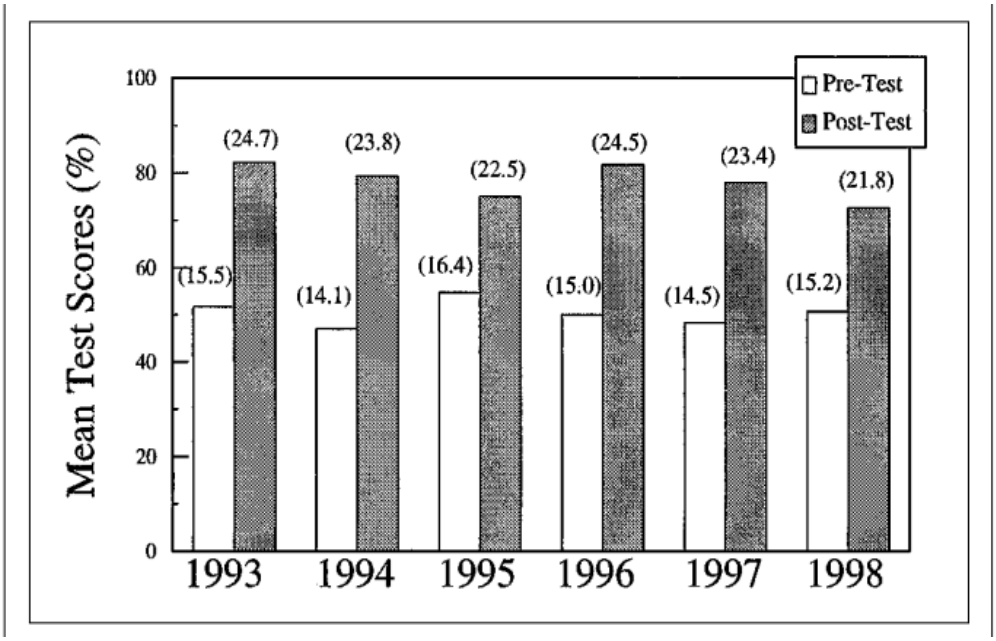


Figure 2. Mean PSVT scores for the experimental group involved in the spatial visualization course.

However, this 10-week course was completed outside of normal classroom hours, and was generally disliked by the student population, especially by those who already felt well-acquainted with spatial visualization. In order to understand more about why students dislike existing spatial visualization practice mediums, we asked 30 of our peers (engineers and non-engineers) to complete a 5-question sample PSVT and provide feedback on their experience. While some respondents seemed comfortable with the questions (reflected in their high scores), the majority indicated that they disagreed with the statement “spatial visualization is a natural skill”. Upon being prompted for suggestions to improve their practice experience, multiple respondents mentioned that a hands-on component would be helpful.

To address these concerns and to explore the efficacy of tactile learning tools for engineering students, the ENGIcation team created a modular learning system consisting of modular blocks, a bluetooth-connected, circuit-embedded baseplate, and an app interface with PSVT-type questions. Specifically, we worked with Rachel Sharpe, a professional engineer in the Integrated Teaching and Learning Program at CU Boulder, as our client to advise project creation and testing.

Final Design Report - Spatial Visualization Blocks

II. DESIGN REQUIREMENTS

A. Qualitative Requirements

As a group, we aimed to create a tactile system that helped the user improve their skills in spatial visualization. In order to create a system that was not only helpful, but also fun and easy to use, we held ourselves to high design standards. We intended to create a seamless connection between the blocks, ensuring connectivity between internal resistors as well as giving the user that satisfaction of the design. Another aspect of the system that was very important to the design was the ability for the baseplate to give the user live feedback. This is a crucial part of being able to see success in real time and it also gamifies the learning experience making it even more fun for users to practice this key skill. In order for the live feedback to be correct, and thus be helpful, it was essential that the mobile app and the base plate have a dependable connection. This meant that the connection between blocks had to be secure so the circuit could read the resistance values correctly, and the bluetooth system had to be able to read the problem number from the app and send it to the circuit to check the resistance values against.

If all went smoothly, and the user built the correct problem on the base plate, then the green light would add up and the user would be informed that they were correct. This interaction is an integral part of our system as being able to understand if the user is correct or not is essential in building spatial visualization intuition.

Final Design Report - Spatial Visualization Blocks

III. OVERVIEW OF DESIGN PROCESS

The final design is made up of 3 main components - modular blocks, a baseplate, and an app.

A. Blocks

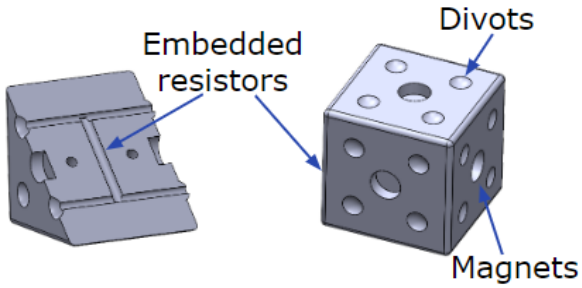


Figure 3. Blocks printed in triangular halves. Divots and magnet holes have tolerance of 0.01 in.

