Handoff Document STEM Outreach

CS410 Capstone Project

3/19/2025

Alea Minar

Repository Overview

Code

Contains all of the code required for installation and use of the application

arduino

- detect_magnet.ino
    - Uses hall effect sensor to determine if a magnet is detected and convert to rpm for digital twin match.
- 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.
- dynamic_base.stl
    - Model of the rotating part of the digital twin (generator, shaft, gear, and turbine).
- static_base.stl
    - Model of the stationary part of the digital twin (the base that supports the model).

Documents

Contains documentation associated with the project.

images

- 3D_prints
    - Images of what has been 3D printed for this project.
- diagrams
    - Diagrams of hydroelectric components.
- measurements

    - Images of Nick's model with associated measurements.
- models
    - Images of the 3D models designed for this project.

weekly_update_slides

The slideshows from (semi) weekly updates on the project.

Models

blend_files

- Contains the blend files for all of the models designed for this project

glb_files_for_ppt

- Contains glb files of some models used for powerpoints.

stl_files

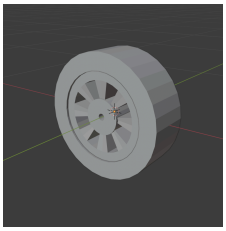
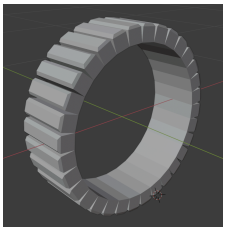
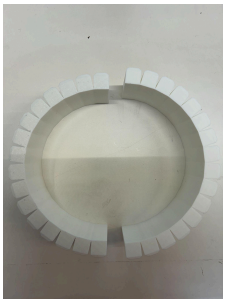
- Contains stl files of models that were either printed or used for visualization

Turntable, 18cm, marble bearings - 2763216

- Turntable sourced from Thingiverse user Zippitybamba, and is licensed under Creative Commons

