Research on an absolute optical encoder

Osman Macit Ali Jaffar Murad Khayr Ada Kanlibas Naghim Ibragimov

Abstract

An absolute encoder is a device that converts the angular position of a disk into an electrical signal. This paper assesses the performance of an encoder that was to be deployed in an industrial automation setting, where robotic mistakes have expensive, time-consuming consequences. The encoder in question performed well, accurately recording position and speed. Common sources of error were discussed and mitigated.

Introduction

The need for industrial automation (IA) devices which can adapt in real time is crucial. These offer vast benefits to manufacturers [1]: enhanced efficiency, safety and ability to categorise various products. They can also be used in dynamic environments that are constantly changing. While current IA systems achieve a high degree of efficiency, improvements can be made.

To achieve these, innovations must be made to the devices which measure a system's state. One of these devices is an encoder, which measures positional changes in the shaft of a device. Current use cases include efficient tracking of end effectors which use revolute joints between their links, for calculating speeds of rotating objects like drills, and more.

An absolute, as opposed to an incremental, encoder records the actual position of a shaft at a given time, i.e. relative to its 0 or resting position [2]. The disk of the encoder has markings through which light shines from transmitters on one side to the receivers on the other side of the disk. As the disk is turned by a DC motor, the beam of light is periodically interrupted by the markings. A single interruption triggers an electrical signal to be registered at the Arduino. Multiple interruptions cause multiple signals, which can then be recorded as binary grey code. Each unique binary number represents a distinct position of the encoder relative to its start. Magnetic or optical sensors can be used to transmit the signal but in this paper an optical encoder was investigated: the transmitters shone an LED light through the disk and interruptions (dark zones) were recorded by the sensor.

State-of-the-art trends in encoders include AI integration streamlining manufacturing processes and data analysis. Another interesting innovation is encoders which can harvest and can power themselves with piezoelectric or solar/light-based means. This has positive environmental impacts and eliminates battery reliance, boosting reliability.

Literature Review

A large amount of research has been done on optical encoders due to their importance in academia and manufacturing.

Since the experimental set-up of this paper was quite rudimentary, an important consideration was how vibrations, such as from nearby footsteps, might affect the accuracy of measurements. A paper by Gurauskis et al., 2022 [3], investigated the accuracy of an optical encoder under dynamic conditions. To be precise, an electrodynamic shaker purposefully introduced vibrations into the system, and the effect of this was investigated for different mechanical constructions. It was found that different designers of the encoder bracket behave differently under the same vibration (including having different resonant frequencies) and thus have different levels of accuracy. The set-up in this investigation is unlikely to experience vibrations

which are so extreme. However, measures were taken to limit vibrations from natural sources – see *Errors*.

Concerning future trends in encoders, another paper by Yu et al., 2024 [4], proposed a novel solution which could reduce the effect of eccentricity in encoder accuracy. Eccentricity is when the geometric centre of the encoder differs from the true centre [5] and can be caused by the encoder spinning too fast and vibrating the bracket, sensors and detectors or by external vibrations into the system and more. Once again, vibrations that are strong enough to cause this are beyond the scope of this paper. Still, measures were taken to limit natural vibrations.

Sustainability considerations are also crucial in the design of future encoders. A paper by Richard Anslow in Analog Devices, 2023 [6], explored how variable, as opposed to fixed, speed encoders provide a path to energy savings. It also detailed current and future use cases of encoders, such as in 'machine health monitoring, intelligence, and robust longer life sensing'. Finally, the paper explained 'why complete signal chain designs are fundamental for designing next-generation motor encoders'.

Design, Set up and Methodology

The enncoder disc design that was selected was an absolute optical encoder – as seen by the technical drawing in the Appendix 2 and image of the laser cut object made from black acrylic. The selected approach involved dividing the encoder disc into 8 distinct sectors, each covering 45 degrees of rotation. This seperation have enabled us to do more accurate angular position tracking using three distinct infrared (IR) sensors. Each unique sensor readings are a combination corresponding to a specific sector. Below, Table 1 illustrates the mapping between states (ON/OFF), related binary values, and the angle ranges for each sector. By encoding angular positions in binary format, the design simplifies digital signal processing and arduino coding, allowing both higher accuracy and ease of implementation.

