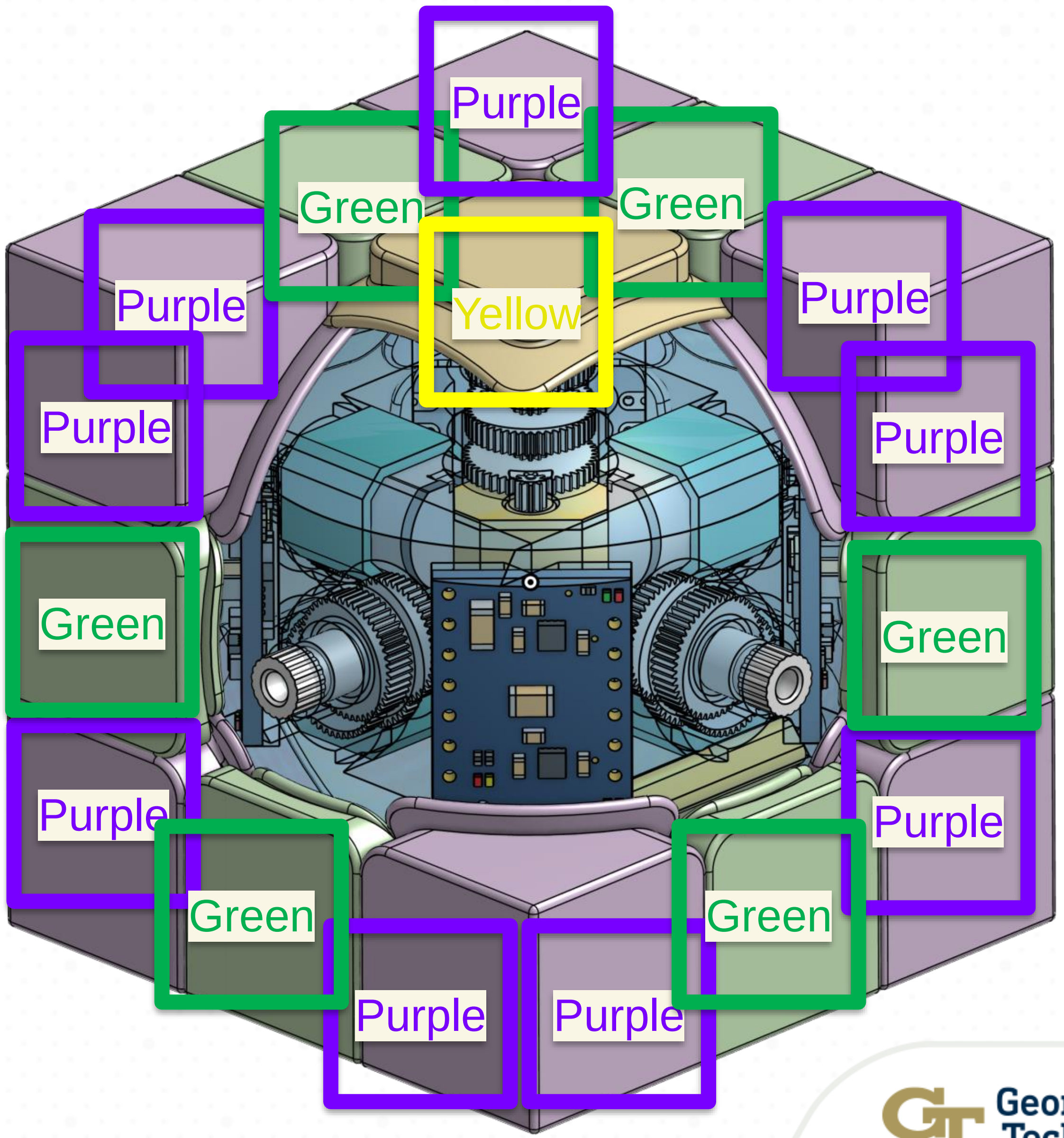
RUBI

Hilmy Rukmana, Shreya Terala

ME4405 – Fall 2024

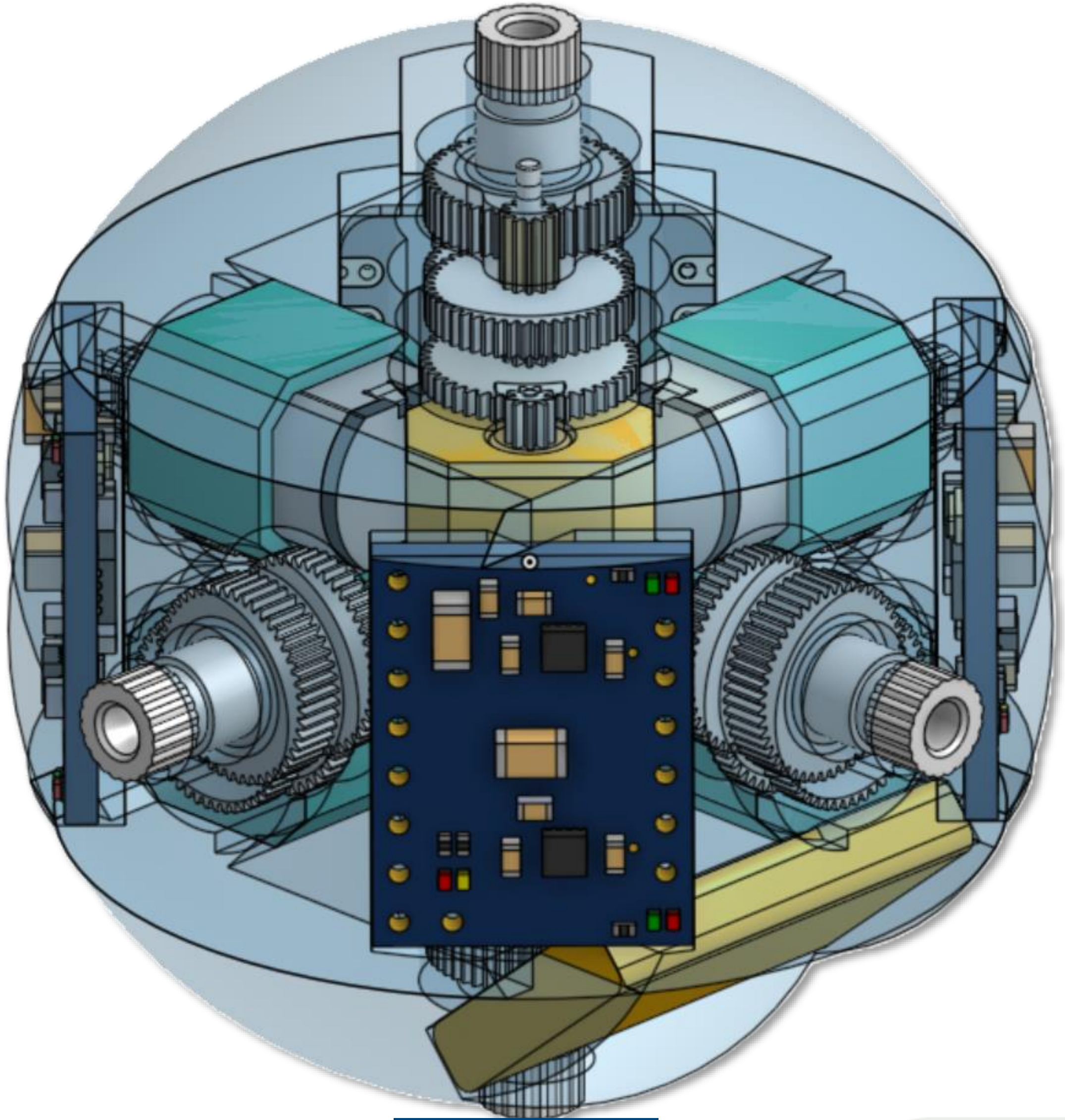


Computer Vision-ed Isometric View

Georgia Tech

Introduction & Motivation

- When first trying to solve a Rubik’s cube, the number of permutations (43,252,003,274,489,856,000) may appear daunting and discouraging to someone who has yet to solve one.
- Hence, creating a self-solving Rubik’s cube that will act as learning tool for users to understand how a Rubik’s cube is solved will reduce the barrier to learning.
- The cube can “assist” the user by showing them the next X amount of moves necessary to solve the cube.
- CV will be used to determine the scrambled position. Hence, the cube will solve based on algorithmic based instructions rather than undoing the moves used to scramble it.



Internal “Core” of RUBI

Georgia Tech

System Snapshot

LAPTOP

RUBI Bot

CV
Identifies colors on each face and generates state string

scrambled state string

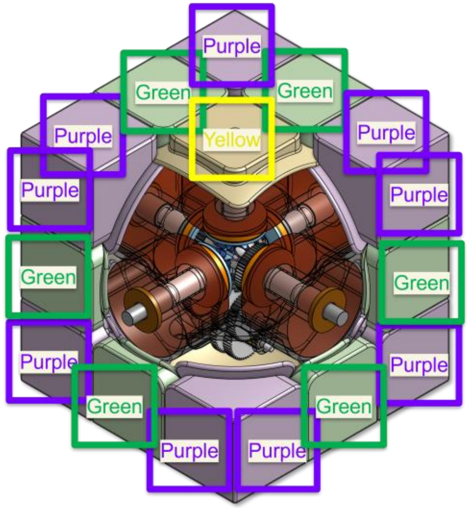
SOLVER
Identifies steps for solution and coverts them into moves

