ระบบป้องกันการเสียหลักของรถ

VEHICLE STABILITY CONTROL SYSTEM

ธีรภัทร์ เกิดไชยศรี
นิธิพัฒน์ ภู่น้อย
ภาคภูมิ สงวนประสิทธิ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมแมคคาทรอนิกส์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

ระบบป้องกันการเสียหลักของรถ

VEHICLE STABILITY CONTROL SYSTEM

ธีรภัทร์ เกิดไชยศรี
นิธิพัฒน์ ภู่น้อย
ภาคภูมิ สงวนประสิทธิ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมแมคคาทรอนิกส์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

VEHICLE STABILITY CONTROL SYSTEM

TEERAPAT KERTCHAISRI
NITIPAT PUNOI
PHAKPHUM SANGUANPRASIT

THIS THESIS IS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN MECHATRONICS ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2014

ปริญญานิพนธ์ปีการศึกษา 2557
ภาควิชาวิศวกรรมการวัดและควบคุม คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบป้องกันการเสียหลักของรถ
VEHICLE STABILITY CONTROL SYSTEM

ผู้จัดทำ นายธีรภัทร์ เกิดไชยศรี 54010637
นายนิธิพัฒน์ ภู่น้อย 54010712
นายภาคภูมิ สงวนประสิทธิ์ 54010981

.....อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์สุมิตร พนาอุดมทรัพย์)

.....อาจารย์ที่ปรึกษา
(รองศาสตราจารย์ ดร.เกียรติศักดิ์ คมวัชระ)

ระบบป้องกันการเสียหลักของรถ

โดย
นายธีรภัทร์ เกิดไชยศรี 54010637
นายนิธิพัฒน์ ภู่น้อย 54010712
นายภาคภูมิ สงวนประสิทธิ์ 54010981

อาจารย์ที่ปรึกษา
ผู้ช่วยศาสตราจารย์สุมิตร พนาอุดมทรัพย์
รองศาสตราจารย์ ดร.เกียรติศักดิ์ คมวัชระ

ปีการศึกษา 2557

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เป็นการนำชุดเซนเซอร์และการควบคุมแบบไร้สายมาประยุกต์ใช้งานกับรถบังคับวิทยุ ซึ่งจะเข้าไปช่วยควบคุมในส่วนของการเข้าโค้งในขณะที่รถมีความเร็วสูง เพื่อให้รถเข้าโค้งได้อย่างมีประสิทธิภาพมากขึ้น และเสริมสร้างความปลอดภัยในการใช้รถ ในส่วนของระบบควบคุมได้นำชุดเซนเซอร์ IMU (Inertial Measurement Unit) มาประยุกต์ใช้งานซึ่งประกอบด้วยเซนเซอร์วัดความเร่งเชิงเส้น เซนเซอร์วัดความเร็วเชิงมุม และเซนเซอร์วัดความเข้มสนามแม่เหล็กโลก และนอกจากนั้นยังได้ศึกษาเกี่ยวกับระบบการสื่อสารไร้สายของรถบังคับวิทยุ

ขั้นตอนดำเนินการ เริ่มจากการศึกษาปัญหาการเคลื่อนที่ของรถ เช่น ลักษณะการหลุดโค้ง การพลิกคว่ำ เป็นต้น จากนั้นทำการศึกษาการทำงานของชุดเซนเซอร์ IMU โดยทำงานร่วมกับวงจรไมโครคอนโทรเลอร์ที่เชื่อมต่อกับคอมพิวเตอร์ผ่านทางยูนิเวอร์แซลซีเรียลบัส (USB) ซึ่งเซนเซอร์ที่ทำการวัดจะแสดงค่าออกมาเป็นตัวเลข จากนั้นทำการลดสัญญาณรบกวนที่อาจเกิดจากสภาพแวดล้อมต่างๆ ที่ไม่สามารถควบคุมได้ และทำการปรับเทียบค่าเพื่อให้ได้ค่าที่มีความแม่นยำมากที่สุด เมื่อได้ค่าที่วัดได้จากเซนเซอร์แล้วได้ศึกษาระบบการสื่อสารไร้สายของรถบังคับวิทยุ ซึ่งทำให้ทราบค่าความกว้างของสัญญาณพัลส์ที่รีโมทส่งให้ตัวรับสัญญาณวิทยุ โดยใช้ออกซิโลสโคปทำการตรวจวัด ซึ่งขนาดความกว้างของสัญญาณพัลส์ที่ส่งจากรีโมทสามารถที่จะนำไปเป็นเซนเซอร์เพื่อทำงานร่วมกับชุดควบคุม นอกจากนั้นยังได้ทำอุปกรณ์ที่ใช้ตรวจวัดความเร็วของรถบังคับวิทยุโดยใช้ Linear Hall Sensor และจากการที่ได้ค่าต่างๆ จากการวัด จึงทำการเขียนโปรแกรมโดยใช้ภาษาซีซึ่งเขียนผ่านโปรแกรม Arduino เพื่อใช้เป็นระบบป้องกันการเสียหลักของรถ

VEHICLE STABILITY CONTROL SYSTEM

By
Mr. Teerapat Kertchaisri 54010637
Mr. Nitipat Punoi 54010712
Mr. Phakphum Sanguanprasit 54010981

Advisors
Asst. Prof. Sumit Panaudomsub
Assoc. Prof. Dr. Kiatisuk Komwatchara

Academic Year 2014

ABSTRACT

This thesis is the conduction of Inertial Measurement Units (IMU) and wireless control applied with the radio controller (RC) which will help to control the part of cornering while the car speeds highly for cornering of car more effectively and strengthening security in the control system of the using car. We has brought Inertial Measurement Units (IMU) for application consisting accelerometer, gyroscope and magnetic field sensor. In addition to IMU, we have also studied about wireless communication system of the radio controller toy.

In procedure, we start from the study of the function of the sensor IMU by collaborating with microcontroller which connected computer through universal serial bus (USB). The sensor of measurement will show the results in number data. Then we reduced the noise that can be caused by different conditions that we cannot control and calibration to get the most value of precision. When we measured values from the sensor, we have studied the wireless communication system of the radio controller which makes us know the width of the pulse signal sent to the remote radio receiver using the oscilloscope to measure. The width of the pulse signal that sent from the remote control can be used as a sensor to work with the controller. We also have a device used to measure the speed of the radio controller toy using Hall effect. From the values of the measurements, we make programming by using C++ which was written through the program Arduino to use as the main protection system of the car.

กิตติกรรมประกาศ

การจัดทำปริญญานิพนธ์ฉบับนี้สามารถสำเร็จลุล่วงไปได้ด้วยดีเพราะได้รับความช่วยเหลือเป็นอย่างดีจากผู้ช่วยศาสตราจารย์สุมิตร พนาอุดมทรัพย์ ที่ได้กรุณาให้คำปรึกษาแนะนำที่ดีมาโดยตลอดตั้งแต่ต้น รวมทั้งเอื้อเฟื้ออุปกรณ์ที่จำเป็นและความช่วยเหลืออื่นๆ ที่เป็นประโยชน์ต่อโครงงาน ผู้จัดทำรู้สึกซาบซึ้งและขอกราบขอบพระคุณอย่างสูง

ขอขอบคุณนายนัทธพงศ์ วัฒนศิริ ที่ให้คำปรึกษาในส่วนของการออกแบบวงจรและนายสมสิน ทองไกรรัตน์ ที่ให้คำปรึกษาในส่วนของการเขียนโปรแกรมและอุปกรณ์ตอบสนองต่างๆ

ขอบคุณเพื่อนๆ ทุกคนที่ให้กำลังใจสนับสนุนอุปกรณ์ที่ขาดเหลือกระตุ้นเตือน รวมทั้งคอยถามไถ่ความคืบหน้าของโครงงานอยู่เสมอ

สุดท้ายนี้ผู้จัดทำขอกราบขอบพระคุณบิดา มารดา และครอบครัว ที่คอยเป็นกำลังใจที่ดีตลอดมารวมถึงการสนับสนุนในเรื่องของงบประมาณที่ขาดเหลือตลอดจนเป็นแรงบันดาลใจที่ดีที่สุดที่ทำให้โครงงานนี้สำเร็จสมบูรณ์ลงได้

ผู้จัดทำ
นายธีรภัทร์ เกิดไชยศรี
นายนิธิพัฒน์ ภู่น้อย
นายภาคภูมิ สงวนประสิทธิ์

สารบัญ

สารบัญ(ต่อ)

สารบัญรูป

สารบัญรูป(ต่อ)

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

การศึกษาในภาควิชาวิศวกรรมการวัดและควบคุม เป็นการศึกษาทฤษฎีที่เป็นพื้นฐานหลักว่าด้วย วิศวกรรมเครื่องกล วิศวกรรมอิเล็กทรอนิกส์ วิศวกรรมระบบควบคุม วิทยาการคอมพิวเตอร์มาบูรณาการเข้าไว้ด้วยกัน เพื่อการออกแบบและพัฒนาระบบอุตสาหกรรมอย่างสมบูรณ์ ดังนั้นเพื่อให้เป็นไปตามวัตถุประสงค์นี้จึงจำเป็นต้องมีการศึกษาทางด้านกลศาสตร์วิศวกรรม วงจรอิเล็กทรอนิกส์ การเขียนโปรแกรมคอมพิวเตอร์ การเลือกใช้อุปกรณ์การวัดและแปลงสัญญาณ รวมทั้งการนำความรู้ที่ศึกษามาผนวกไว้ด้วยกันเพื่อทำการประยุกต์ใช้งานกับระบบอัตโนมัติทางกายภาพจริง

ในโครงงานนี้ได้ทำการศึกษาทดลองเกี่ยวกับระบบควบคุมการเสียหลักของรถ ซึ่งจากสถิติอุบัติเหตุจารจรทางบกของสำนักงานตำรวจแห่งชาติ พบว่าการขับรถเร็วเป็นสาเหตุหลักที่ทำให้เกิดอุบัติบนท้องถนนมากที่สุด ทั้งนี้อุบัติเหตุที่เกิดขึ้นจากการขับรถเร็วส่วนหนึ่งนั้นมาจากปัญหาของการเสียหลักของรถ เช่น การหลุดโค้ง การพลิกคว่ำ เป็นต้น และในปัจจุบันมีการให้ความสำคัญกับความปลอดภัยมากขึ้น จึงมีการนำเทคโนโลยีที่มีในปัจจุบันมาพัฒนาระบบความปลอดภัยเพื่อสร้างความปลอดภัย และลดความสูญเสียที่อาจจะเกิดขึ้น ดังนั้นโครงงานนี้จึงเลือกที่จะศึกษาและออกแบบระบบป้องกันการเสียหลักของรถโดยใช้ IMU เซนเซอร์เป็นอุปกรณ์การวัดเพื่อใช้งานร่วมกับวงจรไมโครคอนโทรลเลอร์ ที่สามารถทำงานคู่กับโปรแกรม Arduino ใช้ในการออกแบบระบบ ซึ่งทำให้สามารถขับรถด้วยความเร็วที่ปลอดภัยได้อย่างเหมาะสมกับสภาวะปัจจุบันของรถ

1.2 วัตถุประสงค์ของโครงงาน

1. ศึกษาการเคลื่อนที่ของรถบังคับวิทยุ และปัญหาทางการเคลื่อนที่ที่เกิดกับรถบังคับวิทยุ
2. ศึกษาไมโครคอนโทรลเลอร์ และอุปกรณ์ตอบสนอง เพื่อใช้ในการวิเคราะห์และควบคุมรถบังคับวิทยุ
3. ศึกษาและออกแบบวิธีลดปัญหาทางการเคลื่อนที่ที่เกิดกับรถบังคับวิทยุ
4. ออกแบบระบบควบคุมการเสียหลักของรถบังคับวิทยุเพื่อลดอุบัติเหตุจากการเสียหลัก

1.3 ขอบเขตของโครงงาน

รถบังคับวิทยุที่มีระบบควบคุมการเสียหลัก โดยใช้ IMU เซนเซอร์เป็นอุปกรณ์การวัดเพื่อใช้งานร่วมกับวงจรไมโครคอนโทรลเลอร์ วงจรอิเล็กทรอนิกส์ ที่สามารถทำงานควบคู่กับการเขียนโปรแกรมผ่านโปรแกรม Arduino ที่ใช้ในการออกแบบระบบ

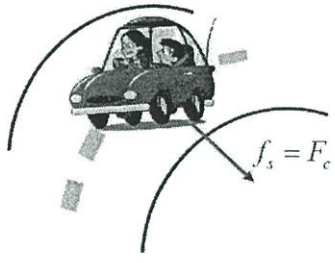
1.4 ขั้นตอนและการจัดทำโครงงาน

เริ่มจากการศึกษาทฤษฎี การใช้ไมโครคอนโทรลเลอร์ และใช้เครื่องมือที่จำเป็นอย่างมีประสิทธิภาพ ทำให้ค่าที่ได้รับจาก IMU เซนเซอร์มีค่าที่แม่นยำและเที่ยงตรงจนสามารถนำไปใช้งานได้ จากนั้นทำการออกแบบวิธีการแก้ปัญหาทางการเคลื่อนที่ และสร้างกลไกตามแบบแผน ทดลองติดตั้งอุปกรณ์ลงบนรถบังคับวิทยุ ทดสอบการเคลื่อนที่ของรถบังคับวิทยุ และแก้ไขปัญหาที่เกิดขึ้น

บทที่ 2
ทฤษฎีและความเกี่ยวข้อง

2.1 หลักการเลี้ยวของรถยนต์

ในชีวิตประจำวันคงคุ้นเคยกับการนั่งรถไม่ว่าจะเป็นรถยนต์ รถประจำทาง หรือรถจักรายานยนต์ ในขณะที่รถวิ่งไปบนท้องถนนที่เป็นทางโค้ง ผู้ขับจะต้องลดความเร็วลงเพื่อให้เข้าโค้งได้อย่างปลอดภัย หรืออาจสังเกตเห็นว่า รถจักรยานยนต์บางคันต้องเอียงทำมุมกับถนนราบในขณะที่เข้าโค้ง หรือเราอาจสังเกตเห็นบริเวณทางโค้งพื้นถนนจะยกตัวให้ลาดเอียง ทั้งนี้เนื่องจากรถวิ่งบนท้างโค้งเป็นการเคลื่อนที่แบบวงกลม จึงมีแรงสู่ศูนย์กลางมากระทำต่อรถนั่นเอง



รูปที่ 2.1 ลักษณะการเข้าโค้งของรถยนต์

ขณะที่รถวิ่งบนทางโค้งซึ่งเป็นการเคลื่อนที่แบบวงกลม ดังนั้นต้องมีแรงสู่ศูนย์กลางกระทำต่อรถ เมื่อพิจารณาแรงที่กระทำต่อรถ พบว่าขณะที่รถเลี้ยวโค้งรถจะพยายามไถลออกจากโค้ง จึงมีแรงเสียดทาน ที่พื้นกระทำต่อล้อรถในทิศทางพุ่งเข้าในแนวผ่านจุดศูนย์กลางความโค้ง ดังนั้นแรงเสียดทานที่พื้นกระทำต่อล้อรถคือแรงสู่ศูนย์กลาง

แรงเสียดทาน = แรงสู่ศูนย์กลาง (2.1)

การหาอัตราเร็วของรถสูงสุดที่รถเข้าโค้งได้อย่างปลอดภัย จากความสัมพันธ์ของสมการที่ (2.1)

$$f_{s(max)} = F_c \tag{2.2}$$

$$\mu mg = \frac{mv_{max}^2}{R} \tag{2.3}$$

$$v_{max} = \sqrt{\mu gR} \tag{2.4}$$

เมื่อ μ คือ สัมประสิทธิ์ของความเสียดทานระหว่างล้อรถกับถนน

v คือ อัตราเร็วสูงสุดขณะเลี้ยวรถได้อย่างปลอดภัย

g คือ ความเร่งโน้มถ่วงของโลก

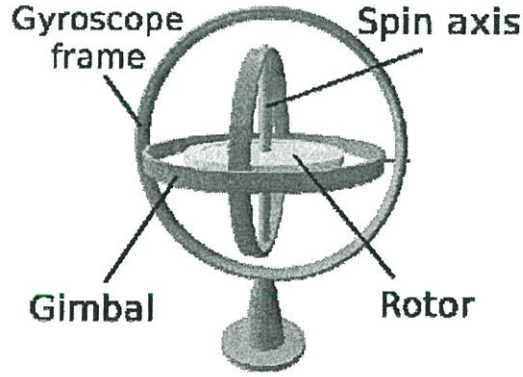
R คือ รัศมีความโค้งของถนน

2.2 Inertial Measurement Unit (IMU)

เซนเซอร์หรืออุปกรณ์ที่ใช้ใน INS (Inertial Navigation System) ถูกเรียกว่า Inertial Measurement Units (IMU) ซึ่งเป็นส่วนประกอบหลักของ INSs ที่ใช้ในเครื่องบินยานอวกาศเรือและจรวดขีปนาวุธ IMU ประกอบด้วย 2 ส่วนหลักคือ เซนเซอร์วัดความเร่ง (Accelerometers) 3 ทิศทางและไจโรสโคป (Gyroscopes) 3 ทิศทางซึ่งรับความเร่งยานพาหนะและความเร็วเชิงมุมตามลำดับ

2.2.1 ไจโรสโคป (Gyroscope)

