Self-Aligning Wireless Charger

Jeremy Noesen, Noah Pritchard, Greg Matson, Remington Greatline, Malakhi Barkley, John Welch

Our Team

Jeremy Noesen Computer Engineering

Noah Pritchard Electrical Engineering

Gregory Matson Computer Engineering

Remington Greatline Electrical Engineering

Malakhi Barkley Software Engineering

John Welch Computer Engineering

Key Contributions

- **Jeremy**
  - Wrote hardware interface, main function, and tested them
- **Noah**
  - Soldered components, measured voltage through test coil, built half-wave rectifier
- **Greg**
  - Modeled various parts in Solidworks, including the rails and charger case
  - Designed CAD drawings of the assembled charger
- **Remington**
  - Built half-wave rectifier, designed power supply, designed wire layout on motor shield
- **Malakhi**
  - Wrote coil scan and coil movement algorithms
- **John**
  - Modeled various parts in Solidworks, 3D printed parts, assembled charger in Solidworks

Introduction: Why

- Current wireless chargers require precise alignment
- Wanted a more user-friendly wireless charger

INIU Wireless Charger

- Very sensitive
- User must manually align phone

JOYROOM Wireless Car Charger

+ Moves the phone using clamps
- Made for cars only

Yoobao Toaster Charger

+ User friendly
- Phone size matters
- Removes many advantages of wireless charging

Introduction: What and How

What?

Create a wireless charger that automatically aligns the charging coil inside the charger with the receiver coil inside a phone.

How?

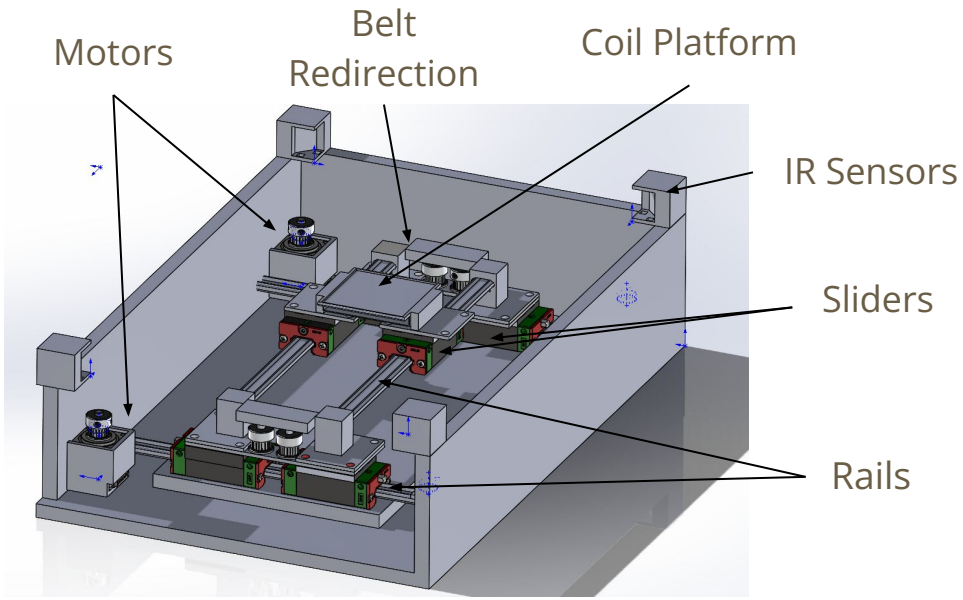
- Two-dimensional sled-rail system to move the coil around
- IR sensors to initially detect the phone
- Qi charging coil voltage to check alignment with devices

Important Note

- We completed every individual component and verified they work
- During integration, we ran into mechanical problems we do not have experience with
- We will show what we completed so far