Sector	**LS1**	**LS2**	**LS3**	**Binary**	**Angle**
0	OFF	OFF	OFF	0,0,0	0 to 45
1	OFF	OFF	ON	0,0,1	45 to 90
2	OFF	ON	ON	0,1,1	90 to 135
3	OFF	ON	OFF	0,1,0	135 to 180
4	ON	ON	OFF	1,1,0	180 to 225
5	ON	ON	ON	1,1,1	225 to 270
6	ON	OFF	ON	1,0,1	270 to 315
7	ON	OFF	OFF	1,0,0	315 to 360

Table 1: sensers received, binary and angle corresponding to each sector position of encoder

The electrical connections for the optical encoder system are intended to allow real-time measurement of the disc's angular location using three IR sensors. Each sensor contains an infrared emitter and a corresponding receiver. The emitters are powered by connecting the red wires to the Arduino's 5V output and the black wires to GND, allowing them to continuously produce infrared light. The receivers, which determine whether the rotating disc interrupts the IR beam, include three wires: red (power), black (ground), and white (signal). The receivers' red wires are also connected to 5V, and the black wires to GND, to ensure correct operation. The white signal wires, which output HIGH or LOW depending on whether light is detected, are wired to Arduino digital input pins 2 for LS1, 3 for LS2, and 4 for LS3. These connections enable the Arduino to monitor each sensor's status in real time. To prevent floating values, add 10kΩ pull-down resistors between each signal wire and ground. This

wiring configuration ensures that the Arduino consistently receives clean digital signals from each sensor, which can then be used to calculate the encoder disc's current sector and angular position.

The emitter is an infrared LED that requires a resistor to limit the current and prevent it from burning out. This is why it is connected in series with a resistor between the 5V supply and GND. The resistor ensures that the emitter receives a safe amount of current while staying on continuously to shine infrared light. The receiver, on the other hand, acts as a sensor that only detects light and does not require a resistor in series because it is not driving current.

Code was written to record and measure several information, and this is shown in Appendix 1. The code began with initialising each sensor reading combination to their sector number, matching the order of which set in Table 1. Data begins to be recorded once the sensors record the sector 0 position. The sequence of ‘x’: 7, 6, 5, 4, 3, 2, 1, 0, allows the data (angle and the position sector) to be recorded in the clockwise position, while the sequence of ‘y’: 13, 14, 15, 16, 17, 18, 19, 20, allows the same data to be recorded in the anticlockwise position. The value ‘d’ determines whether the encoder is rotating clockwise or anticlockwise just before it enters sector 0, so that the direction number and speed can be recorded with respect to which direction the encoder is rotating. ‘Rotation’ is initialised to -1, so that once the encoder position reaches sector 0, rotation count can officially start from then, and one rotation is recorded once the encoder position reaches back to sector 0 again. During this cycle, the time is being measured, and once the disc completes one full rotation, the time of one rotation is measured, and then 60000 is divided by this number to calculate the speed in rpm. The reason why this calculation records the rpm is because, for example, if the disc takes 2 seconds to complete one full rotation, in 60 seconds (60/2), 30 revolutions would occur, meaning that the rpm is 30. Since the function millis() measures in milliseconds, 1 second = 1000 ms, so 60/(time in seconds) is equivalent to 60000/(time in milliseconds). The code in Appendix B is added (the previous part of the code was commented out to be replaced) to measure the position of the encoder with respect to the time.

Experiments, Results and Discussion

Our approach to gather data was to assess the accuracy, repeatability and consistency of the encoder. The first experiment consisted of testing the rotational direction and position. By pushing the switch connected to the DC motor to one side to start the motor in a certain direction, the code was able to identify which direction the motor was spinning based on sequence of the sensor combinations received. The position was also successfully recorded, each sector which was in between the sensor emitters and receivers corresponded to the same position displayed in the Arduino IDE’s serial monitor. By varying the speed, this continued to display the correct rotational direction and position, therefore proving there is no misalignment with the IR sensors.

The second experiment consisted of testing 3 different levels on the speed control potentiometer and examined if they produce the same rpm speeds. The easiest approach for this was to set the potentiometer to its lowest, highest and middle possible rotational levels, and 10 speeds were recorded for each. Table 2 (in Appendix 3) shows the results.