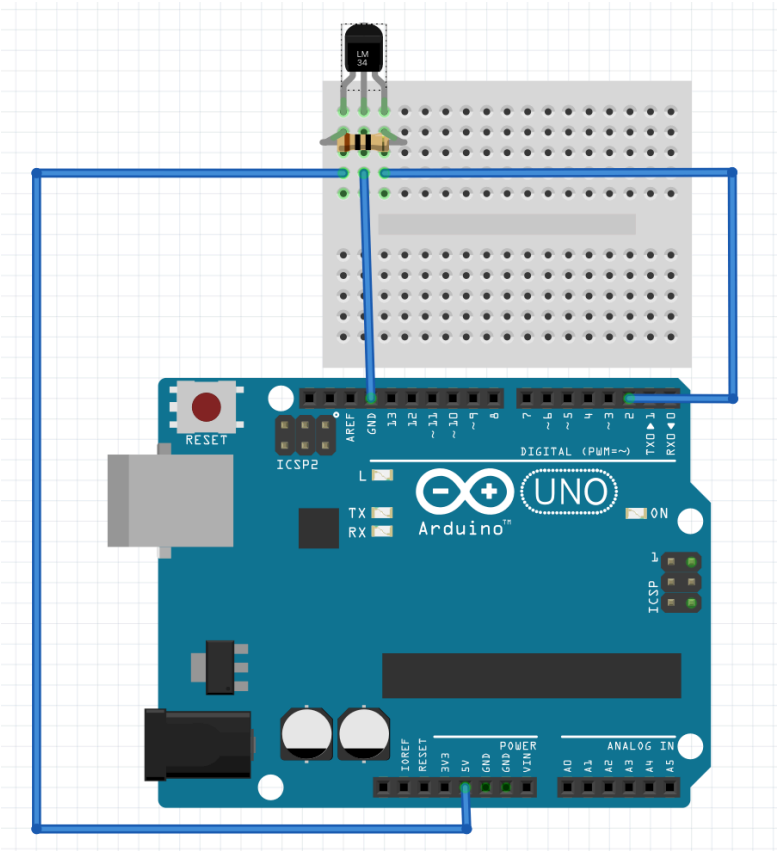
Installation and User Instructions

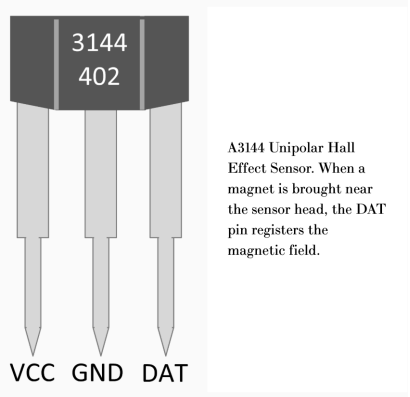
Model Setup

Model name	Model Image	Printed Image	Dimensions (X-Y-Z)	Layer height
generator_support_yoke			127-127-57.2 mm	0.24mm
The generator slides onto the shaft and will need to be secured to the turntable (hot glue would be acceptable).				
magnetic_pole_v1			215.9(8.5")-215.9 (8.5")-63.5 mm (2.5")	0.2mm
The magnetic pole slides into place on the generator.				

Arduino Setup

1. Breadboard diagram





A3144 Unipolar Hall Effect Sensor. When a magnet is brought near the sensor head, the DAT pin registers the magnetic field.

- Connect VCC to 5V
- Connect GND to GND
- Connect DAT to pin 2
- Insert 10k resistor in front of the Hall Effect Sensor VCC and DAT

Note that the Hall Effect Sensor must be quite close to the neodymium magnets.

2. Attach neodymium magnets

- Each prong of the gear should have a neodymium magnet attached (currently set to 8 prongs, if the number of prongs changes, change “magnetsPerRevolution” in code/arduino/detect_magnet/detect_magnet.ino accordingly).
- **NOTE** that the direction of the magnet **MATTERS.** When setting it up, upload code/arduino/magnet_test/magnet_test.ino onto the arduino.

3. Attach arduino and breadboard

- You may either utilize a bluetooth arduino or a corded one (as I did). Ensure that the device you are using to run the application has a serial port (or use a bluetooth arduino).
- You may wish to permanently attach the arduino to the base, in which case hot glue is safe to (carefully) use. Otherwise, you may consider using a rubber band.

- The arduino can be attached anywhere on the base, but for convenience I attached mine near the breadboard on the underside of the cross beam.
- Attach the mini breadboard (mine came with a sticky back) to the underside of the cross beam, carefully lining it up so the **Hall Effect Sensor is as close as possible to the neodymium magnets.**

How to Run The Application

Testing

Test if magnets are detected

- open code/arduino/magnet_test/magnet_test.ino in the Arduino IDE (ver. 2.3.4)
- Upload magnet_test.ino to the arduino
- Hold magnets near the Hall Effect Sensor, “Magnet detected!” will be printed if a magnet is detected and “No magnet detected.” otherwise.

Test if Python application is working

- Download socat (virtual serial port)
  `brew install socat`
- Run socat with
  `socat -d -d pty,raw,echo=0 pty,raw,echo=0`
- Running socat will print two ports. Enter the first port (e.g. ttys003) into code/digital_twin/dt_match.py “SERIAL_PORT”. Enter the second port (e.g. ttys004) into code.digital_twin/simulate_rpm.py “SIMULATED_PORT”.
- In a separate terminal, run simulate_rpm.py
  `python simulate_rpm.py`
- In a 3rd terminal, run dt_match.py
  `python dt_match.py`

dt_match.py will read the randomly generated rpms from the socat serial port and display the model rotating using VTK.

Running the Application

- Change the port as necessary
  - code/digital_twin/dt_match.py “SERIAL_PORT”
  - You can see a list of available serial ports with
    `ls /dev/tty.*`
- Run dt_match.py
  `python dt_match.py`