STEM Outreach

CS410 Capstone Project - PNNL
Last edited 3/19/2025

Alea Minar

In this document:

Project Plan

Management Plan

Team Organization

- Alea Minar - Software Lead
- Hank Childs - Faculty Mentor
- Nick Cramer - Industry Mentor

Work Division

Hardware:

- 3D printing
- Controls

Software:

- Python package
- Digital twin

Team Communication

- Discord for general communication with Nick
- Email for general communication with Hank
- Meetings/Scrums

Work Breakdown Schedule

Milestones

- Simplified 3D model
- Initial 3D print
- Finished 3D model
- Finished print
- Working microcontroller
- Python + Digital twin representation

Project Schedule

- ☑ ~~Contact MakerSpace about 3D printing - 1/15~~
- ☑ ~~Get account number from Hank - 1/23~~

- ☑ ~~Define requirements - 1/23~~
- ☑ ~~Plan project schedule - 1/23~~
- ☑ ~~Research what kind of hardware is necessary - 1/23~~
- ☑ ~~Order hardware - 1/27~~
- ☑ ~~Create simplified 3-D model - 1/24 - 1/31~~
- ☑ ~~Print 3-D model and ensure it meets size requirements - 2/3~~
- ☑ ~~Create final 3-D model - 2/3 - 2/7~~
- ☑ ~~Print 3-D model and ensure it meets size requirements - 2/10~~
- ☑ ~~Implement the hardware - 2/17~~
- ☑ ~~Write software necessary to match speed from hardware - 2/17~~
- ☑ ~~Implement digital twin representation - 2/24~~
- ☑ ~~Ensure documentation is up to date and create deliverable package for Nick - 3/3~~
- ☐ Discuss future check-in and sharing of project with Nick - 3/5

Monitoring and Reporting

- Weekly class-wide scrums on Mondays
- Meetings with Faculty Mentor as needed
- Biweekly meetings with Nick to update
- Email and Discord updates as needed

Implementation Plan

Steps

1. Contact MakerSpace about 3D printing
2. Get account number from Hank
3. Define requirements
4. Plan project schedule
5. Research what kind of hardware is necessary
6. Order hardware
7. Create simplified 3-D model
8. Print 3-D model and ensure it meets size requirements
9. Create final 3-D model
10. Print 3-D model and ensure it meets size requirements
11. Implement the hardware
12. Write software necessary to match speed from hardware
13. Implement digital twin representation
14. Ensure documentation is up to date and create deliverable package for Nick

15. Discuss future check-in and sharing of project with Nick

Intermediate Prototypes

- Initial simplified 3-D print to ensure correct measurements
- Finished 3-D print
- Digital twin representation

Deliverables

- STL/Blender file of the generator model
- STL/Blender file of the base
- Sensor schematics/instructions; Arduino code for revolutions
- Code implementation of sensor and digital twin (Python)
- Project documentation

SDS

Design Description

Hardware

Link to an excellent tutorial:
https://makersportal.com/blog/2018/10/3/arduino-tachometer-using-a-hall-effect-sensor-to-measure-rotations-from-a-fan

Required	Chosen	Purpose
Hall effect sensor x1	A3144	Detects magnetic field changes
Microcontroller x1	Arduino Uno	Processes sensor pulses and calculates rotational speed
Magnet x8	Neodymium magnets	Create the magnetic field detected by the Hall sensor as the gear rotates
Breadboard x1	Mini breadboard	Provides a platform for wiring components
10k resistor x1	10k resistor	stabilizes the Hall sensor's digital output

Total price (from Amazon): $40.25

Hardware Rationale

A3144 Hall effect sensor:
A3144 is a digital hall effect sensor that gives discrete outputs (“on” or “off”) and is a popular choice

Arduino Uno:
- Beginner friendly
- Widely available documentation
- Real-time processing
- Has features for RPM measurement

Neodymium Magnets:
Small neodymium magnets are ideal due to their strong magnetic field and small size.

Software

arduino

- **detect_magnet.ino**
    - To be downloaded to arduino. Uses hall effect sensor to determine if a magnet is detected.
- **magnet_test.ino**
    - Uses hall effect sensor to test if a magnet is detected.

digital_twin

- **dt_match.py**
    - The main application. Can be used to run the digital twin connected to an arduino or used with a simulated port.
- **dt_turbine.py**
    - Uses VTK to display and rotate the models at an rpm specified by user input.
- **simulate_rpm.py**
    - Used to simulate a serial port and randomly generate rpm.