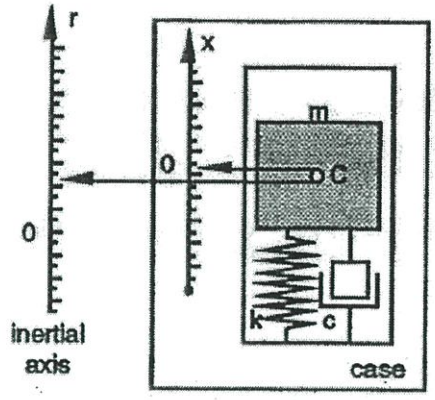
ไจโรสโคป (Gyroscope) เป็นอุปกรณ์สำหรับการวัดหรือการรักษาการปรับทิศทางขึ้นอยู่กับหลักการของการอนุรักษ์โมเมนตัมเชิงมุม



รูปที่ 2.2 Mechanic Gyroscope ซึ่งมี Two-Degree of Freedom

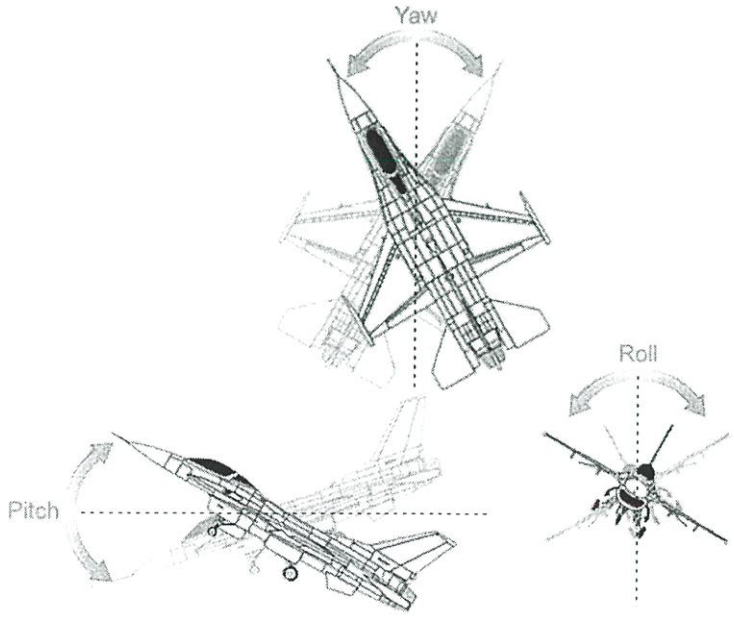
2.2.2 เซนเซอร์วัดความเร่ง (Accelerometer)

เซนเซอร์วัดความเร่ง (Accelerometer) เป็นอุปกรณ์ที่ใช้วัดความเร่งตามแนวแกนที่เฉพาะเจาะจง ตั้งข้อสังเกตได้ว่าเซนเซอร์วัดความเร่งใน IMU รับเพียง Specific Forces แนวคิดนี้เป็นสิ่งสำคัญใน Inertial Navigation หรือระบบนำทางอาศัยแรงเฉื่อย



รูปที่ 2.3 โครงสร้างของเซนเซอร์วัดความเร่ง

2.3 การคำนวณค่ามุม Roll, Pitch และ Yaw



รูปที่ 2.4 มุม Roll มุม Pitch และมุม Yaw

เนื่องจากไจโรสโคปเป็นเซนเซอร์วัดความเร็วเชิงมุม ซึ่งค่าที่ได้จากไจโรสโคปเป็นอัตราการเปลี่ยนแปลงของมุม (Gyro Rate) ทำให้สามารถคำนวณหาค่ามุม Roll, Pitch และ Yaw ได้โดยวิธี Discrete Integral

$$\theta_{roll} = \int gryData_X \, dt \tag{2.5}$$

$$\theta_{pitch} = \int gryData_Y \, dt \tag{2.6}$$

$$\theta_{yaw} = \int gryData_Z \, dt \tag{2.6}$$

ส่วนการคำนวณโดยใช้เซนเซอร์วัดความเร่งสามารถคำนวณหาค่า Roll และ Pitch ได้แต่เนื่องจากข้อจำกัดของ Accelerometer ค่า Yaw ที่วัดได้จะมีความคลาดเคลื่อนสูงจนนำมาใช้ไม่ได้ ถ้าใช้แค่เซนเซอร์วัดความเร่งอย่างเดียวมุม Yaw จะไม่ถูกนำมาใช้ ดังนั้นค่าที่ถูกนำมาใช้คือ Roll และ Pitch เท่านั้น

$$\theta_{roll} = \tan^{-1}\left(-\frac{accData_X}{accData_Z}\right) \tag{2.7}$$

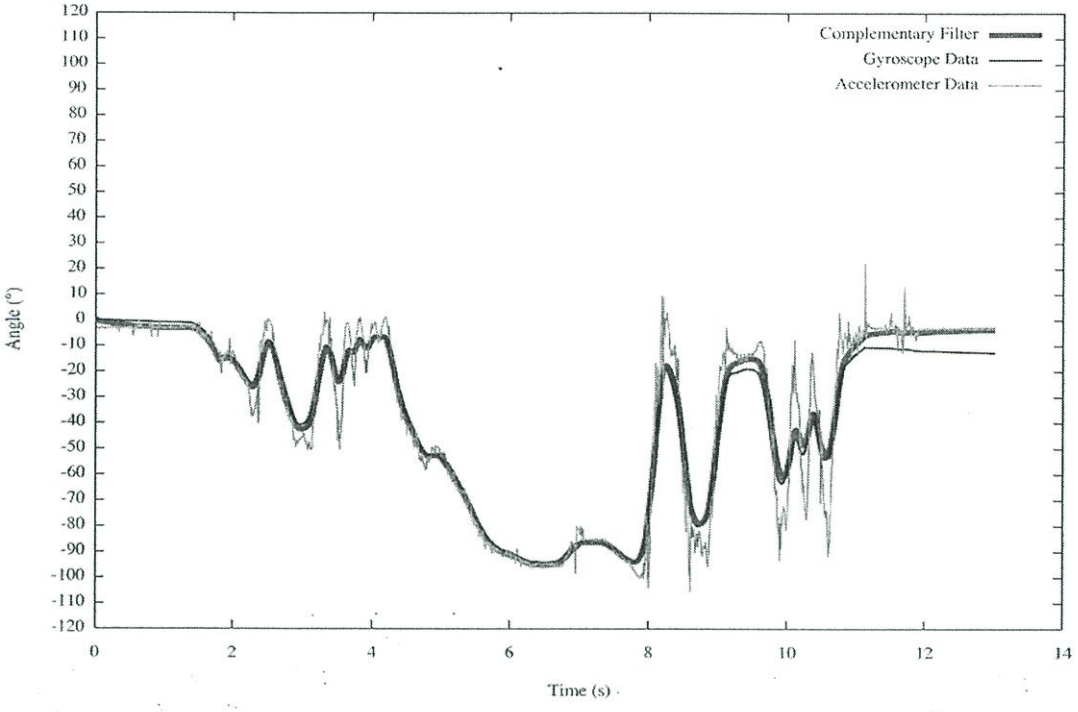
$$\theta_{pitch} = \tan^{-1}\left(-\frac{accData_Y}{accData_Z}\right) \tag{2.8}$$

โดยทั่วไปค่าความเร่งที่วัดได้ในแต่ละแกน (Xg, Yg, Zg) จะมีสัญญาณรบกวนผสมอยู่ จึงควรทำการลดปริมาณสัญญาณรบกวนนี้ออกไป โดยใช้ Complementary Filter

2.4 Complementary Filter

Complementary Filter เป็นตัวกรองดิจิตอลโดยนำค่าจากเซนเซอร์วัดความเร่งและไจโรสโคปมาคำนวณร่วมกันเพื่อปรับปรุงค่าให้มีความแม่นยำมากขึ้น เนื่องจากไจโรสโคปเมื่อมีการใช้งานจะเกิด การดริฟ (Drift) หรือค่าที่อ่านได้ ณ จุดเดียวกันได้ค่าไม่เท่ากันเมื่อเวลาผ่านไปในระยะหนึ่งทำให้ค่าที่อ่านได้ไม่ถูกต้อง และเซนเซอร์วัดความเร่งเมื่ออ่านค่า Output ที่อ่านได้จะไม่นิ่ง เนื่องจากจะมีความเร่งเกิดขึ้นจากค่าความโน้มถ่วงของโลกแต่เซนเซอร์วัดความเร่งไม่เกิดการดริฟ ดังนั้นจึงใช้หลักการ Complementary Filter มาคำนวณค่าของมุมต่างๆ ดังสมการที่ 2.9

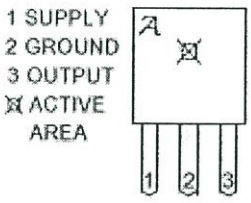
$$angle_n = 0.98(angle_{n-1} + gyrData \, dt) + 0.02(accData) \tag{2.9}$$



รูปที่ 2.5 กราฟเปรียบเทียบมุมวัดได้จากเซนเซอร์วัดความเร่ง ไจโรสโคป และ Complementary Filter

2.5 Hall Effect Sensor

แม่เหล็ก (Magnet) เป็นสิ่งที่สามารถดูดวัสดุบางชนิดได้ เช่น เหล็ก นิกเกิล โคบอลต์ เป็นต้น การที่แม่เหล็กดูดสารบางอย่างได้ เนื่องจากมีสนามแม่เหล็ก (Magnetic Field) ในบริเวณโดยรอบแม่เหล็ก เราสามารถตรวจสอบว่าบริเวณใดมีสนามแม่เหล็กหรือไม่ โดยใช้เข็มทิศ แต่ไม่สามารถทราบได้ว่ามีค่าเท่าใด นักวิทยาศาสตร์พยายามวัดสนามแม่เหล็กด้วยวิธีการต่าง ๆ แต่ในปัจจุบันสามารถวัดสนามแม่เหล็กได้สะดวกและรวดเร็วโดยใช้ Linear Hall Sensor ซึ่งทำงานโดยอาศัยหลักการของปรากฏการณ์ฮอลล์ (Hall Effect) ที่ทำให้เกิดความต่างศักย์ ซึ่งเป็นสัดส่วนตรงกับความเข้มของสนามแม่เหล็กที่ผ่านในแนวดิ่ง เมื่อนำ Linear Hall Sensor ไปต่อกับโวลต์มิเตอร์ แล้วนำไปวางใกล้บริเวณที่มีสนามแม่เหล็กก็จะทำให้ทราบค่าความเข้มของสนามแม่เหล็กได้

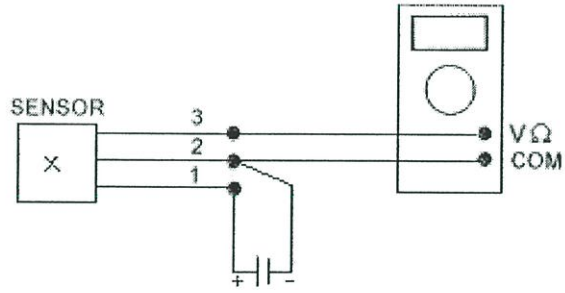


รูปที่ 2.6 ขนาดและลักษณะของ Linear Hall Effect Sensor IC

ในโครงงานนี้เลือกใช้ Linear Hall Effect Sensor IC A1301 มีสมบัติดังนี้

1. Input Voltage 4.5-6
2. Offset Voltage 2.5 V
3. Sensitivity 2.5 mV/G

เมื่อต่อแหล่งจ่ายไฟกระแสตรงหรือเซลล์ไฟฟ้า 4.5-6 โวลต์ เข้ากับขา 1 และขา 2 และ ต่อโวลต์มิเตอร์เข้ากับขา 2 และขา 3 ดังรูปที่ 2 โวลต์มิเตอร์จะแสดงค่าประมาณ 2.5 โวลต์ ค่านี้เป็นความต่างศักย์ขณะที่ไม่มีสนามแม่เหล็ก เรียกว่า Offset Voltage ค่านี้อาจเปลี่ยนแปลงได้เล็กน้อยขึ้นอยู่กับโวลเตจของแหล่งจ่ายไฟกระแสตรงที่ต่อกับขา 1 และขา 2 แต่จะมีค่าประมาณครึ่งหนึ่งของโวลเตจของแหล่งจ่ายไฟกระแสตรง



รูปที่ 2.7 การต่อ Linear Hall Sensor กับแหล่งจ่ายไฟกระแสตรงและโวลต์มิเตอร์

เมื่อนำแม่เหล็กเข้าใกล้ Active Area ของ Linear Hall Sensor ความต่างศักย์จะมีค่าเพิ่มขึ้นหรือลดลงขึ้นอยู่กับทิศของสนามแม่เหล็ก ความต่างศักย์ที่เปลี่ยนไปมีความสัมพันธ์กับความเข้มของสนามแม่เหล็กหรือความหนาแน่นฟลักซ์แม่เหล็ก (Magnetic Flux Density) ดังนี้

$$\Delta V = \frac{V_{out} - V_0}{Sensitivity} \quad (2.10)$$

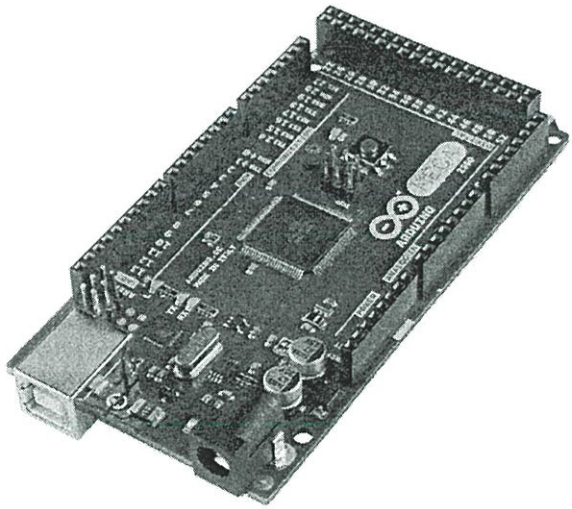
เมื่อ V_{out} คือ ความต่างศักย์ขณะไม่มีสนามแม่เหล็ก (mV)

V_0 คือ ความต่างศักย์ขณะมีสนามแม่เหล็ก (mV)

$Sensitivity$ คือ สัมประสิทธิ์ความไว (mV/G)

2.6 ไมโครคอนโทรลเลอร์ (Microcontroller)

ไมโครคอนโทรลเลอร์เป็นอุปกรณ์ที่มีหน่วยประมวลผล และความจำขนาดเล็กสามารถรับ-ส่งข้อมูลได้ทั้งแบบดิจิตอลและอนาล็อก ใช้พลังงานน้อย ทำให้เป็นที่นิยมในการใช้งานในรูปแบบที่เรียกว่า Embedded

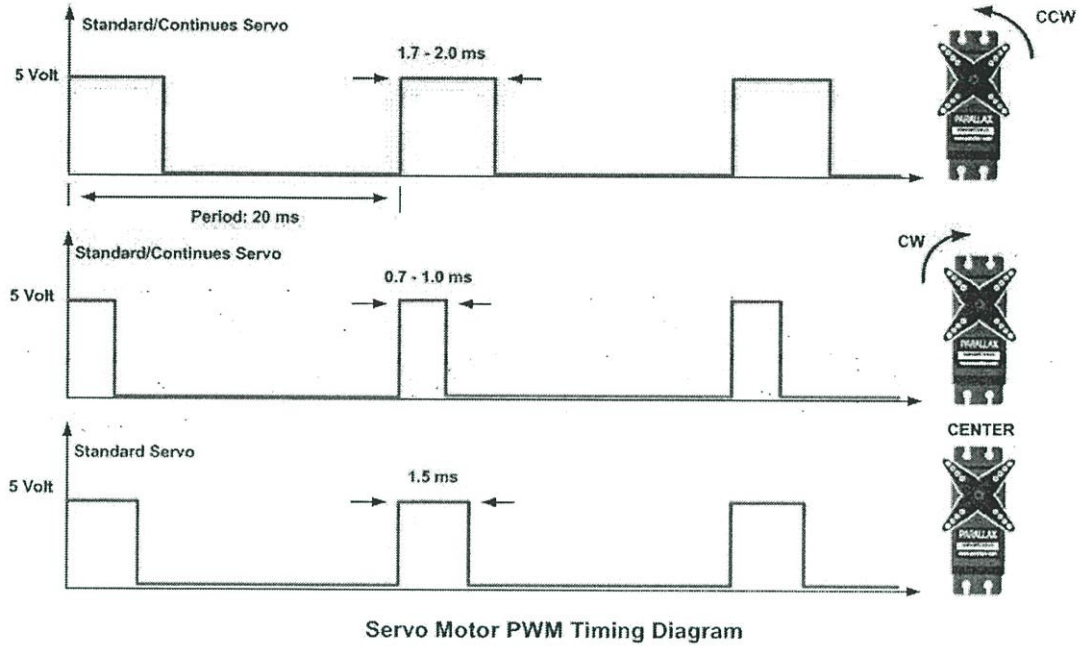


รูปที่ 2.8 Arduino Mega 2560

Arduino เป็นบอร์ดไมโครคอนโทรเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source มีการเปิดเผยข้อมูลทั้งด้าน Hardware และ Software ตัวบอร์ด Arduino ถูกออกแบบมาให้ใช้งานได้ง่าย

