ECE-GY 6483: Real Time Embedded Systems

Final Project Report

Yujie Zhu Siddharth Singh Siyuan Zhu
yz9461 ss16915 sz4277

Introduction:

In this project, we aimed to design a distance recording program, with board STM32F429I with Gyroscope L3GD20. Our report will include the methodology of error measurement, data preprocessing and distance calculation.

Methodology abstract:

1. **Why choose leg instead of arm?**

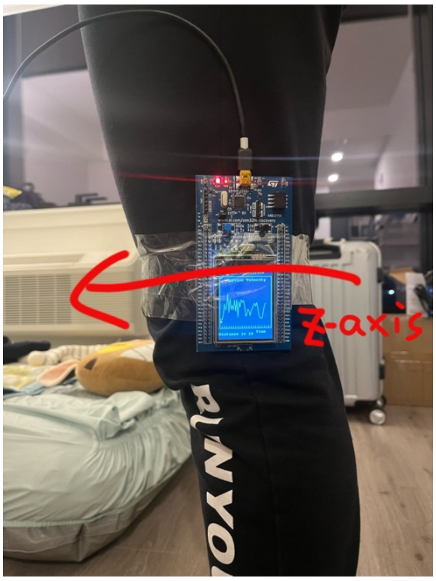
   The arms will swing back and forth when walking, which is not a good choice because the amplitude as well as the directional instability can cause large errors. Ankles are not a good choice either. When walking fast, the different impacts of each landing result in different angular velocities, thus increasing the measurement error. Therefore, the knee is the best choice! There is neither unnecessary wobbling nor much vibration when walking, which has very little effect on the measurement.

2. **How to tie the board:**

   The direction of measurement is very important.

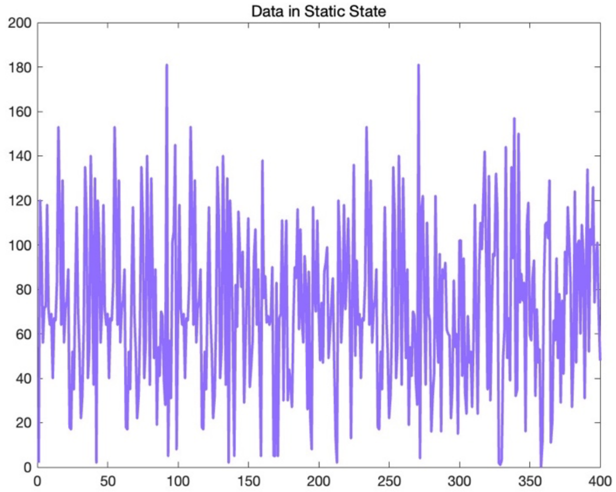
   When tethering directly in front or directly to the side, we only must consider the change in angular velocity in one direction, the other two directions are independent of the forward velocity and do not give rise to more than a possibility of error.

   As shown in the figure, the forward speed of the gyroscope is Z-axis.



3. **Data preprocessing:**

We recorded 400 raw data in the static state (no movement), as shown in the figure.



We noticed that the mean of this set of data is 74.5, and the standard deviation is 37.46.

Therefore, we assume there is a small error ε that ε~N(74.5, 37.46^2), where N is Normal Distribution, and we always let raw data subtract a random error ε that obeys this distribution.

Since C can only generate random numbers that obey a uniform distribution, we use the Box–Muller transform to get random Normal Distribution, that is

Suppose we have two random value u1 and u2. u1~U[0,1] and u2~U[0,1], where u is uniform distribution;

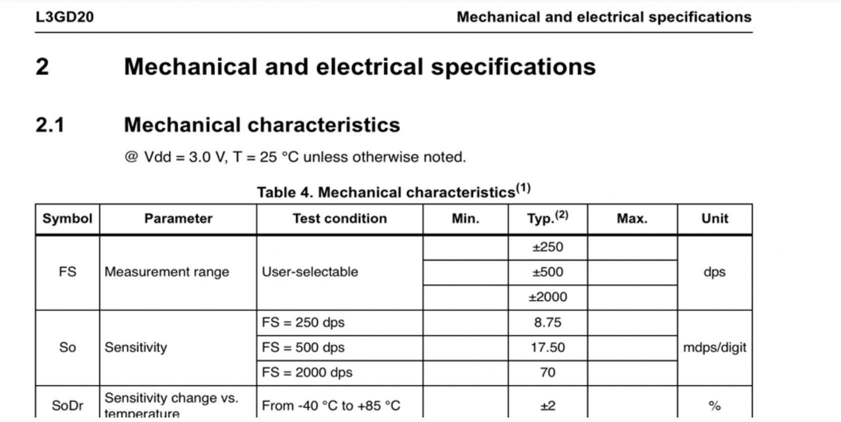
$$X = cos(2\pi u_1)\sqrt{-2lnu_2}$$

$$Y = sinn(2\pi u_1)\sqrt{-2lnu_2}$$

Then X ~N(0,1) and Y~N(0,1)