Diagrams

Youtube link for the current system:
https://youtube.com/shorts/8WicbS5K2uE?si=RDwOSEg-Mbh8U_tr

Youtube link for a demonstration of digital twin and microcontroller:
https://www.youtube.com/watch?v=9bG94OCI_Cw

Diagram of a generator:

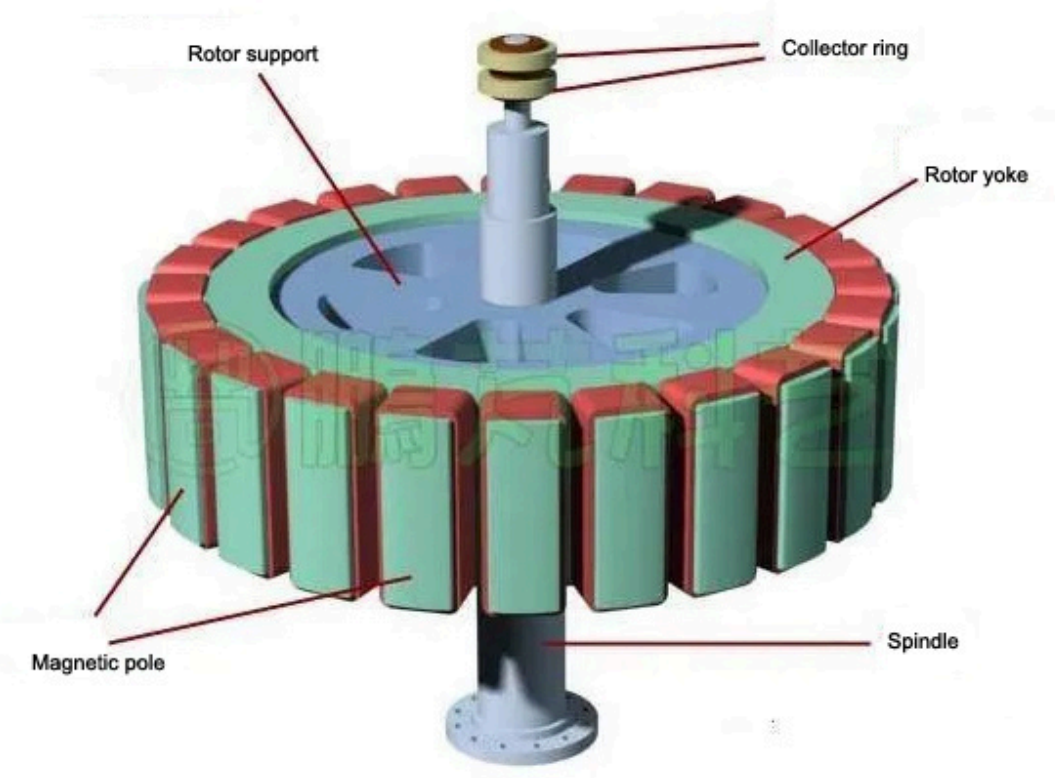
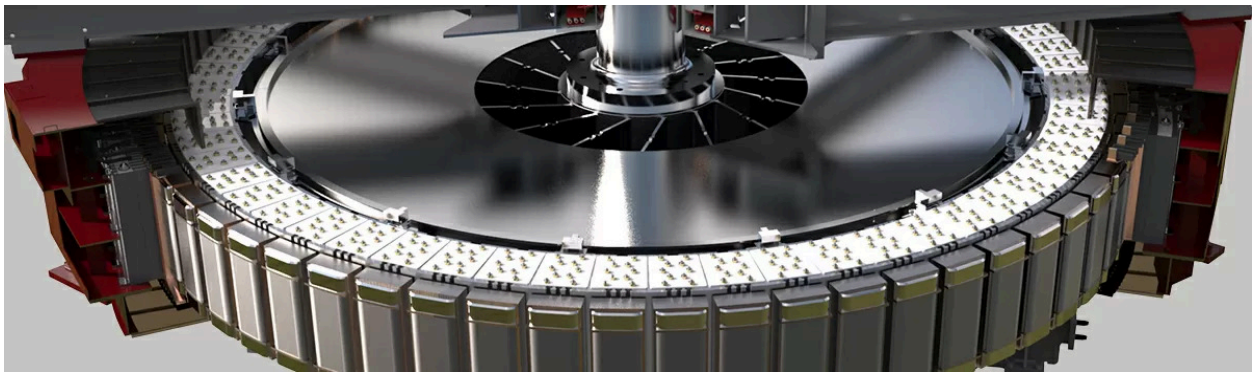
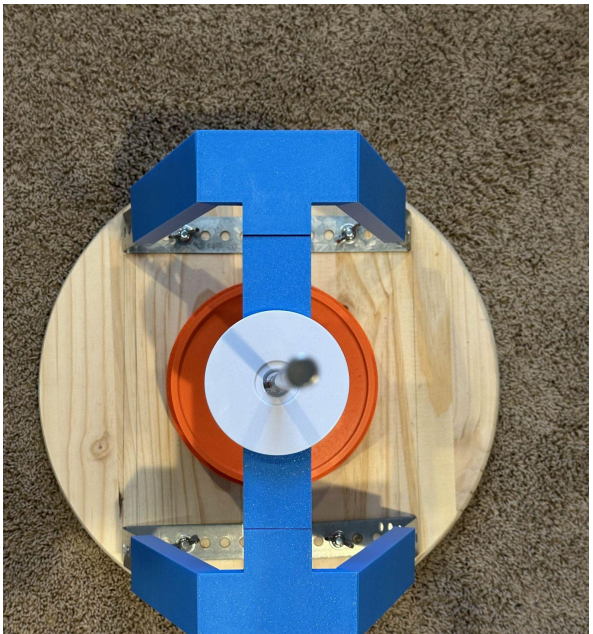
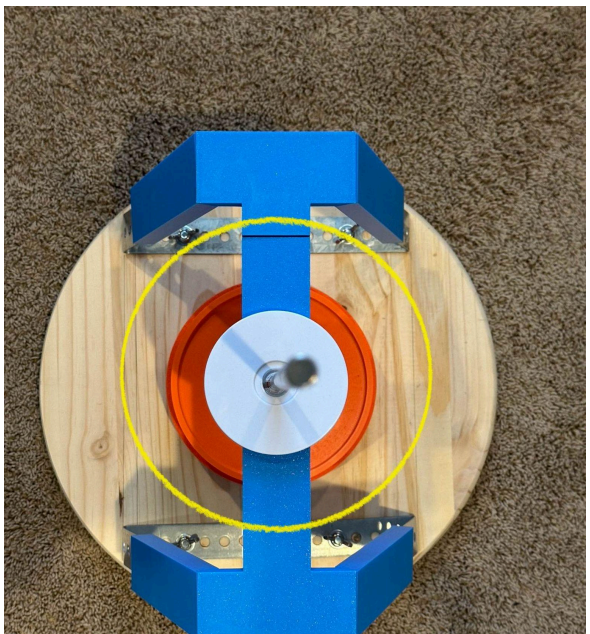


