หุ่นยนต์สำรวจควบคุมด้วยสมาร์ทดีไวซ์ผ่านระบบการเชื่อมต่อแบบไร้สาย

Wireless Explorer Robot Control by Smart Device

กรวีร์ ท่าหิน

กัปตัน บุราณรัตน์

ธนพนธ์ วิสารทกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2557

หุ่นยนต์สำรวจควบคุมด้วยสมาร์ทดีไวซ์ผ่านระบบการเชื่อมต่อแบบไร้สาย
Wireless Explorer Robot Control by Smart Device

กรวีร์ ท่าหิน
กัปตัน บุราณรัตน์
ธนพนธ์ วิสารทกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
สาขาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

WIRELESS EXPLORER ROBOT CONTROL BY SMART DEVICE

KORNRAWEE THAHIN
CAPTAIN BURANRAT
TANAPON VISANTAKUL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2014

ปริญญานิพนธ์ปีการศึกษา 2557
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ใบรับรองปริญญานิพนธ์

.....

หัวข้อปริญญานิพนธ์ หุ่นยนต์สำรวจควบคุมด้วย สมาร์ทดีไวซ์ ผ่านระบบการเชื่อมต่อแบบไร้สาย
Wireless Explorer Robot Control by Smart Device

นักศึกษาผู้จัดทำ

นายธนพนธ์	วิสารทกุล	**รหัสนักศึกษา** 54010545
นายกัปตัน	บุราณรัตน์	**รหัสนักศึกษา** 54010092
นายกรวีร์	ท่าหิน	**รหัสนักศึกษา** 54010033

ปริญญา วิศวกรรมศาสตรบัณฑิต
สาขา วิศวกรรมการวัดคุม
ปีการศึกษา 2557

อาจารย์ผู้ควบคุมปริญญานิพนธ์	ลายมือชื่อ
รศ.ดร.ทวีพล ซื่อสัตย์	

หัวข้อปริญญานิพนธ์	หุ่นยนต์สำรวจควบคุมด้วย สมาร์ทดีไวซ์ ผ่านระบบการเชื่อมต่อแบบไร้สาย Wireless Explorer Robot Control by Smart Device		
นักศึกษาผู้จัดทำ	นายธนพนธ์	วิสารทกุล	รหัสนักศึกษา 54010545
	นายกัปตัน	บุราณรัตน์	รหัสนักศึกษา 54010092
	นายกรวีร์	ท่าหิน	รหัสนักศึกษา 54010033
อาจารย์ที่ปรึกษา	รศ.ดร.ทวีพล	ซื่อสัตย์	
ปีการศึกษา	2557		

บทคัดย่อ

โครงงานฉบับนี้มีวัตถุประสงค์เพื่อศึกษาและสร้างหุ่นยนต์สำรวจ ซึ่งควบคุมด้วยสมาร์ทดีไวซ์ผ่านระบบการเชื่อมต่อแบบไร้สาย เนื่องด้วยในยุคปัจจุบันการสื่อสารแบบไร้สายมีการพัฒนาขีดความสามารถในการส่งข้อมูลระหว่างอุปกรณ์ได้สะดวกและรวดเร็วมากขึ้น ซึ่งสามารถนำมาประยุกต์ใช้ในการควบคุมและส่งข้อมูลระหว่างอุปกรณ์ สมาร์ทดีไวซ์ (สมาร์ทโฟน, tablet, คอมพิวเตอร์)กับอุปกรณ์อื่นๆได้ ทางคณะผู้จัดทำจึงเล็งเห็นถึงประโยชน์ของการสื่อสารแบบไร้สายสามารถนำมาใช้กับหุ่นยนต์ เพื่อเพิ่มความคล่องตัวในการควบคุมหุ่นยนต์ โดยหุ่นยนต์นี้สามารถเดินบนสภาวะภูมิศาสตร์ต่างๆได้อาทิ เช่น พื้นที่ชื้น, พื้นที่กัมมันตภาพรังสีรุนแรง, พื้นที่ประสบภัยพิบัติ, พื้นที่ที่ขรุขระ เป็นต้น ในโครงงานนี้ประกอบไปด้วยการออกแบบโครงสร้างของหุ่นยนต์และการควบคุมการเคลื่อนที่ของหุ่นยนต์ โดยใช้ Raspberry Pi 512MB โมเดล B+ ในการควบคุมการเคลื่อนที่หุ่นยนต์ อีกทั้งใช้ในการเชื่อมต่อสื่อสารข้อมูลระหว่างสมาร์ทดีไวซ์กับตัวหุ่นยนต์ นอกจากนี้ทำการศึกษาและเพิ่มเติมในส่วนของการตรวจจับอุณหภูมิและความชื้นในสถานที่ต่างๆได้ด้วย

Thesis Title	WIRELESS EXPLORER ROBOT CONTROL BY SMART DEVICE	
Authors	Mr. Kornrawee	Thahin
	Mr. Captain	Buranrat
	Mr. Tanapon	Visantakul
Thesis Advisor	Assoc.Prof.Dr. Taweepol	Suesut
Year	2014	

ABSRACT

This project aims to study and built an Explorer Robot with allow the smart device to access and control via wireless technology. In the present day, wireless technology has been improved the limit of ability to be faster data transfer and more convenient. In this project, the advantage of wireless communication is the main factor to make our project using this technology to transfer data and control the device between the smart devices such as smartphone, tablet, computer etc. with other device. In addition, the robot can be operated in any location such as moist area, radiation area, disaster area, rough area. In this project consists of the structural design of the robot and how to control the robot. In this project we use Raspberry Pi 512MB Model B+ as a brain of the robot and connect with smart device. Moreover, additional studies of temperature sensor and humidity sensor are presented.

กิตติกรรมประกาศ

ปริญญานิพนธ์เรื่อง หุ่นยนต์สำรวจควบคุมด้วยสมาร์ทดีไวซ์ผ่านระบบการเชื่อมต่อแบบไร้สาย (WIRELESS EXPLORER ROBOT CONTROL BY SMART DEVICE) สำเร็จลุล่วงไปด้วยดี เนื่องจากได้รับความอุปถัมภ์จากบุคคลหลายฝ่ายที่ให้คำปรึกษา และชี้แนะแนวทาง ทำให้ปริญญานิพนธ์ฉบับนี้บรรลุเป้าหมายตามวัตถุประสงค์ได้เป็นอย่างดี

คณะผู้จัดทำขอขอบพระคุณ รศ.ดร.ทวีพล ซื่อสัตย์ อาจารย์ที่ปรึกษาปริญญานิพนธ์ ที่ให้คำปรึกษา ชี้แนะ และข้อบกพร่องต่างๆเพื่อเป็นแนวทางในการแก้ปัญหาข้อผิดพลาดและอุปสรรคต่างๆ ทำให้ปริญญานิพนธ์ฉบับนี้ประสบผลสำเร็จไปด้วยดี นอกจากนี้คณะผู้จัดทำขอขอบพระคุณ คณะอาจารย์ประจำสาขาวิชาวิศวกรรมการวัดคุม ที่ให้คำปรึกษา

ขอขอบพระคุณภาคสาขาวิชาวิศวกรรมอาหาร ที่ให้เอื้อเฟื้ออุปกรณ์ และเครื่องมือต่างๆในการทำงานจนสำเร็จลุล่วงไปได้ด้วยดี

และสุดท้ายนี้คณะผู้จัดทำขอกราบขอบพระคุณ บิดา มารดา และครอบครัว เป็นอย่างสูงสำหรับความรัก กำลังใจ เป็นที่ปรึกษา และการสนับสนุนในด้านต่างๆ ที่มอบให้อย่างสม่ำเสมอจนเกิดเป็นแรงผลักดัน ทำให้โครงงานนี้ประสบความสำเร็จและผ่านลุล่วงไปได้อย่างสมบูรณ์

คณะผู้จัดทำ

สารบัญ

หน้า

สารบัญรูป

สารบัญตาราง

ตารางที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความสำคัญของปริญญานิพนธ์

เนื่องจากในยุคปัจจุบัน การสื่อสารผ่านระบบไร้สายนั้นมีการพัฒนาไปก้าวไกลทั้งระบบอินเตอร์เน็ต ผ่าน WiFi, เทคโนโลยี 2G, 3G และ 4G ทำให้เราสามารถส่งข้อมูลรวมถึงควบคุมอุปกรณ์ต่างๆ ได้ง่ายและสะดวกมากขึ้นจากอดีตที่ผ่านมา โครงการนี้จึงเล็งเห็นถึงประโยชน์ในการสื่อสารแบบไร้สาย ว่าสามารถนำมาประยุกต์ใช้ในการควบคุมและส่งข้อมูลระหว่างอุปกรณ์สมาร์ทดีไวซ์ เช่น สมาร์ทโฟน, tablet, คอมพิวเตอร์ กับอุปกรณ์อื่นๆได้ เราจึงเล็งเห็นว่าการเชื่อมต่อแบบไร้สายสามารถนำมาใช้กับหุ่นยนต์เพื่อเพิ่มความคล่องตัวในการควบคุมหุ่นยนต์โดยโครงงานนี้ได้เลือกใช้ Raspberry Pi 512MB โมเดล B+ เป็นตัวควบคุมการทำงานของหุ่นยนต์ให้เคลื่อนที่โดยสั่งการทำงานแบบไร้สายผ่านสมาร์ทดีไวซ์ ซึ่งหุ่นยนต์นี้สามารถทำงานต่างๆ ที่มนุษย์ไม่สามารถทำได้ เช่น การเข้าถึงสถานที่ที่มนุษย์ไม่สามารถเข้าถึงได้ เพื่อทำการวัดค่าต่างๆเช่น อุณหภูมิ ความชื้น และเป็นการแสดงผลลัพธ์เป็นแบบทันที

1.2 วัตถุประสงค์ของปริญญานิพนธ์

เพื่อศึกษาและเข้าใจการทำงาน ในการใช้งาน Raspberry Pi 512MB โมเดล B+ มาควบคุมการทำงานของหุ่นยนต์รวมถึงศึกษาการเชื่อมต่อสื่อสารข้อมูลระหว่าง สมาร์ทดีไวซ์ กับ ตัวหุ่นยนต์ และการเขียนโปรแกรมให้สามารถสั่งการหุ่นยนต์ผ่านสมาร์ทดีไวซ์ และสามารถวัดค่าอุณหภูมิและความชื้นตามสถานที่ต่างๆได้

1.3 ขอบเขตของปริญญานิพนธ์

1. สามารถทำให้หุ่นยนต์เคลื่อนที่ตามโปรแกรมที่สั่งได้
2. สามารถควบคุมสิ่งต่างๆ เช่น การเคลื่อนที่ การวัดค่าต่างๆ โดย Raspberry Pi 512MB โมเดล B+

1.4 ขั้นตอนการศึกษศึกษารูปแบบของหุ่นยนต์

1. ศึกษาอุปกรณ์ที่ต้องใช้ในการสร้างหุ่นยนต์
2. ศึกษาการทำงานของ Full H Bridge มอเตอร์ คอนโทรลเลอร์
3. ศึกษาการเขียนโปรแกรม SolidWorks เพื่อออกแบบตัวโครงสร้างและชิ้นส่วนของหุ่นยนต์
4. ศึกษาการเขียนโปรแกรม Altium Designer ในการออกแบบวงจรไฟฟ้า
5. ศึกษาการเขียนโปรแกรม JavaScript เพื่อควบคุมการเคลื่อนที่ของหุ่นยนต์
6. ศึกษาการใช้งาน Raspberry Pi 512MB โมเดล B+

1.5 ประโยชน์ที่คาดว่าจะได้รับ

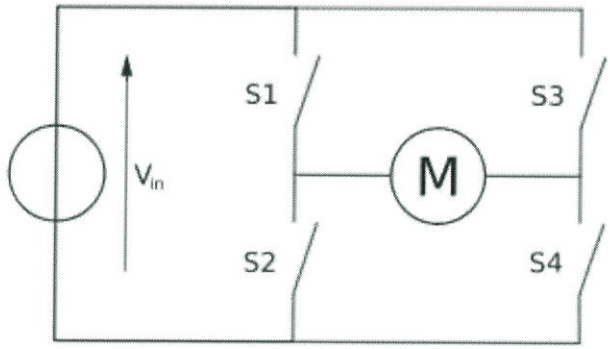
1. เข้าใจหลักการทำงานของวงจรควบคุมมอเตอร์ Full H Bridge มอเตอร์ คอนโทรลเลอร์
2. ได้เรียนรู้การออกแบบตัวโครงสร้างและชิ้นส่วนของหุ่นยนต์
3. ได้เรียนรู้การออกแบบวงจรไฟฟ้า โดยใช้โปรแกรม Altium Designer
4. ได้เรียนรู้ในส่วนของโปรแกรมที่ใช้ในการควบคุมหุ่นยนต์

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 มอเตอร์ คอนโทรลเลอร์

เราต้องการควบคุมทิศทาง และความเร็วรอบของมอเตอร์ ไม่ว่าจะเป็นการขับเคลื่อนสายพาน หรือล้อของหุ่นยนต์ เพราะเราคงไม่ต้องการให้หมุนแบบอิสระควบคุมไม่ได้ ดังนั้นก็เลยมีคนคิดวงจรที่ใช้ในการควบคุมมอเตอร์ขึ้นมาแบบที่นิยมใช้กันเรียกว่าวงจร "H-Bridge"