The results show the consistency in the speed of the encoder. At higher speeds the encoder rotates at more consistent speeds compared to the lowest rotation speed where there is more of a spread. This might be because of the encoder from time to time getting stuck at low speeds while rotating, leading to the rpm to decrease. This experiment was also verified manually where the number of rotations were counted in a minute, and then compared to the

rpm calculated by the code. Since at high speeds manually counting rotations would be difficult and lead to human errors, this was completed at lower speeds, as shown in Table 3.

Rotational speed (rpm)	**Manual count in one minute**
16	16
40	40
74	74

Table 3: Manual count at rotational speed

These results show the repeatability of the number of rotations of the disc within 10 seconds. This is important because one of the main uses of encoders is for measuring angular displacements, so precision is vital for these situations. There are slight differences for the very high speeds, and this could be because of human error as stopping the rotations at 10 seconds might not always be exact. Figure 1 shows the rotation count vs speed, where the mode number of counts were plotted at each speed.

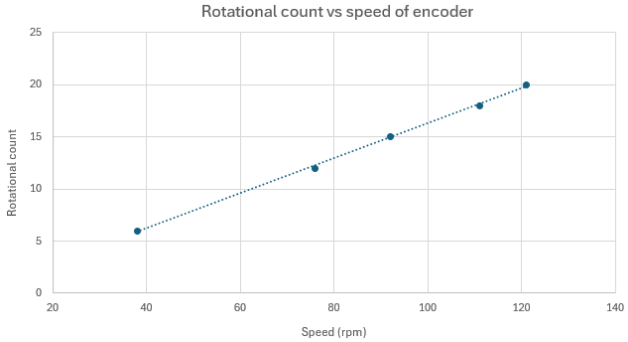


Figure 1: Rotational count vs speed of encoder

The graph shows there is a positive linear relationship between the rotational count and the speed of the encoder. This is expected because as the speed of the DC motor is increased, this would increase the number of full rotations.

The next experiment conducted was similar to the third, where the number of rotations where recorded every 5 seconds from 5 – 25 seconds, for 5 different speeds. Again, the encoder disc was positioned to start at the sector 0 position where the sensors are located, and the speeds were varied at random. Results for this are shown in Table 4 and using these, Figure 2 can be plotted.

	Rotational counts at intervals of 5 seconds				
Speed (rpm)	**5s**	**10s**	**15s**	**50s**	**25s**
16	1	3	4	5	6
51	4	9	13	17	21
80	7	13	20	26	33
107	9	18	27	35	44
118	10	20	30	40	49

Table 4: Rotational counts at intervals of 5 seconds for various speed values

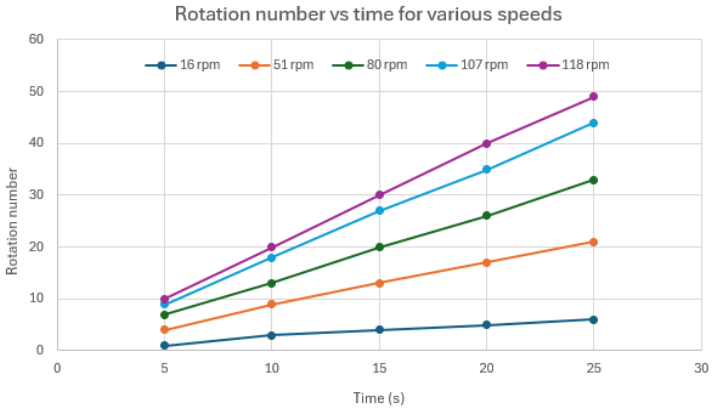


Figure 2: Rotation number vs time for various speeds

It can be determined that there is a positive linear relationship between the rotation number and time, and the gradient increases the higher the speeds, which is expected because more rotations would occur within 5 seconds for higher speeds. This also can conclude there is no (or very negligible) DC motor vibrations which would impact the results as the data gathered is accurate throughout the duration of time. It is not perfectly linear, possibly again because of human error due to the difficulty in measuring at the exact times.

For the fifth experiment, position (angle) vs time was recorded for both clockwise and counterclockwise at different speeds. Angles were recorded for approximately 10 seconds (10000 milliseconds), and the angle speed in deg/s was calculated precisely by dividing 360 by the time it took for one rotation. The time values were also shifted so that they all can start at 0. Figures 3 and 4 show the results.