2.7 Pulse Width Modulation

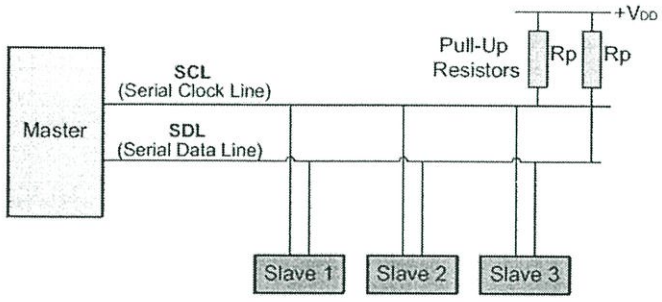
Pulse Width Modulation หรือ PWM จะใช้หลักการสร้างสัญญาณพัลส์แบบสแควร์เวฟ ที่สามารถปรับเปลี่ยนความถี่ และ Duty Cycle เพื่อให้ค่าเฉลี่ยของสัญญาณทั้งหมดออกมาเป็นค่าที่ต้องการ ซึ่งในโครงงานนี้จะนำไปใช้ในการควบคุมเซนเซอร์



รูปที่ 2.9 ตัวอย่างการควบคุมเซอร์โวด้วย PWM

2.8 การเชื่อมต่ออุปกรณ์แบบ I2C (I^2C)

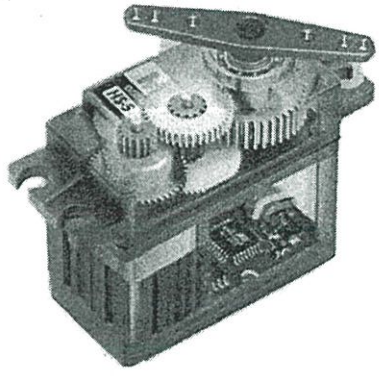
I^2C หรือ I2C Bus ย่อมาจาก Inter Integrate Circuit Bus (IIC) นิยมเรียกสั้นๆ ว่า I^2C BUS เป็นการสื่อสารอนุกรม แบบซิงโครนัส (Synchronous) เพื่อใช้ติดต่อสื่อสารระหว่างไมโครคอนโทรลเลอร์ (MCU) กับอุปกรณ์ภายนอก ซึ่งถูกพัฒนาขึ้นโดยบริษัท Philips Semiconductors โดยใช้สายสัญญาณเพียง 2 เส้นเท่านั้น คือ Serial Data (SDA) และสาย Serial Clock (SCL) ซึ่งสามารถเชื่อมต่ออุปกรณ์ จำนวนหลายๆ ตัวเข้าด้วยกันได้ ทำให้ MCU ใช้พอร์ตเพียง 2 พอร์ตเท่านั้น



รูปที่ 2.10 ลักษณะการการเชื่อมต่ออุปกรณ์แบบ I2C BUS

2.9 เซอร์โว (Servo)

หลักการทำงานของเซอร์โวคือ การเปลี่ยนคำสั่งที่เป็นสัญญาณไฟฟ้าเป็นการเคลื่อนที่ของแขนเซอร์โว

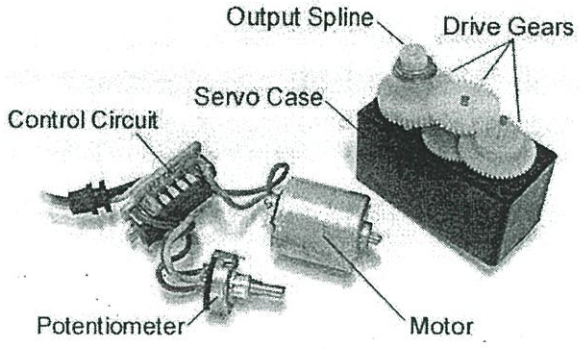


รูปที่ 2.11 เซอร์โวมอเตอร์

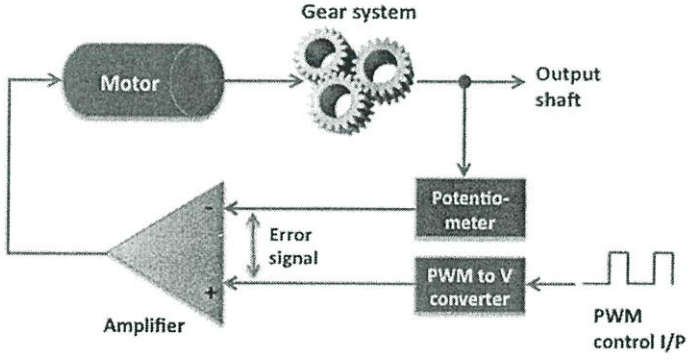
องค์ประกอบพื้นฐานของเซอร์โวมอเตอร์

1. Servo Case ซึ่งส่วนใหญ่จะทำมาจากพลาสติก
2. Motor ซึ่งเป็นส่วนให้กำลังในการหมุนของเซอร์โว
3. Control Circuit มีหน้าที่ในการถอดรหัสสัญญาณควบคุมจากรีซีฟ ซึ่งส่งมาเป็นแบบ PWM และส่งการควบคุมไปสั่งการทำงานของมอเตอร์ให้หมุนแขนของเซอร์โวให้อยู่ในตำแหน่งที่ได้ถอดรหัสมา
4. Potentiometer คือ ส่วนที่ตรวจวัดตำแหน่งของเซอร์โว และส่งสัญญาณกลับไปยัง Control Circuit เพื่อแก้ไขตำแหน่งให้ถูกต้องตามสัญญาณที่ได้เซ็ตไว้
5. Drive Gear คือ ชุดทดรอบจากการหมุนของมอเตอร์เพื่อให้ได้แรงบิดที่สูง
6. Output Spline คือ ส่วนที่ป้องกันการเสียดสีระหว่าง Servo Case และ Output Shaft ซึ่งอาจใช้อุปกรณ์ประเภท Baring เพื่อช่วยลดแรงเสียดทานที่ดี

7. Servo Wire สายไฟของเซอร์โวจะมีอยู่สามเส้นซึ่งจะติดเป็นชุดเดียวกันโดย
เส้นที่ 1 จ่ายไฟกระแส + DC ซึ่งแรงดันปรกติจะอยู่ที่ 5-6 โวลท์
เส้นที่ 2 เป็นสาย Ground หรือเป็นขั้ว DC
เส้นที่ 3 เป็นสายสัญญาณ โดยที่รีซีฟจะส่งสัญญาณลักษณะ On/Off Pulsed



รูปที่ 2.12 องค์ประกอบพื้นฐานของเซอร์โวมอเตอร์

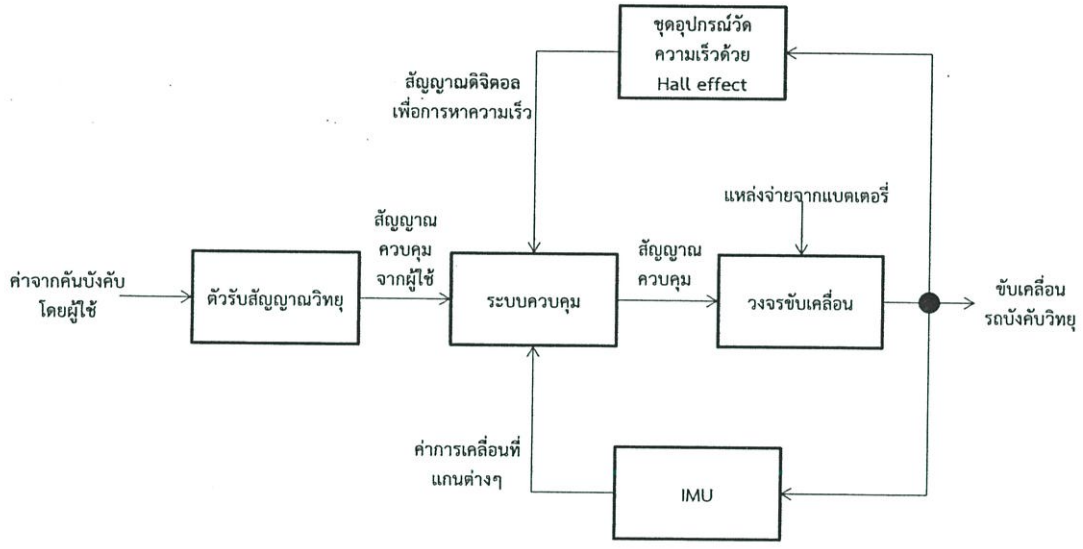


รูปที่ 2.13 ระบบการทำงานของเซอร์โวมอเตอร์

เซอร์โวจะทำงานเมื่อ Control Circuit ของเซอร์โวได้รับสัญญาณการควบคุมตำแหน่งของเซอร์โว โดยสัญญาณที่ส่งมาจะเป็นสัญญาณแบบ PWM (Pulse Width Modulation) จากนั้น Control Circuit จะถอดรหัสสัญญาณ PWM ที่ได้ให้เป็นตำแหน่งของเซอร์โวที่ถูกต้องโดยเปรียบเทียบค่าตำแหน่งปัจจุบันกับสัญญาณกลับจาก Potentiometer แล้วจึงส่งแรงดันไฟฟ้าไปยังมอเตอร์ให้ไปหมุนไปในทิศทางที่จะทำให้ตำแหน่งของ Potentiometer มีค่าที่ถูกต้องเท่ากับค่าที่ได้ถอดรหัสมา ซึ่งขณะที่มอเตอร์หมุนก็จะมีเฟืองที่ไปต่อกับแกนของ Potentiometer ด้วย ดังนั้นกระบวนการนี้จะเกิดขึ้นซ้ำๆจนกว่าค่าของ Potentiometer จะมีค่าเท่ากับการถอดรหัสสัญญาณที่ได้รับมาจากรีซีฟการทำงานของมอเตอร์จึงจะหยุด แต่กระบวนการทำงานของ Control Circuit จะยังทำงานอยู่ตลอดเวลาเพียงแต่หากค่าของ Potentiometer มีค่าเท่ากับสัญญาณที่ถอดรหัสมาจากรีซีฟแล้วก็จะไม่มีการส่งแรงดันไฟฟ้าไปยังมอเตอร์

บทที่ 3
วิธีการดำเนินงาน

ในการทำโครงงานระบบป้องกันการเสียหลักของรถ เริ่มดำเนินโครงงานตั้งแต่การศึกษาปัญหาการเคลื่อนที่ที่เกิดขึ้นกับรถ เช่น การหลุดโค้ง การพลิกคล่ำ เป็นต้น จากนั้นเริ่มสร้างสรรค์วิธีการที่สามารถจำลองปัญหาดังกล่าว โดยคำนึงถึงความใกล้เคียงของปัญหา ทรัพยากร การเข้าถึง และความรู้ในระดับอุดมศึกษา จึงเลือกใช้รถบังคับวิทยุในการจำลองปัญหา ซึ่งสามารถตอบเหตุผลข้างต้นได้เป็นอย่างดี ดังนั้นจึงจำกัดขอบเขตของโครงงานลงมาที่รถบังคับวิทยุ



รูปที่ 3.1 แผนผังการทำงานของสัญญาณควบคุม

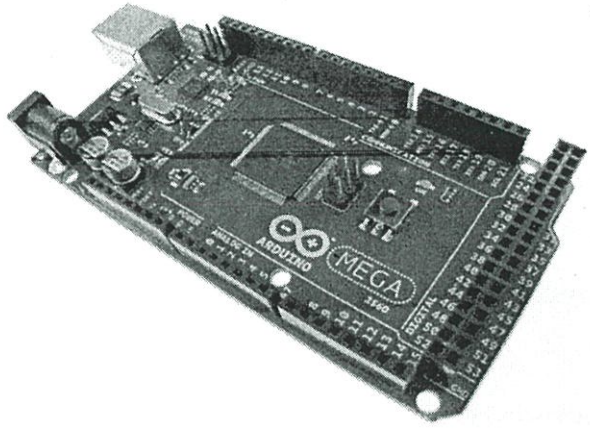
3.1 การศึกษาและเลือกใช้ชุดอุปกรณ์วัดค่าการเคลื่อนที่

ในโครงงานนี้จำเป็นต้องใช้ค่าที่เกี่ยวข้องกับการเคลื่อนที่ และนำค่าเหล่านั้นไปใช้ในการประมวลผล และควบคุมต่อไป ซึ่งการจะได้มาซึ่งข้อมูลที่ต้องการ จึงต้องศึกษาอุปกรณ์ต่างๆ และนำมาเลือกใช้อุปกรณ์ที่เหมาะสม โดยควรคำนึงถึงการทำงานในภาพรวมของทุกอุปกรณ์ และให้ได้ประสิทธิภาพ

3.1.1 การศึกษาและเลือกใช้ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ก็เหมือนแกนกลางของระบบ ซึ่งเป็นจุดศูนย์รวมของการรับค่า การประมวลผล และการส่งค่าออก โดยการจะเลือกใช้ไมโครคอนโทรลเลอร์ ก็ควรคำนึงถึงทั้งสามภาคข้างต้นให้เหมาะสมด้วย ในโครงงานนี้เลือกใช้เป็น Arduino Mega 2560 ไมโครคอนโทรลเลอร์ตัวนี้

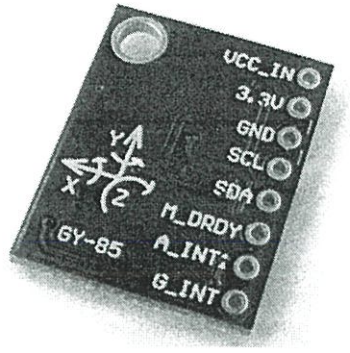
สามารถรับค่าได้หลายแบบคือ การรับค่าแบบดิจิตอล การรับค่าแบบอนาลอก และการรับค่าแบบ Interude ซึ่งการรับค่าทั้งหมดมีความจำเป็นต้องใช้ร่วมกับอุปกรณ์ได้หลายชิ้น และไมโครคอนโทรลเลอร์ตัวนี้สามารถส่งค่าออกเป็น สัญญาณดิจิตอลและเป็นสัญญาณ PWM ได้ ซึ่งต้องใช้ในการควบคุมในส่วนต่อไป การประมวลผลของไมโครคอนโทรลเลอร์ถือว่ามีความเร็วในระดับที่เหมาะสมกับการใช้งาน ภาษาที่ใช้เป็นภาษาซี ซึ่งเป็นภาษาพื้นฐานและไม่ยาก มีฟังก์ชันให้เลือกใช้มากมาย ทั้งฟังก์ชันพื้นฐานและฟังก์ชันเฉพาะทาง ซึ่งจากข้อมูลทั้งหมดจึงเลือกใช้ไมโครคอนโทรลเลอร์ Arduino Mega 2560



รูปที่ 3.2 ไมโครคอนโทรลเลอร์ Arduino Mega 2560

3.1.2 การศึกษาและเลือกใช้ชุดเซนเซอร์วัดค่าการเคลื่อนที่

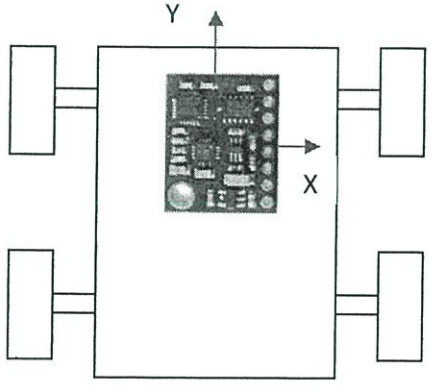
เพื่อที่จะสามารถรู้ถึงสถาวะปัจจุบันของรถได้ จะถูกแสดงเป็นค่าต่างๆ ซึ่งค่าต่างๆ เหล่านั้นจะถูกวัดได้ด้วยชุดเซนเซอร์วัดค่าการเคลื่อนที่ (Inertial Measurement Unit (IMU)) โดยที่เลือกใช้ในโครงงานคือ โมดูล GY-85 โดยส่งข้อมูลแบบ I2C ในโมดูล GY-85 ประกอบไปด้วย Accelerometer(ADXL345) Gyroscope(ITG3205) และ Magnetic Field (HMC5883L) โดย Accelerometer เป็นเซนเซอร์ที่วัดค่าออกมาเป็นค่าความเร่งเชิงเส้นทั้งหมด 3 แกนคือ แกน X แกน Y และแกน Z Gyroscope เป็นเซนเซอร์ที่วัดค่าออกมาเป็นค่าความเร็วเชิงมุมทั้งหมด 3 แกนคือ หมุนรอบแกน X หมุนรอบแกน Y และหมุนรอบแกน Z ส่วน Magnetic Field เป็นเซนเซอร์ที่วัดค่าสนามแม่เหล็กโลก โดยหลักๆ จะนำค่าจาก Accelerometer และ Gyroscope มาประมวลผลต่อไป



รูปที่ 3.3 แบบที่ใช้ยึดอุปกรณ์เข้ากับบอร์ดควบคุม

3.2 การหามุมเอียง

การหามุมเอียงของรถสามารถหาได้จากการนำค่าจาก Accelerometer ทั้ง 3 แกนมาคำนวณเพื่อหาได้ โดย IMU ตั้งแกน Z ตามแรงโน้มถ่วงของโลก ให้ทิศหน้ารถเป็นแกน Y และให้ทิศข้างตัวเป็นแกน X เชื่อมต่อไมโครคอนโทรลเลอร์กับ IMU ใช้โปรแกรมการคำนวณตามการหามุม Roll และมุม Pitch ค่ามุม Roll คือมุมรอบแกน Y ซึ่งถ้า IMU ตั้งฉากกับแรงโน้มถ่วงของโลกจะให้ค่าเป็นศูนย์ และค่ามุม Pitch คือ มุมรอบแกน X ซึ่งถ้า IMU ตั้งฉากกับแรงโน้มถ่วงของโลกจะให้ค่าเป็นศูนย์ ซึ่งเมื่อได้ค่ามุม Roll และมุม Pitch สามารถนำไปใช้ต่อไป