ควบคุมทิศทางการหมุน



รูปที่ 2.1 วงจรควบคุมทิศทางและความเร็วรอบของมอเตอร์

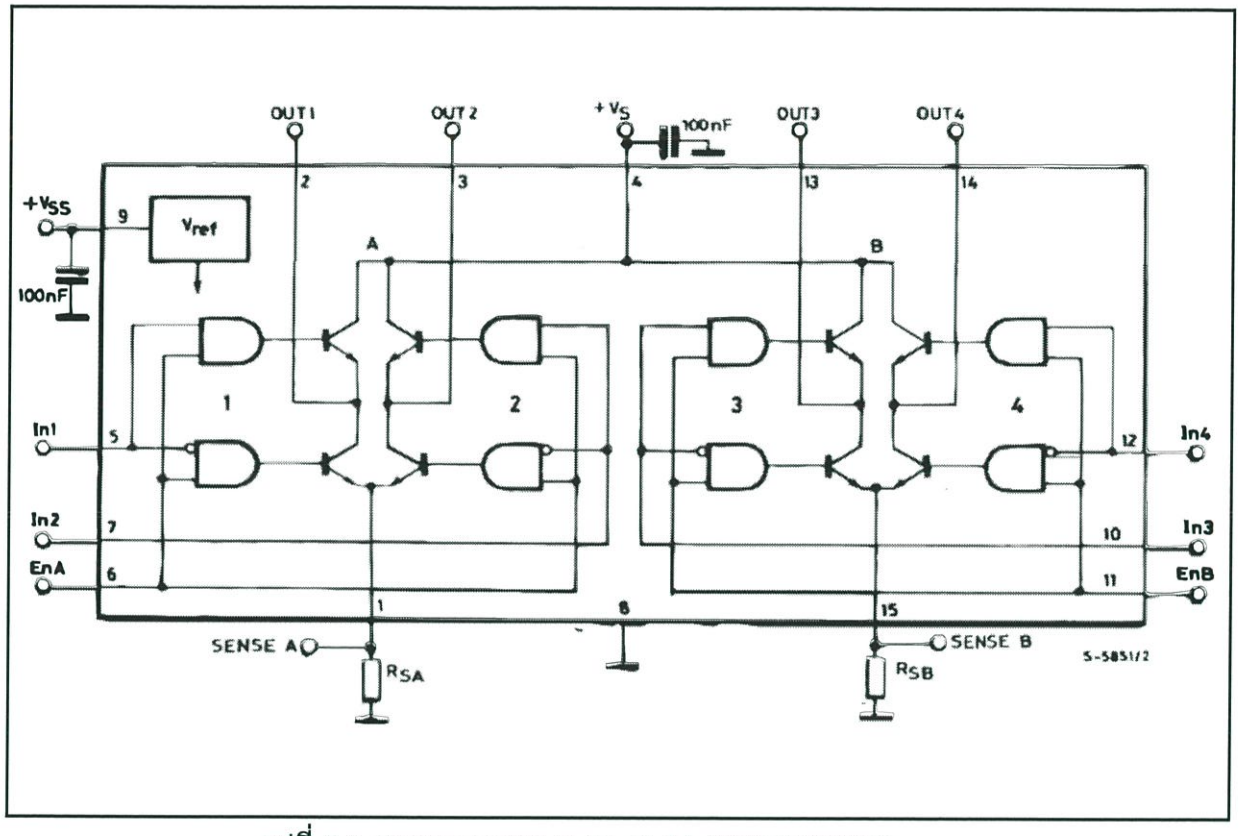
โดยปกติหากต้องการกลับทิศการหมุนของมอเตอร์กระแสตรง วิธีนึงที่ทำได้คือ กลับทิศแหล่งจ่าย ทีนี้ลองดู ที่รูปวงจร H-Bridge ในรูปที่ 1

- หากต้องการให้หมุนตามเข็ม (Clockwise :CW) ก็ให้ S1 และ S4 ปิดวงจร และให้ S2 และ S3 เปิดวงจร
- หากต้องการให้หมุนทวนเข็ม (Counter Clockwise :CCW) ก็ให้ S2และ S3 ปิดวงจร และให้ S1 และ S4 เปิดวงจร

จะเห็นว่าสวิตช์จะทำงานเป็นคู่ S1 คู่กับ S4 และ S2 คู่กับ S3 คู่แรกทำงาน คู่สองต้องเปิดวงจร และในทางตรงข้ามก็คือคู่สองทำงาน คู่แรกต้องเปิดวงจร ทีนี้จะทำอย่างไรให้การเปิดปิดเป็นแบบที่ง่ายกว่านี้ คำตอบก็คือ ใช้อุปกรณ์สารกึ่งตัวนำเช่น MOSFET หรือ IGBT หรือ อื่นๆ แล้วแต่ความเหมาะสม เช่น ขนาดกระแสแรงดันที่ต้องการควบคุม IC ที่มีวงจร H-Bridgeเลือก 2 แบบ หลักๆ คือ L293D และ L298P เราเลือกใช้ L298

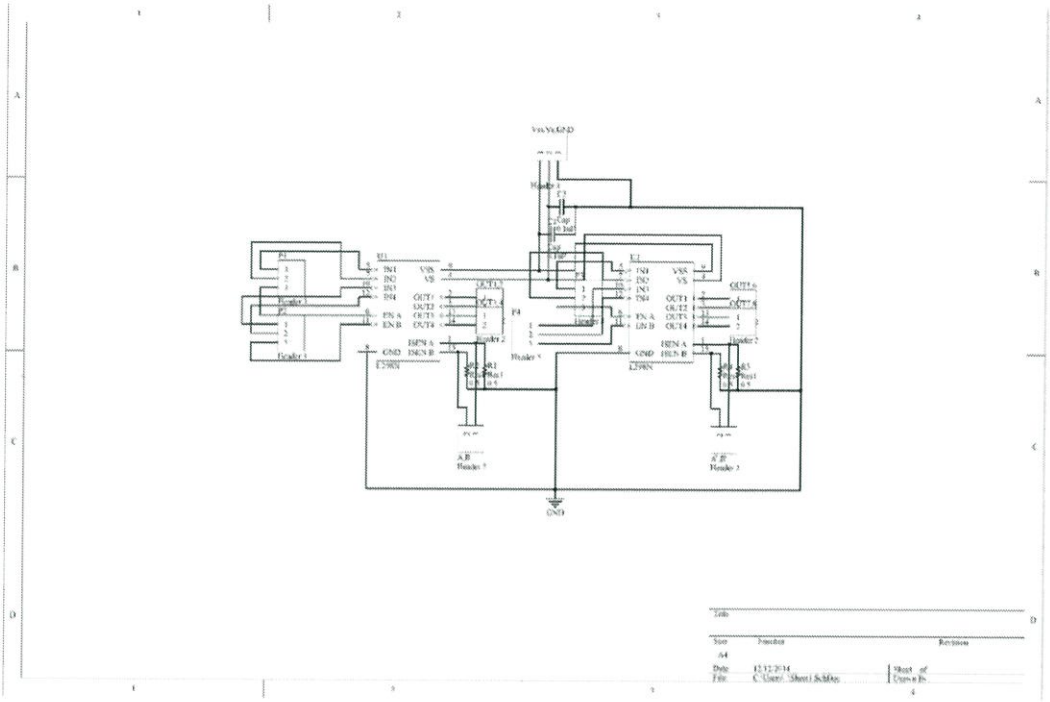
2.1.2 หลักการทำงานของ มอเตอร์ คอนโทรลเลอร์ L298-DUAL FULL-BRIDGE DRIVER

BLOCK DIAGRAM



รูปที่ 2.2 วงจรของ L298-DUAL FULL-BRIDGE DRIVER

วงจรในรูปที่ 2 นั้นเป็นวงจรของ **L298-DUAL FULL-BRIDGE DRIVER** ซึ่งเป็นวงจรที่สามารถควบคุมการการหมุนของมอเตอร์ได้ถึงสองตัว โดยการทำงานหลักๆนั้น คือ เมื่อจ่ายกระแสที่ขา Vs ซึ่งจะจ่ายกระแส 12 V เพื่อจ่ายให้กับมอเตอร์นั้นหมุนได้ โดยขา In1, In2 และ EnA เป็นขาที่ควบคุมทิศทางการหมุนของมอเตอร์ โดยถ้าเราจ่ายกระแส In1 และ EnA จะทำให้มอเตอร์หมุนในทิศทางที่ 1 และถ้าเราจ่ายกระแสที่ขา In2 และ EnA จะทำให้มอเตอร์หมุนไปอีกทิศทางนึง

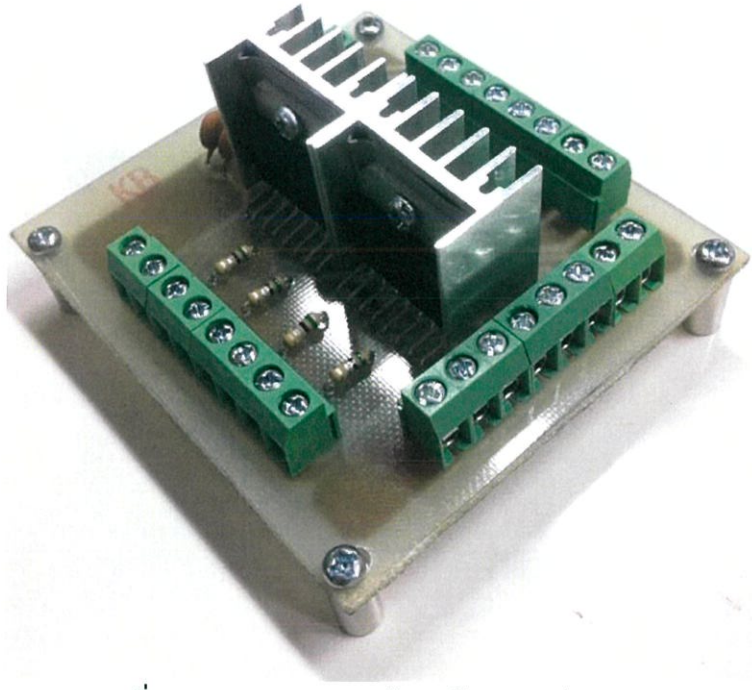


รูปที่ 2.3 แสดง schematic ของ มอเตอร์ คอนโทรลเลอร์

ABSOLUTE MAXIMUM RATINGS

ตารางที่ 2.1 แสดงตาราง Absolute maximum ratings

Symb	Parameter	Value	Uni
VS	Power Supply	50	V
VSS	Logic Supply Voltage	7	V
VI,Ve	Input and Enable Voltage	–0.3 to 7	V
IO	Peak Output Current (each Channel) – Non Repetitive (t = 100ms) –Repetitive (80% on –20% off; ton = 10ms) –DC Operation	 3 2.5 2	 A A A
Vsen	Sensing Voltage	–1 to 2.3	V
Ptot	Total Power Dissipation (Tcase = 75C)	25	W
Top	Junction Operating Temperature	–25 to 130	C
Tstg,	Storage and Junction Temperature	–40 to 150	C

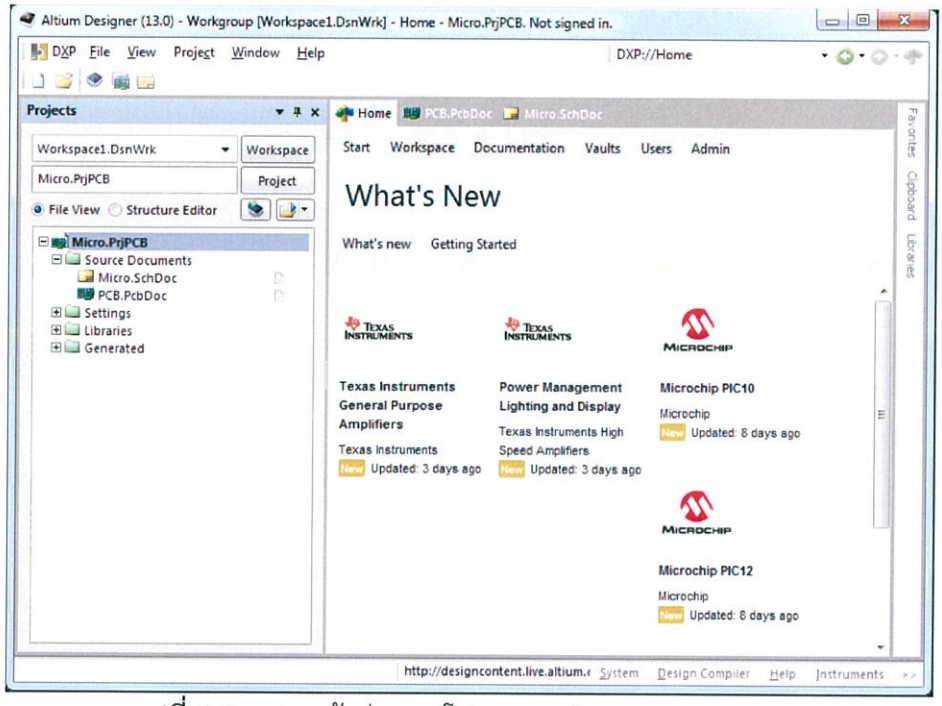


รูปที่ 2.4 รูปวงจร มอเตอร์ คอนโทรลเลอร์

2.2 โปรแกรม Altium Design

โปรแกรม Altium Design คือโปรแกรมทางด้านการออกแบบวงจรไฟฟ้าและสร้างลายวงจรลงบนแผ่นวงจรพิมพ์หรือ PCB (Printed Circuit Board) ที่ไว้ใช้สำหรับลงอุปกรณ์และบัดกรีให้ได้วงจรที่สามารถทำงานได้โดยโปรแกรม Altium Design เป็นโปรแกรมที่ทางบริษัท Altium Limited ได้พัฒนาขึ้นโดยมีฟังก์ชันใช้งานที่มีความหลากหลาย ใช้ง่าย และมีความแม่นยำในการออกแบบวงจรไฟฟ้าสูง

รูปที่ 2.5 แสดงโปรแกรม Altium Design 2013



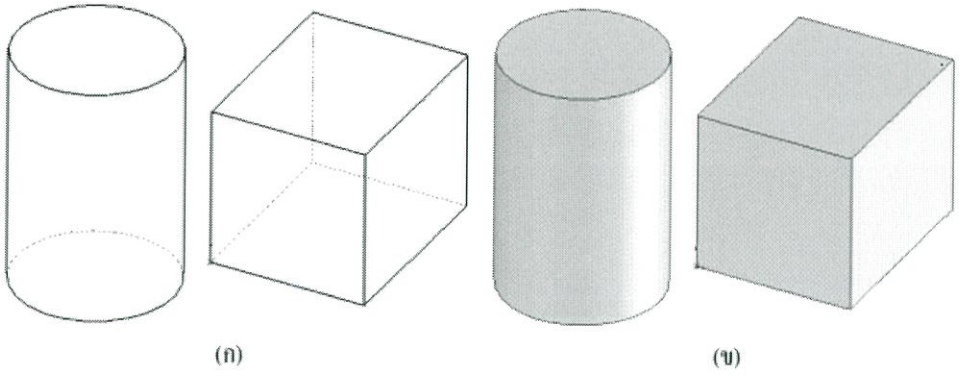
รูปที่ 2.6 แสดงหน้าต่างของโปรแกรม Altium Design 2013