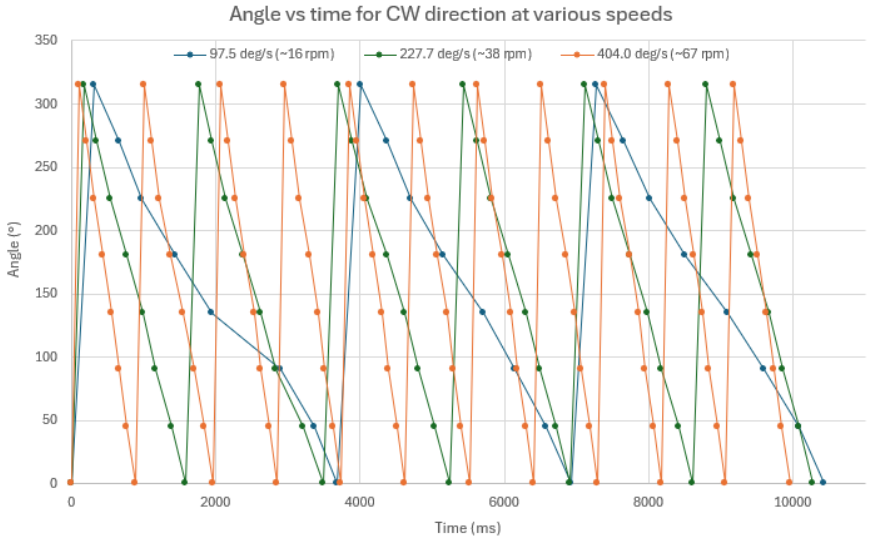


Figure 3: Angle vs time for clockwise direction at various speeds

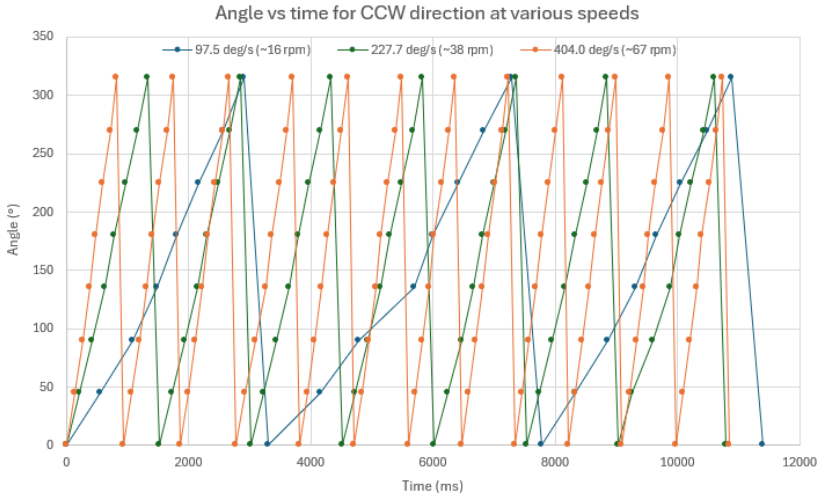


Figure 4: Angle vs time for counterclockwise direction at various speeds

Both the results for CW and CCW have similar shapes but flipped, indicating a consistency within the speed and rotation count. Higher speeds show greater linearity, at the slowest speed it can be seen there is the slight error of the encoder not smoothly rotating. Nevertheless, a high accuracy of the encoder can be concluded from these results since all angles recorded are equally spaced from one another and from each other (in other words equal time values from 45 °, 90° and 135°, and from each 45° recording).

Another type of shaft encoder that could have been used to measure angular displacement and angular velocity is with an incremental encoder. There are two possible configurations for an incremental encoder disk with the capability to sense direction: offset probe configuration and offset track configuration. The quarter pitch offset in the probe location would have to be used to determine direction of rotation by either clock counts to two adjacent rising edges of the two signals or by checking for rising or falling edge of one signal when the other is high. To measure the velocity, either a pulse counting method or a pulse timing method will have to be used.