Image of a generator:

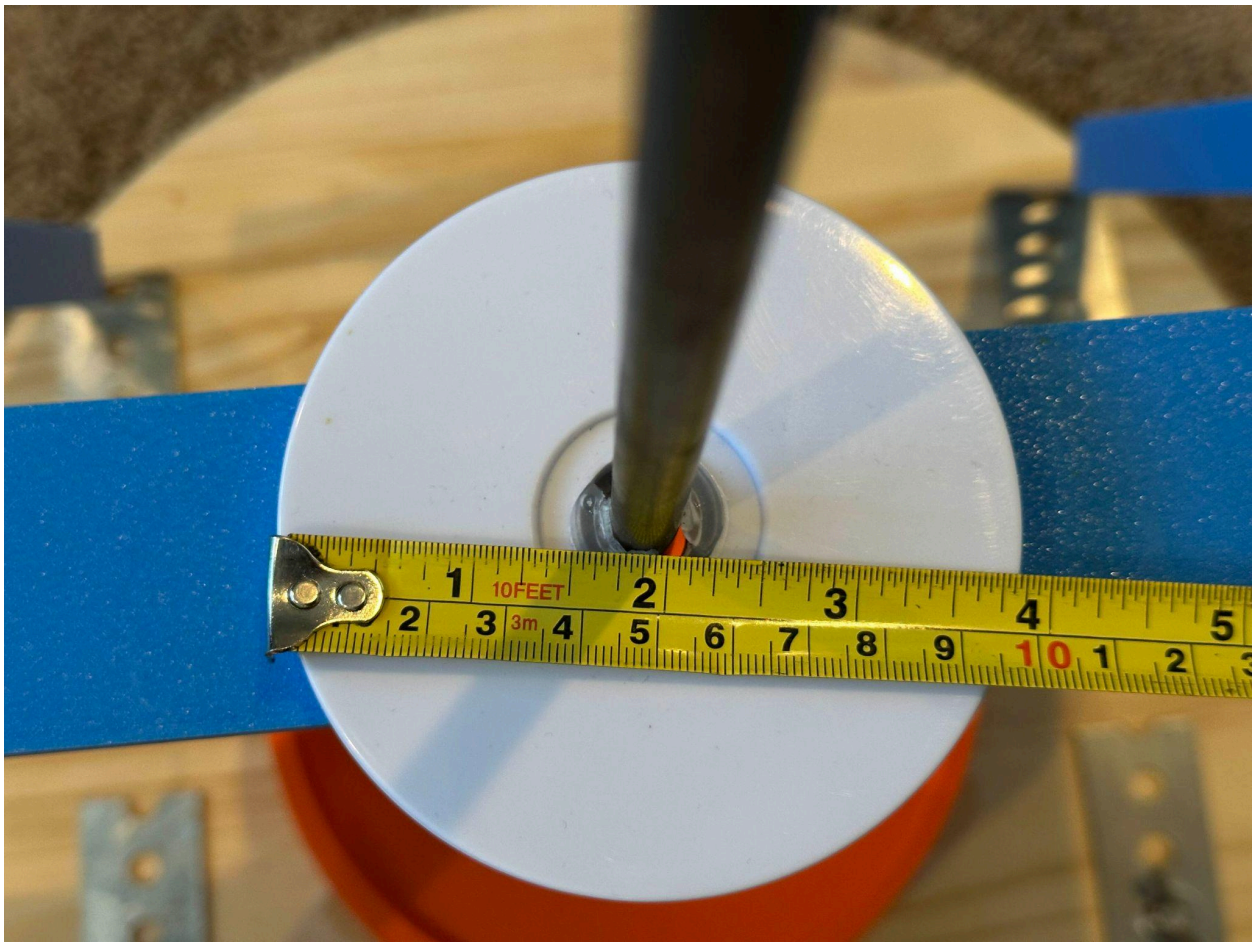


Images of the base, top-down:



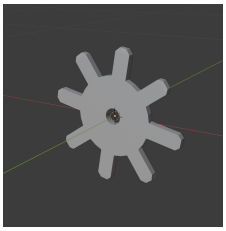
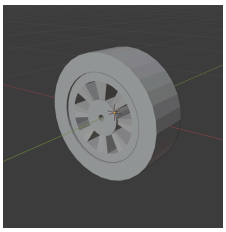
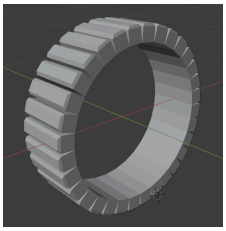
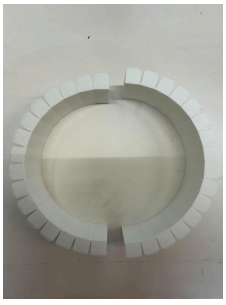
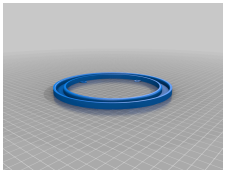
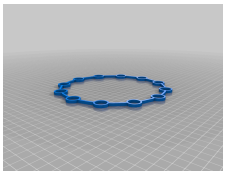


Images of the base detailing the measurements:



Iterations

V.1

Model name	Model Image	Printed Image	Dimensions (X-Y-Z)	Layer height
gear.stl			82.8-79.4-3.88 mm	0.2mm
generator_support_yoke			127-127-57.2 mm	0.24mm
magnetic_pole_v1			215.9(8.5")-215.9 (8.5")-63.5 mm (2.5")	0.2mm
Large_Turntable_Base_v3			180-180-8 mm	0.2mm
Separator_v3			180-180-3.97mm	0.2mm