Software Overview

Year: 2022 Semester: Spring Team: 08 Project: Gimbal Vehicle

Creation Date: Jan 27, 2022 Last Modified: Jan 28, 2022

Author: Zeren Li Email: li3103@purdue.edu

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Software Overview** |  | x2 |  |  |
| **Description of Algorithms** |  | x2 |  |  |
| **Description of Data Structures** |  | x2 |  |  |
| **Program Flowcharts** |  | x3 |  |  |
| **State Machine Diagrams** |  | x3 |  |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** |  | x2 |  |  |
| **Formatting and Citations** |  | x1 |  |  |
| **Figures and Graphs** |  | x2 |  |  |
| **Technical Writing Style** |  | x3 |  |  |
| **Total Score** |  | | |  |

*Table 1.*

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

*Relevant overall comments about the paper will be included here*

1.0 Software Overview

The gimbal vehicle consists of three major parts: The wireless controller that drives the whole vehicle, four motors that drive Mecanum wheels attached to the chassis, and two servos that maintain a desired direction of the gimbal.

When the vehicle is powered up and the switch is set to pure driving mode, the microcontroller on the wireless controller is ready to receive data from the left joystick. The left joystick will control the x and y direction of the vehicle. The left joystick will control the rotation of the vehicle. The data transportation is achieved by analog to digital (ADC). Two joysticks send analog data to the MCU. Then, the MCU will convert the analog data into digital data, which will be stored in a register in the MCU by Direct Memory Access (DMA). The wireless transmitter module will read the digital data from the memory and send them to the receiver end on the vehicle using Serial Peripheral Interface (SPI). The microcontroller on the vehicle is responsible for converting the digital data into a relative velocity and a rotational angle, which will be sent to the corresponding motor drivers by Pulse Width Modulation (PWM). Gimbal is not activated in this mode.

When the switch is set to Gimbal Control mode, instead of controlling the rotation of the vehicle, the right joystick will be responsible for controlling the direction of the Gimbal. Just like how we drive motors in the last paragraph, as we keep holding the right joystick in the desired direction, the servos will keep receiving data from the right joystick about the direction and the angle that we want the Gimbal to face.

When the switch is set to Default Gimbal mode, the Gimbal will be facing directly in front of the vehicle. No matter how the vehicle is shaking or rotating, the Gimbal is stabilized by the MCU with pre-defined parameters. Meanwhile, the right joystick will go back to control the rotation of the vehicle.

2.0 Description of Algorithms

We will implement PID control in our system, for both motors and servos. The existence of the PID control system mainly serves as one reason, to make the system adapt to unexpected exterior change quickly and accurately. Also, PID is helpful for the motor or servo to reach and stay at the desired velocity or angle. PID control contains mainly three parts, P for proportional, I for integral, D for derivative. Each of them serves different usage to accomplish the ultimate goal. Let's say that our servo wants to turn from 0 degrees to 100 degrees. This is a bit exaggerated, but we will say that our system has an initial loss of 40% which is a 40-degree digression in this case.

Firstly, The Proportion (P) is only able to tune the servo to around 90 degrees. Because the closer the servo is approaching 100 degrees (goal), the smaller the digression will be. As a result, the digression is approaching negative infinity, Proportion (P) would have very little effect on the error correction due to the steady-state error. Here, the integral (I) comes in handy because it is able to keep track of errors accumulated from the past. when the servo reaches the steady state, the accumulated errors will make the servo continue rotating. Both P and I will work with each other to drive the angle digression down to zero. When the servo reaches very close to 100 degrees, the P will no longer ask the servo to rotate, but the Integral might have summed too much error, which leads to the case when the servo keeps spinning and generates an angle larger than 100 degrees so that the excessive errors would be canceled out. To avoid overshooting, we would add the Derivative (D) that measures the rate of change of the digression to the system. For example, if the servo is approaching 100 degrees quickly, meaning that the error is quickly shrinking. The negative rate of change of the error will ask the servo to produce a negative offset to bring the overshooting angle back to 100 degrees.

Besides PID control, we also employed Kalman Filter for the acquisition of Roll and Pitch angles of the vehicle. The angle data is later used for controlling the servo motors which keep the balance of the gimbal. First, we calculate the raw roll angle and a pitch angle of the IMU using its acceleration in X, Y, Z directions. Then we pass the angle of acceleration in each direction to a function called Kalman\_getAngle() along with the time interval between this measurement of angle and last measurement and the angular velocity in the corresponding direction measured by a gyroscope. The function will first project the angle using the time interval, the angle from the last measurement, and angular velocity. Next, it will project the error covariance using the error covariance got from the last calculation and process noise of the accelerometer. Then it will calculate the Kalman gain as error covariance divided by error. Finally, the angle will be calculated using the angle difference and Kalman gain.