Errors

The effectiveness of absolute encoders can be limited by a few common sources of error such as ‘environmental factors like dust, dirt, or moisture interfering with the light signals inside the encoder, misalignment of components or poor signal quality can all contribute to errors in the encoder’s output.’ [7]. In an IA context, an important factor is temperature. Too much heat (e.g. from machines that have been working all day) can affect LED performance, cause thermal expansion in components or increase the resistance of circuitry. All of this can lead to incorrect readings or components being unaligned. Also, in systems with lots of gears, backlash can occur. During direction changes, this can lead to motion being lost and not recorded by the encoder. Power supply issues (such as voltage fluctuations) can also affect the encoder output directly (e.g. transmitters don’t work) or indirectly (shaft motor spins irregularly and confuses the encoder computer). The encoder may also change position in small increments due to mechanical vibrations or loose attachment, resulting in misalignment between disc, emitter, and receiver that gives inconsistent results.

In this investigation, the first error occurred immediately after laser printing the encoder disk: the material allowed light to shine through the disk and the receivers could detect no interruptions. This is problematic as it renders the encoder useless at measuring position. This was solved by adding a layer of duct tape to each side of the disk and cutting out the gaps

where the patterns and central attachment point occur [Appendix 2]. Using a thicker piece of material was not an option as the disk would have been too thick to fit in the bracket and would either not spin, spin slower than possible or repeatedly encounter the surrounding components and damage them – or be slowed by them. Using a denser material was not ideal as the disk would be too heavy. It might introduce more vibrations and would have more inertia, causing errors when the disk needed to stop spinning or slow down.

Another source of error was vibrations in the disk and bracket when the disk spun too fast. This is problematic as it could damage components or lead to faulty readings. One way this was mitigated was by ensuring components (transmitters, receivers, etc) were all screwed firmly to the base. Another way was by attaching 4 rubber legs at each corner of the base. This dissipated and made much of the vibrations negligible [Appendix 4].

Another error that occurred was that some of the screws given were too long. When constructing the set-up as instructed, they would scrape against the spinning disk. This is problematic as they would interrupt the spin of the encoder and slow it down. This could potentially confuse the Arduino as it might detect dark spots when expecting light spots () and thus output an incorrect position or angular velocity. This was fixed by using shorter screws and adjusting and repositioning parts (flipping the receiver housing) of the assembly so that the screws and disk wouldn't be in contact [Appendix 5].

Conclusion and Future work

This project observed how well, accurately, and consistently an optical encoder system can measure the position and speed of its disc. Through a set of well-planned tests, it is confirmed that the encoder can accurately measure direction of rotation, speed in RPM, and angular movement. When the results are analysed, a strong relation between the number of encoder rotations and speed, as well as between the amount of angular movement and time were existent. Performance, on the other hand, was definitely more stable at higher speeds rather than lower speeds. This is probably due to some internal mechanical flaws or sensor limits. Nevertheless, the method is proven to be accurate by the fact that same readings can be repeated.

Significant problems could be light leaking through the disc, shaking, and component interference. However, with the right design edits (like blocking light, adding damping supports, and changing the lengths of the screws) we were able to reduce their effect. These new ideas show how important it is for encoders to have both precise mechanics and careful sensor integration.

To further improve the design, more precise time-based metrics and less mechanical error can boost performance. Also by adding more precise sensors, automating data logging in sync, and testing the encoder system in various environmental circumstances (temperature, load, etc.) may also improve reliability. In general, this project shows that the encoder system works well with real-time rotation measurement and has potential for improvement and usage in more complex control systems.

References

[1] Association For Advancing Automation, “Industrial Automation’s Biggest Challenges: Real-Time Adaptation”. 01/09/2025. Accessed 8 Apr 2025.

[2] British Encoder Products Company, “Absolute And Incremental Rotary Encoders: Which Is Best For You?”. 12/09/2024. Accessed 8 Apr 2025.

[3] Donatas Gurauskis et al. “Performance Analysis of an Experimental Linear Encoder’s Reading Head under Different Mounting and Dynamic Conditions”. Published 22/08/2022

[4] Jingyi Yu et al. “Precision encoder grating mounting: a near-sensor computing approach”. Published 27/09/2024

[5] Collins Dictionary. “eccentricity in mechanical engineering”. Accessed 8 Apr 2025.

[6] Richard Anslow. “How to Design Motor Encoders for Next-Generation Sustainable Applications ”. June 2023.

[7] IndMall. “What Is The Optical Encoder Error?”. Accessed 8 Apr 2025.

