3.1.7 - Machine Control Design Challenge

Our team was challenged to design the control system and a physical prototype of an elevator that can go between three floors of a building. Details of the criteria and constraints of this challenge, our ideas and attempts to design the elevator can be found in the pages of this portfolio.

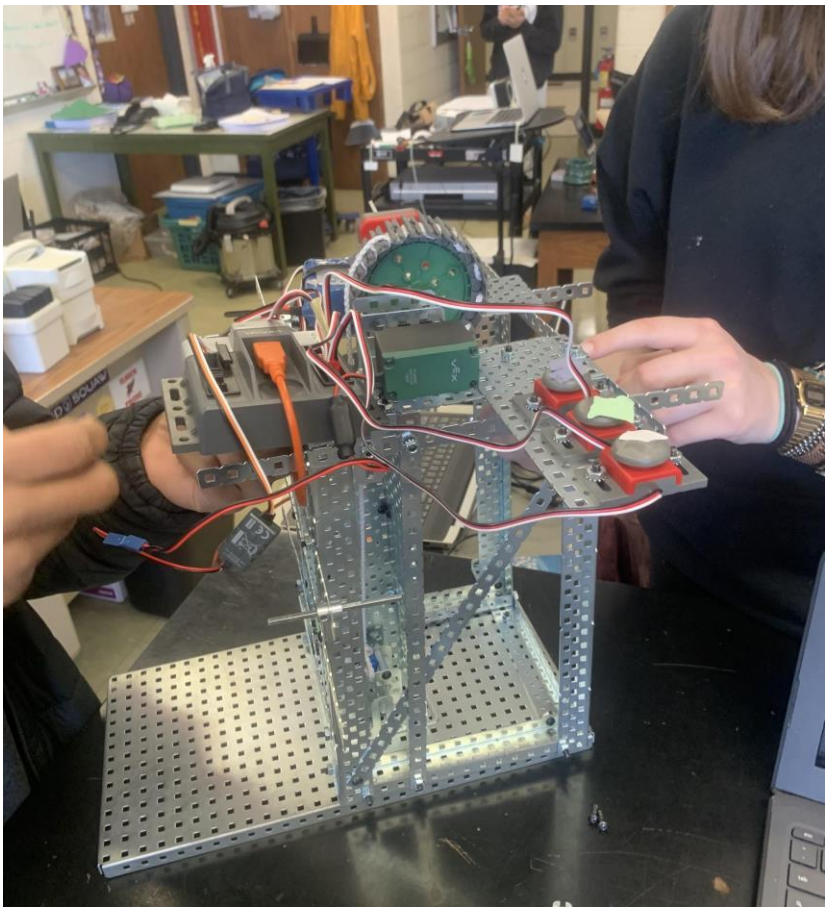


Table of Contents:

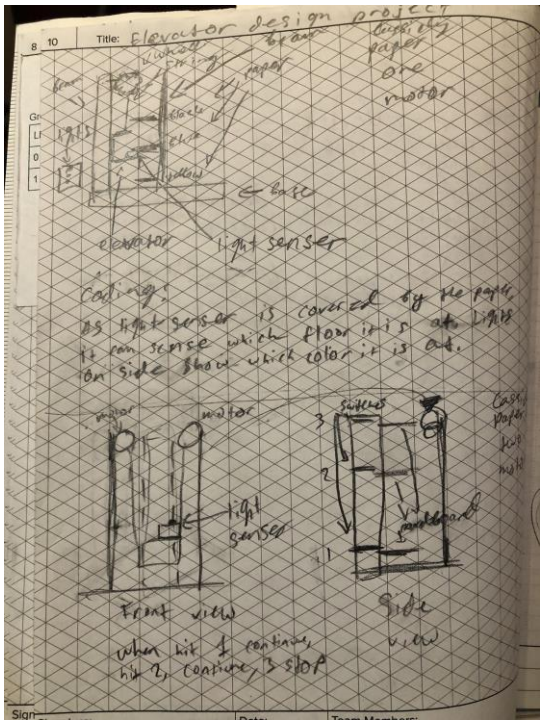
Design Brief

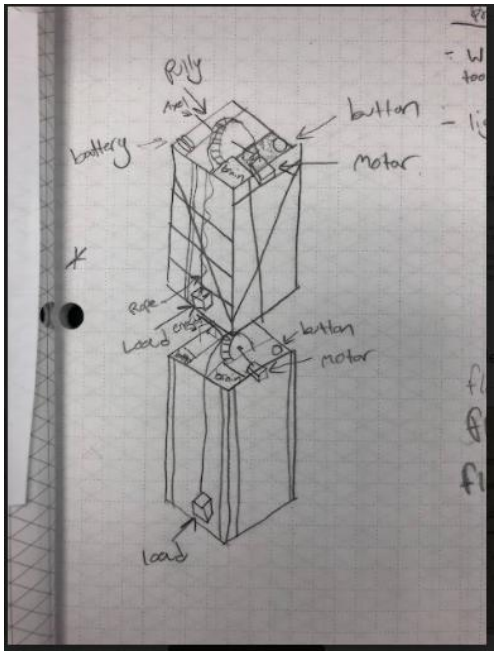
Design Brief Component	Description
Client	The company that produces elevators and has certain qualifications
Target Consumer	A person that is able to get on the elevator and hit a switch to move it to the chosen floor and tell what floor they are at
Designers	
Problem Statement	A company would like to begin producing residential elevators. Your team must design the control system and a prototype of an elevator that can go between three floors in any combination. Our prototype must include a set of three switches to represent each floor of the elevator. Each floor the elevator stops at must have a call button and a set of three lights to indicate where the elevator is currently located. A built-in safety mechanism requires that the elevator normally rest on the ground floor and return to the ground floor after a user-determined period of nonuse.
Design Statement	We will add colored paper to define each floor to the computer as well as have lights that are lit up when a floor is hit. After a certain amount of time, the motor lets the elevator down. The switches are hit by the person, each switch sets off different lines of code.
Criteria	1.Must include a set of 3 switches to show where the elevator is located 2. Be able to go to any of the floors in any order. 3. A built-in safety mechanism that requires it to go to the bottom floor after it hasn't been used for a bit of time. 4. A wait button that adds 10 extra seconds to the time before the safety is activated.

Constraints	1.Time 2. amount of VEX equipment available 3. amount of coding knowledge available to us and our own past experiences 4. Budget

Generate Concepts

First drafts

	Cassidy first draft First: One motor, paper hanging in front of it, light sensor facing up, the elevator stops when light sensor is completely covered by the the paper, waits ten seconds, goes back down Second: Two motors that pulls one string and has an elevator attached, when button pressed light sensor goes up until it hits the corresponding color (construction paper), at that point wait ten seconds go back down

	Ayden first draft Joined the group a little bit late but as an improvement suggested switching the light sensor to an encoder, as it was more reliable.

Develop Solution

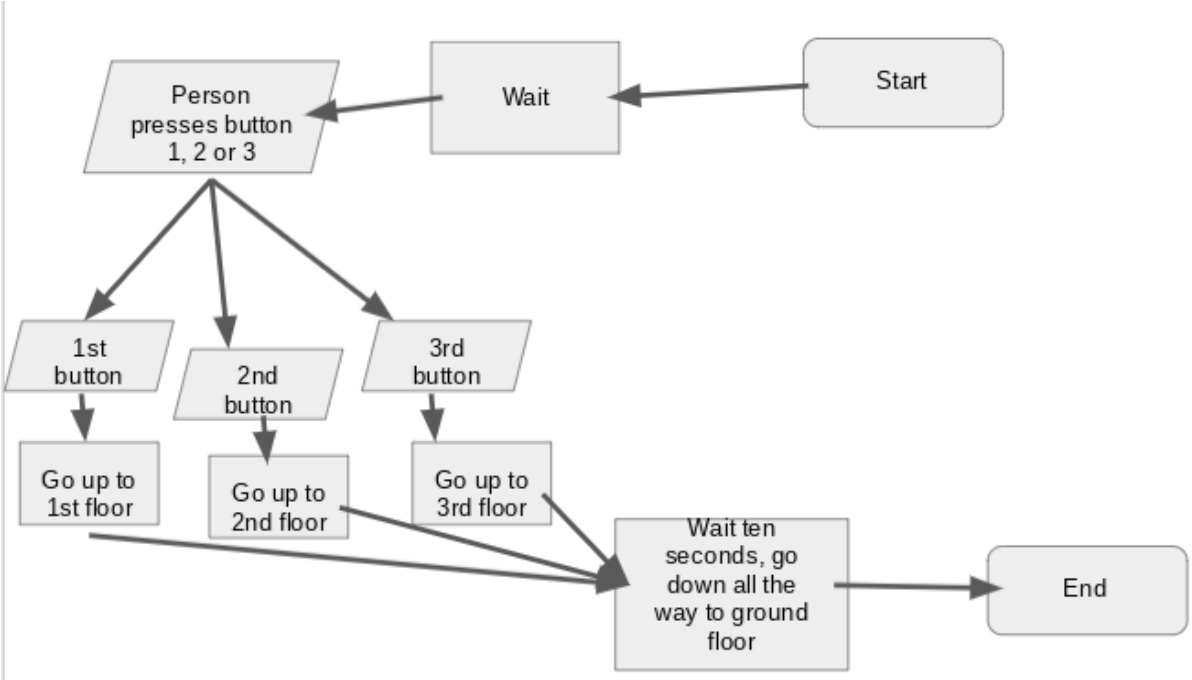
Decision matrix

*Ayden joined our team after this step in our process

Decision Matrix					
Team member names:	Connor and Cassidy				
	Rank all ideas your team is considering for each category in the chart below				