ขั้นตอนการออกแบบ PCB (โดยรวม)

1. วาดลายวงจรในหน้า Schematic
2. อัพเดทใส่หน้า PCB
3. จัดอุปกรณ์และลากลายวงจร (ลากด้วยตัวเอง หรือ โปรแกรม)
4. ปริ้นท์ลายวงจร

2.3 โปรแกรม SolidWorks

โปรแกรม SolidWorks เป็นโปรแกรมช่วยเขียนแบบที่เน้นการเขียนแบบใน 3 มิติหรือแบบ Parametric Solid โมเดลคำว่า Solid โมเดล หมายถึงแบบที่มีทรงตัน ซึ่งหมายถึงแบบใน 3 มิติที่มีเนื้อใน ซึ่งแตกต่างจากแบบ 3 มิติชนิด Wireframe ซึ่งสามารถใช้โปรแกรม CAD 2 มิติบวกกับจินตนาการของคน เขียนแบบก็สามารถเขียนออกมาได้ Wireframe 3-D โมเดล จึงเป็นแบบ 3 มิติที่ประกอบด้วยลายเส้น หรือ line มาต่อๆ กัน ส่วนคำว่า Parametric Solid โมเดล หมายถึงแบบ 3 มิติที่ถูกสร้างขึ้นด้วยความสัมพันธ์ทางคณิตศาสตร์ ซึ่งเป็นการคำนวณภายในตัวโปรแกรม การเขียนแบบ 3 มิติลักษณะนี้จะมีความสะดวกกับ ผู้เขียนแบบมากกว่ารูป 2.5 แสดงแบบ 3 มิติชนิด Wireframe โมเดล และ Solid โมเดล



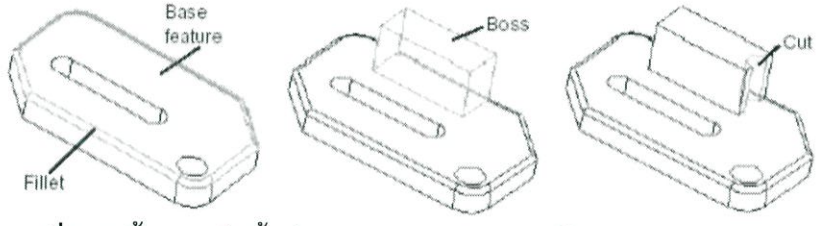
รูปที่ 2.7 แบบจำลอง 3 มิติ (ก) แบบ Wireframe และ (ข) แบบ Solid โมเดล

2.3.1 ขั้นตอนการสร้างชิ้นงาน 3 มิติและแบบสั่งงานในโปรแกรม SolidWorks

กระบวนการสร้างชิ้นงานใน SolidWorks ประกอบด้วย 3 ขั้นตอนหลักได้แก่

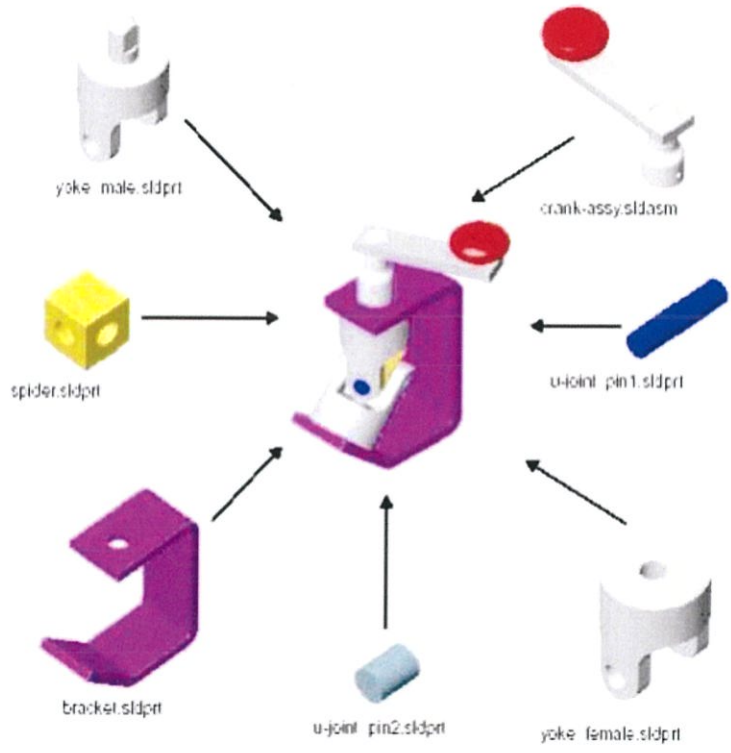
1) การสร้าง Sketch หรือลายเส้นใน 2 มิติบนระนาบ (plane, flat surface) ใดๆ ลายเส้นที่สร้างขึ้นส่วนใหญ่จะเป็นรูปปิด

2) ใช้คำสั่งใดๆ ใน Features ทำให้ Sketch กลายเป็นชิ้นงาน 3 มิติในการสร้างชิ้นงาน 3 มิติที่ ไม่ซับซ้อน Feature จะถูกสร้างซ้อนทับต่อๆ กันไป โดย feature แรกสุดโดยทั่วไปจะใช้ feature ที่มีขนาดใหญ่ เป็นหลักเรียกว่า Base ส่วน feature ที่สร้างต่อๆ มาจะเรียกว่า Boss ให้ดูรูปที่ 2.6



รูปที่ 2.8 ชิ้นงานหรือชิ้นส่วนแบบ Part ประกอบด้วยหลายๆ Features

3) นำ Parts หลายๆ ชิ้นมาประกอบกันเป็น Assembly ขอเรียกเป็นภาษาไทยว่า “ชิ้นงานประกอบ” ใน กรณีที่ชิ้นงานประกอบไม่ซับซ้อน ก็อาจเรียกว่า Subassembly เราสามารถนำ part และ subassembly หลายๆ ชิ้น มาประกอบเป็นชิ้นงานประกอบขั้นสุดท้ายเรียกว่า Full Assembly (ดูรูปที่ 2.7)



รูปที่ 2.9 ชิ้นงานประกอบแบบ Assembly ประกอบด้วย Parts และ Sub-assembly

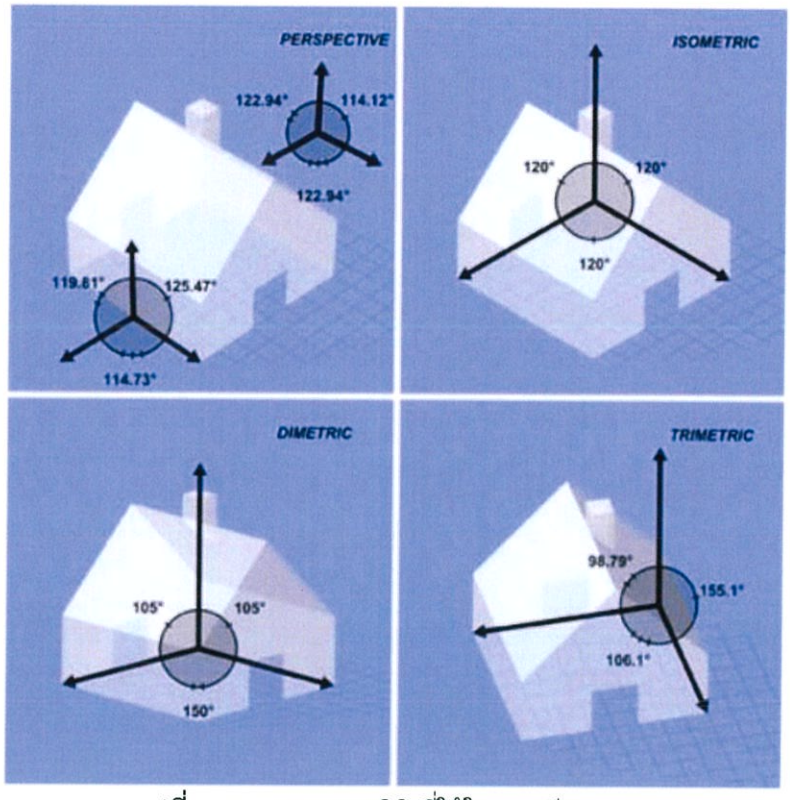
2.3.2 หลักการเขียนแบบพื้นฐานที่ควรรู้

แบบทางวิศวกรรม(Engineering Drawing) เป็นแบบจำลอง (โมเดล) ที่สร้างขึ้นบนระนาบ (e.g. กระดาษเขียนแบบ, จอภาพ) เพื่อเป็นตัวแทนของชิ้นส่วนทางกล (Mechanical Part) แบ่งออกเป็น 2 ประเภท ใหญ่ๆ คือ (1) แบบ 3 มิติและ (2) แบบ 2 มิติในบางครั้งจะใช้คำว่า “มุมมอง” หรือ View เพื่อขยายความว่า แบบในแต่ละประเภท เกิดจากการมองวัตถุในลักษณะใดเพื่อให้ได้แบบที่มีความยาวใน 2 หรือ 3 มิติสำหรับแบบ 2 มิติจะเกิดจากการมองตั้งฉากกับด้านใดด้านหนึ่งของวัตถุเพื่อให้เห็นขนาดที่แท้จริง (true length) แบบ 2 มิติเรานิยมเรียกว่า ภาพฉาย (Projection)

2.3.2.1 มุมมอง 3 มิติ (3-D View)

ที่นิยมใช้ในการเขียนแบบเครื่องจักรกล คือ แบบ Axonometric หมายถึงการมองวัตถุให้เยื้องออกจากแกนหลักทั้ง 3 แกน (ไม่มองขนานกับแกนใดๆ เลย) ผลที่ได้ก็คือ ภาพวัตถุที่มีมิติใน ทั้ง 3 แกน ซึ่งแบ่งออกได้เป็น 3 ประเภท คือ

1. Isometric View คือ ภาพที่มีสามแกนหลักทำมุมเท่ากันทั้งหมด ทุกมุมมีขนาด 120 องศา
2. Dimetric View คือ มีมุมระหว่างแกนหลักในภาพ เท่ากัน 2 มุม
3. Trimetric View คือ ทั้งสามมุมระหว่างแกนหลักไม่เท่ากันเลย

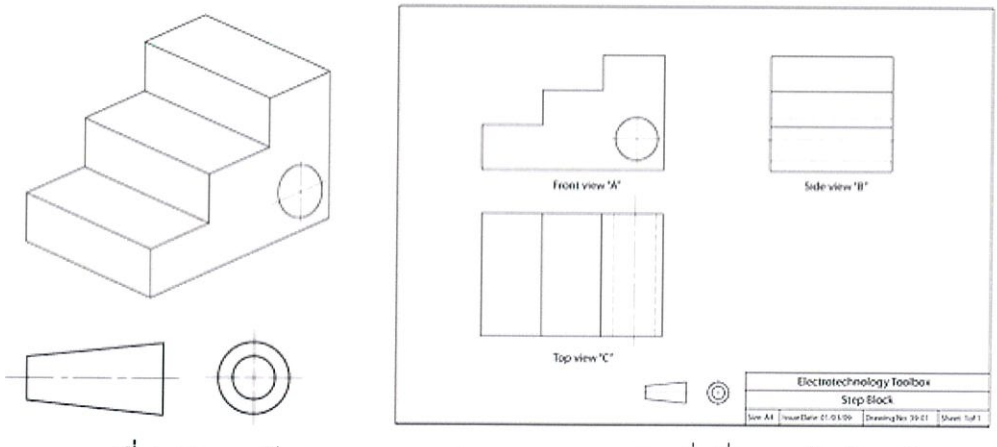


รูปที่ 2.10 มุมมอง 3 มิติ ที่ใช้ในการเขียนแบบ

2.3.2.2 ภาพฉาย 2 มิติ (2-D Projection)

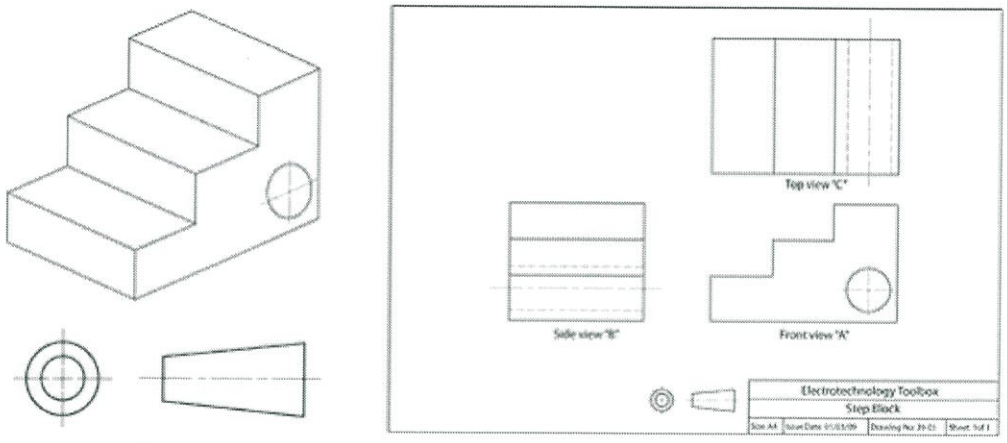
เนื่องจากเป็นการมองตั้งฉากกับด้านใดด้านหนึ่งของวัตถุจึงนิยม เรียกว่า ภาพฉายแบบ orthographic (Orthographic Projection) แบ่งออกเป็น 2 ประเภท คือ

1. แบบมุมที่หนึ่ง (First Angle Projection)



รูปที่ 2.11 การเขียนภาพฉาย orthographic แบบมุมที่หนึ่ง และสัญลักษณ์

2. แบบมุมที่สาม (Third Angle Projection)