Another important algorithm is about decomposing the digital data of the joysticks. The MCU on the vehicle receives digital data x1,y1,x2,y2 (x and y correspond to the horizontal and vertical location of a joystick. 1 is the right joystick and 2 is the left joystick) when we push the right and left joysticks. We design an algorithm to decompose the digital data into four different integers. Each integer will be sent to a motor by PWM to control the speed of its designated Mecanum wheel.

left\_front\_wheel = x1 + x2 + y1 + y2

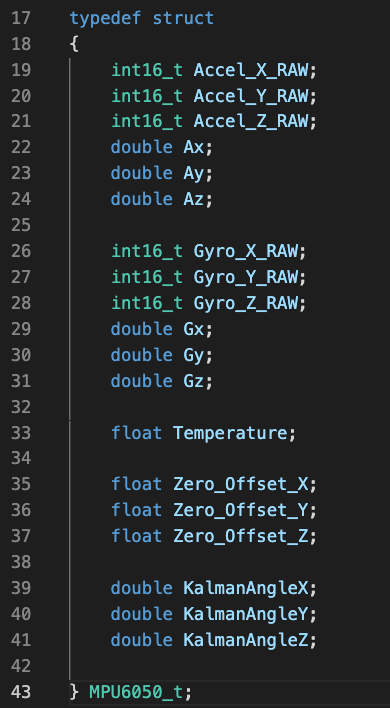
right\_front\_wheel = x1 - x2 - y1 + y2

left\_back\_wheel = x1 + x2 - y1 - y2

right\_back\_wheel = x1 - x2 + y1 - y2

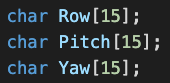
**3.0 Description of Data Structures**

One of our PSSCs describes an ability to obtain data from IMU using I2C. The data structure we used to pass through the I2C is defined as MPU6050\_t (Figure 1). Valid data required for each I2C transaction would be 100 bytes. The information that the struct contains can be described as 5 features, acceleration for XYZ axes, gyroscope position for XYZ axes, temperature, zero offsets for XYZ axes, Kalman angle for XYZ direction. We will not use data from temperature since we don’t have a temperature-related feature. Other than that, all of the required data to control the gimbal servos have been contained in the struct.



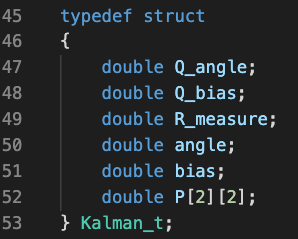
Figure

One of the many features of the Gimbal vehicle is displaying the IMU information on an LCD screen through the I2C protocol. The data structure we used in order to accomplish this feature is the original message strings (Figure 2) that need to be passed.



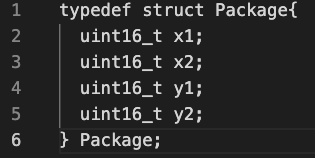
Figure

The Kalman filter algorithm employs a struct called Kalman\_t (Figure 3). It contains six variables of the type double. Namely, they are process noise of accelerometer, gyro, measurement noise variance, gyro bias, 2\*2 error covariance matrix, and angle calculated. The algorithm contains 27 lines of code with no dynamic memory allocation.



Figure

The Joysticks’ data will be stored in a struct called “Package”. It consists of 4 uint16\_t data. x1 and y1 from the right joystick measure how much the joystick shifts horizontally and vertically. x2 and y2 from the right joystick measure how much the joystick shifts horizontally and vertically