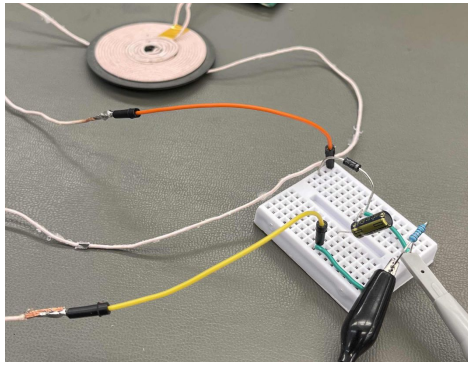
Implementation Architecture

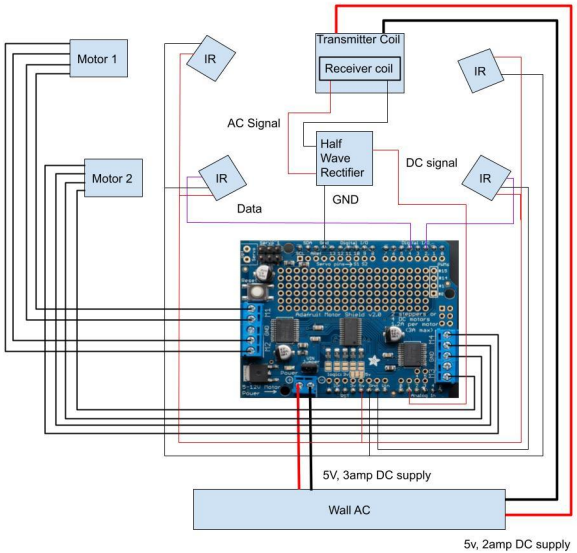
- Charging coil rests on center platform, which rests on 2 rails
- Each rail rests on a set of sliders
- 2 axis movement is achieved through 2 stepper motors, a timing belt, and pulleys
- Motors controlled by Arduino Uno and a stepper motor shield



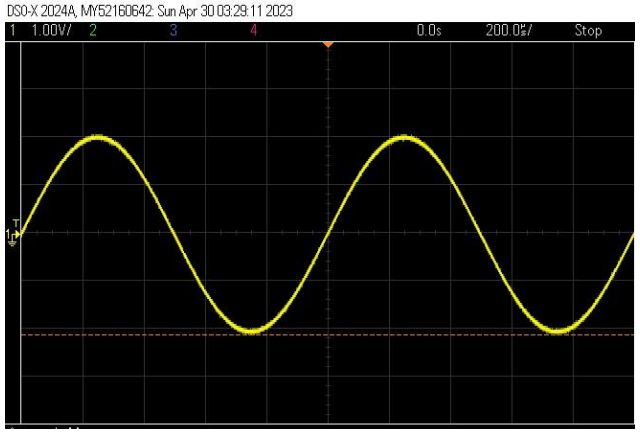
Work Accomplishments: Hardware

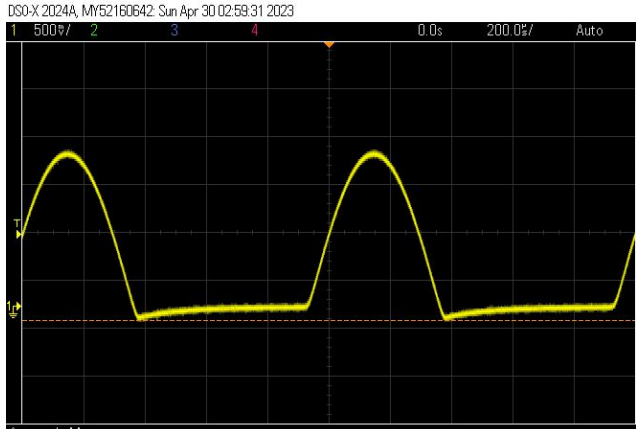
- Built and tested half-wave rectifier
- Constructed a secondary coil to measure the analog voltage of the charging coil
- Assembled interconnects between all components and the Arduino
- Ensured sufficient power supply to all components