Ideas	Creativity of Design	Complexity of Build	Complexity of Code	Likelyhood of Success	Total
Connor2M	7	8	7	9	31
Connor1M	5	7	6	8	26
Cassidy 1 motor	8	8	8	7	31
Cassidy 2 motor	8	9	10	3	30

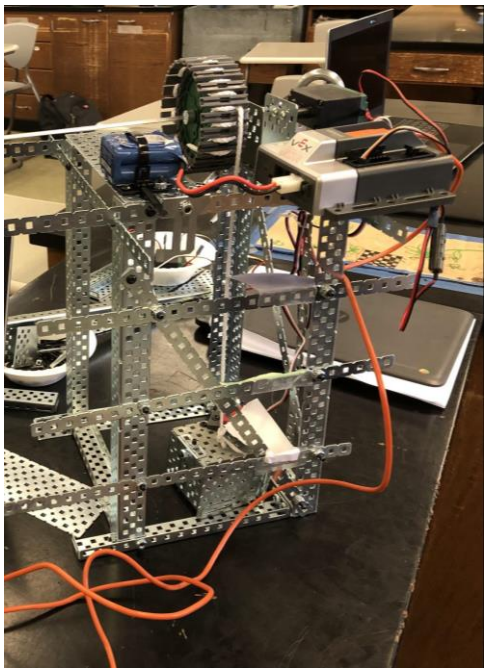
Flow chart

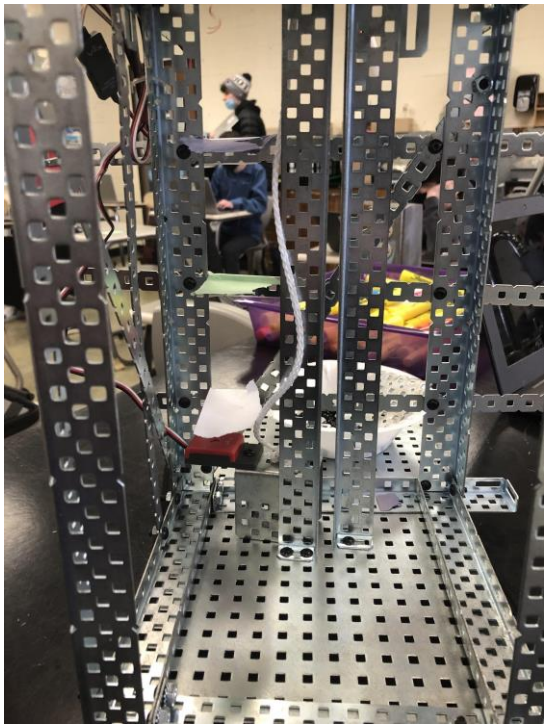


This isn't our first flowchart, but it isn't very different, the first one just relied on the light sensor.