B. Circuit and Base Plate

The circuit design utilizes an Arduino microcontroller to pass power through the blocks - each block containing a 10 kΩ resistor in parallel with the other locks on the space - in order to read the resulting voltage. The circuit accomplishes this by powering each of the blocks individually while iterating through the code. On the input side of the blocks, each Arduino pin is connected to a diode. These diodes are rated for 20V at 1/2W, which is well above the requirements of our project and ensures that no current flows back into the Arduino, effectively cleaning our readings to achieve better accuracy and consistency. In terms of the circuit output, the blocks tie into an analog read pin which is connected across a 10 kΩ resistor to ground. This resistor acts as a voltage separator which provides the Arduino with true voltage values, read as a digitalized value. For our purposes, we convert the digitalized value back into true voltage to compare to known voltage values (determined empirically) for set numbers of blocks on each space.

The code running on the Arduino iterates through each space, utilizing the aforementioned circuit to determine how many blocks reside on each space and creating an array holding the numbers. After this

the Arduino checks the constructed array against a second array containing the expected number of blocks for the given problem number. The problem number is provided by the HM-10 bluetooth module connected through the app, which utilizes one of the digital arduino pins as a serial communication port while also taking power and ground from the arduino. If the constructed object matches the expected values the arduino will power a green LED visible in the baseplate to indicate with instantaneous feedback that the user has attained the correct answer.

The actual baseplate and its construction is very similar to that of the blocks, It consists of a housing for the electronics as well as a building region where small divots and magnets hold the blocks in place. Wires protruding through the divots contact the electrical contact points of the blocks, thereby closing the circuit.

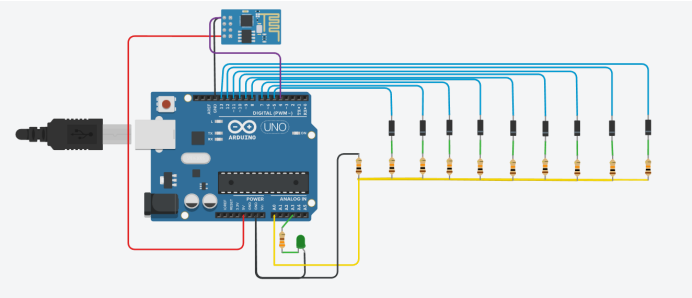


Figure 4. Circuit diagram representing 1 block (one resistor) on each square of the baseplate.

C. App

The app consists of four main components. These are represented by four buttons on the home screen that the user can interact with.

The first is the “Connect/Disconnect” toggle, which when pressed will connect the app via Bluetooth to the arduino circuit’s HM10 bluetooth module. The HM10’s UUID is hardcoded into the app so that the correct module is connected to it consistently. This required the front-end SwiftUI code to be connected to the Swift Bluetooth Manager code via a delegate class. Once set up, the bluetooth manager will send the problem number of the problem the user is looking at on the app whenever a new problem is

shown. This bluetooth manager was the most difficult part of the app to create, due to the fact that it needed to connect Apple's newest framework SwiftUI with its old framework Swift which then had to interact with the Arduino C++ analog language on the circuit side.