via Bluetooth

ESP32
Conveys moves as commands to the motor & identifies when the cube is still

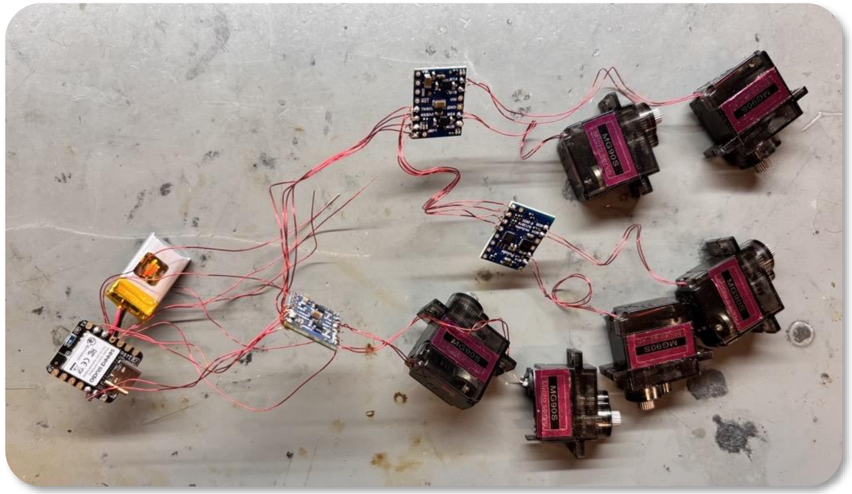
via I2C

DC Motor
Moves one face of RUBI at a time



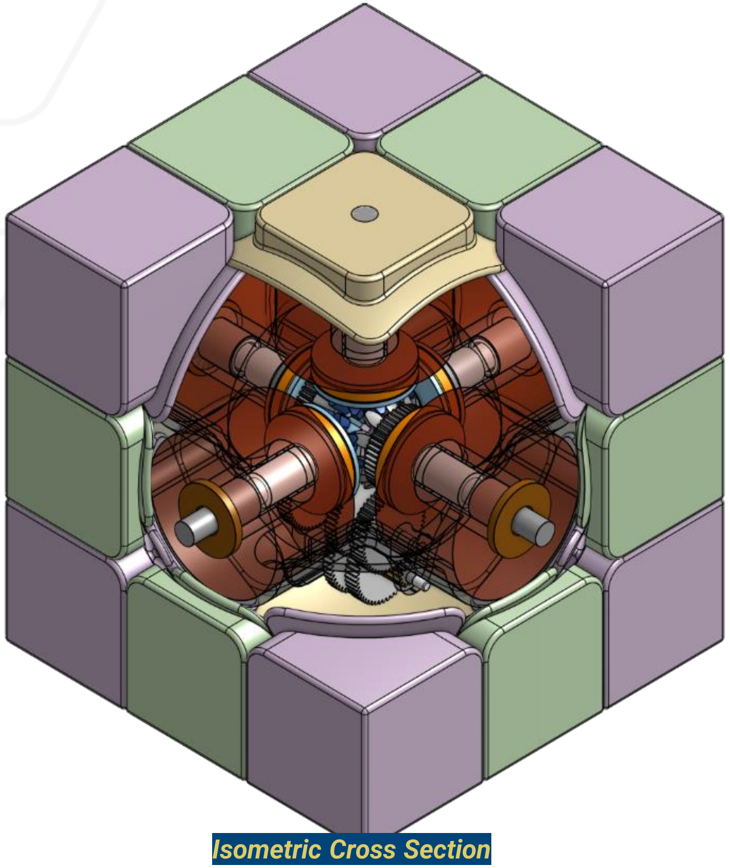

```
Please enter the scrambled Rubik's Cube string (54 characters):
  DFRRUBLBLBLFURRBFBUUURFLUDRRBDRDUUDDLDFLLFRBFDLFUBDLFB
```

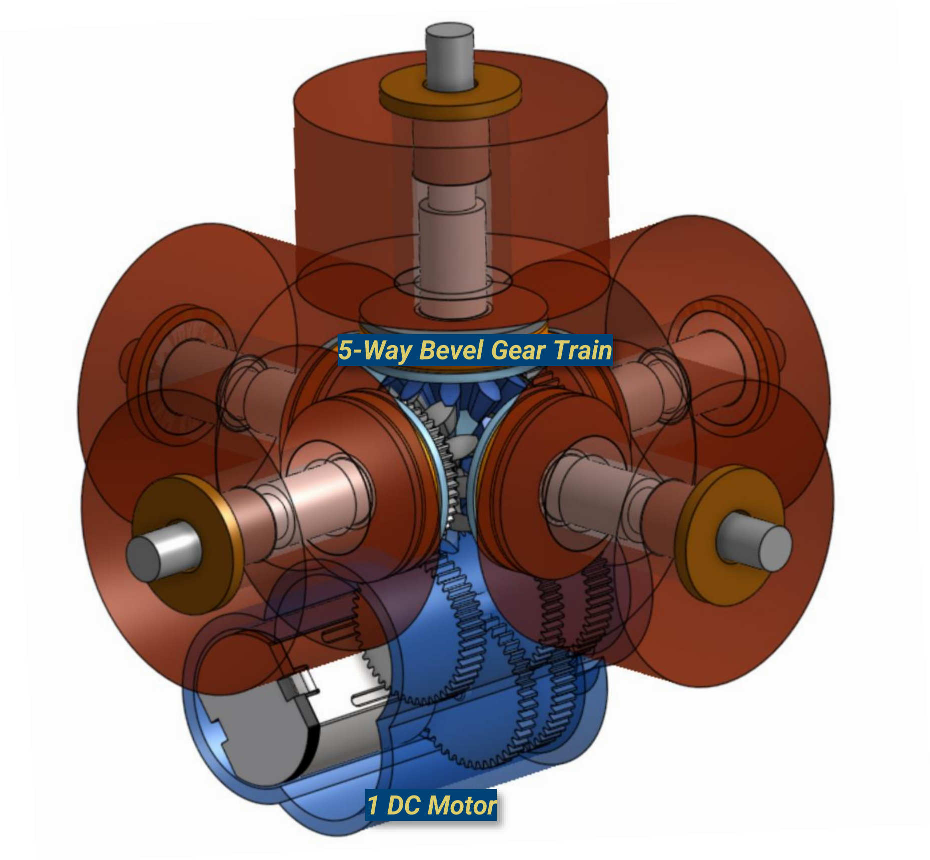
```
Solving the scrambled cube...
Original Solution: U2 B2 U' F U R D' L U' B2 R' B'
                   D2 L2 U2 L2 F2 D F2 U' F2 U2
```