รูปที่ 2.12 การเขียนภาพฉาย orthographic แบบมุมที่สาม และสัญลักษณ์

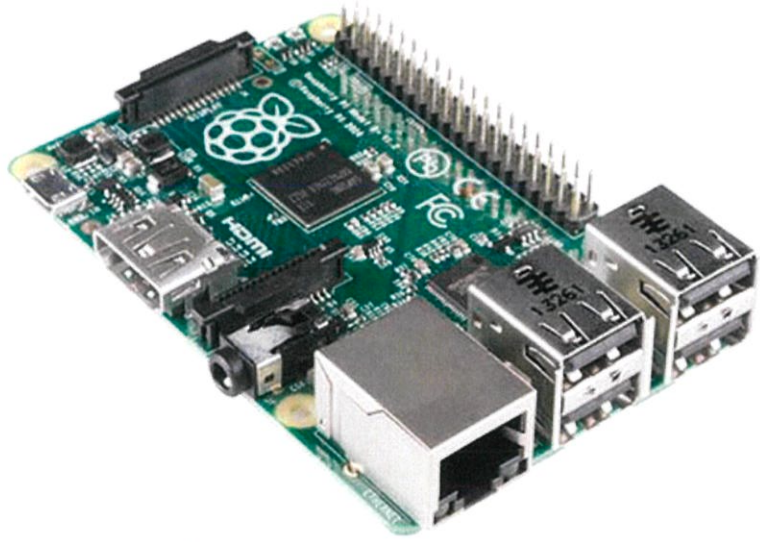
2.3.3 หน่วยในการวัดความยาวของการเขียนแบบทางกล

แบ่งออกเป็น 2 มาตรฐาน คือ

1. หน่วยเมตทริค (Metric Unit) โดยความยาวจะนิยมใช้เป็น มิลลิเมตรตัวย่อคือ มม. (mm) ข้อสังเกต ปกติความยาวในทางวิศวกรรมจะใช้เป็น เมตร (ม. หรือ m) การที่ใช้เป็นมิลลิเมตร เพราะในงานสร้างชิ้นงานทางกล ซึ่งมีขนาดค่อนข้างเล็กเราต้องการความความละเอียดในการวัดเพื่อให้ได้ขนาดที่ถูกต้อง แม่นยำ หน่วย metric นี้อยู่ในกลุ่มเดียวกับ หน่วย S.I. ซึ่งเป็นที่นิยมใช้แพร่หลายมากที่สุด

2. หน่วยแบบอังกฤษ (English Unit) หรือ หน่วยแบบ Old English ความยาวจะเป็น นิ้ว (inch) จะ ใช้สัญลักษณ์ double quote (") ตามหลังตัวเลขความยาวเป็นนิ้วเช่น 1" คือยาว 1 นิ้วและ 3 ½ คือยาวสาม นิ้วครึ่ง ประเทศที่ยังใช้ความยาวเป็นนิ้ว ที่สำคัญ คือ สหรัฐอเมริกาความยาวใน หน่วยนิ้วยังสามารถแบ่งย่อย ให้เล็กลงได้อีก หน่วยย่อยของนิ้วในภาษาไทยคือ "หุน" กาหนดให้ 1 นิ้ว = 8 หุน ฉะนั้น ½ นิ้ว จะเท่ากับ 4 หุน จะพบการใช้งานหน่วยนิ้วและหุน ในการบอกความยาวของน๊อตและสกรู และขนาดของดอกสว่าน เป็นต้น

2.4 บอร์ด Raspberry Pi โมเดล B+

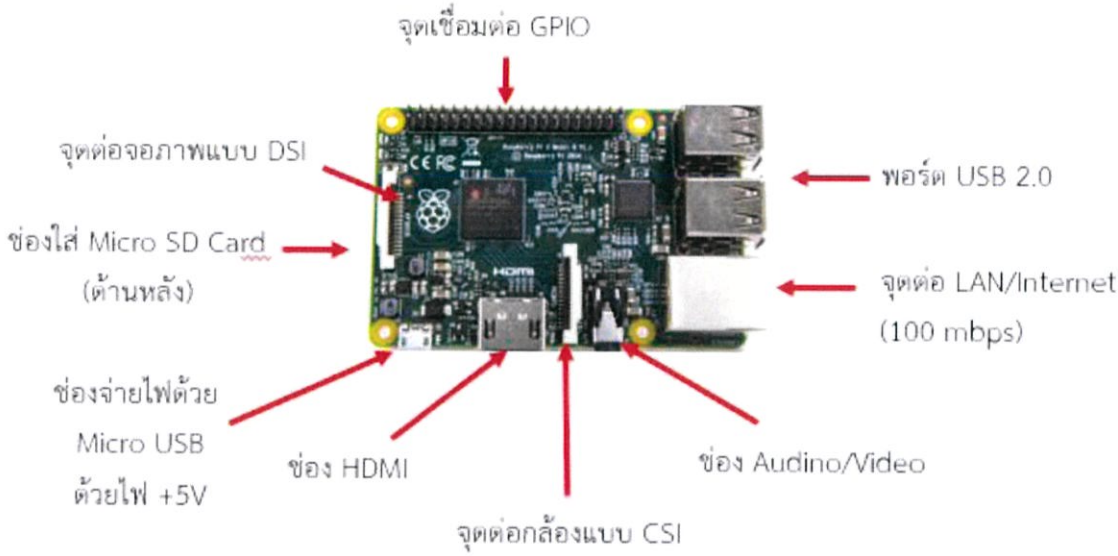


รูปที่ 2.13 บอร์ด Raspberry Pi โมเดล B+

คือบอร์ดคอมพิวเตอร์ขนาดเล็กที่สามารถเชื่อมต่อกับจอมอนิเตอร์ คีย์บอร์ด และเมาส์ได้ สามารถนำมาประยุกต์ใช้ในการทำโครงงานทางด้านอิเล็กทรอนิกส์ การเขียนโปรแกรม หรือเป็นเครื่องคอมพิวเตอร์ตั้งโต๊ะขนาดเล็ก ไม่ว่าจะเป็นการทำงาน Spreadsheet Word Processing ท่องอินเทอร์เน็ต ส่งอีเมล หรือเล่นเกมส์ อีกทั้งยังสามารถเล่นไฟล์วีดีโอความละเอียดสูง (High-Definition) ได้อีกด้วย

บอร์ด Raspberry Pi รองรับระบบปฏิบัติการลินุกซ์ (Linux Operating System) ได้หลายระบบ เช่น Raspbian (Debian) Pidora (Fedora) และ Arch Linux เป็นต้น โดยติดตั้งบน SD Card บอร์ด Raspberry Pi นี้ถูกออกแบบมาให้มี CPU GPU และ RAM อยู่ภายในชิปเดียวกัน มีจุดเชื่อมต่อ GPIO ให้ผู้ใช้สามารถนำไปใช้ร่วมกับอุปกรณ์อิเล็กทรอนิกส์อื่นๆ ได้อีกด้วย

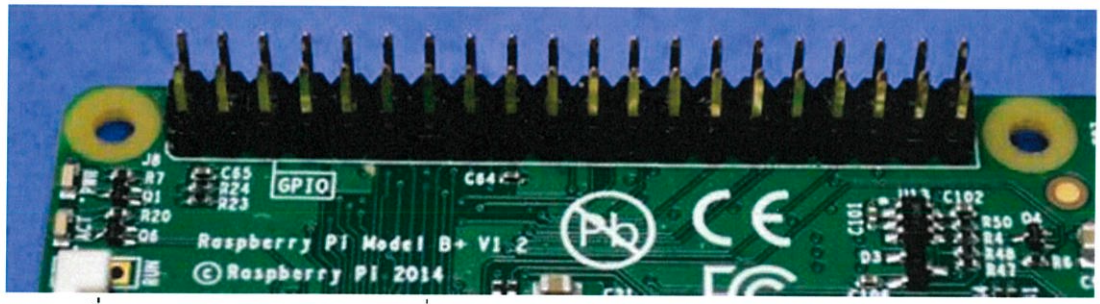
2.4.1 ข้อจำกัดและส่วนประกอบของ Raspberry Pi โมเดล B+



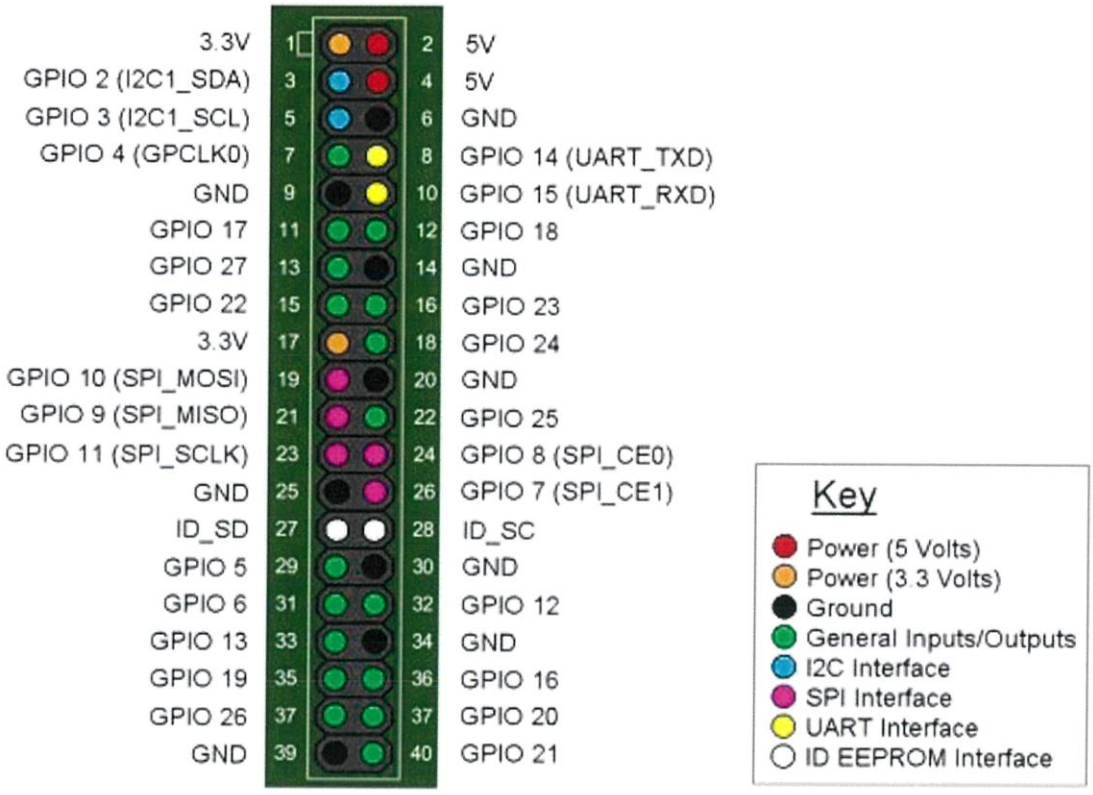
รูปที่ 2.14 ส่วนประกอบของ Raspberry Pi โมเดล B+

ส่วนประกอบหลักของบอร์ด Raspberry Pi โมเดล B+

1) จุดเชื่อมต่อ GPIO ซึ่งในส่วนของจุดเชื่อมต่อ โมเดล B+ นี้จะมีทั้งหมด 40 pins สามารถเพียงพอกับความต้องการใช้งานของผู้ใช้หลายๆรายได้ ซึ่งใน โมเดล รุ่นอื่นๆที่ผ่านมาจะเป็น 26 pins อาจทำให้ไม่เพียงพอต่อการใช้งาน



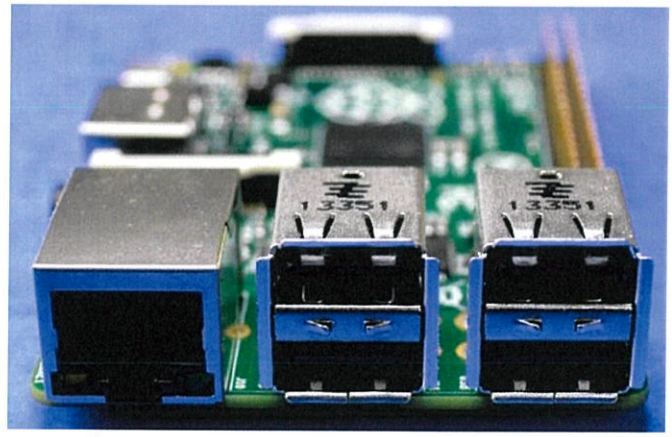
รูปที่ 2.15 แสดงลักษณะของจุดเชื่อมต่อ GPIO แต่ละจุดจำนวน 40 จุด ของ Raspberry Pi



รูปที่ 2.16 แสดงรูปแบบในแต่จุดต่างๆของ GPIO

2) พอร์ต USB 2.0 ซึ่งในส่วนของพอร์ต USB 2.0 ของ โมเดล B+ นั้นจะมีทั้งหมดจำนวน 4 พอร์ต ซึ่งใน โมเดล รุ่นก่อนๆนั้นจะมีเพียงแค่ 2 พอร์ตเท่านั้น

3) จุดต่อ LAN/Ethernet (100 mbps) ใช้สำหรับเป็นจุดต่อของสาย LAN/Ethernet

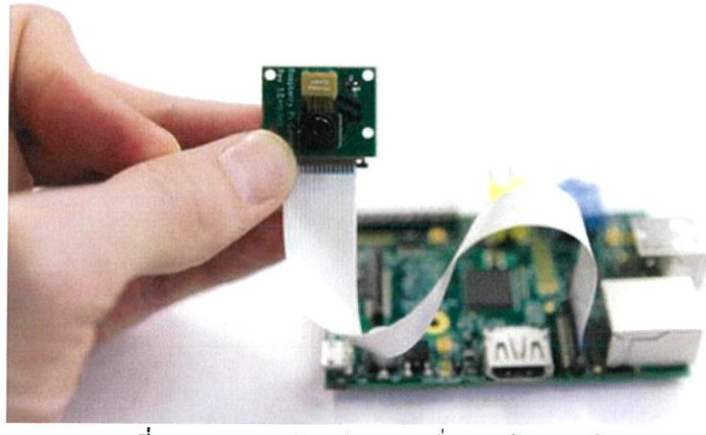


รูปที่ 2.17 พอร์ตต่อ LAN และ พอร์ตสำหรับใส่ USB 2.0 จำนวน 4 พอร์ต

4) พอร์ต HDMI ใช้สำหรับเชื่อมต่อสัญญาพและเสียง

รูปที่ 2.18 แสดงตัวอย่างสาย HDMI

5) พอร์ต CSI (Camera Serial Interface) ใช้สำหรับเชื่อมต่อโมดูลกล้อง



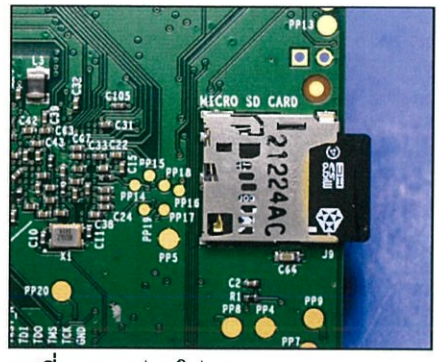
รูปที่ 2.19 แสดงตัวอย่างการเชื่อมต่อโมดูลกล้อง