Appendix 1

A) Code for experiments 1 – 4

```
const int LS1 = 2;
const int LS2 = 3;
const int LS3 = 4;
int x = 0;
int y = 20;
int d;
int rotation = -1;
int rotation2 = -1;
unsigned long time = 0;
unsigned long time2 = 0;
unsigned long I_time;
unsigned long N_time;
unsigned long D_time;
unsigned long I_time2;
unsigned long N_time2;
unsigned long D_time2;

void setup() {
  pinMode(LS1, INPUT);
  pinMode(LS2, INPUT);
  pinMode(LS3, INPUT);
  Serial.begin(9600);
}

void loop() {

  int zero = digitalRead(LS1) == 0 && digitalRead(LS2) == 0 && digitalRead(LS3) == 0;
  int one = digitalRead(LS1) == 0 && digitalRead(LS2) == 0 && digitalRead(LS3) == 1;
  int two = digitalRead(LS1) == 0 && digitalRead(LS2) == 1 && digitalRead(LS3) == 1;
```

```
int three = digitalRead(LS1) == 0 && digitalRead(LS2) == 1 && digitalRead(LS3) == 0;
int four = digitalRead(LS1) == 1 && digitalRead(LS2) == 1 && digitalRead(LS3) == 0;
int five = digitalRead(LS1) == 1 && digitalRead(LS2) == 1 && digitalRead(LS3) == 1;
int six = digitalRead(LS1) == 1 && digitalRead(LS2) == 0 && digitalRead(LS3) == 1;
int seven = digitalRead(LS1) == 1 && digitalRead(LS2) == 0 && digitalRead(LS3) == 0;

if ((x == 0 || y == 20) && zero) {

  if (x == 0 && d == 0){
    Serial.print("Direction: CW\n");
    rotation = rotation + 1;
    Serial.print("CW rotation # = ");
    Serial.println(rotation);
    if (rotation == 0){
      I_time = millis();
    }
  else{
    N_time = millis();
    D_time = N_time - I_time;
    Serial.print("speed (rpm) =");
    Serial.println(60000/(D_time));
    I_time = N_time;
  }
}

if (y == 20 && d == 1){
  Serial.print("Direction: CCW\n");
  rotation2 = rotation2 + 1;
  Serial.print("CCW rotation # = ");
  Serial.println(rotation2);
  if (rotation2 == 0){
    I_time2 = millis();
```

```
  }
  else{
    N_time2 = millis();
    D_time2 = N_time2 - I_time2;
    Serial.print("speed (rpm) =");
    Serial.println(60000/(D_time2));
    I_time2 = N_time2;
 } }

 Serial.print("Sector = 0\n");
 Serial.print("Angle = 0 to 45\n");

 x = 7;
 y = 13;

}

 if ((x == 1 || y == 13) && one) {
  Serial.print("Sector = 1\n");
  Serial.print("Angle = 45 to 90\n");
  x = 0;
  y = 14;
  d = 0;
 }

 if ((x == 2 || y == 14) && two) {
  Serial.print("Sector = 2\n");
  Serial.print("Angle = 90 to 135\n");
  x = 1;
  y = 15;
 }
```

```
if ((x == 3 || y == 15) && three) {
  Serial.print("Sector = 3\n");
  Serial.print("Angle = 135 to 180\n");
  x  = 2;
  y = 16;
}

if ((x == 4 || y == 16) && four) {
  Serial.print("Sector = 4\n");
  Serial.print("Angle = 180 to 225\n");
  x = 3;
  y = 17;
}

if ((x == 5 || y == 17) && five) {
  Serial.print("Sector = 5\n");
  Serial.print("Angle = 225 to 270\n");
  x = 4;
  y = 18;
}

if ((x == 6 || y == 18) && six) {
  Serial.print("Sector = 6\n");
  Serial.print("Angle = 270 to 315\n");
  x = 5;
  y = 19;
}

if ((x == 7 || y == 19) && seven) {
  Serial.print("Sector = 7\n");
  Serial.print("Angle = 315 to 360\n");
  x = 6;
```