Prototype

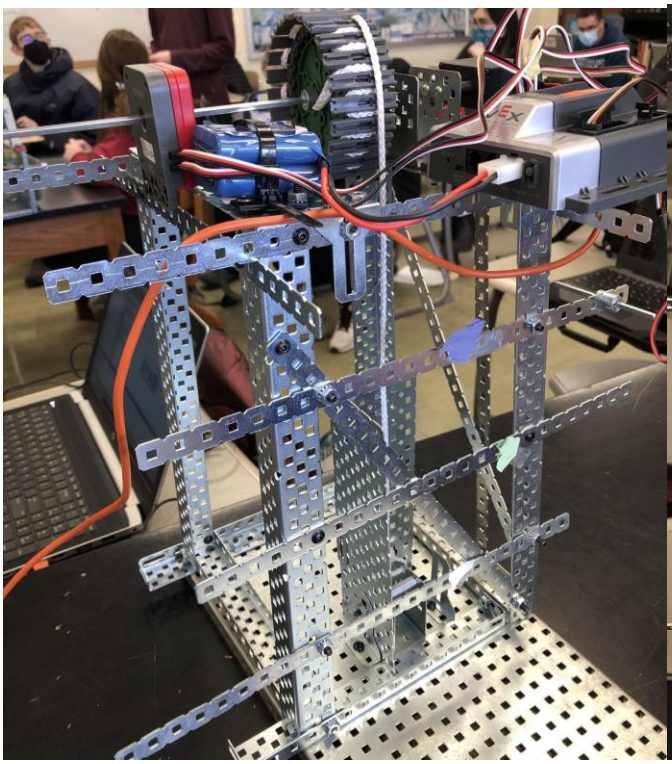
First Models

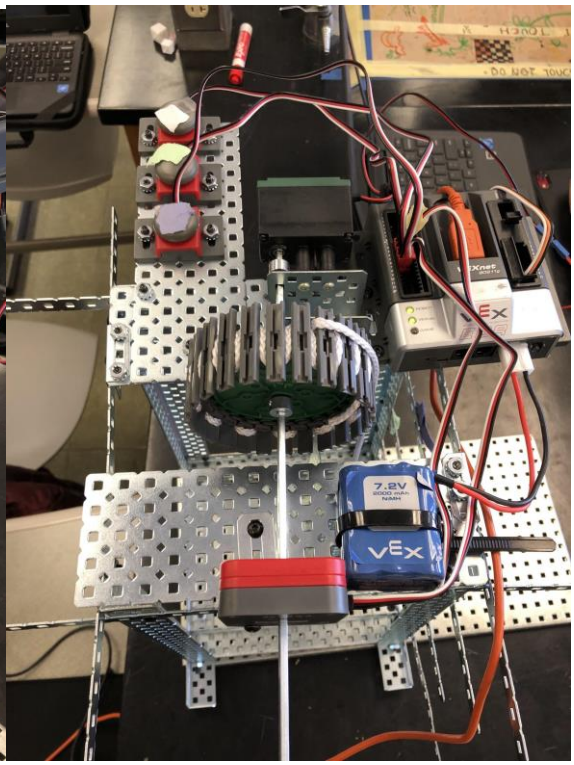
	First picture : The elevator was made of multiple boxes and had a hinge door this got scrapped because it was too heavy, and the motor couldn't pull it up very controlled. Base was made and kept the entire time, a couple adjustments for stability but that is it.

	Second picture: Something that stayed from the first to the second was the light sensor on top, and the paper was laying so that as the elevator moved up, it would cover the light sensor fully. This did not really work, it also meant that the elevator could only go up, not back down. We also had a problem with the accuracy of the light sensor, it was never covered exactly the same each time, and you had to reprogram it each time it went up. Not only that but the wire from the light sensor kept getting in the way and we scrapped that entire set up, keeping only the base. Right after this, we tried to have the light sensor press up against paper, and be on the side of the elevator, but we only got 30 mins into that design.

Testing

Video link





Our final design, instead of using a light sensor, one of our group members thought to use an encoder. The encoder could sit at the top (red box in photos) and be attached to the motor's axel. So, at a certain amount of rotations, it will stop when the code tells it to. This means we could assign specific numbers to different floors. This meant it would always be accurate, and the wires wouldn't get in the way. We added buttons, which we had always planned on adding, but hadn't gotten to yet, and focused on the coding (of course we had been coding this time, but we hadn't gotten very far yet).

Code

```
                                        //Program begins
{
while (1==1){
if (SensorValue[floor1]==1)
{
  startMotor(Motor, 15);//goes up
untilEncoderCounts(90, tape);// to first floor
stopMotor(Motor);//stops
wait(3);//wait 3 seconds
startMotor(Motor, -10);//goes back down
untilEncoderCounts(-90, tape);//to ground floor
stopMotor(Motor);//stops;
}

if (SensorValue[floor2]==1)
{
  startMotor(Motor, 15);
untilEncoderCounts(200, tape);
stopMotor(Motor);
wait(3);
startMotor(Motor, -10);
untilEncoderCounts(-200, tape);
stopMotor(Motor);//stops;
}

if (SensorValue[floor3]==1)
{
startMotor(Motor, 15);
untilEncoderCounts(300, tape);
stopMotor(Motor);
wait(3);
startMotor(Motor, -10);
untilEncoderCounts(-300,tape);
stopMotor(Motor);//stops

}

}
```

Reflection

The code, after the light sensor was out of the running, was the hardest part of this process. Our first problem was that this code would skip over the second start motor, so it wouldn't go back down to the ground floor. We figured out this was because everytime the encoder stops, it re-zeros, so when we told it to go back to zero, it already had. We changed that so it went back to the negative value of what it went up by. The second big thing that went wrong was the loops. At first, we didn't have any, so you could only press the buttons in the order they were coded in. So, when we added these, it would only work when a button was held down, not when it was pressed. It also started skipping around to different bits of the code, and I still don't honestly know what happened. When we came back after a weekend, I opened it back up, got rid of some pseudo code, put a section back into the code and it started working. So, now it can go to any floor, come back down to the ground floor, and repeat.