To get Z~N(μ, σ^2), we just need z = x*σ + μ.

Also, from manual, we need to let data multiply 0.00875 for sensitivity as figure below.



L3GD20 Mechanical and electrical specifications

2 Mechanical and electrical specifications

2.1 Mechanical characteristics

@ Vdd = 3.0 V, T = 25 °C unless otherwise noted.

Table 4. Mechanical characteristics(1)

Symbol	Parameter	Test condition	Min.	Typ.(2)	Max.	Unit
FS	Measurement range	User-selectable		±250		dps
				±500		
				±2000		
So	Sensitivity	FS = 250 dps		8.75		mdps/digit
		FS = 500 dps		17.50		
		FS = 2000 dps		70		
SoDr	Sensitivity change vs. temperature	From -40 °C to +85 °C		±2		%

This is function *double ConvertRealAngularVelocity(int16_t x);* in our code.

4. Convert Angular Velocity to Linear Forward Velocity.

The core formula is **v = ωr**; where v is linear velocity; ω is angular velocity and r is radius, here it is leg length.

Notice, the unit of data we capture is degrees per second, and here ω is radians, so we need to use data/360*2*π

And this is function *double ConvertForwardVelocity(double x);* in our code.

5. Distance Calculation:

Here, we use three different methods to calculate the distances.

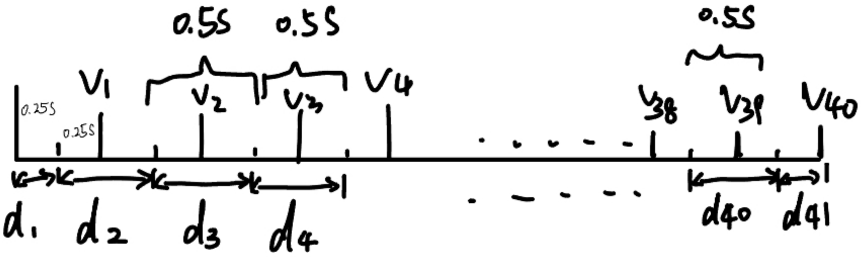
Method 1

We add all forward velocity together, and the data is stored in *SumVelocity*. *numberVelocity* stores the amount of data collected.

Therefore, Average Speed = SumVelocity/ numberVelocity;

Distance = Average Speed * timeInterval;

Method 2



$$d_1 = 0.25 \times V_1$$

$$d_2 = \frac{V_1+V_2}{2} \times 0.5$$

$$d_3 = \frac{V_2+V_3}{2} \times 0.5$$

$$\dots$$

$$d_{40} = \frac{V_{39}+V_{40}}{2} \times 0.5$$

$$d_{41} = V_{40} \times 0.25$$

$$\text{distance} = d_1 + d_2 + d_3 + \dots + d_{39} + d_{40}$$

As shown in the picture, we divided the total distance into 41 pieces based on time. For d_1 and d_{40}, they are for 0.25s each, so $d_1 = 0.25*v_1$, $d_{41} = 0.25*v_{40}$. For d_i, i = 2, 3 ….40, $d_i = (v_{i-1}+v_i)/2*0.5$, and the total distance is $\sum_{i=1}^{41}(d_i)$