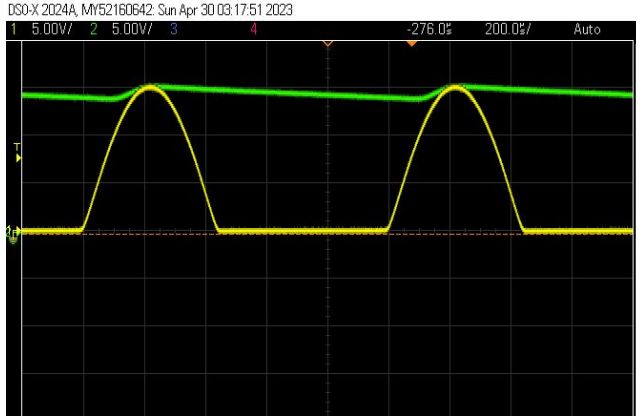




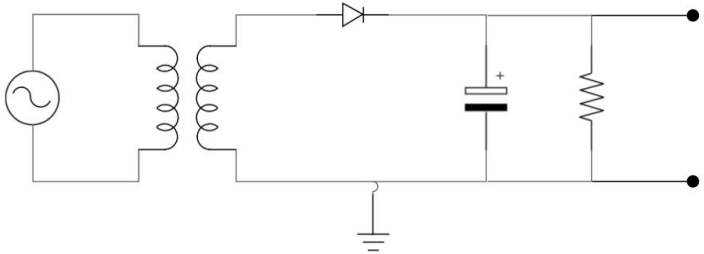
Half-Wave Rectifier



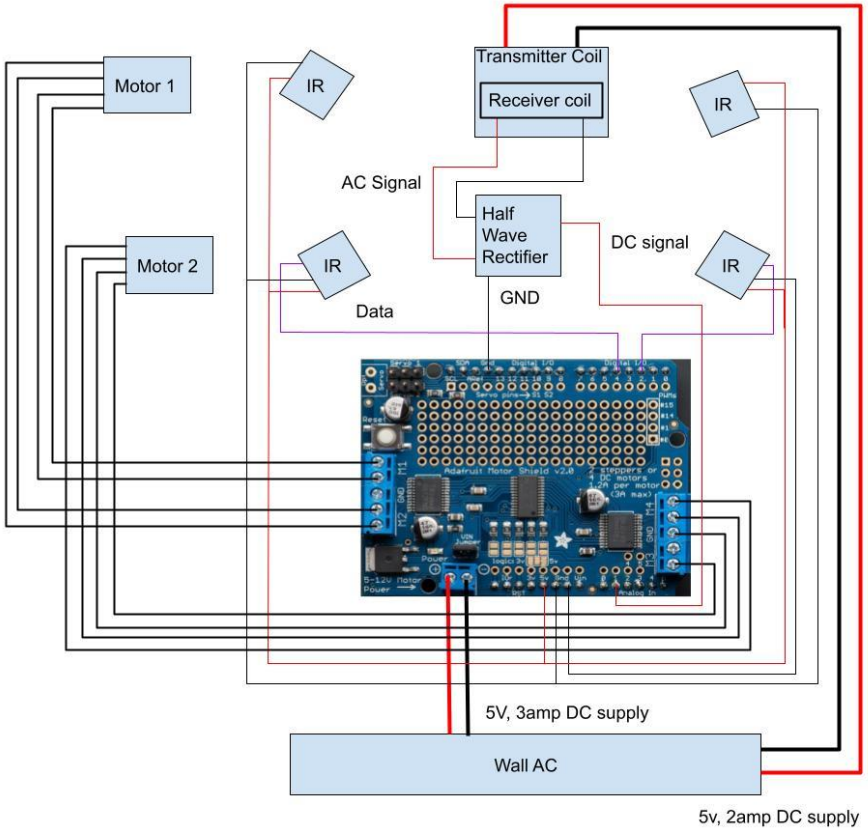




Schematic



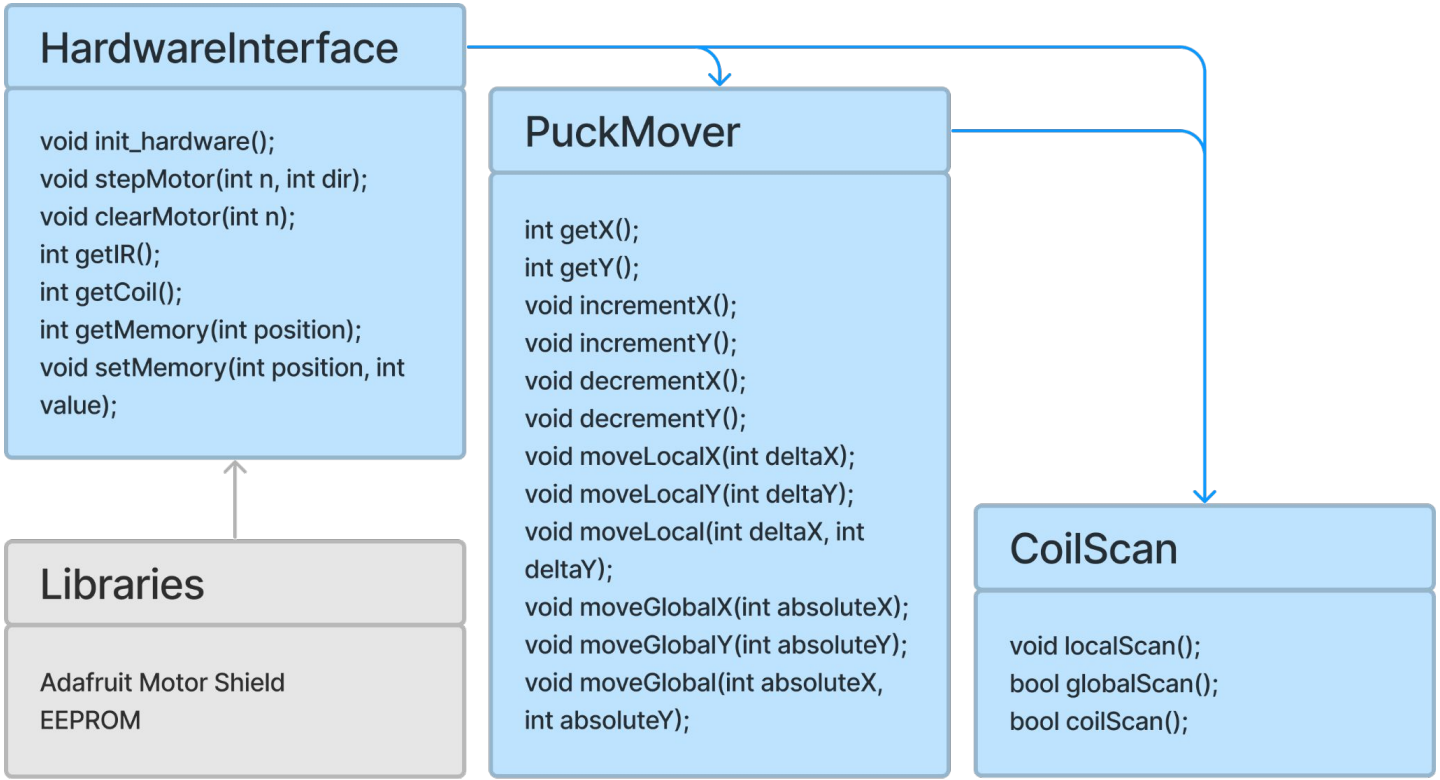
Arduino Motor Shield Interconnects



Work Accomplishments: Software

- All hardware components can be interfaced with the software
- Coil movement saved to EEPROM memory for persistent state
- Coil scanning algorithm would move coil in a spiral starting from center
- Unable to test movement and scanning algorithms with hardware
    - System was never fully and successfully assembled