รูปที่ 3.4 ลักษณะการติดตั้งชุดเซนเซอร์กับตัวรถบังคับวิทยุ

3.3 การเลือกใช้รถบังคับวิทยุ

รถบังคับวิทยุมีเลือกใช้อยู่หลากหลายรูปแบบ ทั้งรถวิ่งทางลาด รถบัคกี้ และรถทรัก รถบังคับวิทยุที่ใช้ในโครงงานคือ รถทรัก รุ่น Massive สเกล 1 : 10 มอเตอร์ไฟฟ้าไร้แปรงถ่าน ขับเคลื่อนสี่ล้อ ความถี่วิทยุ 2.4 GHz ซึ่งเลือกใช้โดยการเปรียบเทียบราคา และความคุ้มค่า

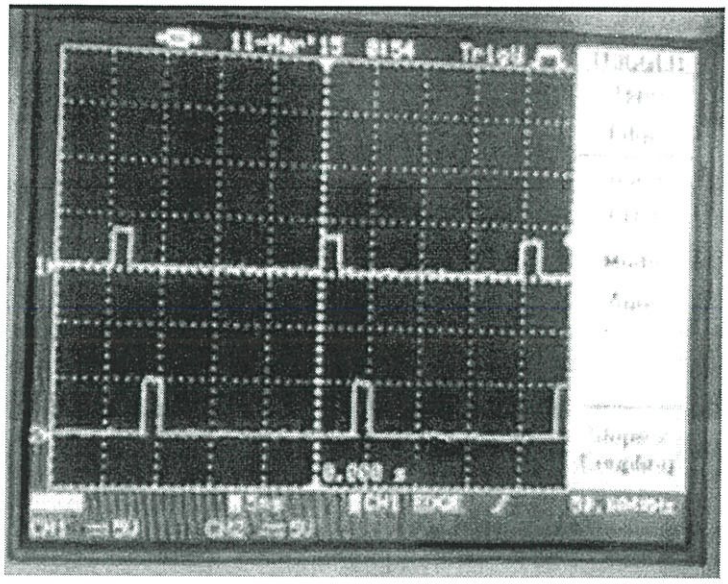
รูปที่ 3.5 Truck รุ่น Massive

3.4 การศึกษาและควบคุมรถบังคับวิทยุ

เนื่องจากโครงงานนี้ต้องการสร้างสรรค์ระบบควบคุมที่มีประสิทธิภาพ จึงต้องศึกษาการทำงานของตัวรถ และศึกษาการควบคุมอุปกรณ์ภายในตัวรถ ในตัวรถประกอบด้วยอุปกรณ์หลักๆ 3 ส่วน คือ ตัวรับสัญญาณวิทยุ เซอร์โวมอเตอร์และมอเตอร์ไร้แปรงถ่าน

3.4.1 การศึกษาตัวรับสัญญาณวิทยุ

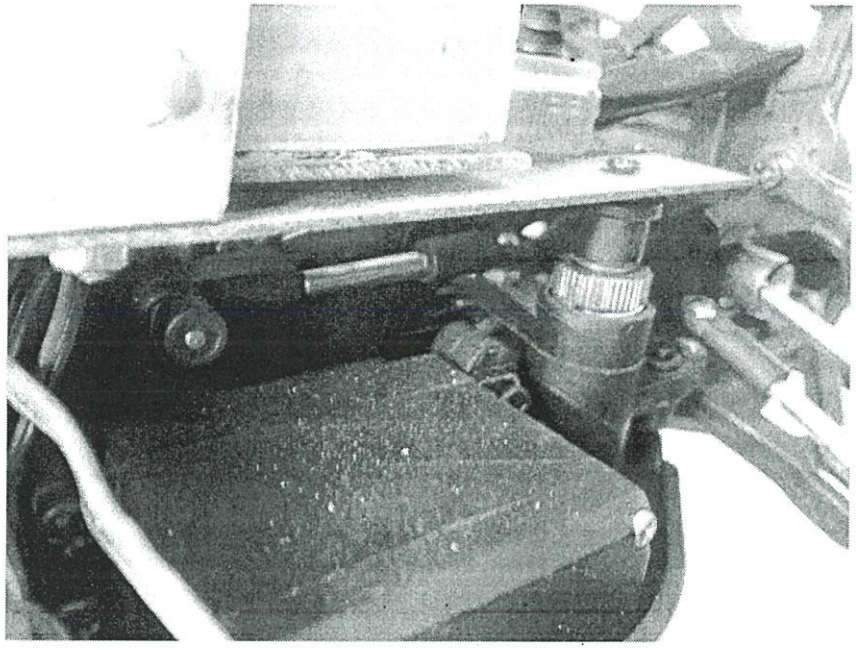
ตัวรับสัญญาณวิทยุเป็นอุปกรณ์ที่รับค่าจากคันบังคับเป็นสัญญาณ แล้วประมวลผลเป็นสัญญาณควบคุม โดย Channel 1 ส่งสัญญาณออกไปเพื่อควบคุมเซอร์โวมอเตอร์และ Channel 2 ส่งสัญญาณออกเพื่อควคุมมอเตอร์ไร้แปรงถ่าน โดยสัญญาณควบคุมนั้นเป็นการส่งค่าแบบ PWM มีคาบเป็น 20 มิลลิวินาที การหมุนคันบังคับจะเปลี่ยนแปลงช่วง High ให้มีความกว้างเปลี่ยนแปลงในระดับไมโครวินาที โดยความกว้างของช่วง High ของทั้ง 2 Channel มีช่วงอยู่ระหว่าง 600-2500 ไมโครวินาที จากจุดนี้ทำให้รู้วิธีการควบคุมเซอร์โวมอเตอร์และมอเตอร์ไร้แปรงถ่าน



รูปที่ 3.6 การวัดสัญญาณจากตัวรับสัญญาณวิทยุทั้ง Channel 1 และ Channel 2

3.4.2 การควบคุมเซอร์โวมอเตอร์

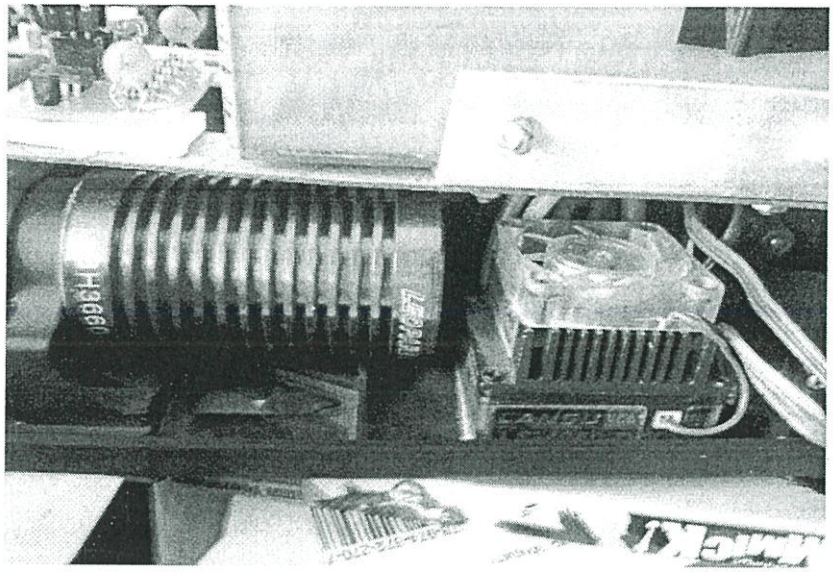
เซอร์โวมอเตอร์ที่อยู่ในตัวรถเป็นมอเตอร์เซอร์โวสเตอร์ริ่ง ซึ่งเป็นต้นกำลังของชุดคันชักเลี้ยว โดยค่าความกว้างของช่วง High ของสัญญาณพัลส์ ที่ระหว่าง 1650-1750 ไมโครวินาที จะอยู่ในช่วงล้อตรงและเป็นค่ากลาง ถ้าค่าความกว้างของช่วง High ของสัญญาณพัลส์ มากกว่าค่ากลางล้อจะโยกไปทางซ้าย และการโยกมากน้อยขึ้นอยู่กับค่าที่แตกต่างจากค่ากลาง ถ้าค่าความกว้างของช่วง High ของสัญญาณพัลส์ น้อยกว่าค่ากลางล้อจะโยกไปทางซ้าย และโยกมากน้อยขึ้นอยู่กับค่าที่แตกต่างจากค่ากลางเช่นกัน ซึ่งเมื่อรู้ถึงวิธีควบคุมจึงสามารถใช้ไมโครคอนโทรลเลอร์ในการสร้างสัญญาณดังกล่าวได้



รูปที่ 3.7 มอเตอร์เซอร์โวและชุดคันชักเลี้ยว

3.4.3 การควบคุมมอเตอร์ไร้แปรงถ่าน

ในตัวรถมีวงจรสำหรับขับเคลื่อนอยู่ โดยวงจรดังกล่าวก็รับสัญญาณพัลส์ แบบเดียวกับมอเตอร์เซอร์โวเช่นกัน การควบคุมมอเตอร์ไร้แปรงถ่านนั้นถ้าค่าความกว้างของช่วง High ของสัญญาณพัลส์ ที่ประมาณ 1500 ไมโครวินาที จะอยู่ในช่วงล้อไม่หมุนและเป็นค่ากลาง ถ้าค่าความกว้างของช่วง High ของสัญญาณพัลส์ มากกว่าค่ากลางรถจะขับเคลื่อนเดินหน้า และความเร็วมากน้อยขึ้นอยู่กับค่าที่แตกต่างจากค่ากลาง ถ้าค่าความกว้างของช่วง High ของสัญญาณพัลส์ น้อยกว่าค่ากลางรถจะขับเคลื่อนเดินหน้าและความเร็วมากน้อยขึ้นอยู่กับค่าที่แตกต่างจากค่ากลางเช่นกัน



รูปที่ 3.8 มอเตอร์ไร้แปรงถ่านและวงจรขับเคลื่อน

3.5 การหามุมเอียงด้วยวิธี Complementary Filter

Complementary Filter เป็นวิธีการคำนวณหามุมโรล มุมพิท และมุมยอร์ โดยใช้ค่าทั้งหมดของ IMU มาใช้ในการคำนวณ โดยวิธีจะทำการเฉลี่ยค่าที่คำนวณได้เป็นมุมโรลจาก Accelerometer และค่าที่คำนวณได้เป็นมุมโรลจาก Gyroscope เพื่อให้ได้ค่าที่เหมาะสมขึ้น ส่วนมุมพิทก็ใช้หลักการเดียวกับมุมโรล มุมยอร์ต้องใช้ค่าจาก Magnetic Field แต่ Magnetic Field ถูกรบกวนจากสัญญาณภายนอกได้ง่าย ซึ่ง IMU เซนเซอร์ต้องถูกประกอบลงตัวรถ สัญญาณรบกวนจากมอเตอร์ไร้แปรงถ่านมีผลทำให้การคำนวณผิดพลาดอย่างมาก จึงต้องการชิลทั้งวงจรด้วยตัวนำ แต่ผลที่ได้ก็ยังคงไม่เปลี่ยนแปลง จึงเลือกที่จะไม่ใช้มุมยอร์ ซึ่งไม่ค่อยมีความจำเป็นต้องใช้มากนัก เนื่องจากได้ค่าที่มีจากคำนวณที่เป็นระบบ และละเอียดอ่อนกว่า จึงเลือกใช้ค่ามุมโรลและมุมพิทจากวิธี Complementary Filter

3.6 การรับค่าสั่งการของผู้ใช้

ระบบทีมีประสิทธิภาพควรจะตอบสนองความต้องการที่เหมาะสมได้ จึงควรรู้ถึงการสั่งการปัจจุบันของผู้ใช้ การรับค่าแบบ Interrupt คือการดักจับขอบขาขึ้นหรือลงของสัญญาณดิจิตอล เมื่อพบจะทำการคำนวณหนึ่งครั้ง จึงใช้การรับค่าแบบดังกล่าวในการดักจับสัญญาณพัลส์ ที่ได้จากตัวรับสัญญาณวิทยุ โดยดักจับขอบขาขึ้นและดักจับขอบขาลง นำเวลาที่ต่างกันนั้นคือค่าความกว้างของช่วง High ของสัญญาณพัลส์ ซึ่งจะสามารถทำให้รู้ได้ถึงความต้องการของผู้ใช้ได้

3.7 การหาความเร็ว

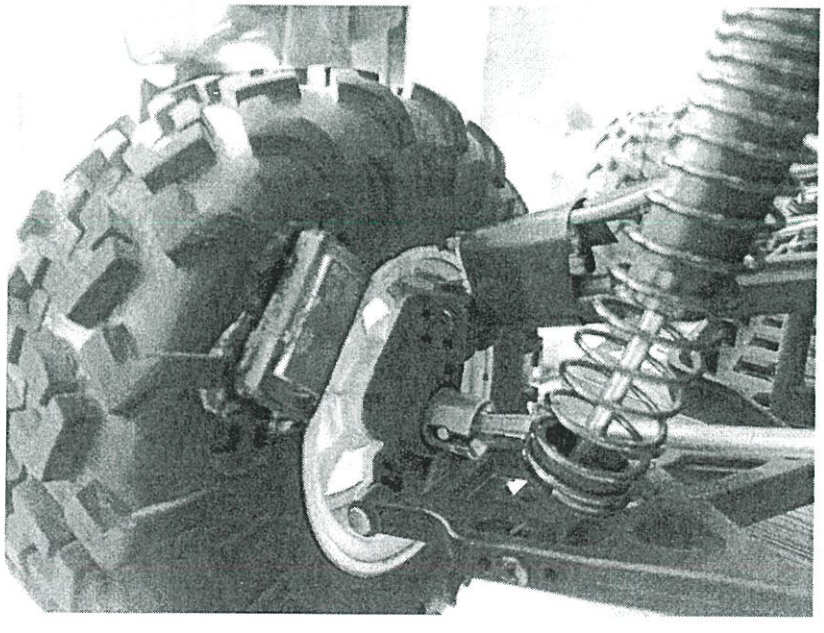
ความเร็วเป็นค่าที่มีผลกับการเคลื่อนที่ และมีผลกับการเกิดปัญหาการเคลื่อนที่

3.7.1 การหาความเร็วด้วย Accelerometer แกน Y

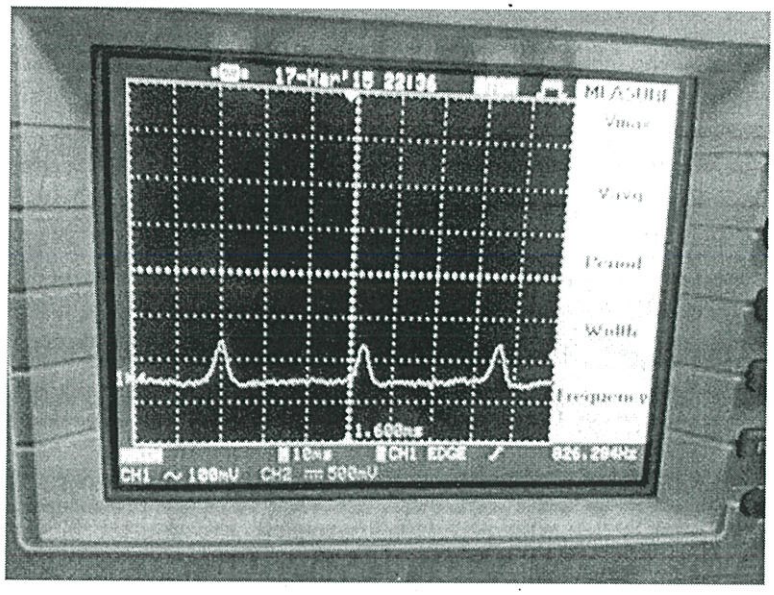
สามารถหาด้วยการอินติเกรตความเร่งเชิงเส้นแกน Y กับเวลา แต่วิธีนี้มีข้อเสียที่ชัดเจนคือมีค่าผิดพลาดสะสมตลอดการคำนวณ และเมื่อรถเกิดการเอียงจะคำนวณผิดพลาดเช่นกัน แม้ว่าจะใช้มุมคำนวณเพื่อแก้ค่าแล้ว ผลก็ยังค่อยไม่ดี