Method 3

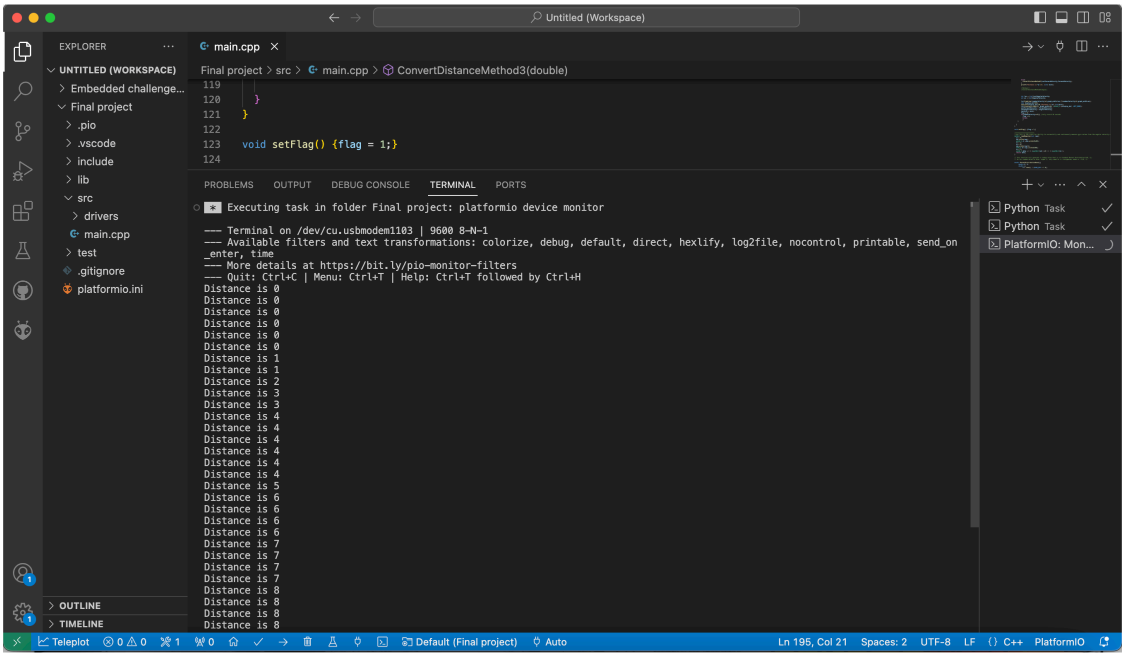
We try to use two formulas:

θ = ω * t, where θ is angle, ω is angular velocity and t is time interval.

L=2*r*sin(θ/2), where L is chord, r is radius and θ is angle.

The sum of the chord is the distance.

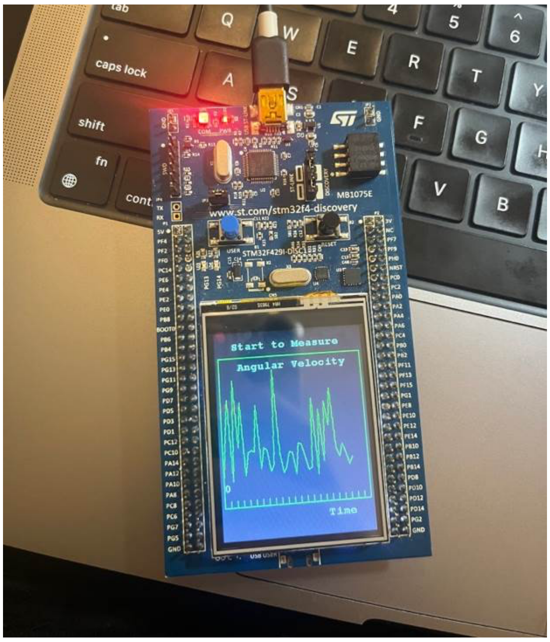
Print out of the results:



```cpp

    }
}

void setFlag() {flag = 1;}

```

```
 *  Executing task in folder Final project: platformio device monitor

--- Terminal on /dev/cu.usbmodem1103 | 9600 8-N-1
--- Available filters and text transformations: colorize, debug, default, direct, hexlify, log2file, nocontrol, printable, send_on
_enter, time
--- More details at https://bit.ly/pio-monitor-filters
--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H
Distance is 0
Distance is 0
Distance is 0
Distance is 0
Distance is 0
Distance is 0
Distance is 1
Distance is 1
Distance is 2
Distance is 3
Distance is 3
Distance is 4
Distance is 4
Distance is 4
Distance is 4
Distance is 4
Distance is 4
Distance is 5
Distance is 6
Distance is 6
Distance is 6
Distance is 6
Distance is 7
Distance is 7
Distance is 7
Distance is 7
Distance is 8
Distance is 8
Distance is 8
Distance is 8
```

Display the angular velocity on the board:



Link to Youtube - link