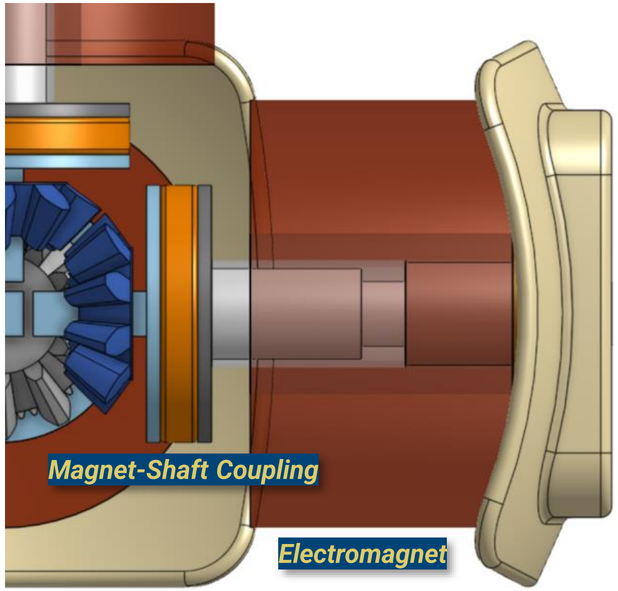


Georgia Tech

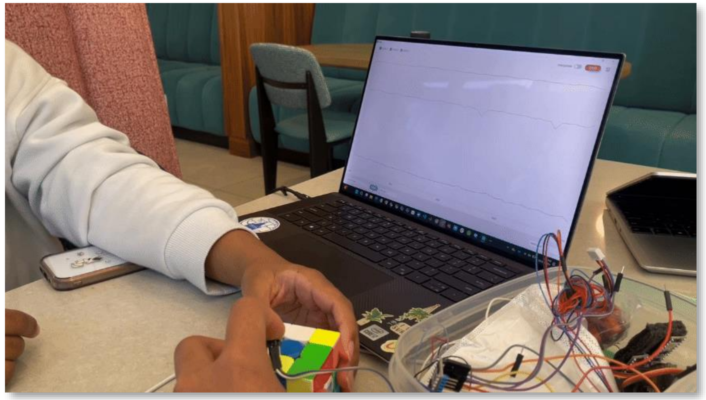
System Snapshot – Ver. 1



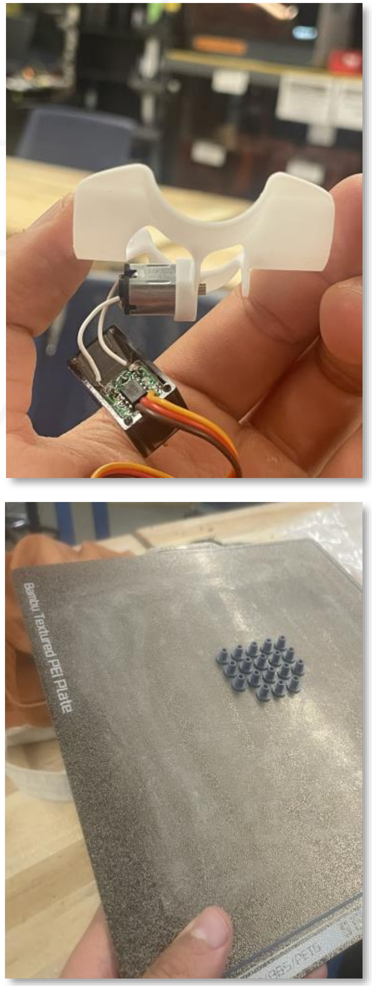




System Snapshot – Ver. 1

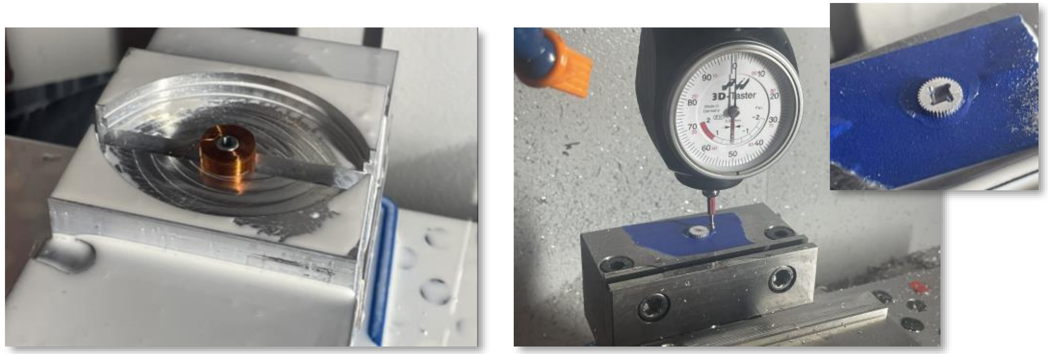


Accelerometer over I2C Works!

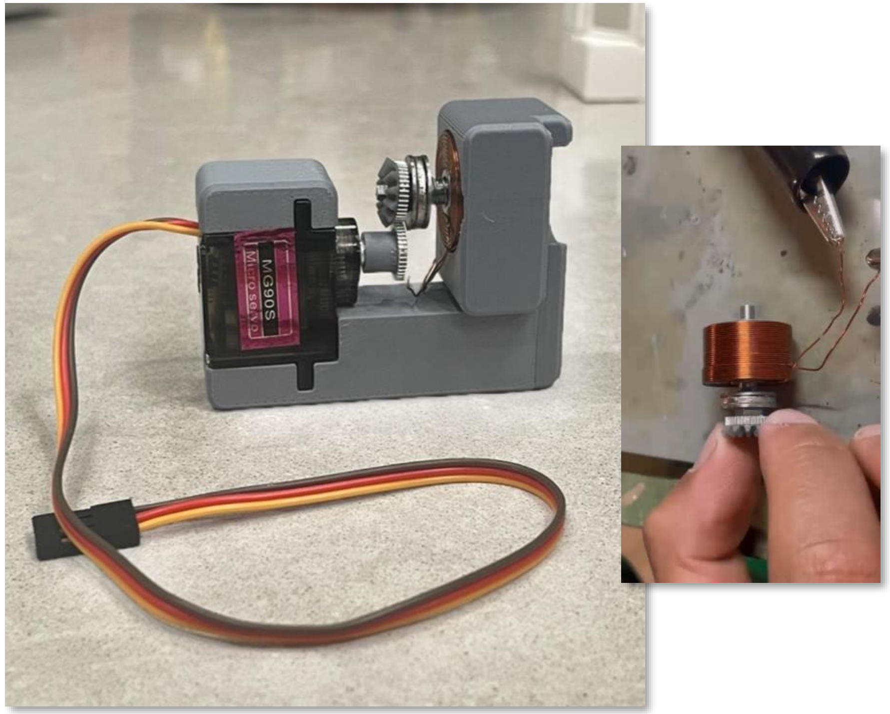


3D Printing Mfg.

Is manufacturing possible at this scale? | CNC Mfg.