6) ชิพ Boardcom BCM2835 SoC

7) พอร์ตช่องจ่ายไฟด้วย Micro USB ใช้สำหรับเป็นไฟเลี้ยงวงจร Raspberry Pi

8) พอร์ต DSI (Display Serial Interface) ใช้สำหรับต่อจอแสดงผล เช่น จอแสดงผลแบบ TFT Touch Screen เป็นต้น

9) ช่องใส่ Micro SD Card ซึ่งจะอยู่บริเวณด้านล่างของบอร์ด Raspberry Pi



รูปที่ 2.20 ช่องใส่ Micro SD Card

2.4.2 อุปกรณ์เสริมสำหรับ Raspberry Pi

1. Case สำหรับ Raspberry Pi โมเดล B+ เพื่อป้องกันอันตรายและการกระแทกจากสิ่งต่างๆไปยัง Raspberry Pi

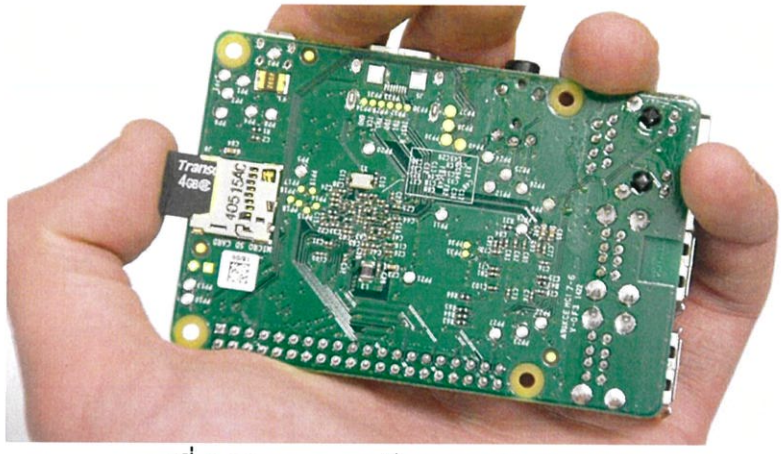
รูปที่ 2.21 Case สำหรับ Raspberri Pi โมเดล B+

2. สาย HDMI จำเป็นที่ต้องใช้สำหรับเชื่อมต่อ Raspberry Pi เข้ากับจอมอนิเตอร์
3. สาย Ethernet ใช้สำหรับเชื่อต่อ Raspberry Pi เข้ากับเราท์เตอร์
4. จอมอนิเตอร์ ,คีย์บอร์ด และเมาส์

2.4.3 การเริ่มต้นการใช้งาน Raspberry Pi โมเดล B+

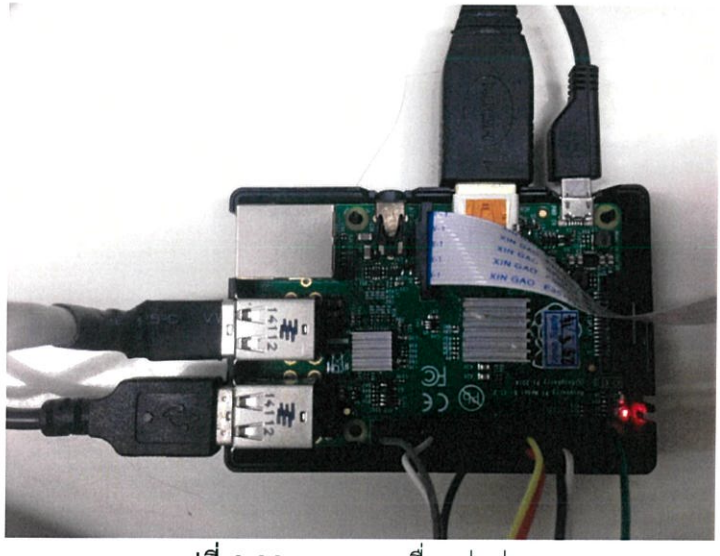
จากการใช้งานบอร์ด ซึ่งมีขั้นตอนการใช้งานเริ่มต้น ดังนี้

1) นำ MICRO SD CARD ที่ลงระบบปฏิบัติการเรียบร้อยแล้ว เสียบช่อง MICRO SD CARD ของบอร์ด Raspberry Pi ดังรูป



รูปที่ 2.22 แสดงการเสียบ MiCRO SD CARD

2) ทำการเชื่อมต่อคีย์บอร์ด และเมาส์เข้าที่ช่อง USB 2.0 ต่อสาย HDMI เข้าจอภาพ ในกรณีที่จอภาพไม่มีขั้ว HDMI อาจต่อสัญญาณภาพผ่านทางขั้ว RCA ก็ได้ แต่คุณภาพของภาพจะด้อยลงไปด้วย จากนั้นให้เปิดจอภาพ และจ่ายไฟเลี้ยงเข้าบอร์ด Raspberry Pi ผ่านทางขั้ว Micro USB ดังรูป

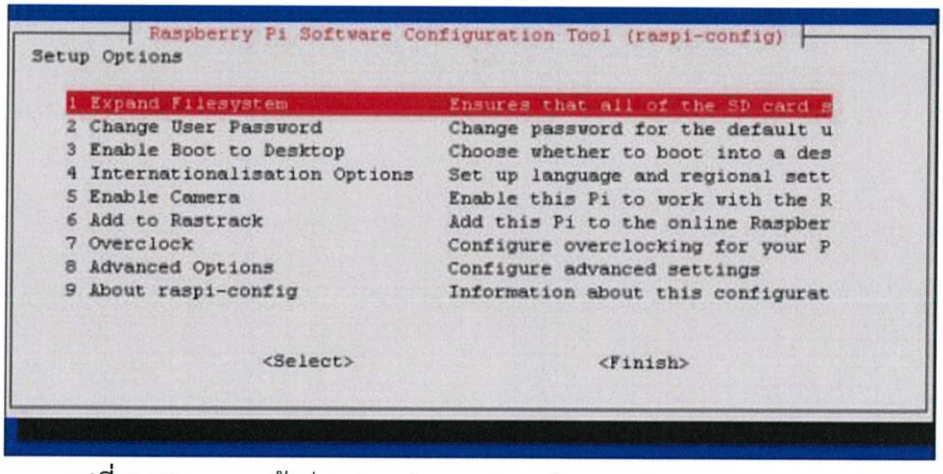


รูปที่ 2.23 แสดงการเชื่อมต่อต่างๆ

3) จากนั้นจะแสดงข้อความการบูตระบบของบอร์ด Raspberry Pi ดังรูป

รูปที่ 2.24 แสดงการบูตของบอร์ดบนจอ

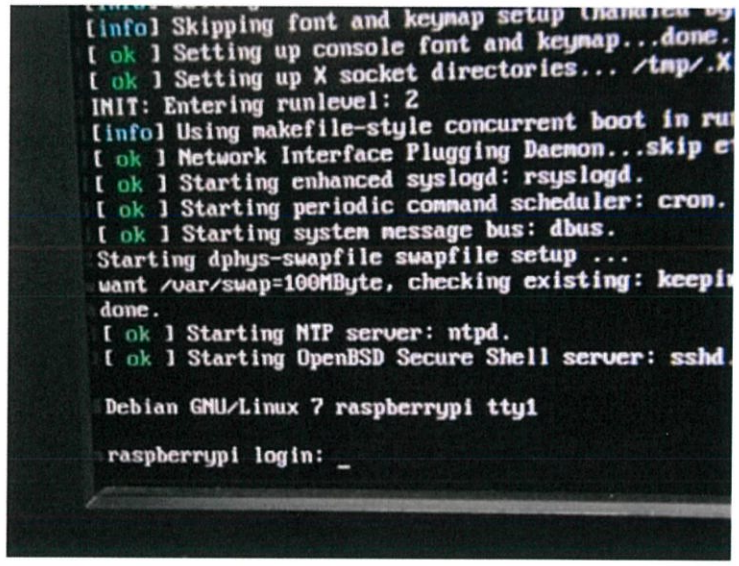
4) หลังจากนั้นจะปรากฏหน้าต่าง Raspberry Pi Software Configuration tools (แต่เนื่องจาก SD CARD ที่ทางคณะผู้จัดทำได้ทำการซื้อมานั้น ซึ่งมีระบบปฏิบัติการแล้ว จึงสามารถข้ามขั้นตอนไป ขั้นตอนต่อไปได้เลย)



รูปที่ 2.25 แสดงหน้าต่าง Raspberry Pi software Configuration tools

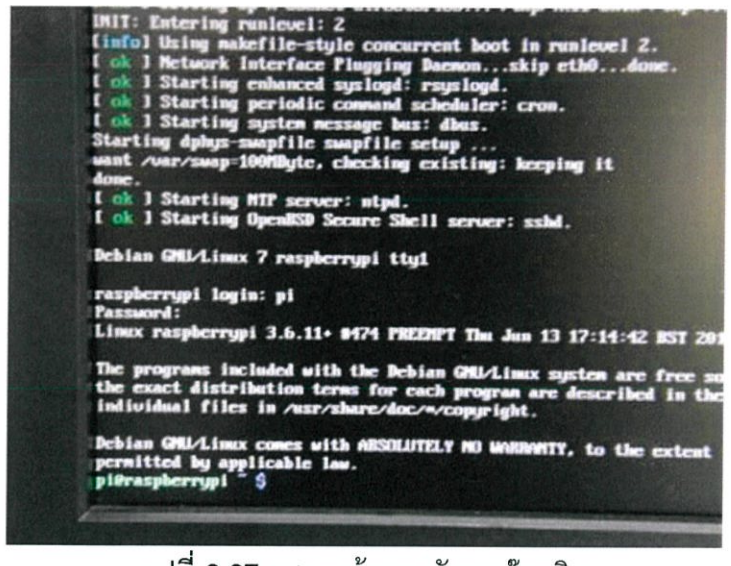
5) และหลังจากนั้นบอร์ดจะทำการรีบูตระบบใหม่และให้ใส่ล๊อคอิน Raspberry Pi login ให้ใส่ pi กดปุ่ม Enter และใส่ Password เป็น Raspberry กดปุ่ม Enter

Raspberry login: **pi**
Password: **raspberry**



รูปที่ 2.26 แสดงหน้าจอเมื่อทำการรีบูตใหม่

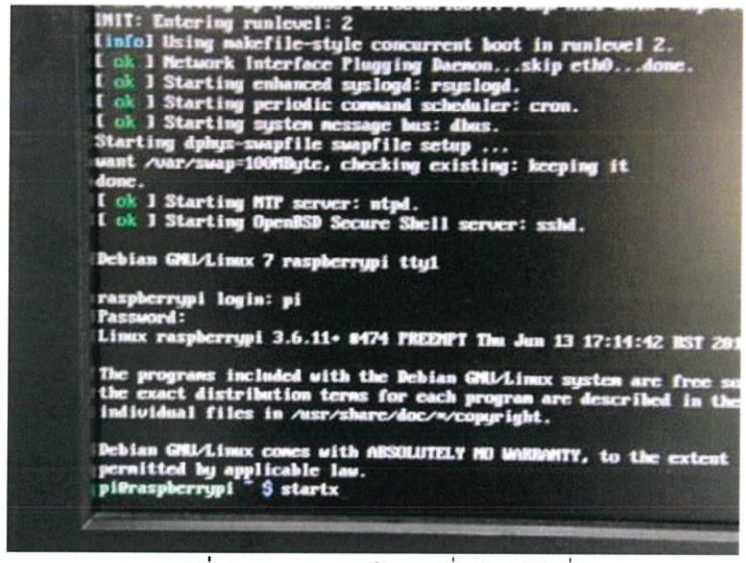
6) ถ้าการล็อคอินถูกต้อง ก็จะปรากฏข้อความดังรูป



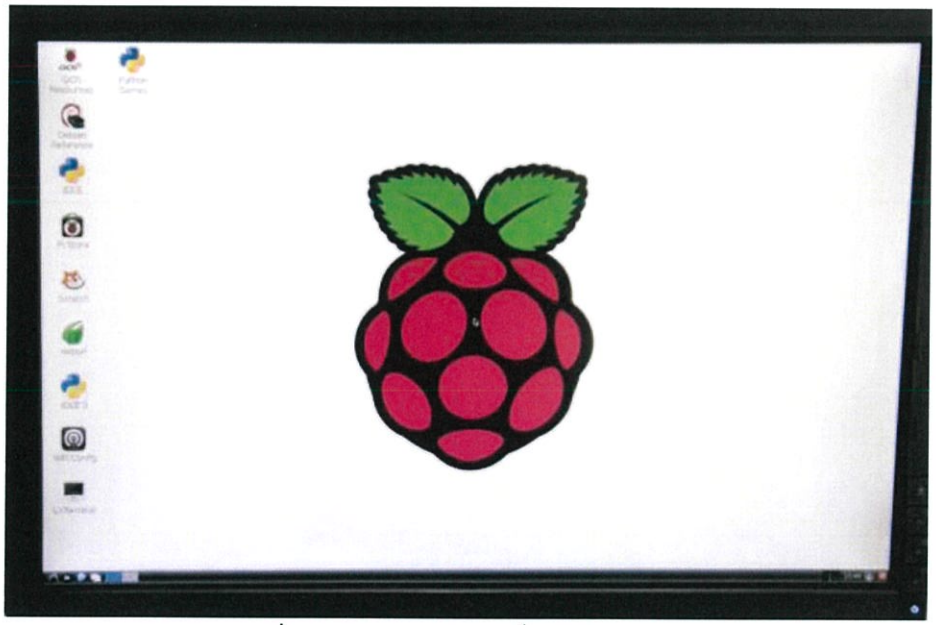
รูปที่ 2.27 แสดงหน้าจอหลังจากล็อคอิน

ปกติ Raspberry Pi software Confliguration Tool จะขึ้นมาครั้งแรกหลังจากเอา SD CARD ที่ลงระบบปฏิบัติการเสร็จที่ยังไม่ผ่านการใช้งาน มาใส่เข้าบอร์ดครั้งแรกเท่านั้น ในกรณีที่ต้องการตั้งค่าต่างๆอีกครั้ง ก็สามารถกลับไปตั้งค่าใหม่ได้อีกรอบ โดยใช้คำสั่ง sudo raspi-config