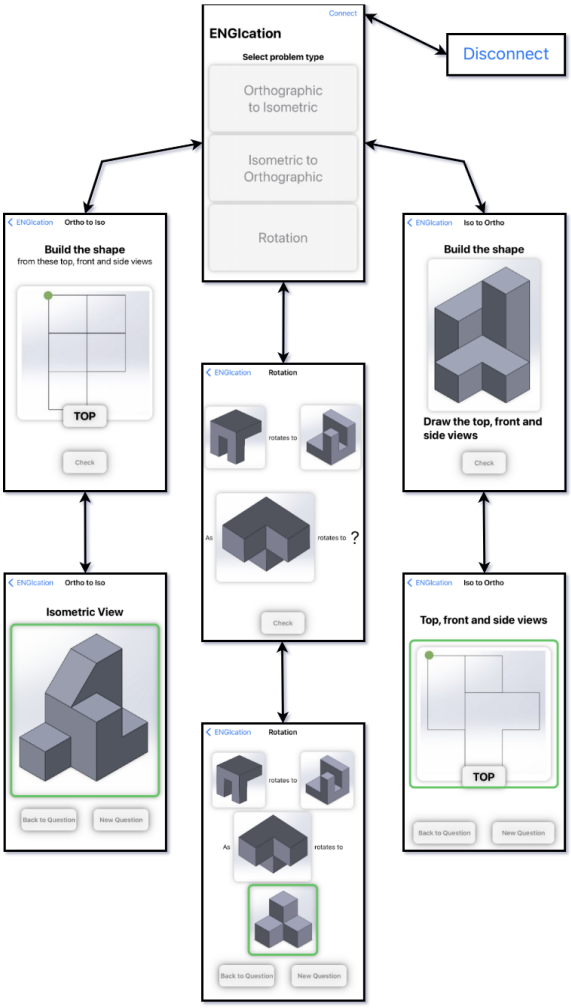
The next two buttons on the home screen bring the user to the "Isometric to Orthographic" and "Orthographic to Isometric" problem types, respectively. When the "Isometric to Orthographic" is pressed, a random problem model is selected, and its isometric views are shown to the user as the prompt. The user can then select the "Check" button, and the corresponding isometric view of the model will be shown as the answer to the problem. From here, the user can either choose to go back to the orthographic views, or move on to a different problem. At any time during this, the user can choose to navigate back to the app's home screen via the back button in the top left corner. When the "Orthographic to Isometric" button is pressed, a random model is selected, just as with the previous problem type, but an isometric view is shown to the user as a prompt instead. When the user presses "Check," the corresponding orthographic views are displayed. Just like with the Orthographic to Isometric problem type, the user can choose to go back to the problem statement (the isometric view this time), show a different problem, or return to the app home screen.

The final button on the home screen is represented by a button that says "Rotation," signifying the rotation problem type. When this is pressed, the user is brought to a screen with three images. The first two are isometric views of the same model, but rotated in different orientations. The third is an isometric view of a different model, and the user is prompted to think of and draw the orientation of this model if it was rotated in the same way the first model was. When the user presses the "Check" button, the correct orientation for the second model is shown, and the user again has the option to go back to the problem statement, see a new problem, or return to the home screen.

These models are pre-loaded as a set of four images each: a top, front, side, and isometric view. These can be referenced by the app using the format <imageType><problemNumber>. This means that

Final Design Report - Spatial Visualization Blocks

the Orthographic to Isometric and Isometric to Orthographic problem types had as many problems as there were models loaded. For the rotation problem type, eight images for each model are loaded - one isometric view from each vertice of the model. This translates to the Rotation problem type having 56*n! unique problems able to be shown when there are n models loaded. Our functional app had three models loaded into the rotation problem, and so 336 individual problems were able to be shown. Below is a schematic of the main functions of the app described above.



Linked here is the full code for the application:

https://drive.google.com/file/d/1_jnXCemZSpJYzd3wiz9tBt_0uBhoP_pe/view?usp=share_link

Final Design Report - Spatial Visualization Blocks

IV. TESTING

In order to justify the design for first-year engineering students at CU Boulder, the team consulted previous SV research and conducted a survey of current students' SV competency and feedback from paper-and-pencil exam methods. The team has tested the circuitry aspect of the system by ensuring connectivity between blocks and analyzing the necessary voltage ranges.

Because the primary function of the live feedback system relies on the blocks making connections to read the correct resistance, it is most important to test the consistency of these block connections. It is also crucial to perform qualitative analysis to determine the expected voltages from the Arduino's Analog Read pin, and adjust the ranges accordingly.

First, to create the necessary voltage ranges denoting 0, 1, 2, or 3 blocks, the team used the Serial Print function in the Arduino integrated development environment to display the voltage drop across 0, 1, 2, and 3 resistors. The Arduino is powered by 5V, and there is an additional 10 k resistor before the Analog Read and ground, so the Analog Read will never read exactly 5V. Upon building mini-circuits with 1 resistor, 2 resistors in parallel, and 3 resistors in parallel, we arrived at these ranges for the corresponding voltages:

Number of Blocks	Analog Voltage Range (V)
0	$V < 2$
1	$2 \leq V < 2.5$
2	$2.5 \leq V < 3.1$
3	$3.1 \leq V < 4$

Table 1. Experimental voltage ranges corresponding to the amount of resistors in parallel, including the voltage drop caused by the grounding resistor.

It should be noted that after experiencing some inconsistencies in these voltage readings despite a consistent number of blocks, the team decided to include diodes into each “stack”, ensuring that the current could not move backwards into another stack and introduce other resistors into the system. Diodes are simple circuit components that only allow current to flow in one direction.

V. PERSONAL CONTRIBUTION

My main contribution to this project was coding the iPhone application and setting up the connection between the app and the baseplate via Bluetooth. The app was coded using the Swift and SwiftUI frameworks in the XCode IDE. In the past I had made apps using Swift and SwiftUI, so I was comfortable with creating interactive interfaces, displaying images, and moving data around within the app. This meant that making the visual interface for the entire app, including problem screenshots and randomization took less than 10 hours in total.

