Department of Mechanical & Mechatronics Engineering
MTE 201 – Experimental Measurement and Statistical Analysis

Project – Measurement System Project

Professor Peter Teerstra
March 25th, 2019

ZiXuan Gao, z72gao@edu.uwaterloo.ca
Vu Julian Hoa-Chau Huynh, vjhhuynh@edu.uwaterloo.ca
Heechul Jung, hcjung@edu.uwaterloo.ca
Seonghan Cho, sh8cho@edu.uwaterloo.ca

Design

For this lab, we decided to measure thickness of a coin using a rotary encoder. The rotary encoder has 2 sensors that output pulses as the shaft rotates. This means we have 4 states (00, 01, 10, 11), with which we can determine the direction of rotation and its number of transitions. Arduino is used as the user interface, and an LCD display is used to output converted displacement of the clamp.

As the user spins the control wheel, the rotational motion of the control shaft was delivered to the encoder shaft through regular gears. The encoder shaft comprises a regular gear as well as a worm gear. The worm gear drives an intermediate gear that meshes with the final clamp rack. When the encoder shaft rotates, the rotational displacements of the worm gear drives the intermediate gear which is eventually transformed into rectilinear motion (vertical) of the clamp rack. This rectilinear motion is manifested through the raising and lowering of the clamp head. The whole mechanism is shown in the figures below.

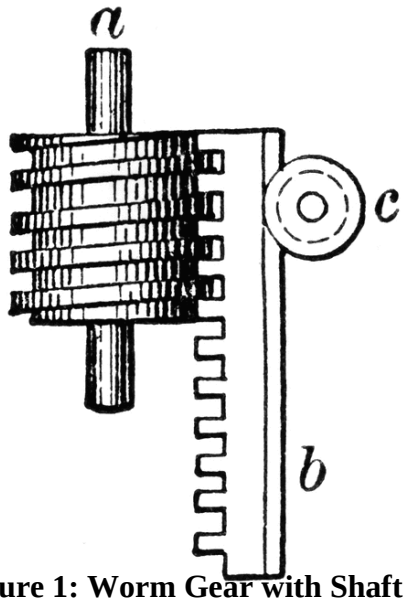

Figure 1: Worm Gear with Shaft

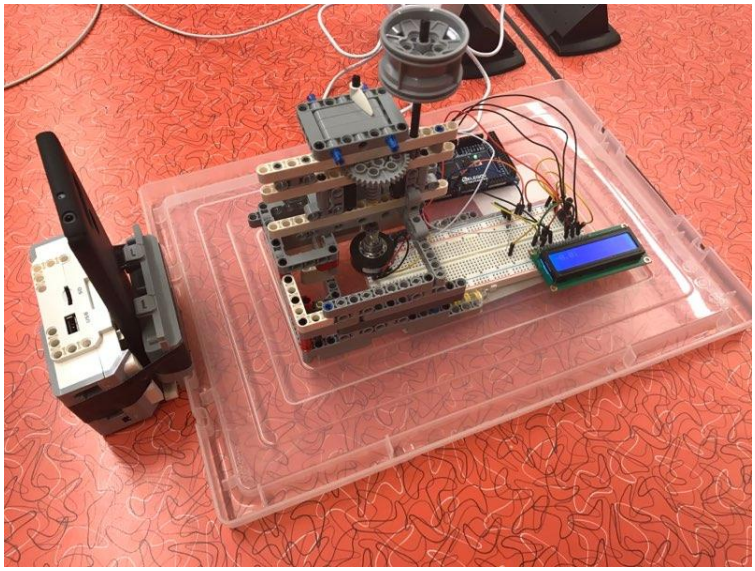

Figure 2: Photo of Final Design

When building and testing the final design, several problems occurred. Firstly, the surface that the coin rests on was not stable, and we had to devise mechanisms to stabilize it. Secondly, we had difficulties interfacing the encoder shaft with the LEGO shaft (linked to the rack). We ended up using tape. Thirdly, the Lego pieces did not fit the Arduino breadboard width well, and we had to use Lego rods as spacers to stabilize it. Lastly during testing, each state transition count occurred at a very small rotational angle. It was very easy to overshoot the measurement or overtighten the clamp head.