7) จากที่ผ่านมา เราสามารถเข้าระบบโหมด Command Line ได้แล้ว ถ้าต้องการใช้งานในโหมด X Window ก็สามารถทำได้โดยพิมพ์คำสั่ง startx แลกดปุ่ม Enter ซึ่งจะเข้าในโหมด X Window ดังรูป

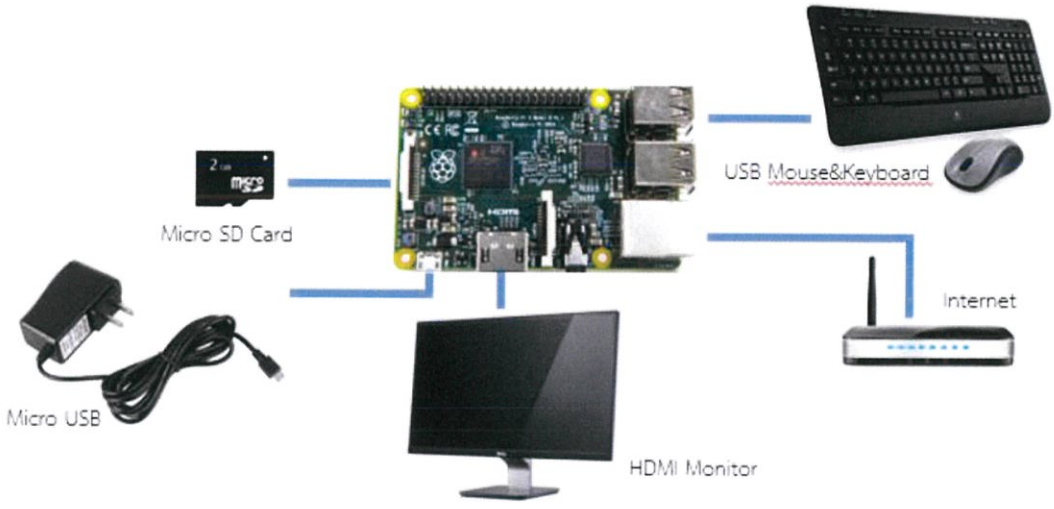


รูปที่ 2.28 แสดงหน้าจอเมื่อพิมพ์คำสั่ง



รูปที่ 2.29 แสดงหน้าจอเมื่อเข้าโหมด X Window

2.4.4 การเชื่อมต่อ



รูปที่ 2.30 แสดงแบบแผนการเชื่อมต่อของอุปกรณ์

2.5 ภาษา JavaScript

JavaScript เป็นภาษายุคใหม่สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ตที่กำลังได้รับความนิยมอย่างสูงเราสามารถเขียน โปรแกรม JavaScript เพิ่มเข้าไปในเว็ปเพจเพื่อใช้ประโยชน์สำหรับงานด้านต่างๆ ทั้งการคำนวณ การแสดงผล การรับส่งข้อมูล และที่ สำคัญคือ สามารถโต้ตอบกับผู้ใช้ได้อย่างทันทีทันใดนอกจากนี้ยังมี ความสามารถด้านอื่น ๆอีกหลายประการที่ช่วยสร้างความน่าสนใจให้กับเว็บเพจ ได้อย่างมาก ภาษาจาวาสคริปต์ถูกพัฒนาโดยเน็ตสเคปคอมมิวนิเคชันส์)Netscape Communications Corporation) โดยใช้ชื่อว่า Live Scriptออกมาพร้อมกับ Netscape Navigator 2.0 เพื่อใช้สร้างเว็บเพจโดยติดต่อกับเซิร์ฟเวอร์แบบ Live Wire ต่อมาเน็ตสเคปจึงได้ร่วมมือกับบริษัทซันไมโครซิสเต็มส์ปรับปรุงระบบของบราวเซอร์เพื่อให้สามารถติดต่อใช้งานกับภาษาจาวาได้และได้ปรับปรุง LiveScript ใหม่เมื่อ ปี แล 2538ว้ตั้งชื่อใหม่ว่า JavaScript

2.5.1 ลักษณะการทำงานของ JavaScript

JavaScript เป็นภาษาสคริปต์เชิงวัตถุหรือเรียกว่าอ็อบเจ็กโอเรียลเต็ด)Object Oriented Programming) ที่มีเป้าหมายในการออกแบบและพัฒนาโปรแกรมในระบบอินเทอร์เน็ต สำหรับผู้เขียนเอกสารด้วยภาษา HTML สามารถทำงานข้ามแพลตฟอร์มได้ทำงานร่วมกับ ภาษา HTML และภาษาจาวาได้ทั้งทางฝั่งไคลเอนต์) Client) และ ทางฝั่งเซิร์ฟเวอร์) Server) โดยมีลักษณะการทำงานดังนี้

1) Navigator JavaScript เป็น Client-Side JavaScript ซึ่งหมายถึง JavaScript ที่ถูกแปลทางฝั่งไคลเอนต์หมายถึงฝ)ั่งเครื่อง คอมพิวเตอร์ของผู้ใช้ไม่ว่าจะเป็นเครื่องพีซีเครื่องแมคอินทอช หรืออื่น ๆ(จึงมีความเหมาะสมต่อการใช้งานของผู้ใช้ทั่วไปเป็นส่วนใหญ่2) LiveWire JavaScript เป็น Server-SideJavaScript ซึ่งหมายถึง JavaScript ที่ถูกแปลทางฝั่งเซิร์ฟเวอร์หมายถึงฝั่งเครื่อง คอมพิวเตอร์ของผู้ให้บริการเว็บ โดยอาจจะเป็นเครื่องของซัน ซิลิคอมกราฟิกส์หรืออื่นๆสามารถใช้ได้ (เฉพาะกับ LiveWire ของเน็ตสเคปโดยตรง

2.5.2 JavaScript กับ HTML

การเขียน JavaScript เราอาจเขียนรวมอยู่ในไฟลเดียวกันกับ HTML ได้ซึ่งแตกต่างจากการเขียนโปรแกรมภาษา Java ที่ต้องเขียนแยกออกเป็นไฟล์ต่างหากไม่สามารถเขียนรวมอยู่ในไฟล์เดียวกับ HTML ได่วิธีการเขียน JavaScript เพื่อสั่งให้เว็ปเพจทำงาน มีอยู่ด้วยกัน วิธีดังนี้ 2

1) เขียนด้วยชุดคำสั่งและฟังก์ชันของ JavaScript เอง

2) เขียนตามเหตุการณ์ที่เกิดขึ้นตามการใช้งานจากชุดคำสั่งของ HTML

2.6 Node.js

Node.js นิยามไว้ว่ามันคือ environment เพื่อให้เราเขียน ซอฟแวร์ มารันตามที่เราต้องการภายใต้ environment ของ node นี้ โดยภาษาที่เราใช้เขียน มันคือภาษา Javascript เหมือนที่เราใช้เขียนบนหน้าเว็บ โดยจะมีตัว complier คือ google javascript engine V8 ก็คือตัวประมวลผลภาษา Javascript ที่ทาง google พัฒนาขึ้นมานั่นเอง (เป็น open source) สรุปคือเราเขียนโปรแกรมด้วยภาษา Javascript โดยความสามารถมันทำได้หลายอย่างมาก ไม่ว่าจะเป็นประมวลผลงานต่างๆ การติดต่อผ่าน socket หรือไปจนถึงสั่ง command line บนเครื่องเลย

การนำมาใช้ประโยชน์ของ Node.js

ส่วนใหญ่จะนิยมใช้ node.js ในงานที่ใช้ทำเป็นเบื้องหลัง ก็คือ งานที่ต้องประมวลผลสั่ง server ซึ่งเป็นงานที่อาจจะต้อง interface กับผู้ใช้ หรือไม่ต้อง interface เลยก็ได้ ตัวอย่างที่ต้อง interface กับคนใช้ก็คือ การทำตัวเองเป็น http server ในการดึงหน้าเว็บมาแสดงผลให้กับ user หรือว่าการเปิด socket เพื่อรับส่งข้อมูลกันระหว่าง server กับคนใช้ ซึ่งอาจจะเอาไปทำเป็นระบบที่ป้อนข้อมูลเพื่อคำนวณเอาผลลัพธ์ ในส่วนของจุดนี้จะเป็นในส่วนของการประมวลผล แต่ในส่วนของเรื่องหน้าตาก็จะเป็นหน้าที่ของ http + css + Javascript ที่เขียนหน้าเว็บตามปกติ

การใช้งาน Node.js

ซึ่งการใช้งาน node.js เวลาจะใช้งาน จะเหมือนภาษา script ที่ต้องใช้ command เพื่อสั่งให้มันทำงาน เพราะมันไม่มีตัวกลางที่คอยสร้าง request เหมือน web browser ที่ไปทำหน้าที่เรียกหาเว็บต่างๆ

ดังนั้นคนที่จะใช้งาน node.js จำเป็นที่จะต้องเรียนรู้เรื่องของ linux command เบื้องต้น เพื่อให้เข้าใจและเรียกใช้งานได้ โดยประมาณว่าจะใช้ command ของ linux เช่น cd, ls, ll, &, kill, nano

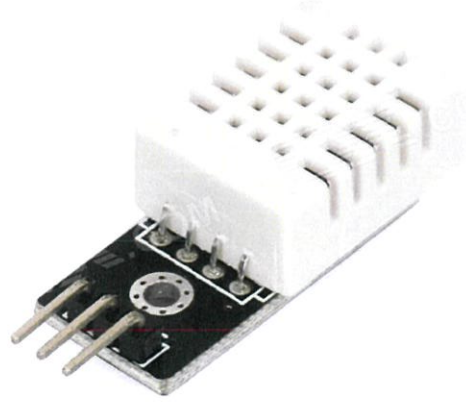
2.7 อุปกรณ์ตรวจรู้(เซนเซอร์) DHT22

อุปกรณ์ตรวจรู้สำหรับวัดอุณหภูมิและความชื้นสัมพัทธ์ (Temperature & Relative Humidity Sensor) เป็นอุปกรณ์ที่สามารถนำมาประยุกต์ใช้งานทางด้านระบบสมองกลฝังตัวได้หลากหลาย เช่น การวัดและควบคุมอุณหภูมิและความชื้น ระบบบันทึกข้อมูลเกี่ยวกับอุณหภูมิและความชื้นในห้อง เป็นต้น อุปกรณ์ประเภทนี้แตกต่างกันตามผู้ผลิต ราคา ความแม่นยำ ความละเอียดในการวัด การให้ค่าแบบดิจิทัลหรือแบบแอนะล็อก

การทดลองใช้งานโมดูล DHT22 ให้ค่าเป็นแบบดิจิทัล ใช้ขาสัญญาณดิจิทัลเพียงเส้นเดียวในการเชื่อมต่อแบบบิตอนุกรมสองทิศทาง (serial data, bi-directional) โดยนำมาเชื่อมต่อกับ Raspberry Pi เพื่ออ่านค่าจากอุปกรณ์ตรวจรู้

2.7.1 ข้อมูลเชิงเทคนิคตัวอุปกรณ์ตรวจรู้ DHT22

1) ใช้แรงดันไฟเลี้ยงได้ในช่วง: 3.3V ถึง 5.5V DC (ดังนั้นจึงใช้ได้กับ 3.3V และ 5V)
2) วัดอุณหภูมิได้ในช่วง: -40 to 80 °C (±0.5 °C accuracy)
3) วัดความชื้นสัมพัทธ์ได้ในช่วง: 0 - 100 RH% (2 - 5% accuracy)
4) อัตราการวัดสูงสุด: 0.5Hz
5) คอนเนกเตอร์แบบ 4 ขา (0.1" / 2.54mm spacing)
   Pin 1 = VCC
   Pin 2 = SDA (Serial data, bidirectional)
   Pin 3 = N.C. (Not Connect)
   Pin 4 = GND



รูปที่ 2.31 แสดงลักษณะของตัวอุปกรณ์ตรวจรู้ DHT22

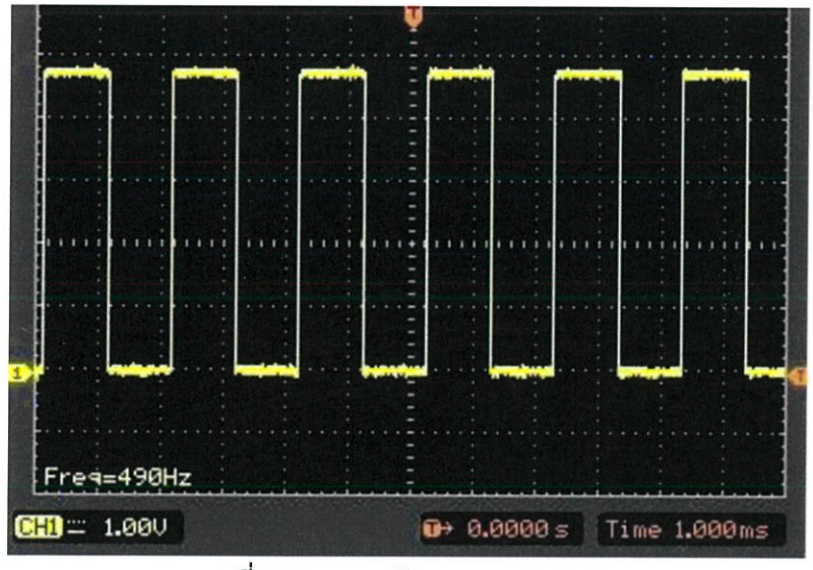
2.8 Pi-Blaster

Pi-Blaster เป็นซอฟแวร์ที่ช่วยให้ Raspberry Pi ส่ง PWM ผ่าน node.js เพื่อสั่งการไปยัง GPIO เพื่อทำให้ Raspberry Pi สามารถเพิ่มหรือลดความเร็วของมอเตอร์ ได้ตามค่าความยาวของสัญญาณไฟฟ้า

2.9 PWM

คือ การมอดูเลตสัญญาณให้มีความกว้างตามสัดส่วนที่กำหนด บนความถี่ Carrier ที่ต้องการใช้งาน ซึ่งโดยปกติแล้วจะใช้ประโยชน์ในการควบคุมการเปิดปิดของวงจรอิเล็กทรอนิกส์กำลัง เช่น วงจรบักวงจรบูส วงจรบักบูส เป็นต้น นอกจากนี้ยังใช้สามารถใช้ประกอบกับวงจร H - Bridge เพื่อควบคุมความเร็วรอบของมอเตอร์ หรือว่าแต่ต้องการจะหรี่หลอด LED ก็ได้อีกด้วย