Overview

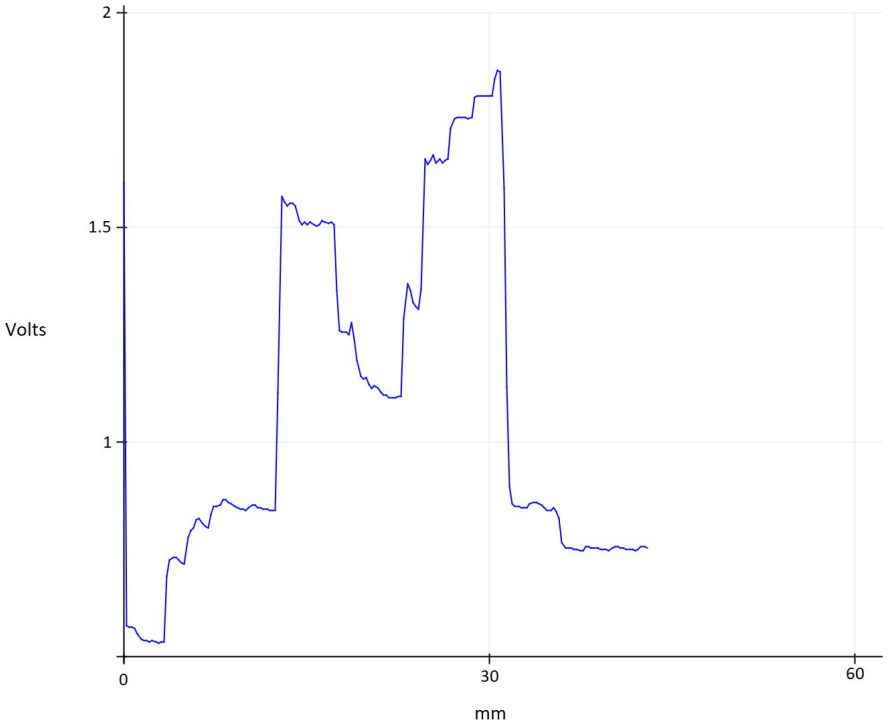

HardwareInterface
void init_hardware();
void stepMotor(int n, int dir);
void clearMotor(int n);
int getIR();
int getCoil();
int getMemory(int position);
void setMemory(int position, int value);
PuckMover
int getX();
int getY();
void incrementX();
void incrementY();
void decrementX();
void decrementY();
void moveLocalX(int deltaX);
void moveLocalY(int deltaY);
void moveLocal(int deltaX, int deltaY);
void moveGlobalX(int absoluteX);
void moveGlobalY(int absoluteY);
void moveGlobal(int absoluteX, int absoluteY);
CoilScan
void localScan();
bool globalScan();
bool coilScan();
Libraries
Adafruit Motor Shield
EEPROM

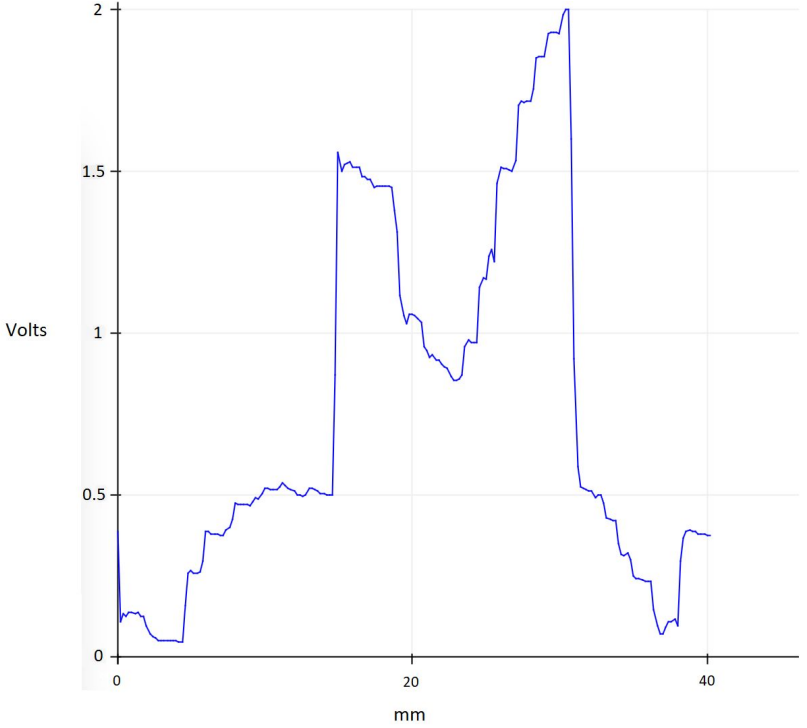
How It Works

```
if (getIR()) {
    if (!successful) {
        if (coilScan()) {
            successful = true
        } else {
            moveGlobal(0, 0);
            successful = false;
        }
    }
} else {
    stopScan();
    moveGlobal(0, 0);
    successful = false;
}
```

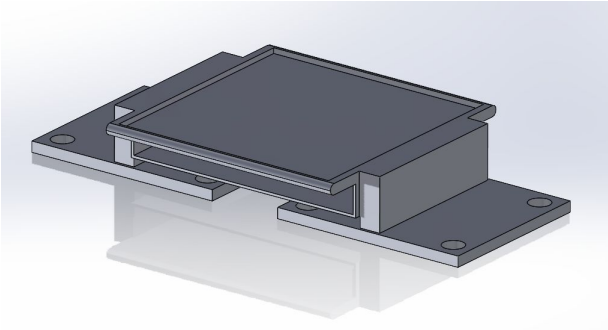
- If the IR beams are broken, and a scan has not completed, do a scan.
- If the scan fails or if the phone is removed from the charger, stop the scan and reset coil position.