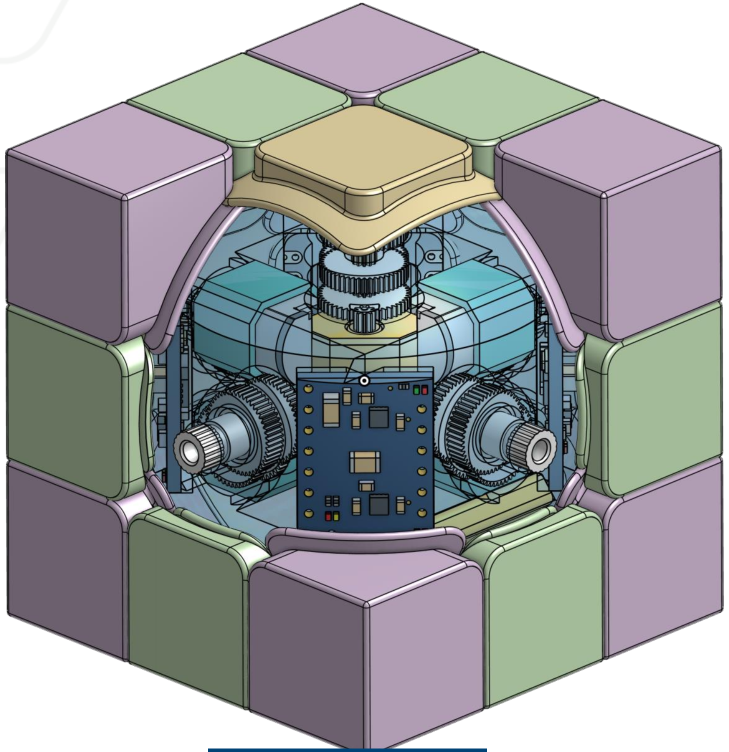
Modifying Off-the-Shelf Components

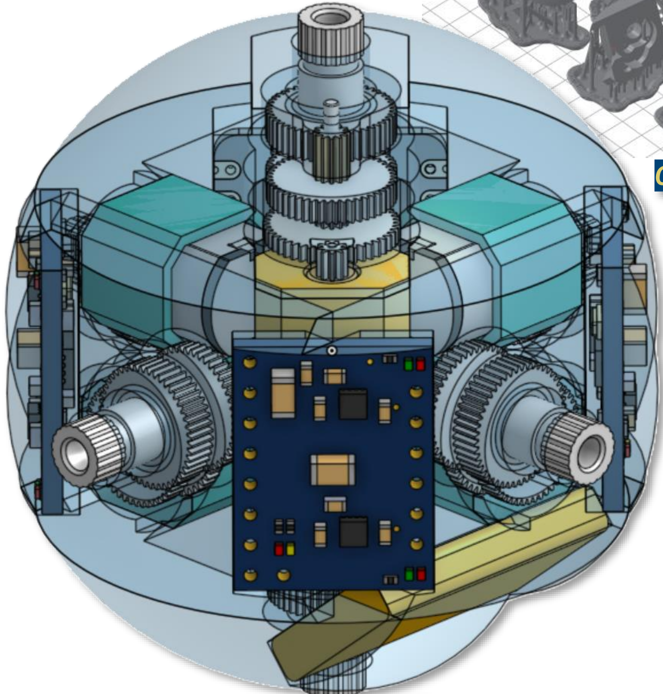


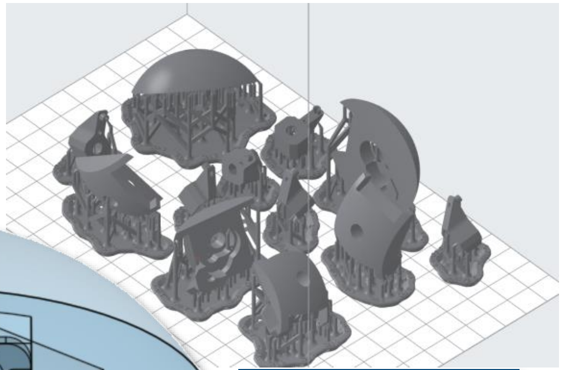
Electromagnet Test Platform

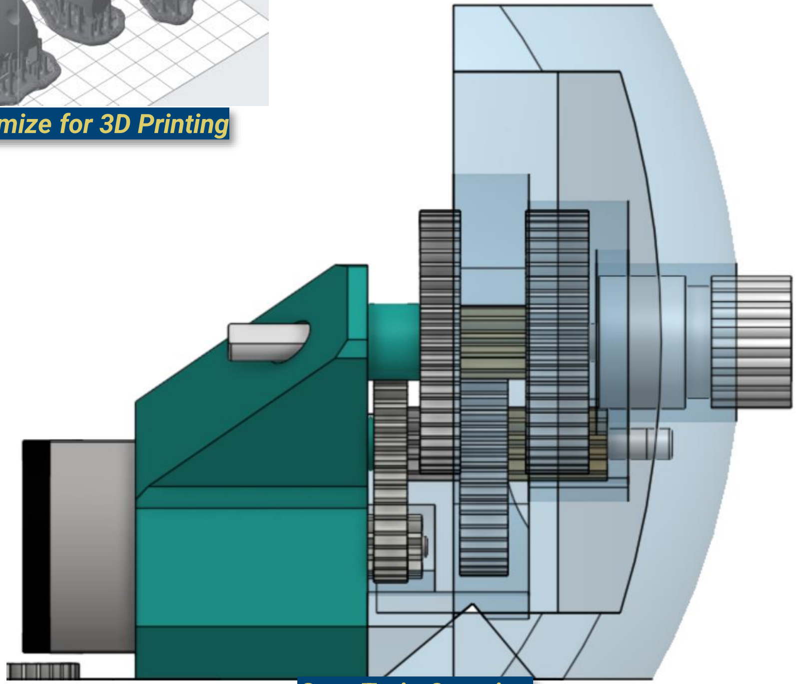
Georgia Tech

System Snapshot – Ver. 2


Isometric Cross Section


6 DC Motors

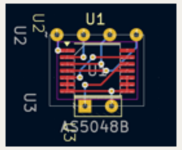
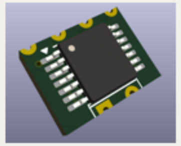