3.7.2 การหาความเร็วด้วย Linear Hall Sensor

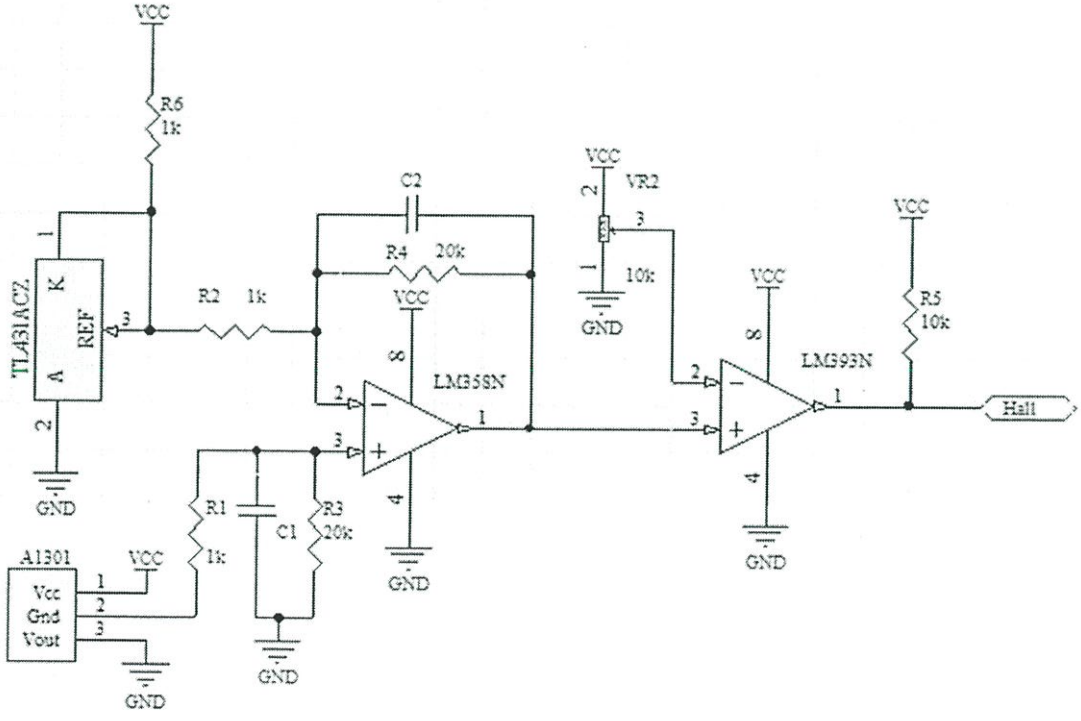
การหาความเร็ววิธีนี้ใช้หลักการเดียวกับ Encoder ซึ่งจะต้องเพิ่มเซนเซอร์ Hall Effect ลงไปบริเวณตัวรถ และเพิ่มแม่เหล็กไว้ที่ล้อ โดยที่ตัวเซนเซอร์กับแม่เหล็กต้องมาใกล้กันทุกๆ การหมุนของล้อหนึ่งรอบ ซึ่งเมื่อ Hall Effect พบแม่เหล็ก แรงดันที่ขาสัญญาณจะสูงขึ้น ซึ่งสัญญาณเปลี่ยนไปมากจึงต้องใช้วงจรขยายสัญญาณ และใช้วงจรเปรียบเทียบสัญญาณ ทำให้สัญญาณกลายเป็นสัญญาณ Digital ใช้การรับค่าแบบ Interude ดักจับขอบขาขึ้น และเก็บค่าเวลาไว้ เมื่อการ Interude ทำงานอีกครั้งนำความต่างของเวลาปัจจุบันกับค่าเวลาที่เก็บไว้มาคำนวณ จะได้ค่าความเร็วในระดับที่มีความแม่นยำในระดับนึง



รูปที่ 3.9 แม่เหล็กและ Linear Hall Sensor



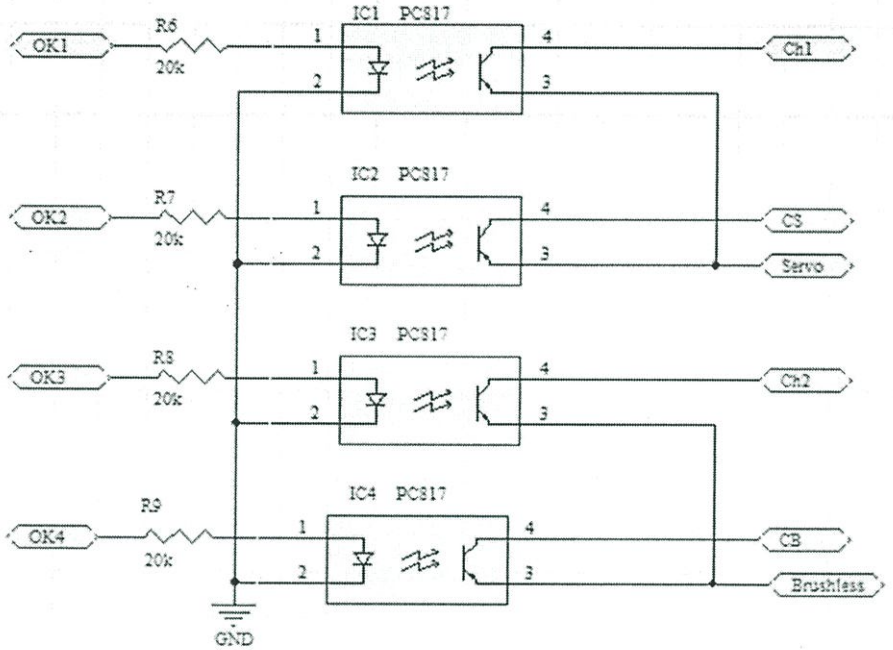
รูปที่ 3.10 สัญญาณอนาลอกจาก Linear Hall Sensor



รูปที่ 3.11 วงจรไฟฟ้าของ Linear Hall Sensor

3.8 การสลับสัญญาณส่งออก

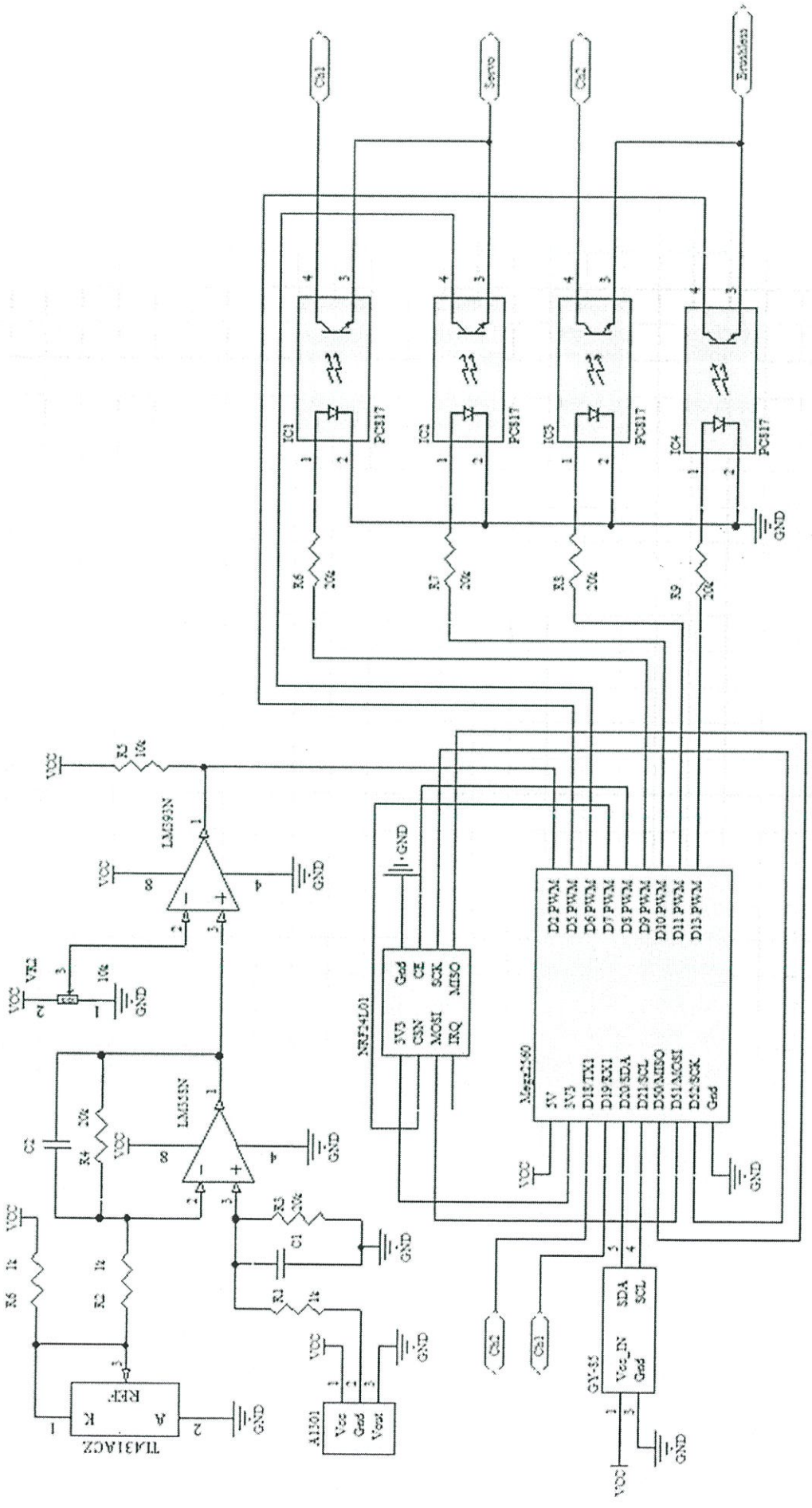
โดยถ้ารถอยู่ในสภาวะปกติระบบจะให้ผู้ใช้ควบคุมโดยตรงโดยไม่ผ่านการควบคุม แต่ถ้าอยู่ในสภาวะไม่ปกติระบบจะตัดการควบคุมที่ไม่เหมาะสมของผู้ใช้ และเข้าไปควบคุมแทน จึงได้ออกแบบวงจรที่สามารถสลับสัญญาณออก โดยใช้ Optocoupler และควบคุมสัญญาณสำหรับสลับโดยใช้ไมโครคอนโทรลเลอร์หรือสวิทช์ก็ได้ สุดท้ายการใช้วงจรนี้ทำให้สามารถเขียนโปรแกรมได้ง่ายขึ้น



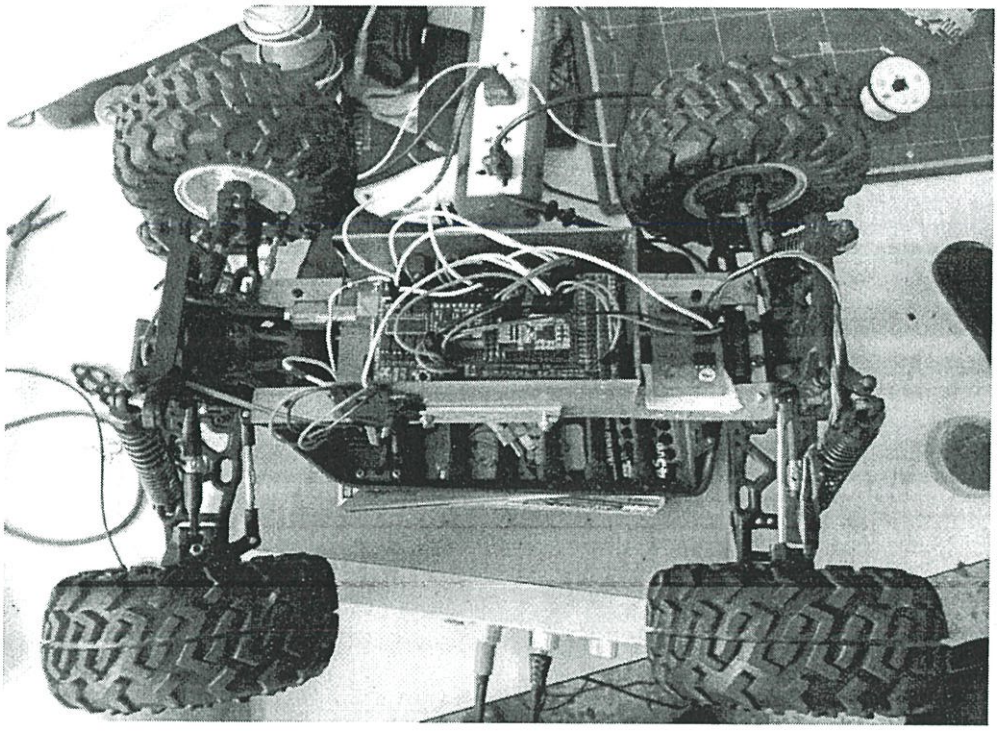
รูปที่ 3.12 วงจรไฟฟ้าของวงจรสลับสัญญาณส่งออก

3.9 การประกอบวงจรต่างๆ ไมโครคอนโทรลเลอร์ และรถบังคับวิทยุเข้าด้วยกัน

เมื่อรวบรวมวงจรอุปกรณ์ต่างๆ และไมโครคอนโทรลเลอร์เรียบร้อย จากนั้นทำการเดินสายเชื่อมต่อวงจรต่างๆ สร้างโครงโลหะสำหรับยึดไมโครคอนโทรลเลอร์ไว้กับตัวรถ สร้างหัวปลั๊กสำหรับจ่ายไฟฟ้าให้แก่ไมโครคอนโทรลเลอร์ ซ่อมบำรุงตัวรถ



รูปที่ 3.13 วงจรรวมของรถ



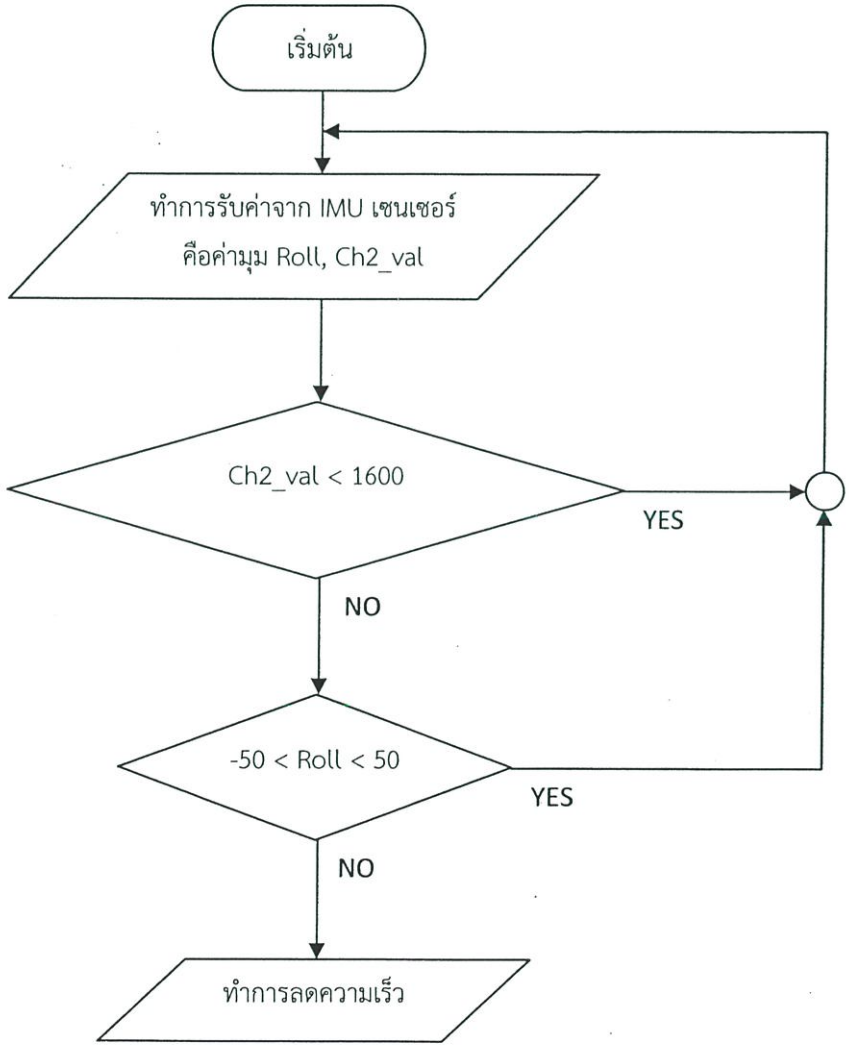
รูปที่ 3.14 รถบังคับวิทยุซึ่งประกอบอุปกรณ์และวงจรเข้าด้วยกัน

3.10 การนำค่าต่างๆ มาวิเคราะห์และสร้างสรรค์รูปแบบการควบคุม

นำโปรแกรมในแต่ละภาคส่วนมาทำงานร่วมกัน ซึ่งในขั้นตอนนี้เป็นขั้นตอนที่ต้องใช้สิ่งที่ทำมาทั้งหมดนำมาประสานกัน โดยค่าที่สามารถนำไปใช้ได้ ประกอบไปด้วยค่าความเร่งเชิงเส้น 3 แกน ความเร็วเชิงมุม 3 แกน มุม Roll มุม Pitch ค่าความกว้างของช่วง High ของสัญญาณพัลส์ ของตัวรับสัญญาณวิทยุของ Channel 1 และ Channel 2 โดยความเร่งเชิงเส้นบอกถึงทิศทางความเร่งปัจจุบันและความเร่งสู่ศูนย์ที่เกิดจากการเลี้ยว ความเร็วเชิงมุมบอกถึงการหมุนตัวของรถ ค่าความกว้างของช่วง High ของสัญญาณพัลส์ ของตัวรับสัญญาณวิทยุของ Channel 1 และ Channel 2 บอกถึงความต้องการ ณ สภาวะปัจจุบันว่าต้องการกระทำอย่างไร ความเร็วเชิงเส้นแกน Y บอกถึงความเร็วของรถ ณ ทิศชี้ไปหน้ารถ มุม Roll มุม Pitch บอกถึงการเอียงปัจจุบันของตัวรถ โดยเมื่อนำค่าเหล่านี้มาวางเงื่อนไขผ่านการเขียนโปรแกรมในการควบคุมระบบ และเมื่อถึงขอบเขตที่กำหนดไว้ก็จะทำการตัดสัญญาณการสั่งการของผู้ควบคุมออกไป และส่งสัญญาณพัลส์ ที่เหมาะสมไปแทนที่ เพื่อแก้ไขสภาวะให้อยู่ในขอบเขตที่กำหนดไว้ โดยระบบจะตรวจสอบค่าต่างๆ อยู่ตลอดเวลาเมื่อเปิดการใช้งาน