Coil as an Analog Sensor



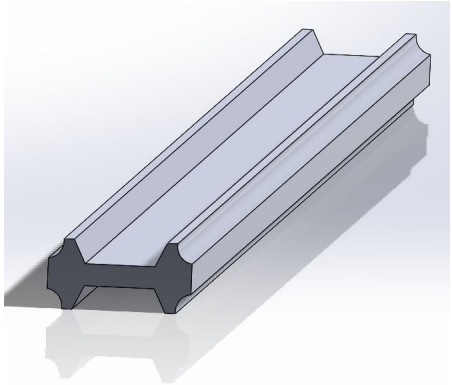


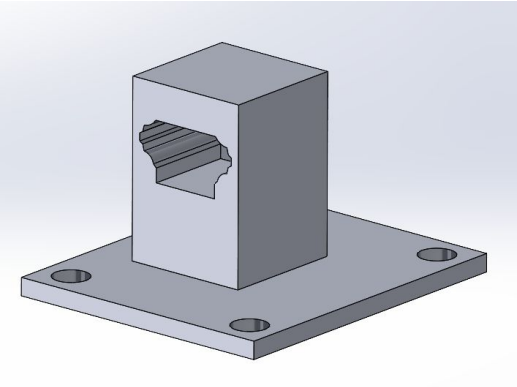
Work Accomplishments: 3D Models



Coil Platform

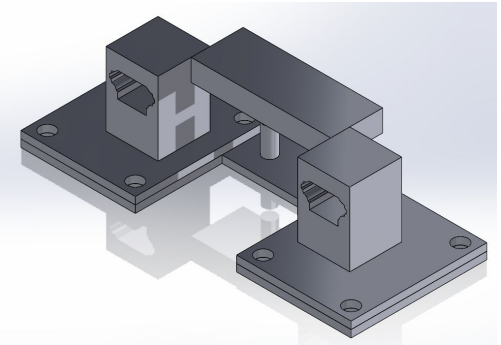
Rail





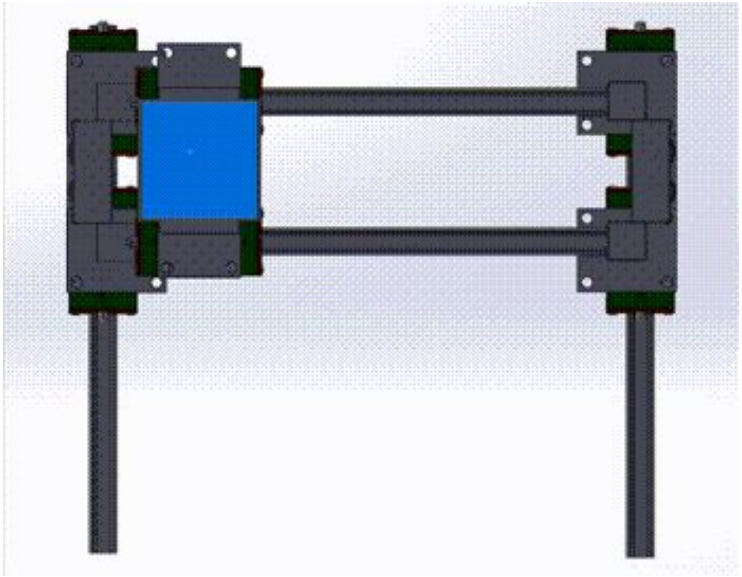
Slider Cover

Belt Redirection

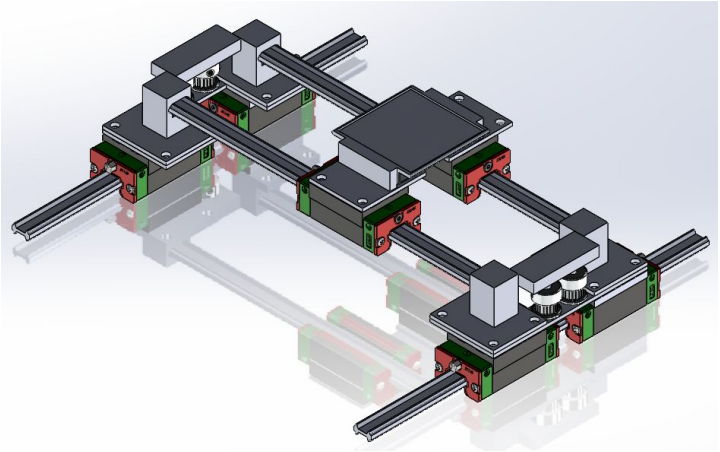


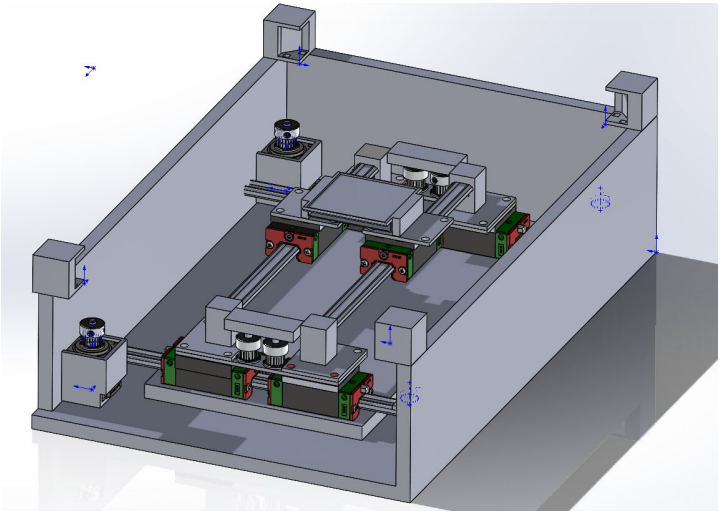
3D Models (cont.)