Optimize for 3D Printing

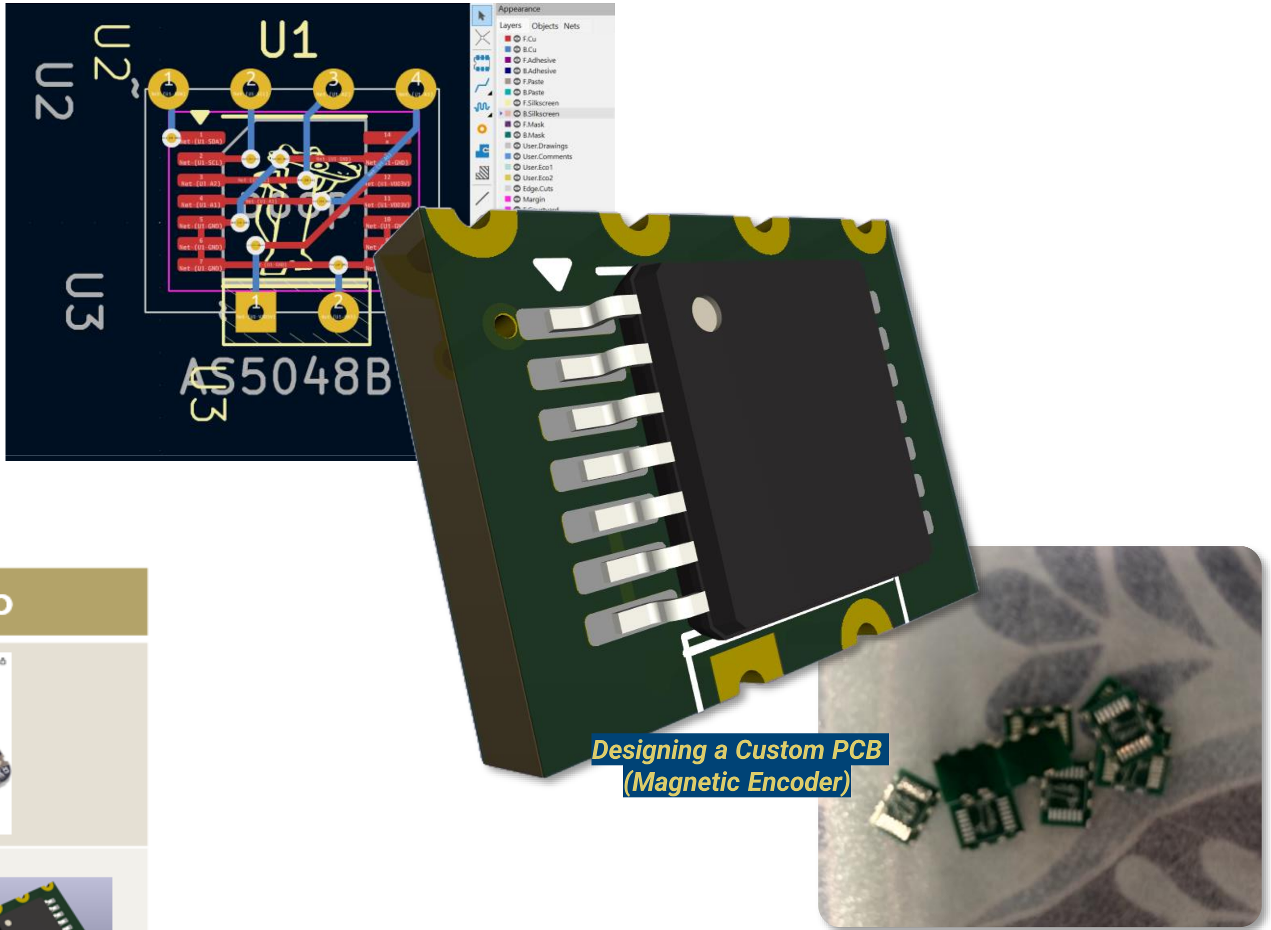

Gear Train Overview

Sensor & Actuator selection

MICROCONTROLLER & SENSORS

Component	Description
XIAO nRF52840 Sense	Microcontroller & Accelerometer
AS5048B	Magnetic Encoder

Requirements	Specifications	Product Photo
Size: Smaller than 30 x 30 mm (fit inside Rubik's Cube Core) **Bluetooth:** YES	**Size:** 21 x 17.5 mm **Bluetooth:** YES	
Comm: I2C/SPI **Size:** As small as possible	**Comm:** I2C **Size:** 10 x 7 mm (custom PCB)	 



Designing a Custom PCB (Magnetic Encoder)

Sensor & Actuator selection

ACTUATORS

Component	Description
M2T550	Motor Controller
MG90s	“DC” Motors

Requirements	Specifications	Product Photo
Comm: I2C/SPI **Control:** 2 motors **Size:** Smaller than 30 x 30 mm	**Comm:** I2C **Control:** 2 motors **Size:** 15.3 x 20.4 mm	
Min Torque: 0.0883 Nm **Size:** As small as possible	**Min Torque:** 0.19 Nm @ 4.8 V **Size:** 8 x 10 x 12 mm	

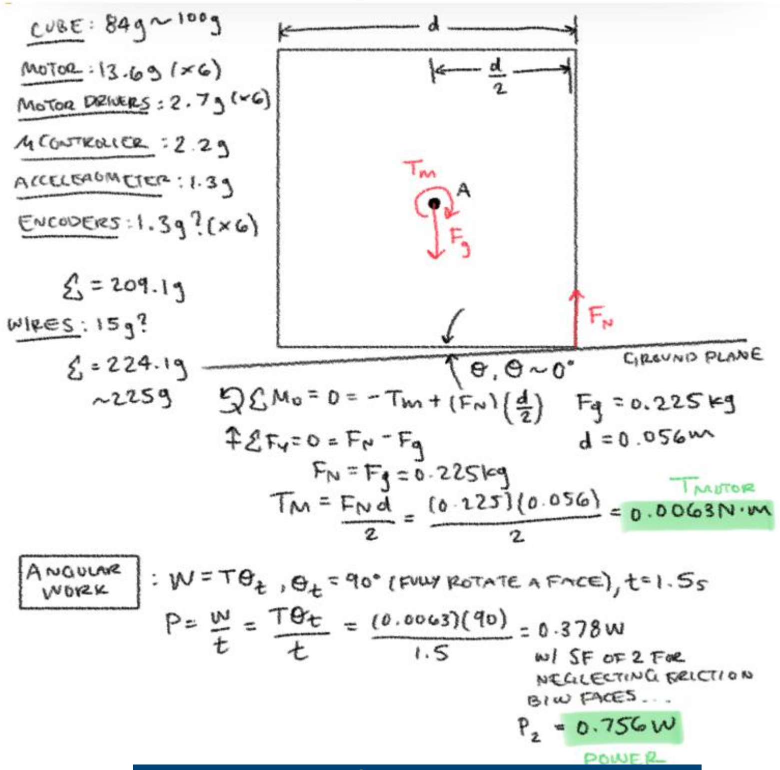
JeremyG • 5y ago • Edited 5y ago
Sub-practice(CN Roux) PB: 5.06

Roux is very efficient, I think I averaged around 45 moves last time I checked.