3.11 การเขียนโปรแกรมในการควบคุม

โปรแกรมที่ใช้ในการออกแบบระบบควบคุมนี้คือ โปรแกรม Arduino โดยใช้พื้นฐานของภาษาซีในการเขียนโปรแกรม จากที่ได้ปรับแต่งค่าตัวแปรต่างๆ ที่ได้จาก IMU เซนเซอร์และค่าสัญญาณพัลส์ จะนำค่าเหล่านั้นมาใช้ในการเขียนโปรแกรม ซึ่งค่าที่นำมาเป็นเงื่อนไขคือค่ามุม Roll ซึ่งวางเงื่อนไขไว้ให้อยู่ในช่วง (-50 ,50) องศา และอีกเงื่อนไขคือค่าของขนาดความกว้างของสัญญาณพัลส์จากการบังคับรีโมท Channal 2 (Ch2_val) ซึ่งวางเงื่อนไขไว้ให้น้อยกว่า 1600 ไมโครวินาที ค่าที่นำมาวางเป็นเงื่อนไขนี้เป็นค่าที่ผู้ทำการทดลองนำมาปรับแบบลองผิดลองถูก ซึ่งโปรแกรมเป็นไปตามโฟล์วชาร์ตดังนี้



รูปที่ 3.15 โฟล์วชาร์ตของโปรแกรมในการควบคุมระบบการเสียหลักของรถ

3.12 การทดลองผลและปรับปรุงให้ระบบดีขึ้น

หลังจากที่ได้ทำการติดตั้งระบบควบคุมการเสียหลักของรถ จึงได้ทดลองระบบด้วยการให้รถบังคับวิทยุวิ่งทดสอบเพื่อดูผลตอบสนองของระบบ ผลปรากฏว่าระบบสามารถทำให้รถเข้าโค้งได้แคบลงในระดับหนึ่ง และเนื่องจากตัวผู้ขับเองอาจมีผลต่อการควบคุมรถ ทำให้อาจจะเกิดปัญหาทางการเคลื่อนที่ได้ โดยระบบสามารถช่วยลดผลได้ในระดับนึง เมื่อทำการทดลองผลทำให้สามารถเห็นแนวทางในการปรับปรุงให้ดีขึ้นได้

บทที่ 4
การทดลอง

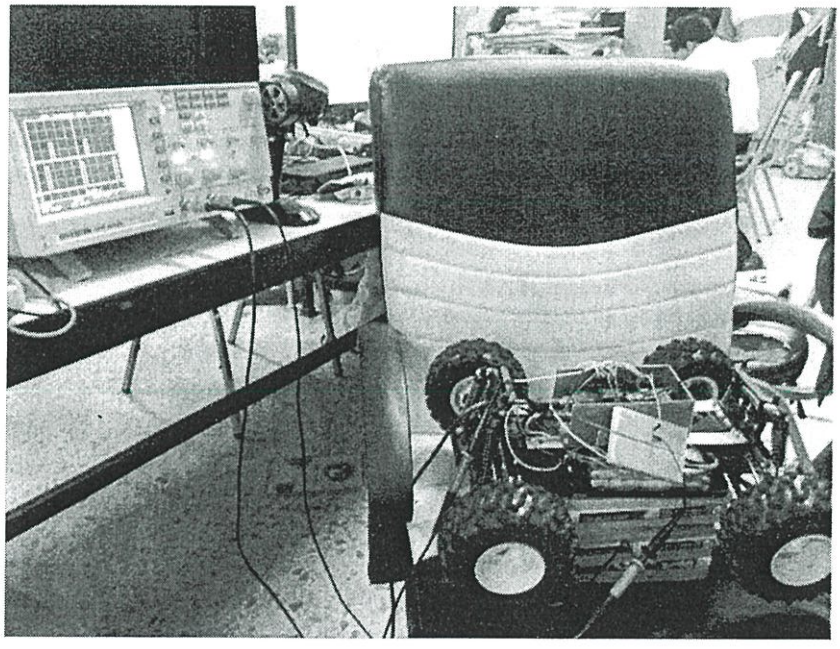
ในบทนี้จะกล่าวถึงการทดลองที่เกี่ยวกับค่าตัวแปร ที่วัดได้จากเซนเซอร์และตัวแปรต่างๆ ที่มีส่วนเกี่ยวข้องในการออกแบบระบบป้องกันการเสียหลักของรถ

4.1 การทดลองภาคขับของรถบังคับวิทยุ

ในการทดลองนี้ได้ศึกษาเกี่ยวกับการส่งสัญญาณจากรีโมทไปยังตัวรับสัญญาณวิทยุ เพื่อไปควบคุมการทำงานของรถบังคับวิทยุ บันทึกผลดังตารางที่ 4.1

4.1.1 การทดลองภาคขับเซอร์โวมอเตอร์

การทดลองนี้เป็นการเก็บค่าจากการขับเซอร์โวมอเตอร์ด้วยคำสั่งจากโปรแกรม Arduino โดยทำการสั่งให้เซอร์โวมอเตอร์เคลื่อนที่เพิ่มขึ้นทีละ 10 องศา จากนั้นนำออกซิโลสโคปไปวัดค่าความกว้างของสัญญาณพัลส์ และสังเกตลักษณะการหักเลี้ยวของล้อรถ



รูปที่ 4.1 การทดลองวัดค่าความกว้างของสัญญาณพัลส์ด้วยการขับเซอร์โวมอเตอร์

ตารางที่ 4.1 ผลการทดลองวัดค่าที่เกิดจากการขับเซอร์โวมอเตอร์

เซอร์โวมอเตอร์ (องศา)	สัญญาณ PWM (ไมโครวินาที)	ลักษณะการเลี้ยวของล้อ
40	1000	ขวาหักสุด
50	1100	ขวา
60	1200	ขวา
70	1300	ขวา
80	1400	ขวา
90	1500	ขวา
100	1600	ขวา
110	1700	ล้อตั้งตรง
120	1800	ซ้าย
130	1900	ซ้าย
140	2000	ซ้าย
150	2100	ซ้าย
160	2200	ซ้าย
170	2300	ซ้าย
180	2400	ซ้ายหักสุด

จากผลการทดลองเมื่อองศาของเซอร์โวมอเตอร์เพิ่มขึ้น ขนาดความกว้างของสัญญาณพัลส์ก็จะเพิ่มขึ้นตามเป็นลำดับ และเมื่อองศาของเซอร์โวมอเตอร์อยู่ที่ 110 องศา เทียบเท่ากับความกว้างของสัญญาณพัลส์ 1700 มิลลิวินาที ลักษณะของล้อจะตั้งตรง ซึ่งจากผลการทดลองนี้ทำให้ทราบขนาดความกว้างของสัญญาณพัลส์ที่สั่งเข้าไป ไม่ว่าจะสั่งผ่านโปรแกรม Arduino หรือรีโมท

4.1.2 การทดลองภาคขับมอเตอร์ไร้แปรงถ่าน

การทดลองนี้จะทำการทดลองคล้ายกับการทดลองที่ 4.1.1 เพียงแต่เปลี่ยนเป็นการขับมอเตอร์ไร้แปรงถ่าน ซึ่งมอเตอร์ไร้แปรงถ่านเป็นต้นกำลังที่ทำให้รถบังคับวิทยุเคลื่อนที่ โดยทำการทดลองด้วยคำสั่งจากโปรแกรม Arduino ทำการสั่งมอเตอร์ไร้แปรงถ่านด้วยสัญญาณพัลส์ ซึ่งจะทำการเพิ่มความกว้างของสัญญาณพัลส์ทีละ 100 มิลลิวินาที จากนั้นสังเกตค่าความเร็วที่เกิดขึ้นโดยให้ Linear Hall Sensor เป็นตัววัดความเร็วเปรียบเทียบ ดังแสดงในตารางที่ 4.2

ตารางที่ 4.2 ผลการทดลองวัดค่าที่เกิดจากการขับมอเตอร์ไร้แปรงถ่าน

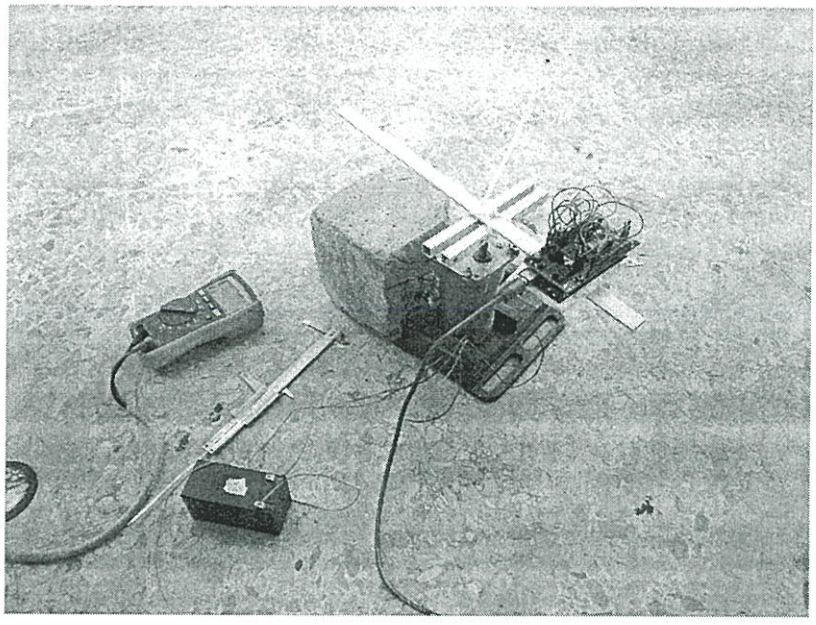
สัญญาณ PWM (ไมโครวินาที)	ความเร็ว (เมตร/วินาที)
1500	0
1600	6.38
1700	7.72
1800	8.46
1900	10.14
2000	11.17
2100	11.85

จากผลการทดลอง เมื่อความกว้างของสัญญาณพัลส์เพิ่มขึ้น ความเร็วที่วัดได้ก็มีค่าเพิ่มขึ้นเช่นกัน ทำให้เห็นถึงประโยชน์ในการนำค่าความกว้างของสัญญาณพัลส์ ที่สามารถจะนำมาสร้างเงื่อนไขในการเขียนโปรแกรมเพื่อทำการตรวจวัดความเร็วของตัวรถบังคับวิทยุ เพื่อป้องกันการเสียหลัก

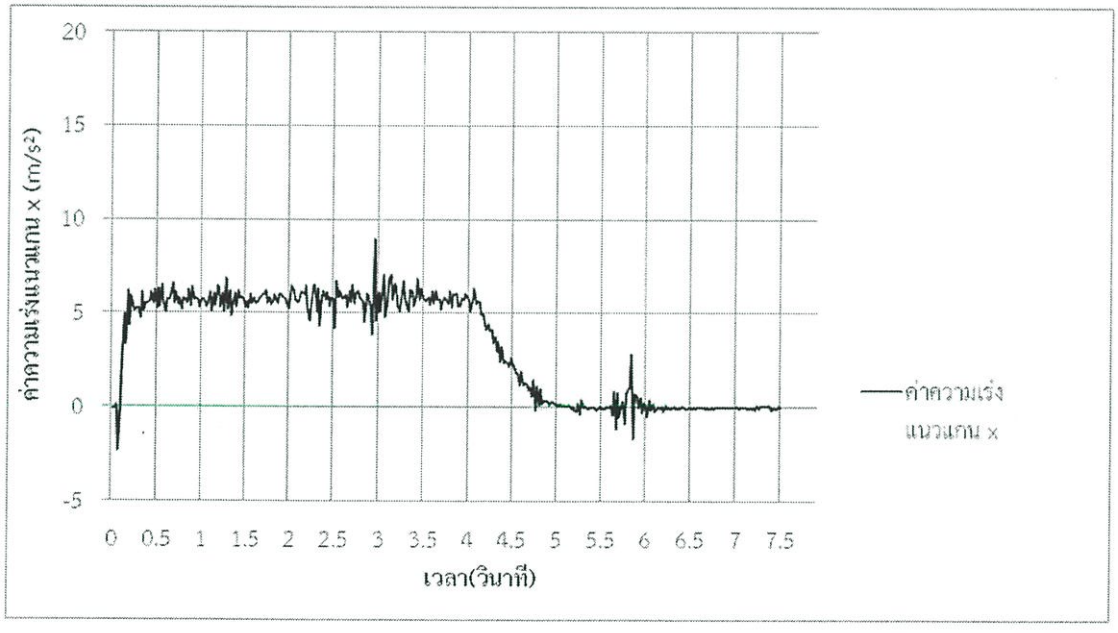
สัญญาณพัลส์ที่วัดได้ จะทำการปรับแต่งเมื่อนำไปใช้จริง โดยทำการปรับแต่งโดยใช้โปรแกรม Arduino สร้างเงื่อนไขกำหนดย่านขนาดความกว้างของสัญญาณพัลส์ เพื่อที่จะได้สัญญาณที่มีช่วงความกว้างคงที่

4.2 การทดลองการเปลี่ยนแปลงค่าความเร่งในแนวแกน X เมื่อรถเข้าโค้ง

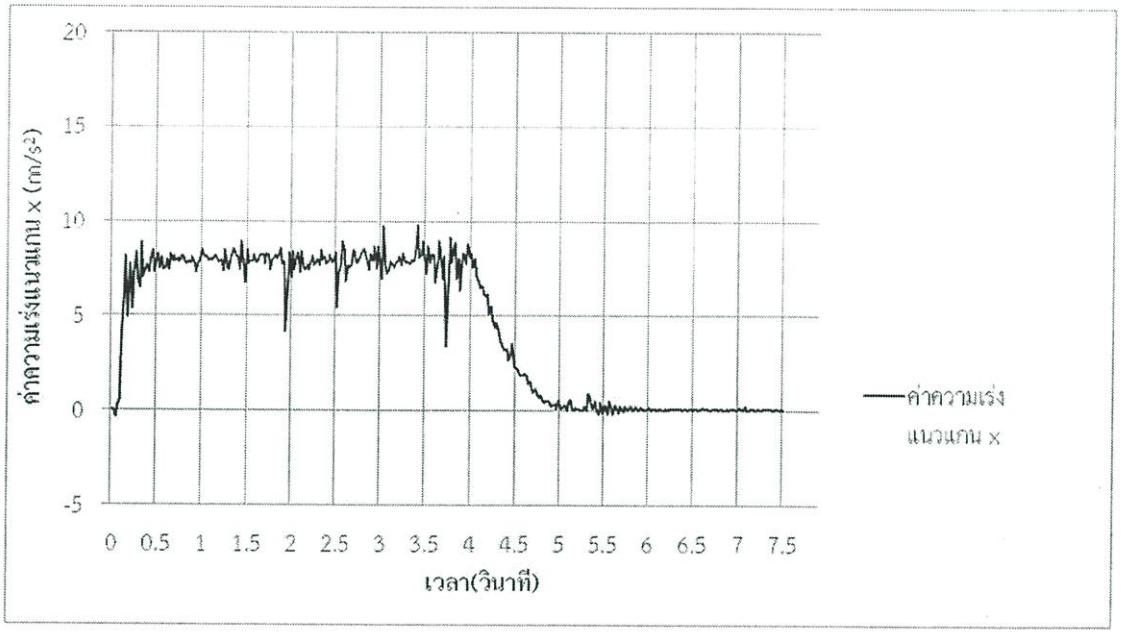
ในการทดลองนี้ทำการศึกษาเกี่ยวกับการเปลี่ยนแปลงค่าความเร่งในแนวแกน X ในขณะที่รถเข้าโค้ง ซึ่งในการทดลองนี้ได้ทำการจำลองการเคลื่อนที่แบบวงกลมเพื่อวัดค่าความเร่งในแนวแกน X โดยจะทำการเพิ่มรัศมีการติดตั้งชุดเซนเซอร์ให้ห่างออกไปทีละ 40 มิลลิเมตร ทำการหมุนโดยใช้มอเตอร์ไฟฟ้ากระแสตรงเพื่อจะได้ควบคุมตัวแปรความเร็วเชิงมุมให้คงที่ จากนั้นสังเกตค่าและบันทึกผล ดังรูปที่ 4.2