Calibration

The calibration begins by raising the clamp to an arbitrary height, which is achieved by spinning the control wheel clockwise. We then lower the clamp head until it contacts the grey table beneath it. Live camera feed was utilized to monitor the exact moment of contact between the clamp and the lower surface. We then press the red “reset” button on the Arduino board to reset the output to 0. To measure the coin’s thickness, we raise the clamp head again, providing enough space for the coin to fit in the gap between the clamp and the lower surface. We then lower the clamp until the exact moment that the clamp head touches the coin. Finally, the number of state transitions shown on the LCD display is recorded. We repeated this procedure thrice for each coin type. To measure the coin’s true thickness, we used Mitutoyo Vernier micrometer. The measurement was also repeated thrice, and a mean true thickness was derived.

The results recorded were plotted against the true value, and a line of best fit was computed.

	1	2	3	Mean True Thickness [mm]	1	2	3	Mean Count
5 Cents	1.77	1.77	1.78	1.77	293	294	290	292.33
10 Cents	1.19	1.18	1.20	1.19	190	190	198	192.67
25 Cents	1.58	1.58	1.58	1.58	260	258	260	259.33
1 Dollar	1.95	1.93	1.91	1.93	311	339	332	327.33
2 Dollars	1.74	1.74	1.74	1.74	282	278	272	277.33

Table 1: Recorded True Thickness Values and State Transition Counts

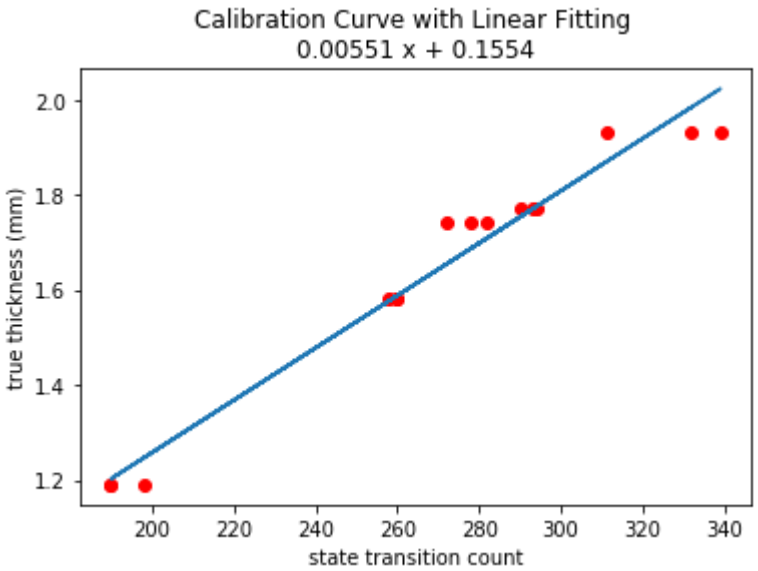

Figure 3: Calibration Curve with State Transition Counts vs True Thickness [mm]

Uncertainty Analysis

The linear function obtained from the calibration curve in **Figure 3** was used to compute the measured thickness values. We use “Y” to denote the measured thickness in mm, and “X” to denote the state transition counts, and get

$$y = 0.00551x + 0.1554$$

The results were plotted against their associated true thickness values, from which the maximum positive and negative uncertainties were found.