E: just did 7 solves at 1 TPS (metronome) and got 42, 48, 35, 40, 40, 43, 40 which gives a 41ish average

Source: [Reddit]

60 seconds / ~40 moves → 1.5 sec / 90 degrees

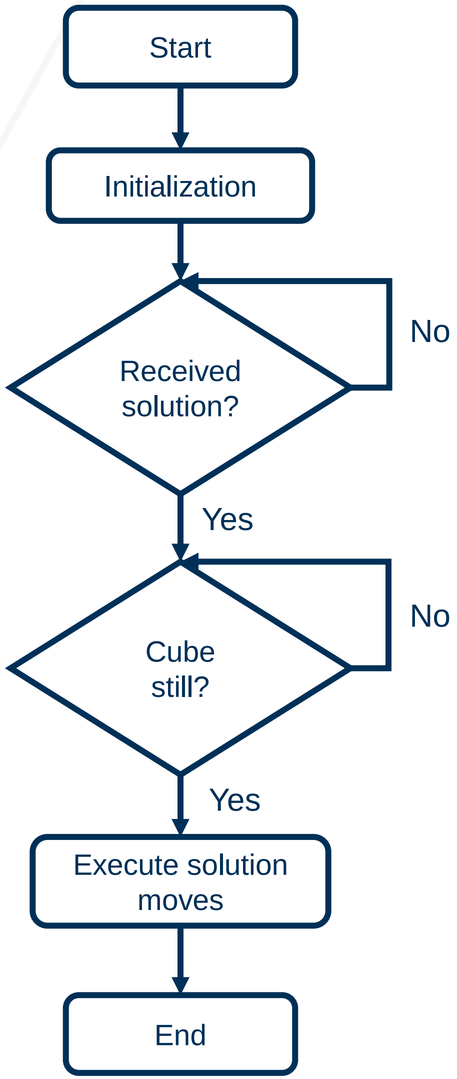


Hand Calculations for Minimum Torque Required

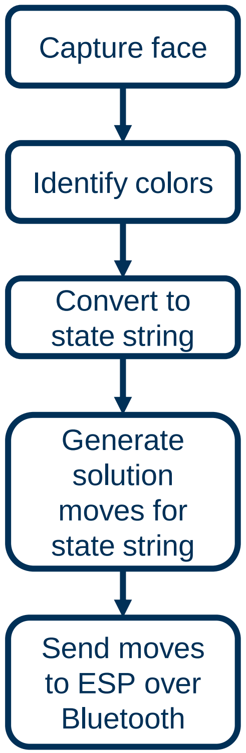
Georgia Tech

State Machine Diagram

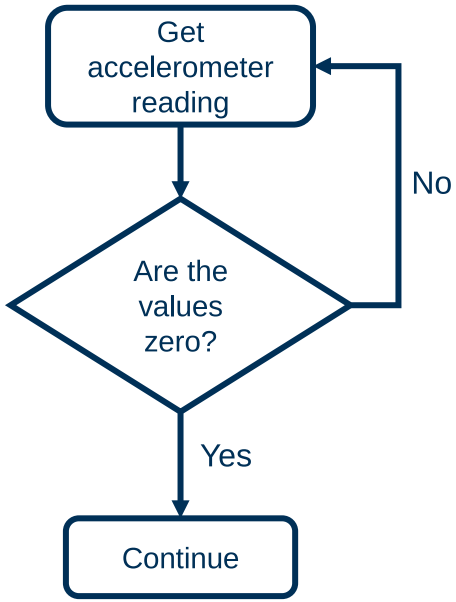
RUBI BOT HIGH LEVEL



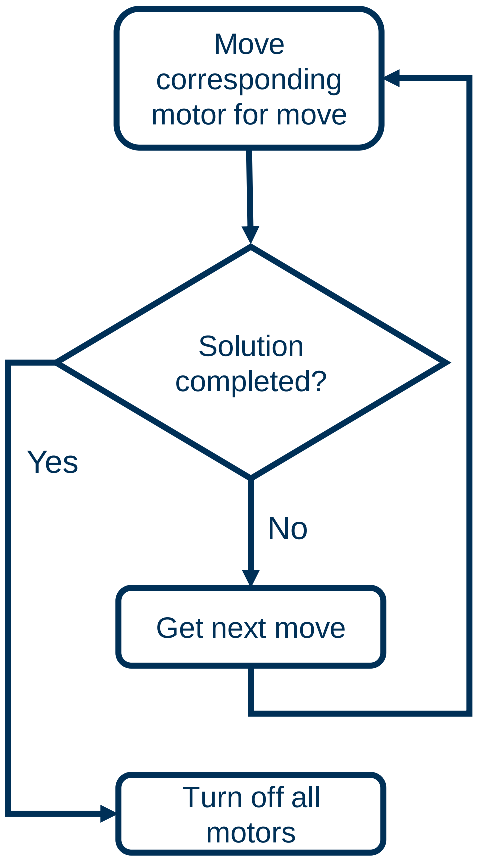
RECEIVING SOLUTION



ACCELEROMETER

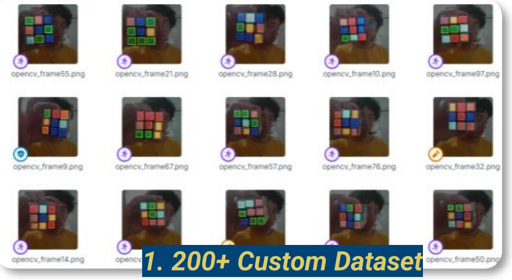


EXECUTING SOLUTION

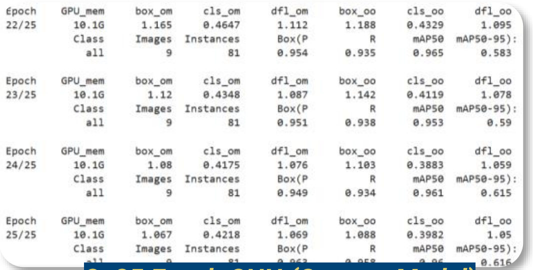


Demonstration

CV + Solution Solver

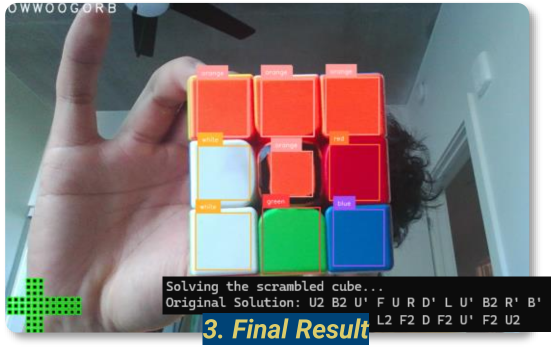


1. 200+ Custom Dataset



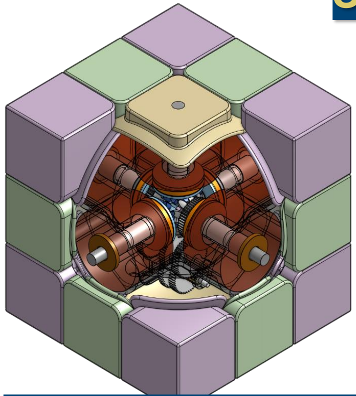
```
Epoch    GPU_mem   box_om   cls_om      dfl_om   box_oo   cls_oo   dfl_oo
22/25    10.1G     1.165    0.4647      1.112    1.188    0.4329   1.095
         Class     Images   Instances   Box(P    R        mAP50    mAP50-95):
         all       9        81          0.954    0.935    0.965    0.583

Epoch    GPU_mem   box_om   cls_om      dfl_om   box_oo   cls_oo   dfl_oo
23/25    10.1G     1.12     0.4348      1.087    1.142    0.4119   1.078
         Class     Images   Instances   Box(P    R        mAP50    mAP50-95):
         all       9        81          0.951    0.938    0.953    0.59

Epoch    GPU_mem   box_om   cls_om      dfl_om   box_oo   cls_oo   dfl_oo
24/25    10.1G     1.08     0.4175      1.076    1.103    0.3883   1.059
         Class     Images   Instances   Box(P    R        mAP50    mAP50-95):
         all       9        81          0.949    0.934    0.961    0.615

Epoch    GPU_mem   box_om   cls_om      dfl_om   box_oo   cls_oo   dfl_oo
25/25    10.1G     1.067    0.4218      1.069    1.088    0.3982   1.05
         Class     Images   Instances   Box(P    R        mAP50    mAP50-95):
                                                                   0.616
```

2. 25 Epoch CNN (Custom Model)

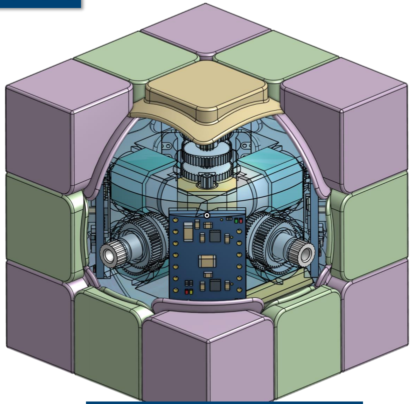


3. Final Result