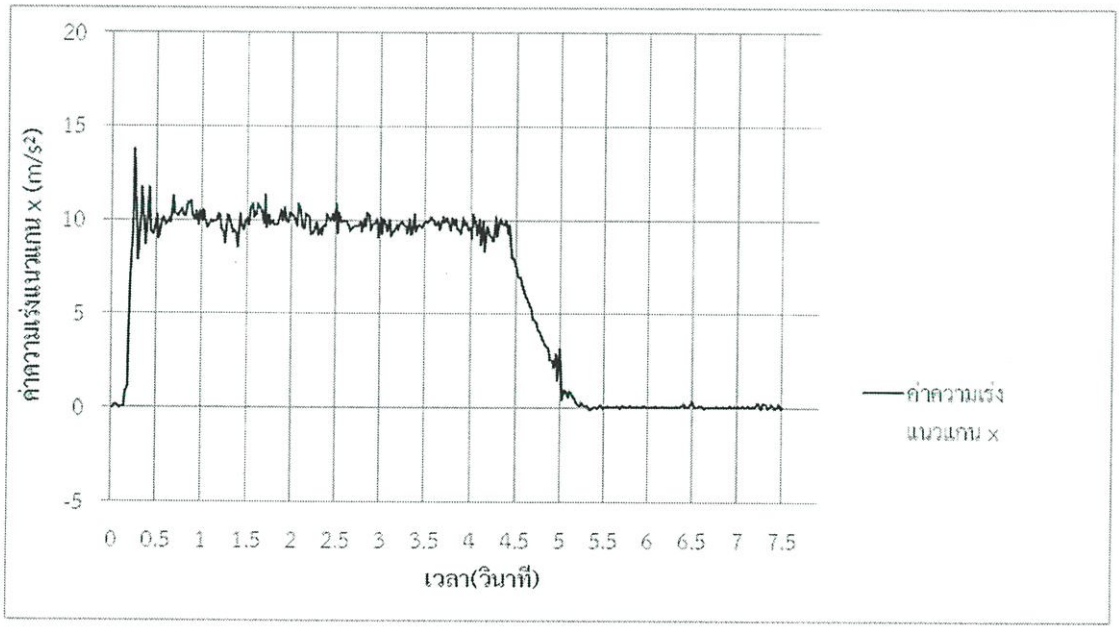
รูปที่ 4.2 การทดลองวัดค่าความเร่งในแนวแกน X โดยจำลองการเคลื่อนที่แบบวงกลม



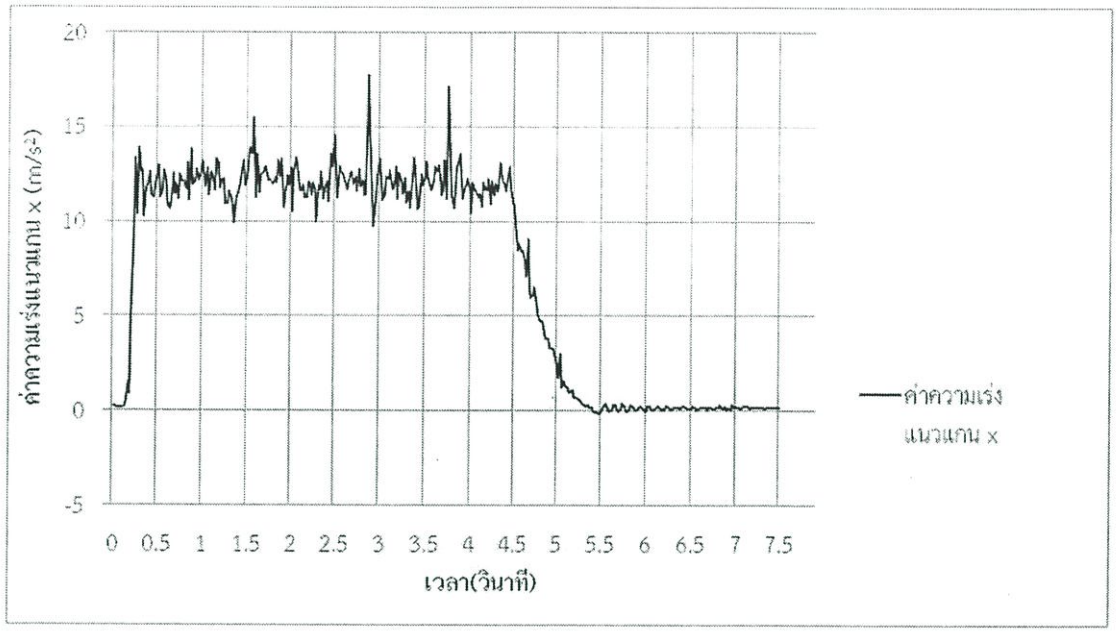
รูปที่ 4.3 ผลการทดลองเมื่อติดตั้งชุดเซนเซอร์ห่างจากจุดศูนย์กลางของการหมุน 110 มิลลิเมตร



รูปที่ 4.4 ผลการทดลองเมื่อติดตั้งชุดเซนเซอร์ห่างจากจุดศูนย์กลางของการหมุน 150 มิลลิเมตร



รูปที่ 4.5 ผลการทดลองเมื่อติดตั้งชุดเซนเซอร์ห่างจากจุดศูนย์กลางของการหมุน 190 มิลลิเมตร



รูปที่ 4.6 ผลการทดลองเมื่อติดตั้งชุดเซนเซอร์ห่างจากจุดศูนย์กลางของการหมุน 240 มิลลิเมตร

จากการทดลองจะสังเกตเห็นได้ว่าเมื่อเพิ่มรัศมีการติดตั้งชุดเซนเซอร์ ค่าความเร่งแนวแกน X จะมีค่าเพิ่มขึ้น ซึ่งในที่นี้ค่าความเร่งแนวแกน X คือค่าความเร่งเข้าสู่ศูนย์กลาง โดยกำหนดให้ความเร็วเชิงมุมคงที่ เมื่อรถเกิดการเข้าโค้ง รถจะมีลักษณะการเคลื่อนที่เป็นวงกลม ซึ่งการเข้าโค้งก็จะขึ้นกับรัศมีการเข้าโค้งมากหรือน้อย ซึ่งถ้ารัศมีในการเข้าโค้งมาก ก็จะยิ่งทำให้ค่าความเร่งแนวแกน X เปลี่ยนแปลงมากขึ้นด้วย ดังสมการที่ 4.1 ดังนั้นค่าความเร่งแนวแกน X จึงสามารถที่จะเป็นตัวบอกสภาวะที่รถเข้าโค้งได้ ซึ่งทำให้สามารถนำค่าความเร่งแนวแกน X ไปกำหนดเงื่อนไขในการเข้าโค้งของระบบป้องกันการเสียหลักของรถ ซึ่งจะทำงานควบคู่กับการตรวจสอบเงื่อนไขความกว้างของสัญญาณพัลส์

$$a_c = \omega^2 R \tag{4.1}$$

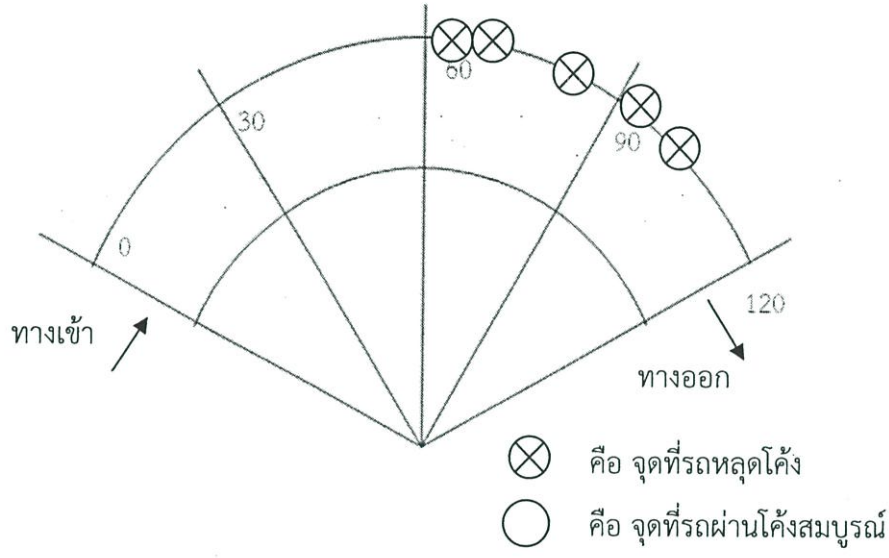
a_c คือ ความเร่งเข้าสู่ศูนย์กลาง

ω คือ ความเร็วเชิงมุม

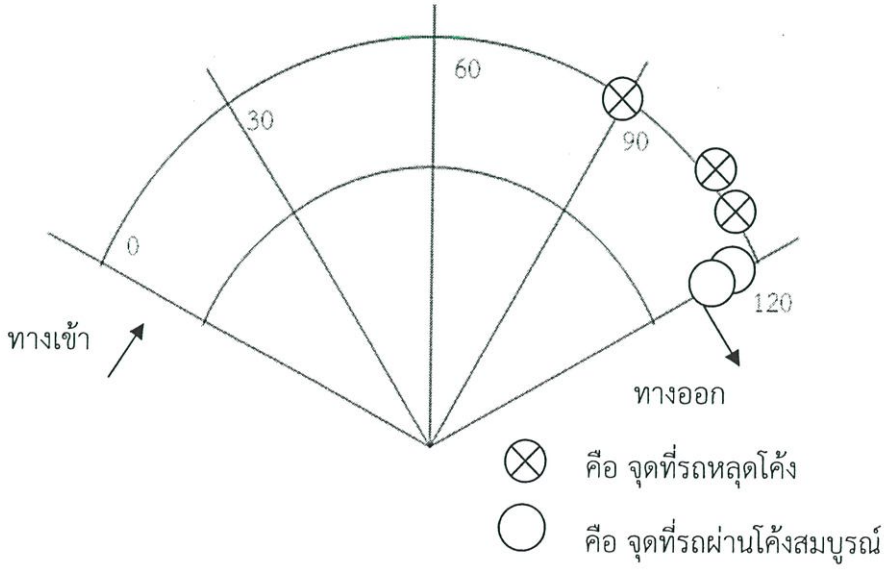
R คือ รัศมีวงกลมหรือรัศมีโค้งถนน

4.3 การทดลองประสิทธิภาพในการเข้าโค้งของรถ

ในการทดลองส่วนนี้เป็นการนำค่าตัวแปรที่วัดได้จากเซนเซอร์ทั้งหมด รวมทั้งการเขียนโปรแกรม Arduino ด้วยภาษาซีมาทำการประยุกต์ใช้งานในการสร้างระบบป้องกันการเสียหลักของรถบังคับวิทยุ ซึ่งการทดลองนี้จะทำการจำลองโค้งถนน 120 องศา พื้นเรียบ ที่มีรัศมีภายใน 1.3 เมตร และรัศมีภายนอก 3 เมตร โดยทำการเปรียบเทียบระหว่างรถบังคับวิทยุที่ไม่มีระบบป้องกันการเสียหลักกับรถบังคับวิทยุที่มีระบบป้องกันการเสียหลัก ในที่นี้จะมีผู้บังคับเพียงคนเดียวเท่านั้นซึ่งจะทำการเข้าโค้งกรณีละ 5 ครั้ง โดยเร่งความเร็วสูงสุดในการเข้าโค้ง



รูปที่ 4.7 ผลการทดลองการเข้าโค้งแบบไม่มีระบบป้องกันการเสียหลัก



รูปที่ 4.8 ผลการทดลองการเข้าโค้งแบบมีระบบป้องกันการเสียหลัก

จากผลการทดลองจะสังเกตเห็นได้ว่าเมื่อรถไม่มีระบบป้องกันการเสียหลัก ในการทดลองทั้ง 5 ครั้ง รถบังคับวิทยุไม่สามารถที่จะวิ่งผ่านโค้งได้อย่างสมบูรณ์ และจะหลุดโค้งที่ประมาณมุม 60 - 110 องศา ส่วนในการทดลองรถที่มีระบบป้องกันการเสียหลัก จะเห็นได้ว่า ในการทดลองทั้ง 5 ครั้ง รถบังคับวิทยุสามารถที่จะวิ่งผ่านโค้งได้อย่างสมบูรณ์ 2 ใน 5 ของการเข้าโค้ง และจะหลุดโค้งที่ประมาณมุม 90 - 115 องศา ซึ่งแสดงให้เห็นว่าระบบนี้สามารถเพิ่มองศาของการเลี้ยวโค้งได้ดีขึ้น ในขณะที่รถเข้าโค้งด้วยความเร็ว และระบบนี้น่าจะเหมาะสมกับการเข้าโค้งที่มีรัศมีการเข้าโค้งมากกว่านี้ ทั้งนี้ยังมีตัวแปรควบคุมที่อาจไม่คงที่ในการทดลองด้วย อย่างเช่น ความเร็วในการเข้าโค้งของผู้บังคับ ลักษณะของการหักเลี้ยวแต่ละจังหวะ

บทที่ 5
บทวิจารณ์และสรุป

5.1 สรุปผลการทดลอง

จากการทดลองระบบป้องกันการเสียหลักของรถ ในที่นี้จะทำการป้องกันการเสียหลักในกรณีที่รถเกิดการเข้าโค้งด้วยความเร็วสูง ซึ่งโดยปกติทั่วไปคงไม่มีรถคันไหนที่จะสามารถเข้าโค้งด้วยความเร็วสูงแล้วออกจากโค้งได้อย่างสมบูรณ์แบบ ยกเว้นจะเป็นรถที่ไว้ใช้สำหรับแข่งขันจริงๆ เป้าหมายคือ ให้รถบังคับวิทยุสามารถเคลื่อนที่ผ่านโค้งได้อย่างสมบูรณ์แบบ และจากผลการทดลองผลปรากฏว่า เมื่อได้ทำการใช้ระบบป้องกันการเสียหลักแล้ว รถบังคับวิทยุก็ยังไม่สามารถวิ่งเข้าโค้งได้อย่างสมบูรณ์อย่างที่ได้ตั้งเป้าหมายไว้ แต่จะสังเกตเห็นได้ว่าเมื่อใช้ระบบนี้ ตัวรถบังคับวิทยุจะมีวิถีการเลี้ยวโค้งที่ดีขึ้น ซึ่งการทำงานของระบบจะไปช่วยลดความเร็วในขณะที่เกิดการเข้าโค้ง ทำให้ผู้บังคับสามารถที่จะหักเลี้ยวโค้งได้ดีขึ้น ในการลดความเร็วรถได้กล่าววิธีการทำงานดังบทที่ 3 วิธีดำเนินงาน

5.2 ปัญหาที่พบและแนวทางแก้ไข

จากการทำโครงงานเรื่องนี้ ปัญหาที่พบในช่วงแรกคือ การเกิดสัญญาณรบกวนต่อชุดเซนเซอร์ ซึ่งทำให้ค่าเซนเซอร์คลาดเคลื่อน จึงแก้ปัญหาโดยการใช้ตัวกรองสัญญาณแบบ Complementary เพื่อให้ได้ค่าจากเซนเซอร์มีความคลาดเคลื่อนน้อยที่สุด ปัญหาที่สองคือ การควบคุมรถบังคับวิททยุ เนื่องจากแต่ละคนไม่มีพื้นฐานในการบังคับมาก่อน ทำให้ก่อให้เกิดอุบัติเหตุจากการชน กระแทก และต้องทำการซ่อมแก้ไข เปลี่ยนอะไหล่อยู่บ่อยครั้ง ซึ่งทำให้เกิดความล่าช้าในการทำงาน แก้ปัญหาโดยการฝึกการบังคับให้คุ้นมืออยู่บ่อยๆ เพื่อที่จะได้ลดอุบัติเหตุ ปัญหาที่สามคือเรื่องแบตเตอรี่ในการขับเคลื่อนรถบังคับวิทยุ เนื่องจากรถบังคับวิทยุนี้ค่อนข้างมีความเร็วพอสมควรทำให้ใช้พลังงานในการขับเคลื่อนค่อนข้างมาก และแบตเตอรี่สามารถใช้งานได้ประมาณ 30 นาทีเท่านั้น แต่ต้องทำการชาร์จแบตเตอรี่เป็นเวลาถึง 3 ชั่วโมง ทำให้เกิดความล่าช้าในการทำงาน จึงต้องแก้ปัญหาด้วยการมีแบตเตอรี่ 2 ก้อน เพื่อสลับการใช้งานและสามารถทำงานได้อย่างต่อเนื่อง ปัญหาที่สี่คือ ในการติดตั้งชุดเซนเซอร์ทั้งหมดบนรถบังคับวิทยุ ในขณะที่รถเคลื่อนที่อาจมีการสั่นสะเทือนทำให้เซนเซอร์ต่างๆ ได้รับผลกระทบไปด้วย และอาจทำให้การเชื่อมต่อสายสัญญาณต่างมีปัญหา เช่น เกิดการหลวม เป็นต้น ทำการแก้ปัญหาโดยการยึดชุดเซนเซอร์ทั้งหมดให้แน่นขึ้น และต้องมีการตรวจสอบการขันยึดอยู่บ่อยๆ และปัญหาสุดท้ายคือ รถบังคับวิทยุคันนี้ไม่สามารถที่จะพลิกคว่ำได้ ซึ่งทีแรกได้ตั้งเป้าหมายด้วยว่าจะทำการแก้ไขปัญหาการพลิกคว่ำ เนื่องด้วยตัวรถบังคับวิทยุนี้มีระบบโช๊คอัพค่อนข้างดีทำให้มีการยืดหยุ่นในขณะทำการเข้าโค้ง ดังนั้นจึงเน้นไปยังการแก้ปัญหาการเข้าโค้งด้วยความเร็วสูง

5.3 ข้อเสนอแนะและแนวทางในการค้นคว้าพัฒนา

ในการออกแบบระบบป้องกันการเสียหลักของรถ สามารถทำได้หลายวิธี ซึ่งการควบคุมโดยใช้ชุดโมดูลเซนเซอร์ IMU นี้ก็เป็นอีกหนึ่งวิธี และในทางการเขียนโปรแกรม Arduino ก็ยังสามารถประยุกต์ใช้งานจากค่าที่วัดได้จากเซนเซอร์ต่างๆ มาใช้ในการออกแบบโปรแกรมเพื่อควบคุมระบบการทำงานให้ดีขึ้น ซึ่งในการออกแบบระบบควบคุมครั้งนี้ ผลลัพธ์ที่ได้อาจจะไม่บรรลุเป้าหมายที่แท้จริงนัก ซึ่งแนวทางในการการพัฒนาเพื่อให้ได้ผลลัพธ์ที่ชัดเจนอาจจะต้องทำการเพิ่มรัศมีความโค้งในการทดลอง เพื่อให้เห็นว่ารัศมีขนาดใดที่รถบังคับวิทยุจะสามารถวิ่งเข้าโค้งได้อย่างสมบูรณ์แบบ รวมทั้งต้องวิเคราะห์อุปกรณ์ทั้งฮาร์ดแวและซอฟร์แวในการออกแบบด้วย เพื่อให้ได้ผลลัพธ์ที่ดีขึ้น

เอกสารอ้างอิง

[1] วรพงศ์ ตั้งศรีรัตน์. **เซนเซอร์และทรานสดิวเซอร์ ทฤษฎีและการประยุกต์ใช้ในระบบการวัดและระบบควบคุม**. พิมพ์ครั้งที่ 9. กรุงเทพมหานคร : สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น). 2548.