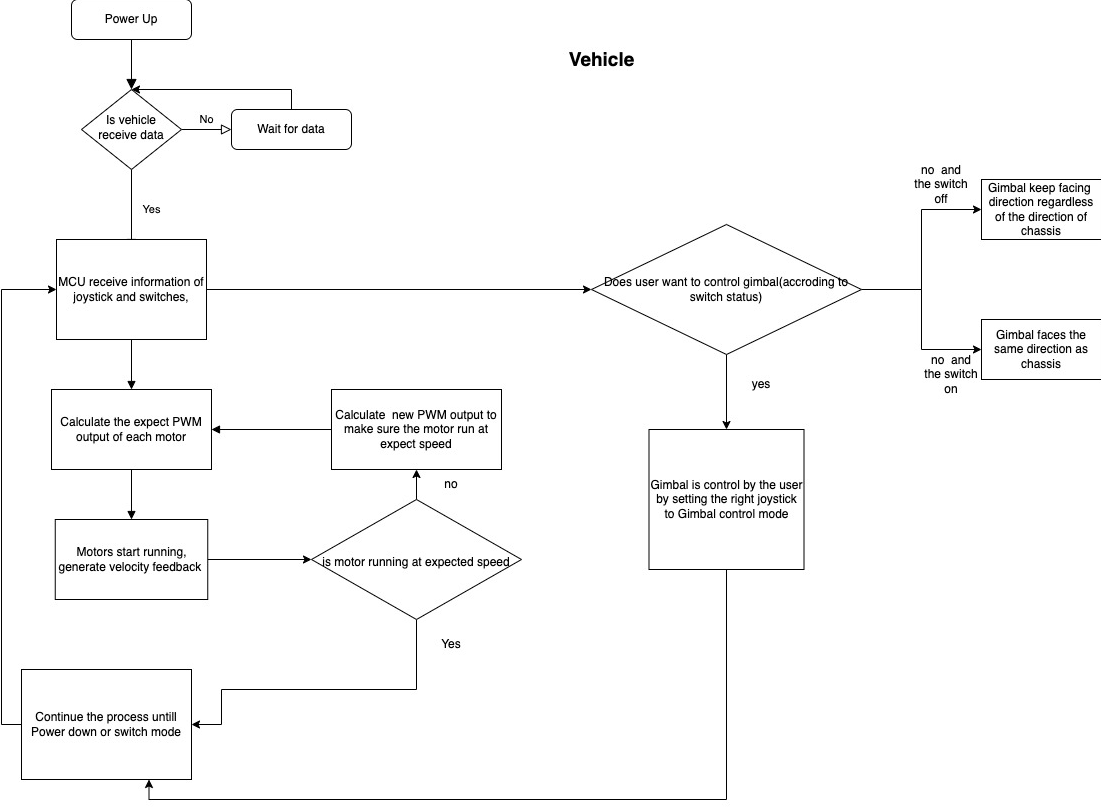


Figure

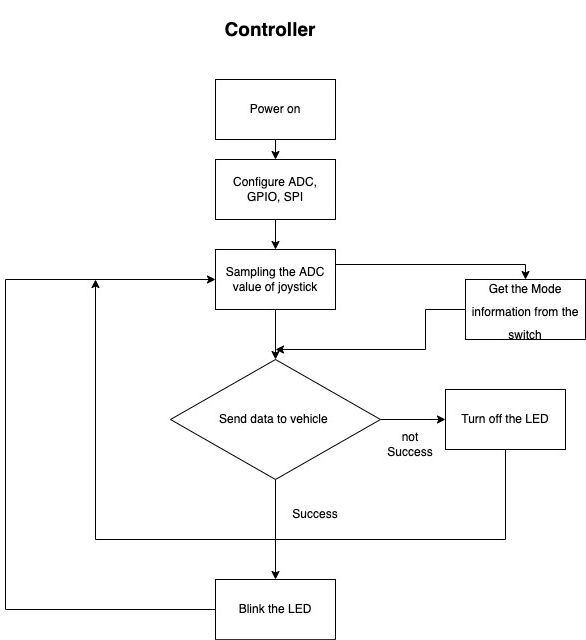
4.0 Sources Cited:

[1] “Understanding PID Control, Part 1: What Is PID Control?” MATLAB, 22 May 2018, [Online] Available: <https://www.youtube.com/watch?v=wkfEZmsQqiA&ab_channel=MATLAB>.

Appendix 1: Program Flowcharts

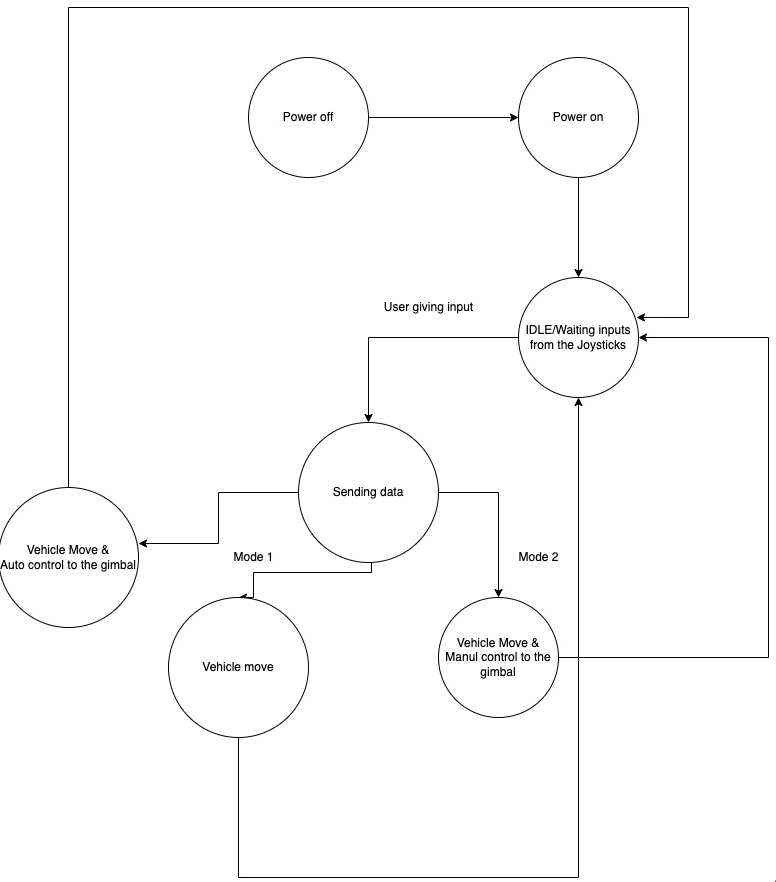
**

Figure

**

Figure

Appendix 2: State Machine Diagrams

**

Figure