Solving Cube

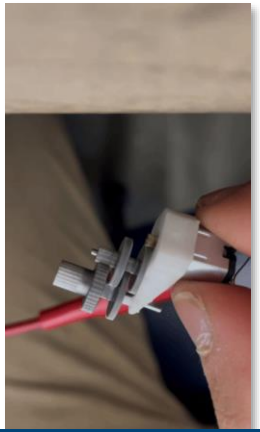


V1. Transistors + Electromagnets (not enough pinouts)



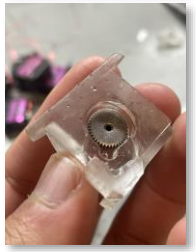
V2. Motor on Each Face

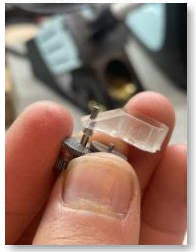
Controlling Motor Over I2C



DC Motor & Gear Train Works in More Compact Enclosure

DC Motors (partially) Work in Sphere Enclosure





Mfg!

How well did the system perform?

- Control of all 6 motors w/ 3 Drivers over I2C?
  - Why? Minimizing the number of drivers provides more space for bigger motors or batteries. Communicating over I2C provides pins for other non-I2C devices (e.g., status LED’s).
  - How? Daisy-chaining motor drivers to minimize form factor and changing addresses of the driver boards to enable this form of I2C communication
- Speed of solving Rubik’s cube?
  - Why? The faster the Rubik’s cube can solve and spin its faces, the faster a user can understand how to solve it. Providing a higher top speed affords lower speeds to operate at a longer duration.
  - How? Maximizing the motor RPM of the necessary, minimum torque.
- Battery life?
  - Why? Although obvious, the longer the battery lasts, the more solves can be completed
  - How? Minimizing stalling of the motor through proper tolerancing means lower current draw from battery

Conclusion

- Challenges
  - Minimizing size while retaining performance of electronics
  - Packaging and tolerancing
  - Training CV to work in a variety of environments
- Takeaways
  - I2C is incredible (communicated with encoders, accelerometers, and motor drivers concurrently).
  - Dealing with small mechanical components is a pain.
  - Dealing with small electrical components is a pain.
  - Dealing with both is awful.
  - Side projects aren't as important as the main task at hand.
  - Using CV and communicating via Bluetooth at the same time using Python threading
- Future Steps
  - Electronics packaging iteration
  - Integrating sensors (i.e. encoders) for accurate face movement tracking
  - Implementing "buddy" mode

Georgia Tech