ทีนี้มารู้จักกับพารามิเตอร์ที่ใช้ระบุรูปร่างหน้าตาของ PWM กันบ้าง ที่สำคัญมี 2 ค่าด้วยกันคือ ความถี่ของคลื่นพาหะ (Carrier Frequency) และ อัตราส่วนหน้าที่ (Duty Ratio) ซึ่งการเปลี่ยนแปลง อัตราส่วนหน้าที่ ทำให้เกิดการเปลี่ยนแปลงแรงดันไฟฟ้าเฉลี่ยที่ มอเตอร์ ทำให้เกิดความแตกต่างของความเร็วรอบของมอเตอร์



รูปที่ 2.32 แสดงลักษณะกราฟ PWM

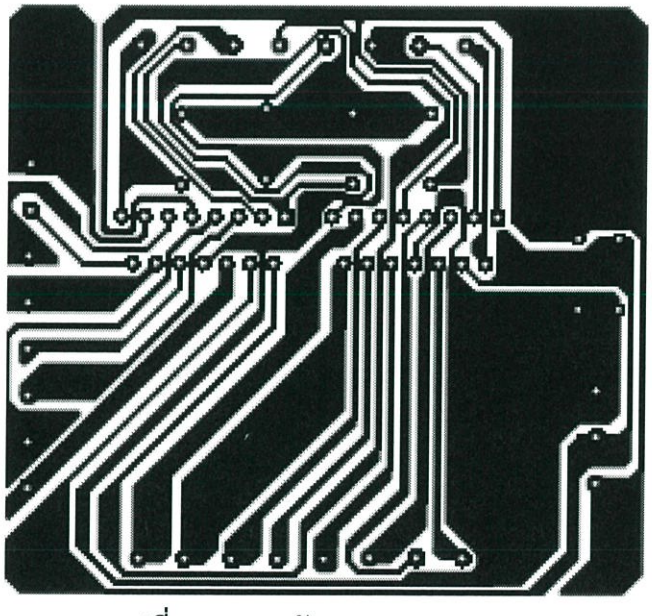
บทที่ 3
การออกแบบและการสร้าง

3.1 กล่าวนำ

ในส่วนของบทนี้ ได้กล่าวถึงส่วนของการออกแบบและการสร้าง โดยขั้นตอนต่างๆได้อ้างอิงจากทฤษฎีในบทที่2 นำมาเป็นข้อมูลประกอบการสร้างหุ่นยนต์ ซึ่งทางคณะผู้จัดทำได้ออกแบบในส่วนต่างๆ แบ่งออกเป็น ส่วนหลักๆ ได้แก่ การควบคุมวงจรมอเตอร์แบบ “Full H Bridge มอเตอร์ ” โดยทำการออกแบบแผงวงจร มอเตอร์ คอนโทรลเลอร์ ขึ้นมาเองด้วยโปรแกรม Altium Design ,การออกแบบอุปกรณ์ชิ้นงานและโครงสร้างหุ่นยนต์เพื่อเพิ่มความแม่นยำและลดข้อผิดพลาดในการสร้างโดยใช้โปรแกรม Solidworks ,การเลือกวัสดุอุปกรณ์ต่างๆเพื่อให้เหมาะสมในการสร้างหุ่นยนต์ ,การตัดชิ้นงาน รวมถึงการประกอบชิ้นส่วนต่างๆเพื่อให้เกิดโครงสร้างของหุ่นยนต์ , ออกแบบการติดต่อสื่อสารและการส่งข้อมูลระหว่างอุปกรณ์ต่างๆ

3.2 การออกแบบลาย PCB (Printed Circuit Board)

ได้ทำการออกแบบวงจรไฟฟ้าและสร้างลายวงจรบนแผ่นวงจรพิมพ์ หรือ Printed circuit board โดยโปรแกรม Altium Design เพื่อใช้สำหรับลงอุปกรณ์และบัดกรีให้ได้วงจรที่สามารถทำงานได้ และสามารถนำมาใช้งานเพื่อการควบคุมมอเตอร์ ซึ่งลักษณะที่ทำการออกแบบลาย Printed circuit board ของหุ่นยนต์สำรวจ นั้นมีลักษณะดังนี้



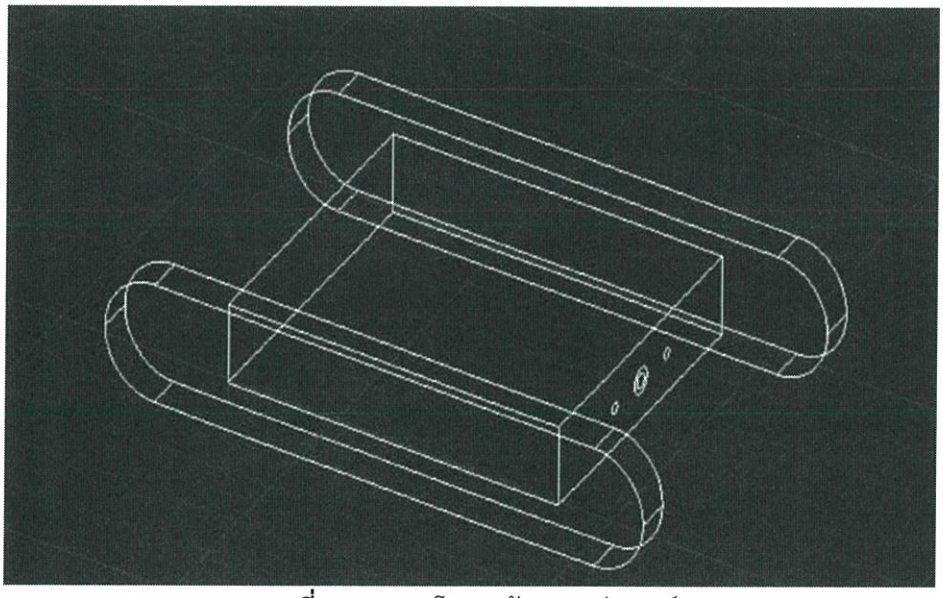
รูปที่ 3.1 แสดงลักษณะของลายวงจร

3.3 การออกแบบโครงสร้าง

การออกแบบโรงสร้างหุ่นยนต์นี้ทางคณะผู้จัดทำได้ทำการออกแบบ 2 ครั้งโดยใช้โปรแกรม AutoCad ในการออกแบบเบื้องต้น เพื่อให้ทราบขนาดของวัสดุที่เราต้องใช้แต่ละชิ้นส่วนงานต่างๆและ ใช้โปรแกรม SolidWorks ในการออกแบบโครงสร้างเพื่อความเสมือนจริงของหุ่นยนต์ ซึ่งโครงสร้างของหุ่นยนต์หลักๆที่จะทำการออกแบบประกอบไปด้วย มอเตอร์ จำนวน 2 ชิ้น, เพืองโซ่ปีก ขนาด 18 ฟัน เส้นผ่านศูนย์กลาง 30 มิลลิเมตร ขนาดรู เส้นผ่านศูนย์กลาง 8 มิลลิเมตร ทั้งหมดจำนวน 4 ชิ้น, ชิ้นส่วนต่างๆของโซ่ปีก ,แผ่นอลูมิเนียม ,ฉากอลูมิเนียม ,ตัว Wireless USB Adapter 1 ชิ้น, แบตเตอรี่ 1 ชิ้น, ตลับลูกปืนอลูมิเนียมจำนวน 4 ชิ้น, ตัว Raspberry Pi โมเดล B+ จำนวน 1 ชิ้น เมื่อทำการออกแบบแต่ละส่วนเสร็จแล้ว เราจึงทราบขนาดของแต่ละชิ้นส่วนที่ต้องการแล้ว แล้วจึงนำชิ้นส่วนงานทั้งหมดที่ได้ทำการออกแบบมาประกอบเข้าด้วยกันเป็นโครงสร้าง เพื่อดูขนาดแต่ละส่วนว่าสามารถประกอบเข้ากันได้ไหมก่อนตัดสินใจในการจัดซื้อ

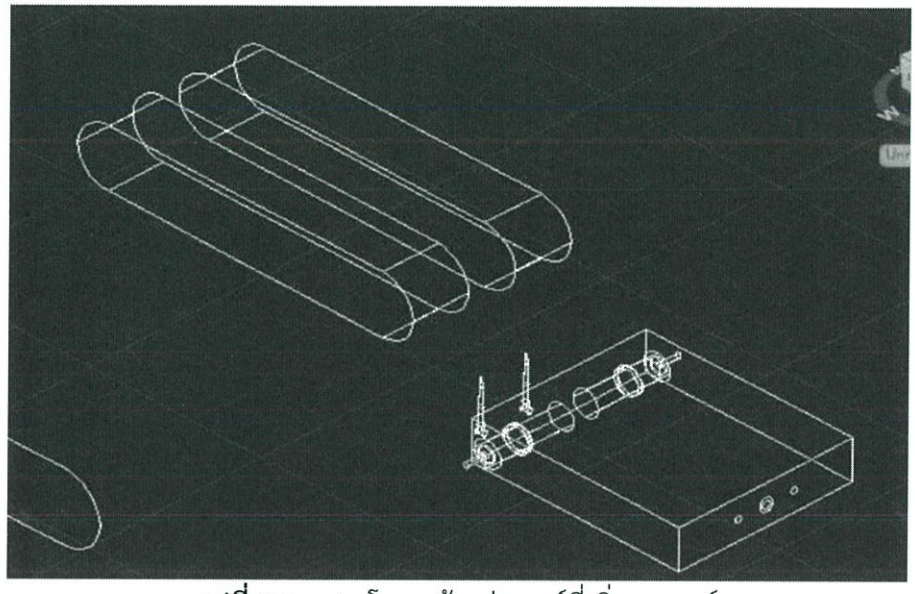
3.3.1 การออกแบบโครงสร้างเบื้องต้นด้วยโปรแกรม AutoCAD

การออกแบบครั้งแรกนั้นเราได้ทำการออกแบบเฉพาะในส่วนของโครงสร้างของตัวหุ่นยนต์เท่านั้น โดยหลักการนั้น คือ การออกแบบหุ่นยนต์ที่สามารถเคลื่อนที่ได้ไปในทุกสภาพพื้นที่ และมีโครงสร้างที่ง่ายต่อการทำ ฮาร์ดแวร์ รวมทั้งง่ายต่อการควบคุม



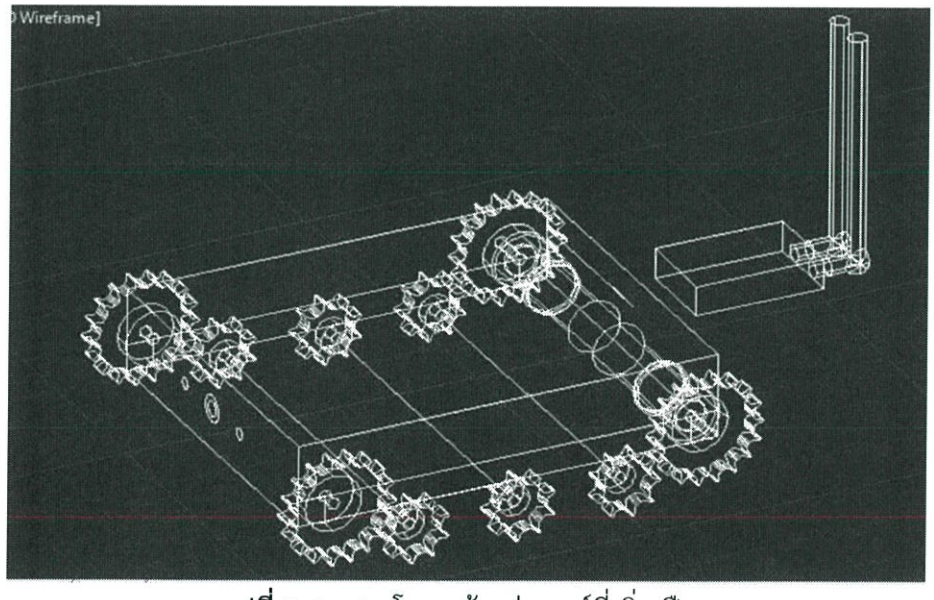
รูปที่ 3.2 แสดงโครงสร้างของหุ่นยนต์

ในการออกแบบต่อมาเราได้ทำการเพิ่มมอเตอร์ลงไปยังตัวโครงสร้างของหุ่นยนต์ เพื่อที่จะดูว่าเราสามารถใส่มอเตอร์ลงไปได้หรือไม่ โดยมีการแก้ไขตัวโครงสร้างของหุ่นยนต์เล็กน้อย โดยเพิ่มความกว้างของตัวหุ่นยนต์เพื่อที่จะให้พอดีกับมอเตอร์ทั้งสองตัว



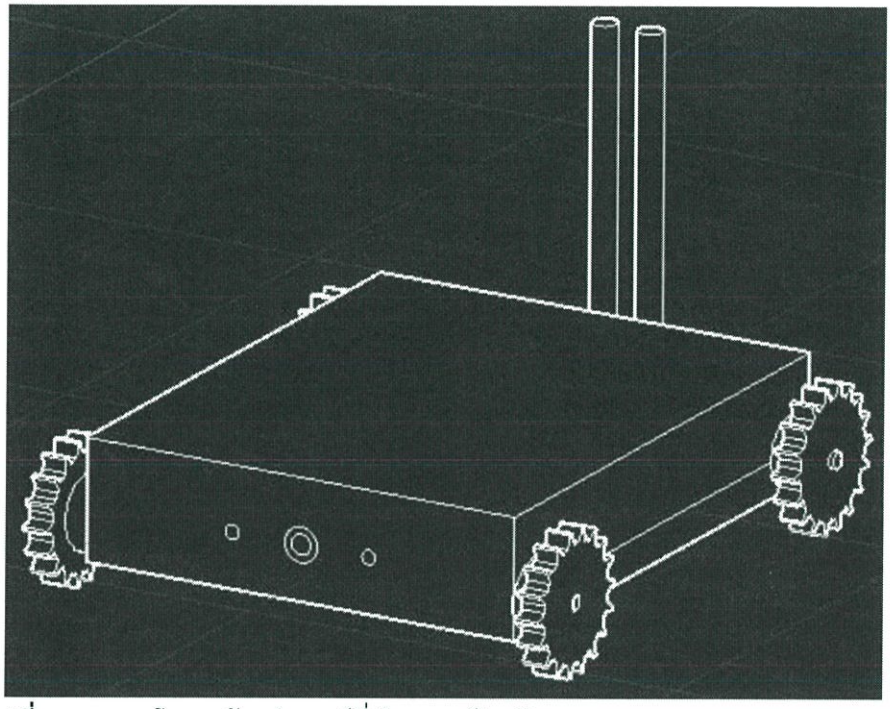
รูปที่ 3.3 แสดงโครงสร้างหุ่นยนต์ที่เพิ่มมอเตอร์