The challenge for me to overcome was figuring out how to send the problem number the user was viewing from the app to the Arduino base plate circuit, on command. Since Apple’s Core Bluetooth framework is integrated into Swift, but most of my UI is made with SwiftUI, I first needed a bridge between Swift and SwiftUI to manage the connection between the interface’s Bluetooth “Connect/Disconnect” button and the baseplate’s bluetooth module. This was done by creating a delegate class called BluetoothManager. I called functions from an object of this class in order to accomplish all of the Bluetooth connection functionality I needed.

Once this was all coded, I still needed to get the HM10 Bluetooth module connected somehow. The first step was setting up the module alone on an Arduino, and then connecting it to a BLE Scanner app to find its UUID. Once I had its UUID, I hardcoded it into the connection function of the BluetoothManager so that other nearby HM10’s wouldn’t be connected instead of ours. I set up some code on the Arduino side to listen for problem numbers sent to the HM10 and store them in a variable to

be used by the circuit to check if what the user built on the baseplate is correct. Once this was working by itself, I integrated the "send data" function into the SwiftUI so that whenever a new problem was shown, its problem number would be sent.

Through this project I learned a wide range of both soft and hard skills. I learned better communication about design iterations and system connections through the multidisciplinary nature of the project, and I learned the importance of failing early and often. I improved my skill in Arduino programming and also my Swift knowledge, specifically with Bluetooth.

VI. CONCLUSION

Throughout this project, our team faced many challenges, setbacks and successes that all not only helped further develop our engineering skills but also taught us what it would be like to work with a client and bring our idea to life. Though our system was not functionable at the expo, all of the individual components worked separately. If time was managed slightly differently or if we had more time, we could have constructed a functional system. We hope to keep working on this project, as we are all passionate about the idea of spatial visualization and realize the need for beneficial resources in this field. Spatial visualization is a key skill in engineering and can predict success in more complex courses so it is imperative that we continue to develop a system that helps students who may not already have good spatial visualization intuition. If we were to further develop this product, we would focus on ensuring the reliability of our system as well as add triangular blocks to incorporate more complex problems.

This class, and thus the project associated with it, has a large emphasis on teamwork. Being able to work with a diverse group of people and ideas is a key part of being a successful engineer and our group came out of this project knowing how to successfully navigate a team dynamic.

Final Design Report - Spatial Visualization Blocks

VII. APPENDICES

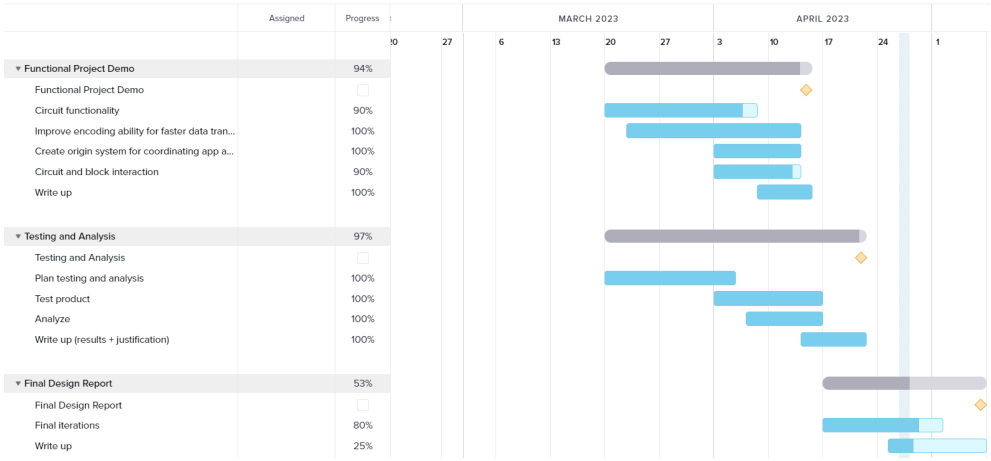
A. Budget

The budget for the project was $375.00 ($75.00 from each team member). The project's total cost ended up being $173.00, with the breakdown of the key costs of the project in the table below.

High-Definition 3D Prints	$70.00
Magnets	$12.00
Arduino Uno R3	$27.00
HM-10 Bluetooth	$9.00
Other costs	$55.00
Total	$173.00

B. Timeline

In order to stay on track with assignments throughout the semester, the team created and followed a Gantt chart and a Notion to-do list. The Gantt chart provided an upper-level overview of deadlines and deliverables, while the Notion page served as a detailed plan of action and assigned action items to each team member.



University of Colorado **Boulder**

Final Design Report - Spatial Visualization Blocks

C. Supporting Files

- Arduino base plate code:

  https://drive.google.com/file/d/1HpP6Ve1Fm2N_Hyk_YUOsBix72U2zSB0u/view?usp=share_link