Ideal Movement

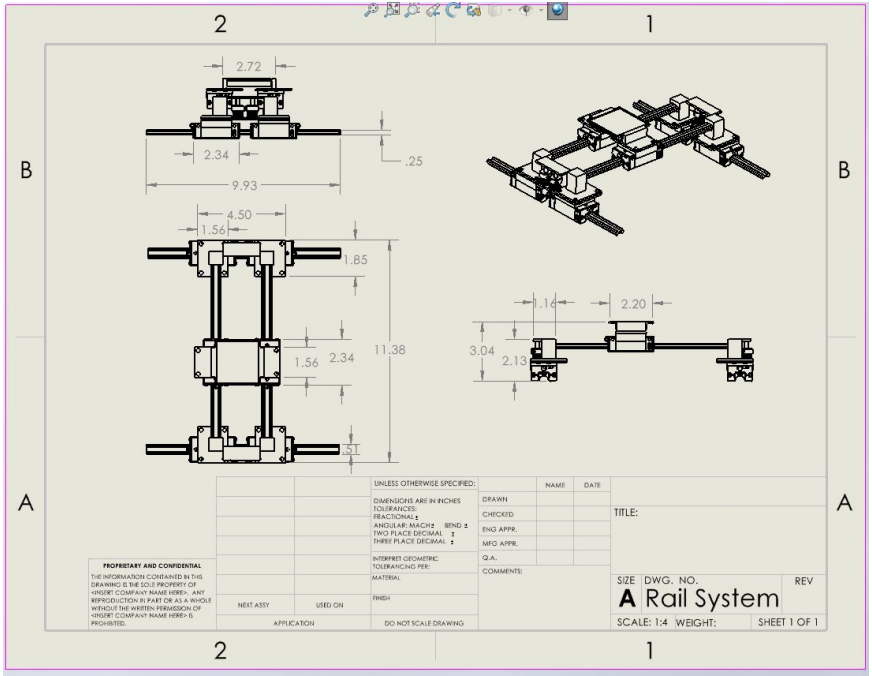


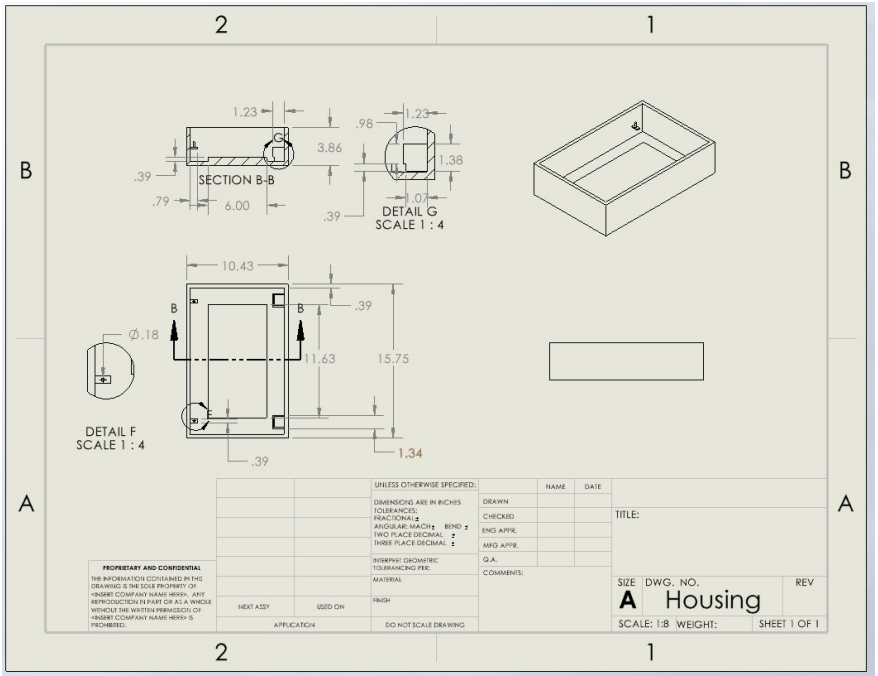
Assembled Charger





CAD Drawings





Estimated Cost

Parts	Prototype Cost
Arduino UNO Microcontroller	$34.99
Stepper Shield	$24.95
Mini Stepper Motor	$19.95
Transmitter Coil & PCB Board	$26.95
IR Sensors	$5.95
2D Rail System	$253.20
Total	**$365.99**

Challenges and Solutions

- Challenge: Measurements of 3D prints were slightly off
  - Solution: Print a custom part to test measurements, test, then utilize the results into the final print
  - Solution: Sand off edges of parts
- Challenge: Current cost of design is expensive
  - Solution: 3D print various parts
- Challenge: Motor recalibration is difficult
  - Solution: Use EEPROM to store position every move

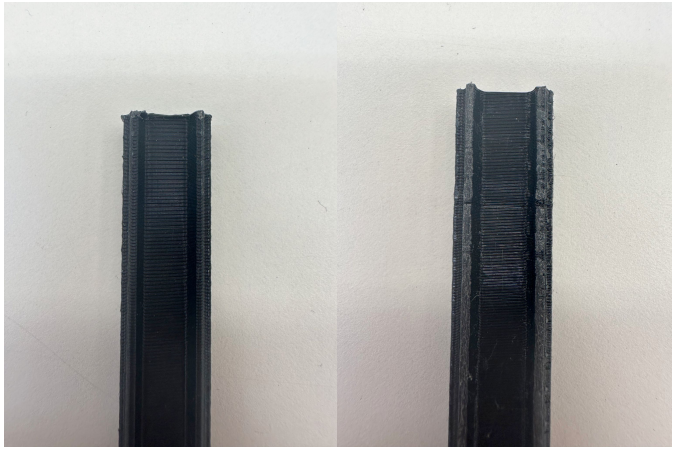