[2] อรพิน ประวัติบริสุทธ์. **คู่มือเรียนภาษาซี ฉบับปรับปรุงใหม่**. พิมพ์ครั้งที่ 10. กรุงเทพมหานคร : โปรวิชั่น. 2554.

[3] **การเคลื่อนที่บนทางโค้ง**. [ออนไลน์]. สืบค้นจาก : http://www.scimath.org/socialnetwork/groups/viewbulletin/679-การเคลื่อนที่บนทางโค้ง?groupid=162. 2554.

[4] **Gyroscope**. [ออนไลน์]. สืบค้นจาก : http://touch.exteen.com/blog/zygomatica/20100811/gyro-sensor-gyroscope

[5] **Accelerometer**. [ออนไลน์]. สืบค้นจาก : https://sites.google.com/site/thaimulticopter/acc

[6] **Accelerometer**. [ออนไลน์]. สืบค้นจาก : http://www.hobbytronics.co.uk/accelerometer-info

[7] **Accelerometer**. [ออนไลน์]. สืบค้นจาก : http://www.hobbytronics.co.uk/accelerometer-info

[8] **หามุม Pitch และ Roll จากเซ็นเซอร์วัดความเร่ง**. [ออนไลน์]. สืบค้นจาก : http://www.arduitronics.com/article/inertial-sensors-หามุม-pitch-และ-roll-จากเซ็นเซอร์วัดความเร่ง

[9] **การเชื่อมต่ออุปกรณ์แบบ I2C**. [ออนไลน์]. สืบค้นจาก : http://www.thaimicrotron.com/CCS-628/Referrence/I2CBUS.htm

ภาคผนวก

Code ในการเขียนโปรแกรมควบคุมระบบป้องกันการเสียหลัก

```
#include <Wire.h>
  int error = 0;

//Accelerometer
#include <ADXL345.h>
  ADXL345 accel;
  AccelerometerRaw Araw;
  int xAxisRawData;
  AccelerometerScaled Ascaled;
  float Ax;
  float Ay;
  float Az;
  float Av;
  float Ay1;
  float Ay_N;
  float Ay_B;
  float Ay_avg;
  int count;

//Velocity & Radius
  float Vmax;
  float Vt;
  float V;
  float V2;
  float R;
  float L;
  float D;
  int Si;
  int So;
```

```
//Gyro
#include <ITG3205.h>
  ITG3205 itg3205;
  float Gx;
  float Gy;
  float Gz;

//Compass
#include <HMC5883L.h>
  HMC5883L compass;
  MagnetometerRaw Mraw;
  MagnetometerScaled Mscaled;
  int MilliGauss_OnThe_XAxis;
  float heading;
  float declinationAngle;
  float headingDegrees;

//Angle
//Roll,Pitch > Accelerometer
  float roll_R;
  float pitch_R;

  float roll_A;
  float pitch_A;

//Roll,Pitch > Gyro
  float roll_G;
  float pitch_G;
  float yaw_G;

//Roll,Pitch > Complementary
  float roll_C;
  float pitch_C;
```

```
  float yaw_C;

//Roll,Pitch > Scale
  float roll_S;
  float pitch_S;

//Roll; > Ref
  float roll_ref;
  float roll_del;

//NRF
#include <SPI.h>
#include <nRF24L01p.h>
  nRF24L01p transmitter(7,8);
  String message;
  String temp_str;

//Interrupt
int Ch1Interrupt = 4; //Pin 19 : servo (5)
int Ch2Interrupt = 5; //Pin 18 : brushless (6)
int hallInterrupt = 0; //Pin 2 : Hall effect

unsigned long Ch1_startPulse;
unsigned long Ch2_startPulse;
unsigned long N;
volatile double Ch1_val;
volatile double Ch2_val;
volatile double Ch1_val_last;
volatile double Ch2_val_last;
volatile double T;
```

```
//Servo&Brushless
#include <Servo.h>
  Servo myservo;
  Servo mybrushless;
  int data[] = {1680,1680,1680,1680,1680};
  int servo_angle;
  int servo_pulse;
  int servo_count;
  int Ch_servo;
  int turning;
  int Speed;

//Blink without deley
  const long interval_print = 100;
  unsigned long previousMillis_print = 0;

  const long interval_sensor = 20;
  unsigned long previousMillis_sensor = 0;

//EEPROM
#include <EEPROM.h>
int addr = 0;
float val_1;
float val_2;
float val_3;
float val_4;
int sensorVal;
int sensorVal1;
int sensorVal2;

//TEST
  int M_roll;
  int M_Ax;
```

```
  int M_Gz;
  int Adog;

void setup()
{
  Serial.begin(115200);
  Wire.begin();

  pinMode(4,INPUT);
  pinMode(9,OUTPUT);
  pinMode(10,OUTPUT);
  pinMode(11,OUTPUT);
  pinMode(12,OUTPUT);
  pinMode(13,OUTPUT);

 //Accelerometer
 accel = ADXL345();
 if(accel.EnsureConnected())
  {
    Serial.println("Connected to ADXL345.");
  }
 else
  {
    Serial.println("Could not connect to ADXL345.");
  }
 accel.SetRange(2, true);
 accel.EnableMeasurements();

 Ay1 = 0;
 count = 0;

 //Velocity & Radius
 Vt = 0;
```

```
V = 0;
L = 0.275;
D = 0.255;

//Gyro
delay(100);
itg3205.itg3205initGyro();
delay(100);
itg3205.itg3205CalGyro();
delay(100);

//Compass
  Serial.println("Starting the I2C interface.");
  Serial.println("Constructing new HMC5883L");
compass = HMC5883L();
  Serial.println("Setting scale to +/- 1.3 Ga");
error = compass.SetScale(1.3);
if(error != 0)
  Serial.println(compass.GetErrorText(error));
  Serial.println("Setting measurement mode to continous.");
error = compass.SetMeasurementMode(Measurement_Continuous);
if(error != 0)
  Serial.println(compass.GetErrorText(error));

//Angle
Araw = accel.ReadRawAxis();
roll_R = atan2(Araw.XAxis,Araw.ZAxis);
pitch_R = atan2(Araw.YAxis,Araw.ZAxis);

roll_A = (roll_R/PI)*180;
pitch_A = (pitch_R/PI)*180;

roll_G = (roll_R/PI)*180;
```

```
pitch_G = (pitch_R/PI)*180;

roll_C = (roll_R/PI)*180;
pitch_C = (pitch_R/PI)*180;
yaw_C = (heading/PI)*180;

roll_ref = roll_C;

Mscaled = compass.ReadScaledAxis();
heading = atan2(Mscaled.YAxis, Mscaled.XAxis);

//NRF24L01
delay(150);
SPI.begin();
SPI.setBitOrder(MSBFIRST);
transmitter.channel(90);
transmitter.TXaddress("ALL");
transmitter.init();

//Interrupt
attachInterrupt(Ch1Interrupt, Ch1_begin, RISING);
attachInterrupt(Ch2Interrupt, Ch2_begin, RISING);
attachInterrupt(hallInterrupt, Hall, RISING);

//Servo
servo_count = 0;
turning = 1680;
Ch_servo = 1680;
servo_angle = 110;
servo_pulse = 1680;
myservo.attach(5);
mybrushless.attach(6);
```

```
  delay(100);
}

void loop()
{
  //Blink without deley (sensor)
  unsigned long currentMillis = millis();
  if(currentMillis - previousMillis_sensor >= interval_sensor)
  {
    previousMillis_sensor = currentMillis;
    digitalWrite(13,HIGH);

    //Accelerometer
    Araw = accel.ReadRawAxis();
    xAxisRawData = Araw.XAxis;
    Ascaled = accel.ReadScaledAxis();

    Ax = Ascaled.XAxis * 9.81;
    Ay = Ascaled.YAxis * 9.81;
    Az = Ascaled.ZAxis * 9.81;

    if (Ax < 0)
    {
      Av = -1 * Ax;
    }
    else Av = Ax;

    //Gyro
    itg3205.itg3205ReadGyro();
    if(itg3205.itg3205GyroX() > -0.07 && itg3205.itg3205GyroX() < 0.07)
        Gx = 0.00;
    else Gx = itg3205.itg3205GyroX();
    if(itg3205.itg3205GyroY() > -0.07 && itg3205.itg3205GyroY() < 0.07)
```

```
    Gy = 0.00;
else Gy = itg3205.itg3205GyroY();
if(itg3205.itg3205GyroZ() > -0.21 && itg3205.itg3205GyroZ() < 0.21)
    Gz = 0.00;
else Gz = itg3205.itg3205GyroZ();

//Compass
Mraw = compass.ReadRawAxis();
Mscaled = compass.ReadScaledAxis();
MilliGauss_OnThe_XAxis = Mscaled.XAxis;
heading = atan2(Mscaled.YAxis, Mscaled.XAxis);
declinationAngle = 0.0457;
heading += declinationAngle;
if(heading < 0)
  heading += 2*PI;
if(heading > 2*PI)
  heading -= 2*PI;
headingDegrees = heading * 180/M_PI;

//Angle
roll_R = atan2(Araw.XAxis,Araw.ZAxis);
pitch_R = atan2(Araw.YAxis,Araw.ZAxis);

roll_A = (roll_R/PI)*180;
pitch_A = (pitch_R/PI)*180;

roll_G -= itg3205.itg3205GyroY() *0.020 * 1.5;
pitch_G -= itg3205.itg3205GyroX() *0.020 * 1.5;
yaw_G += Gz *0.020;
if(yaw_G < 0)
  yaw_G += 360;
if(yaw_G > 360)
  yaw_G -= 360;
```

```
  roll_C = 0.98*(roll_C - itg3205.itg3205GyroY() *0.020 * 1.5) + 0.02*(roll_A);
  pitch_C = 0.98*(pitch_C + itg3205.itg3205GyroX() *0.020) + 0.02*(pitch_A);

  roll_S = roll_C / 10;
  pitch_S = pitch_C / 10;

  roll_del = roll_C - roll_ref;


 turning = Ch1_val;
 if (Ch1_val < 1050)
 {
   servo_angle = 40; // Pulse 1000 ms (Min)
   servo_pulse = 1000;
   Si = 21;
   So = 21;
 }
 else if (1050 >= Ch1_val && Ch1_val < 1150)
 {
   servo_angle = 50; // Pulse 1100 ms
   servo_pulse = 1100;
   Si = 18;
   So = 18;
 }
 else if (1150 >= Ch1_val && Ch1_val < 1250)
 {
   servo_angle = 60; // Pulse 1200 ms
   servo_pulse = 1200;
   Si = 15;
   So = 15;
 }
 else if (1250 <= Ch1_val && Ch1_val < 1350)
```

```
 {
   servo_angle = 70; // Pulse 1300 ms
   servo_pulse = 1300;
   Si = 12;
   So = 12;
 }
 else if (1350 <= Ch1_val && Ch1_val < 1450)
 {
   servo_angle = 80; // Pulse 14000 ms
   servo_pulse = 1400;
   Si = 9;
   So = 9;
 }
 else if (1450 <= Ch1_val && Ch1_val < 1550)
 {
   servo_angle = 90; // Pulse 1500 ms
   servo_pulse = 1500;
   Si = 6;
   So = 6;
 }
 else if (1550 <= Ch1_val && Ch1_val < 1650)
 {
   servo_angle = 100; // Pulse 1600 ms
   servo_pulse = 1600;
   Si = 3;
   So = 3;
 }
 else if (1650 <= Ch1_val && Ch1_val < 1750)
 {
   servo_angle = 110; // Pulse 1700 ms (Middle)
   servo_pulse = 1700;
   Si = 0;
   So = 0;
```

```
 }
 else if (1750 >= Ch1_val && Ch1_val < 1850)
 {
   servo_angle = 120; // Pulse 1800 ms
   servo_pulse = 1800;
   Si = 3;
   So = 3;
 }
 else if (1850 >= Ch1_val && Ch1_val < 1950)
 {
   servo_angle = 130; // Pulse 1900 ms
   servo_pulse = 1900;
   Si = 6;
   So = 6;
 }
 else if (1950 >= Ch1_val && Ch1_val < 2050)
 {
   servo_angle = 140;  // Pulse 2000 ms
   servo_pulse = 2000;
   Si = 9;
   So = 9;
 }
 else if (2050 >= Ch1_val && Ch1_val < 2150)
 {
   servo_angle = 150; // Pulse 2100 ms
   servo_pulse = 2100;
   Si = 12;
   So = 12;
 }
 else if (2150 >= Ch1_val && Ch1_val < 2250)
 {
   servo_angle = 160; // Pulse 2200 ms
   servo_pulse = 2200;
```

```
    Si = 15;
    So = 15;
  }
  else if (2250 >= Ch1_val && Ch1_val < 2350)
  {
    servo_angle = 170; // Pulse 2300 ms
    servo_pulse = 2300;
    Si = 18;
    So = 18;
  }
  else if (2350 >= Ch1_val)
  {
    servo_angle = 180; // Pulse 2400 ms (Max)
    servo_pulse = 2400;
    Si = 21;
    So = 21;
  }

  R = (((L/sin((PI/180) * So)) + (sqrt (pow(L,L) + pow( (L/tan((PI/180) * Si))+
D,2))))/2) ;
  Vmax = sqrt(Av * R * cos((PI/180) * roll_C));

   if (8 < V && Ch2_val < 1600)
  {
    Vt = V2;
  }
  else
  {
    Vt = V;
  }

  V2 = Vt;
```

```
Speed = Ch2_val;

//Multiplexer
if ( (-50 > roll_A || roll_A > 50) && Ch2_val > 1600)
{
  digitalWrite(9,LOW);
  digitalWrite(10,HIGH);

  Adog = 1 ;
  mybrushless.write(1600);
}
else
{
  digitalWrite(10,LOW);
  digitalWrite(9,HIGH);
  Adog = 0 ;
  mybrushless.write(Speed);
}

 if (roll_A < 0)
 {
   M_roll = -1 * roll_A;
 }
 else M_roll = roll_A ;

 M_Ax = Av * 10 ;

 if (Gz < 0)
 {
   M_Gz = Gz / -10 ;
 }
 else M_Gz = Gz / 10 ;
```

```
  digitalWrite(13,LOW);
}

///Blink with deley (print)
if(currentMillis - previousMillis_print >= interval_print)
{
  previousMillis_print = currentMillis;
  //digitalWrite(13,HIGH);

  //EEPROM
  sensorVal = digitalRead(4);
  if (sensorVal == 1)
  {
    val_1 = (Ch1_val/10);
    val_2 = (Ch2_val/10);
    val_3 = (M_roll);
    val_4 = (Vt * 10);

    EEPROM.write(addr, val_1);
    addr = addr + 1;
    EEPROM.write(addr, val_2);
    addr = addr + 1;
    EEPROM.write(addr, val_3);
    addr = addr + 1;
    EEPROM.write(addr, val_4);
    addr = addr + 1;

    if (addr == 2000)
    {
       addr = 0;
    }

  }
```

```
    //Test
    Serial.println(Vt);

  }
 }

 //Innterrupt
 void Ch1_begin()
 {
   Ch1_startPulse = micros();
   detachInterrupt(Ch1Interrupt);
   attachInterrupt(Ch1Interrupt, Ch1_end, FALLING);
 }

 void Ch1_end()
 {
   Ch1_val = micros() - Ch1_startPulse;
   detachInterrupt(Ch1Interrupt);
   attachInterrupt(Ch1Interrupt, Ch1_begin, RISING);
   if (Ch1_val < 600 || Ch1_val > 2500) { Ch1_val = Ch1_val_last;}
   else {Ch1_val_last = Ch1_val;}
 }

 void Ch2_begin()
 {
   Ch2_startPulse = micros();
   detachInterrupt(Ch2Interrupt);
   attachInterrupt(Ch2Interrupt, Ch2_end, FALLING);
 }

 void Ch2_end()
 {
```

```
  Ch2_val = micros() - Ch2_startPulse;
  detachInterrupt(Ch2Interrupt);
  attachInterrupt(Ch2Interrupt, Ch2_begin, RISING);
  if (Ch2_val < 600 || Ch2_val > 2500) { Ch2_val = Ch2_val_last;}
  else {Ch2_val_last = Ch2_val;}
}

void Hall ()
{
  T = micros()- N ;
  if (25000 < T)
  {
    V = 0.43/(T/1000000) ;
    N = micros() ;
  }
}
```