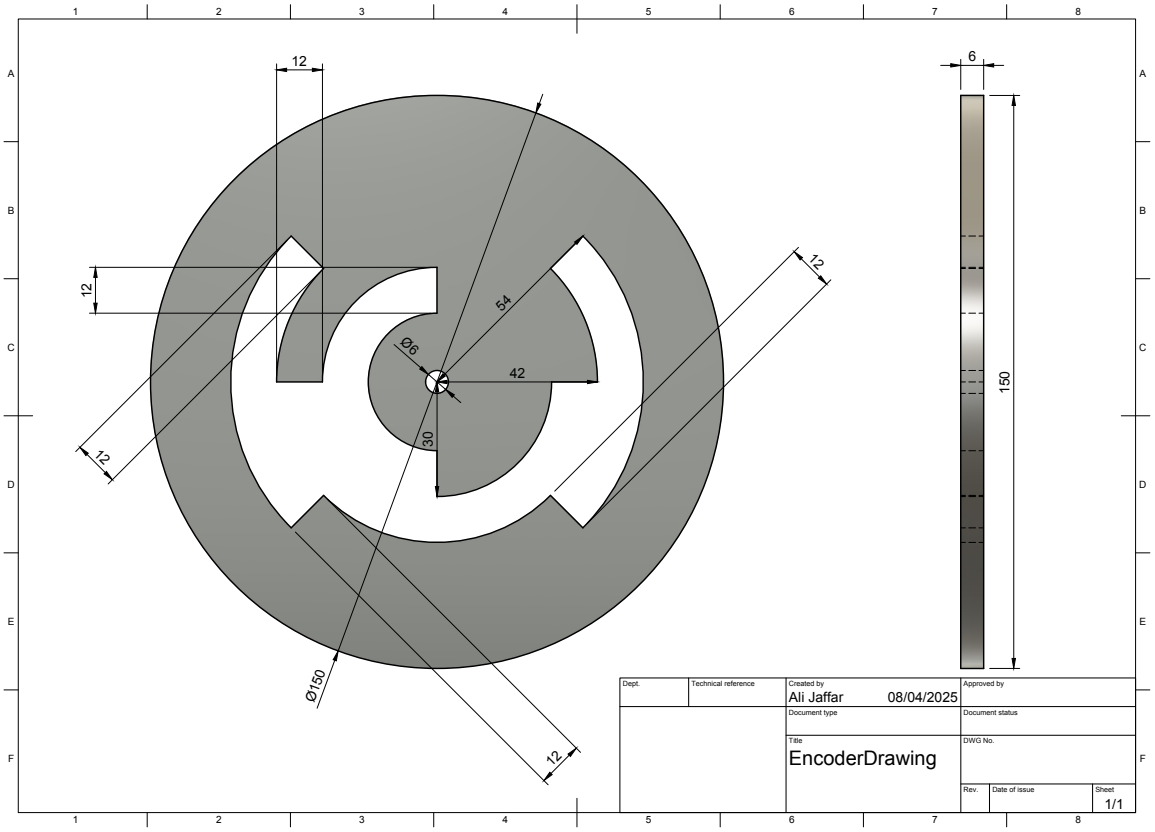
```
    y = 20;
    d = 1;
  }
}
```

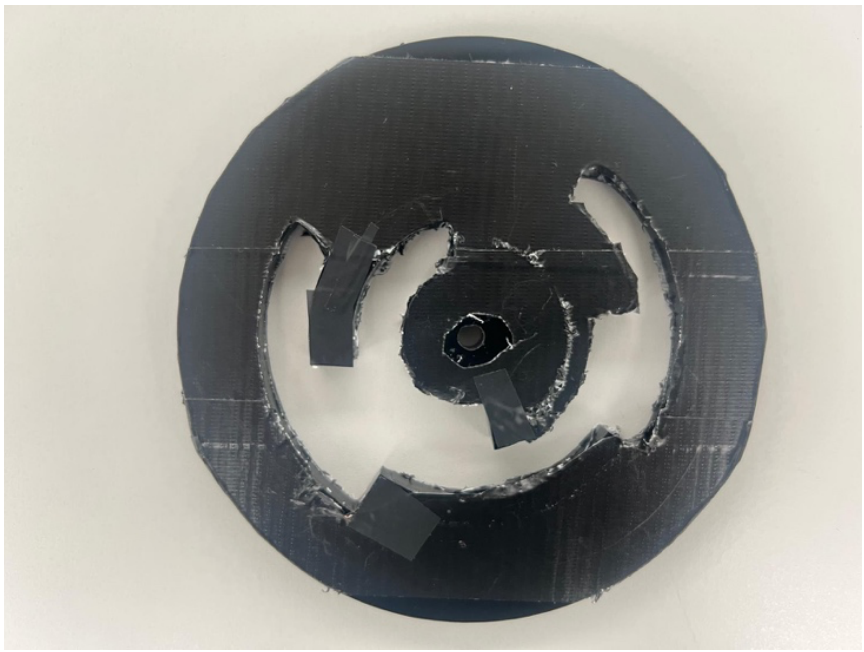
B) Code implemented for experiment 5

```
if ((x == 0 || y == 20) && zero) {
    Serial.print(millis());
    Serial.println(",0");
    x = 7;
    y = 13;
  }

  if ((x == 1 || y == 13) && one) {
  Serial.print(millis());
  Serial.println(",45");
  x = 0;
  y = 14;
  }

  if ((x == 2 || y == 14) && two) {
    Serial.print(millis());
    Serial.println(",90");
    x = 1;
    y = 15;
  }

  if ((x == 3 || y == 15) && three) {
    Serial.print(millis());
    Serial.println(",135");
    x = 2;
    y = 16;
```

```
}

if ((x == 4 || y == 16) && four) {
  Serial.print(millis());
  Serial.println(",180");
  x = 3;
  y = 17;
}

if ((x == 5 || y == 17) && five) {
  Serial.print(millis());
  Serial.println(",225");
  x = 4;
  y = 18;
}

if ((x == 6 || y == 18) && six) {
  Serial.print(millis());
  Serial.println(",270");
  x = 5;
  y = 19;
}

if ((x == 7 || y == 19) && seven) {
  Serial.print(millis());
  Serial.println(",315");
  x = 6;
  y = 20;
}
```