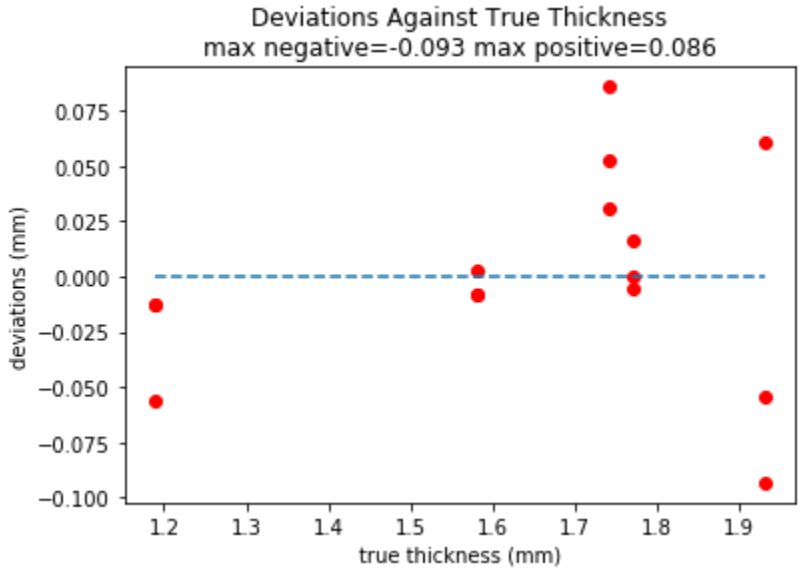

Figure 4: Deviations vs True Thickness [mm]

Based on the found values and the plot in **Figure 4**, the maximum negative uncertainty is -0.093mm and the maximum positive uncertainty is 0.086mm. When looking at the deviations for each measurement, it can be seen that the uncertainty is random, as there is no visible pattern occurring with the deviation values. The random error is expected, because it was difficult to monitor the exact moment of contact between the upper clamp and the rim surface of the coin. The uncertainty of the measurement is highly dependent on the consistency that we determine the exact moment of contact.

Conclusion and Recommendations

Looking at the calibration curve, the linear line of the best fit mostly matches with the measured data points. The magnitude of the uncertainties was also very small compared to their associated true values. We could therefore conclude that our measurement system is accurate since the small uncertainties indicate a very close match to the true value obtained from a micrometer. However, our measurement system is much bulkier compared to the commercial digital micrometer that is widely available.

Using the conclusion and the problems encountered during the construction and testing of the system, there are a few recommendations that the team could think of:

1. We should have devised a better interface between the encoder shaft and the LEGO shaft. Instead of using the tape, we could have 3D printed a custom interface that would properly grips the notch of the encoder shaft.
2. We should have machined a flat metal surface to act as the table that measured item rests on.
3. We should have ensured that the clamp head was mounted such that the clamp head's lower surface was as horizontal as possible. Or alternatively, the 'clamp head' could be replaced with a small pointer tip.
4. Another improvement would be to have a better organization of wiring, for aesthetic and less potential chance for wire-detaching accidents.

Appendix

Code for Calibration Graph.

```
import numpy as np
import matplotlib.pyplot as plt

x = [
   293, 190, 260, 311, 282,
   294, 190, 258, 339, 278,
   290, 198, 260, 332, 272
] # digital encoder count
y = [1.77, 1.19, 1.58, 1.93, 1.74] * 3 # true length (mm)

fitln = np.poly1d(np.polyfit(x, y, 1))

plt.plot(x, y, 'ro', x, fitln(x))
plt.ylabel('true length (mm)')
plt.xlabel('digital encoder count')
plt.title("Calibration Curve with Linear Fitting" +str(fitln))

plt.show()
```

Uncertainty Graph:

```
uncertainties = [ y[index]-fitln(x[index]) for index in range(len(x)) ]

plt.plot(y, uncertainties, 'ro', [min(y), max(y)], [0, 0], '--')
plt.xlabel('true length (mm)')
plt.ylabel('uncertainties (mm)')
plt.title("Uncertainties Against True Length\nmin=%.3f max=%.3f"
          %(min(uncertainties), max(uncertainties)))

plt.show()
```

Code for Arduino

```
#include <LiquidCrystal.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
const int A = 9, B = 10;
int counter = 0;
bool prevA, prevB, curA, curB;


LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {

       pinMode(A, INPUT);
       pinMode(B, INPUT);
```

```
        prevA = digitalRead(A);
        prevB = digitalRead(B);
        lcd.begin(16, 2);

}

void loop() {
        curA = digitalRead(A);
        curB = digitalRead(B);

        if (curA == prevA && curB == prevB) return;

        if (curA == !prevB && curB == prevA) ++counter;
        else if (curA == prevB && curB == !prevA) --counter;


        prevA = curA;
        prevB = curB;
        double dist = 0.005608*counter*-1 + 0.129;

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print(dist);



}
```