What Worked? What Didn't?

What worked?

- Charging coil
- Motor code
- IR sensor code
- 3D printing*

What didn't?

- System design

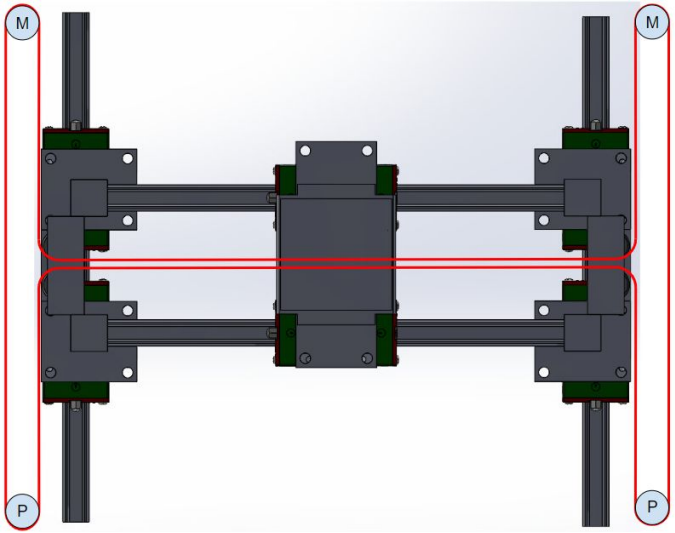


Issues to be Addressed

- Charging case
- Power supply
- Charging efficiency
- 3D printing quality
- Charger weight

Future Work

- Rework the rail system design
  - Could be more of a mechanical engineering project
  - Make design cost-efficient
  - Make design lighter
- More thorough coil testing post-assembly
  - Confirm that the phone charges optimally once hardware is completed



Conclusion

- We were able to prove that coil alignment can be found through the use of a secondary coil.
- There are many mechanical issues that are beyond our knowledge.
- Verified separate components
    - Integration of said components, on the other hand...