Appendix 2 – The encoder technical drawing and physical disk, covered in duct tape to make it opaque. All units in mm.



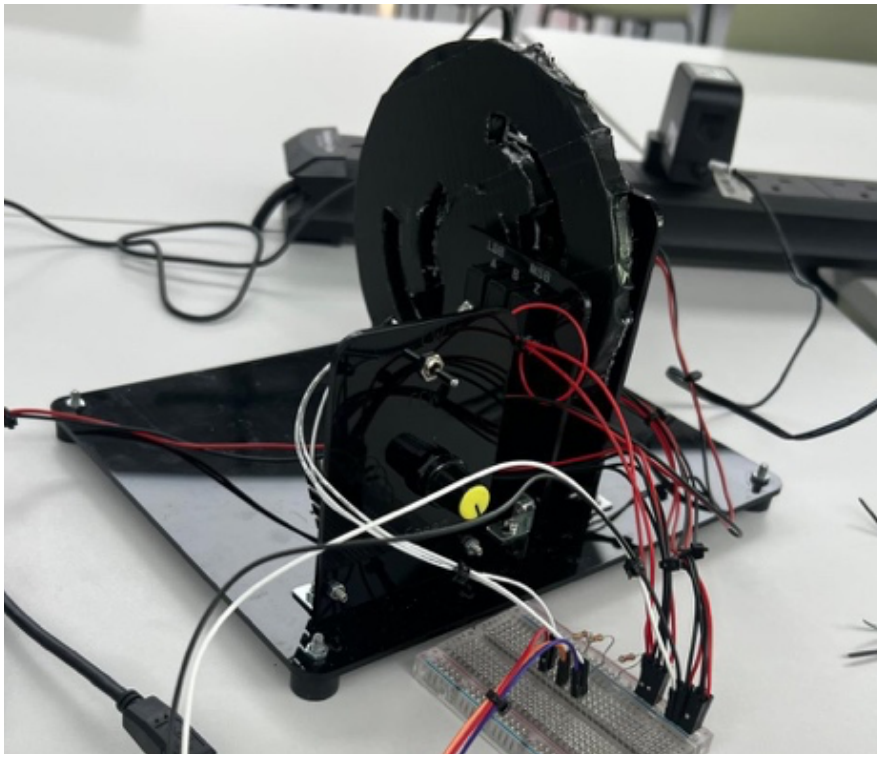


Appendix 3

Test	Lowest rotation speed (rpm)	Middle rotation speed (rpm)	Highest rotation speed (rpm)
1	16	102	131
2	17	101	131
3	17	101	132
4	16	101	132
5	17	101	131
6	15	101	132
7	16	101	132
8	16	101	132
9	16	101	132
10	15	101	132
Mean	16.1	101.1	131.7

Table 2: Speeds of different potentiometer levels

Appendix 4



Appendix 5