ในการออกแบบต่อมาเราได้ทำการใส่เฟืองลงไป และได้ทำการออกแบบ Wireless USB Adapter โดยออกแบบตามเสป็คของจริง และทำการออกแบบเพิ่มรูของโครงสร้างด้านข้างของตัวหุ่นยนต์เพื่อที่จะให้เพลาของมอเตอร์ต่อเข้ากับรูของเฟืองได้ แสดงดังรูป 3.3



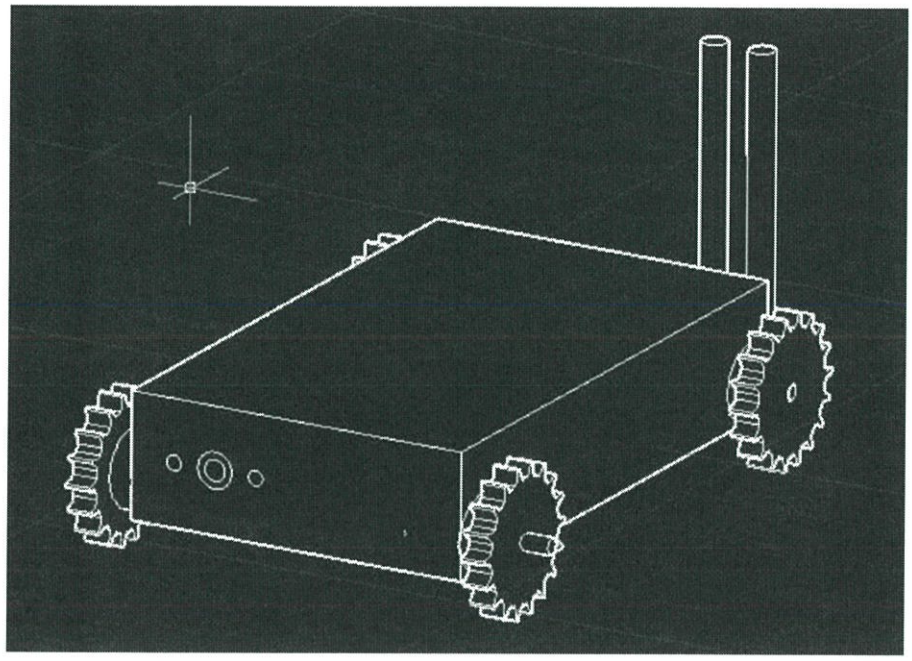
รูปที่ 3.4 แสดงโครงสร้างหุ่นยนต์ที่เพิ่มเฟือง

ต่อมานำตัว Wireless USB Adapter มาใส่ในหุ่นยนต์ดังรูป 3.4 โดยมีการปรับปรุงโครงสร้างของตัวหุ่นยนต์เพื่อให้เหมาะสมกับ Wireless USB Adapter โดยให้เสาของตัว Wireless USB Adapter ออกมาภายนอกของตัวหุ่นยนต์ และทำการยกเลิกการใช้เฟืองไซต์เล็ก ตามคำแนะนำของผู้มีประสบการณ์

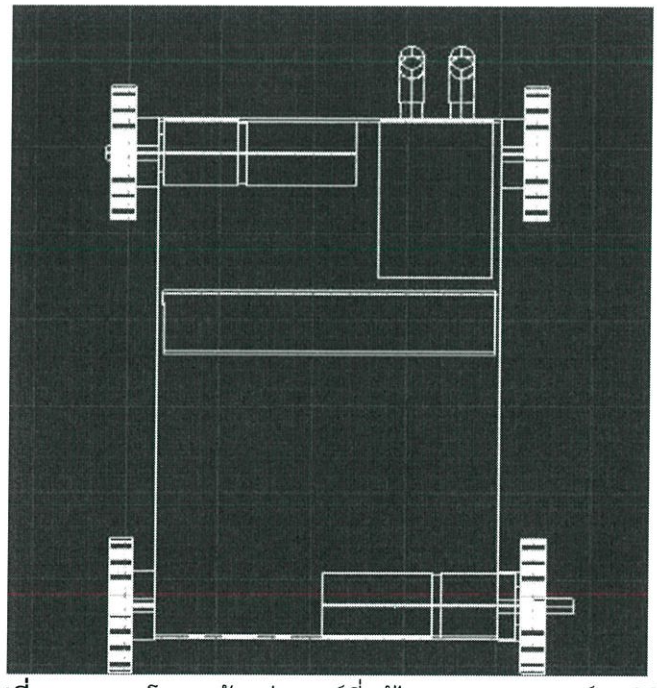


รูปที่ 3.5 แสดงโครงสร้างหุ่นยนต์ที่ทำการแก้ไขเฟือง และ Wireless USB Adapter

การออกแบบครั้งนี้ได้มีการเปลี่ยนแปลงโครงสร้างของตัวหุ่นยนต์โดยลดขนาดโครงสร้างลงโดยย้ายมอเตอร์ จากที่วางคู่กันไว้ด้านหลัง เราจึงเปลี่ยนโดยการเลือกที่จะวางมอเตอร์ตัวหนึ่งไว้ด้านหลังข้างขวา และอีกหนึ่งตัวถูกวางไว้ด้านหน้าข้างซ้ายของตัวหุ่นยนต์ โดยดูการวางมอเตอร์ได้จากรูป 3.6 อีกทั้งเราได้ทำการย้าย Wireless USB Adapter ไว้ทางซ้ายด้านหลังของตัวหุ่นยนต์



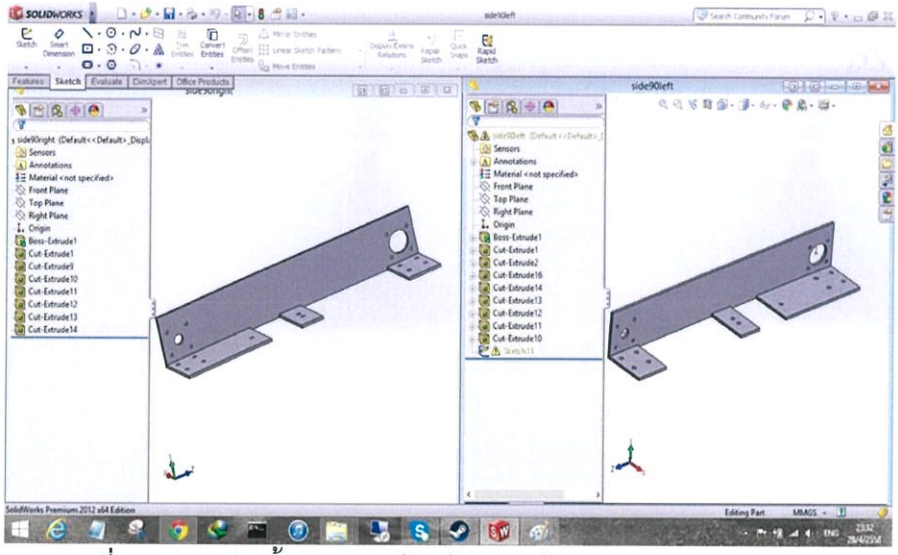
รูปที่ 3.6 แสดงโครงสร้างหุ่นยนต์ที่แก้ไขการวางมอเตอร์ 3 มิติ



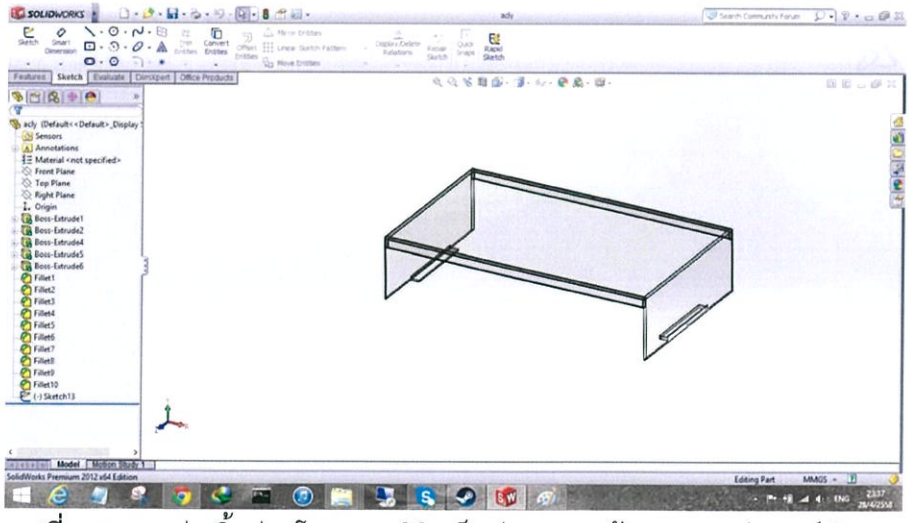
รูปที่ 3.7 แสดงโครงสร้างหุ่นยนต์ที่แก้ไขการวางมอเตอร์ 2 มิติ

3.3.2 การออกแบบโครงสร้างเบื้องต้นด้วยโปรแกรม SolidWorks

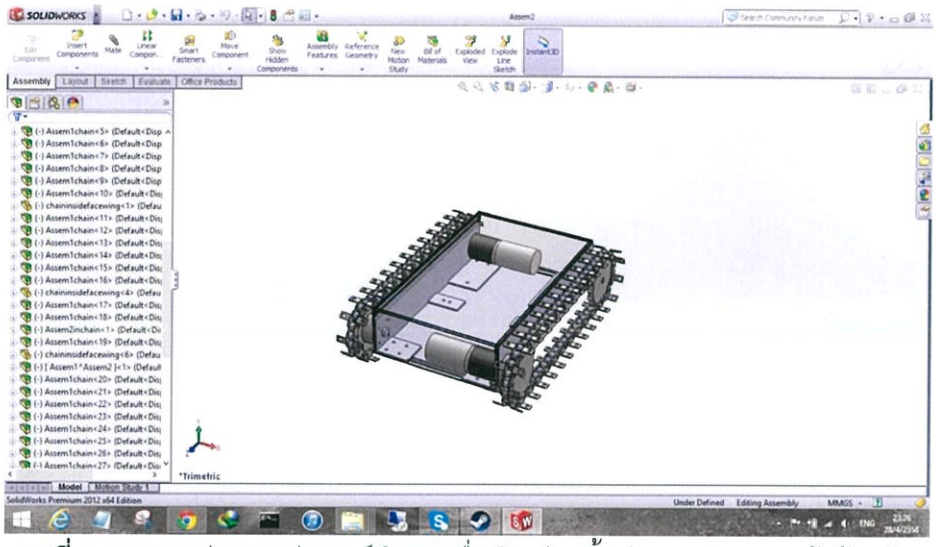
ขั้นตอนต่อไปผู้จัดทำได้ทำการออกแบบชิ้นงานบางส่วน เช่น ฉากอะลูมิเนียมทั้งสองด้าน ด้านซ้าย และด้านขวา จำนวน 2 ชิ้น และแผ่นครอบโครงสร้างอะคริลิค จำนวน 1 ชิ้น แต่เนื่องจากชิ้นงาน จำพวกมอเตอร์ ,โซ่ปีก ,เฟือง เป็นรูปแบบที่ทางผู้จัดทำไม่ต้องคิดออกแบบใหม่ เพราะได้ทำการจัดซื้อมาเป็นรูปแบบสำเร็จรูปแล้ว แต่เราจะทำการสร้างโครงร่างตามรูปแบบที่จัดซื้อนั้น เพื่อให้สามารถทำการออกแบบดูโดยรวมและนำมาใช้งานได้จริงได้ และจากนั้นนำมาประกอบเข้าด้วยกันก่อนทำการสร้างการใช้งานจริง



รูปที่ 3.8 แบบร่างชิ้นส่วนฉากด้านซ้ายและด้านขวาของหุ่นยนต์สำรวจ



รูปที่ 3.9 แบบร่างชิ้นส่วนโครงอะคริลิคเป็นส่วนครอบด้านบนของหุ่นยนต์สำรวจ



รูปที่ 3.10 แบบร่างของหุ่นยนต์สำรวจเมื่อนำแต่ละชิ้นส่วนมาประกอบเข้าด้วยกัน

3.4 รายการอุปกรณ์ที่ใช้ในการสร้างหุ่นยนต์

การเลือกใช้วัสดุอุปกรณ์ต่างๆทางคณะผู้จัดทำได้ทำการออกแบบเลือกวัสดุอุปกรณ์ที่ทนทานและเหมาะสมกับโครงสร้างที่จะใช้ในการประกอบหุ่นยนต์ ซึ่งในแต่ละชิ้นส่วนของอุปกรณ์ชิ้นงาน ทางคณะผู้จัดทำได้ทำการออกแบบร่างที่ได้จากโปรแกรม Solidworks ก่อนที่จะตัดสินใจจัดซื้อหรือตัดชิ้นส่วนอุปกรณ์ อย่างเช่น แผ่นเหล็กอะลูมิเนียม และแผ่นอะคริลิค

ทางคณะผู้จัดทำจึงต้องได้ทำการจัดซื้ออุปกรณ์ต่างๆที่จำเป็นต้องใช้ เพื่อการดำเนินงานอย่างเหมาะสม

รูปที่ 3.11 แสดงการดำเนินการจัดซื้ออุปกรณ์

3.4.1 อุปกรณ์ที่ได้มาจากการทำการจัดซื้อ

มีดังนี้

1) DC มอเตอร์ 12 โวลต์ 400rpm จำนวน 2 ชิ้น

รูปที่ 3.12 แสดงรูปมอเตอร์

2) USB 2.0 WIFI Dongle for Raspberry Pi
   - Chipset : **Ralink RT5370**
   - Wireless Speed **:** 11n: Up to 150Mbps
   - Frequency Range : 2.4-2.4835GHz

รูปที่ 3.13 แสดงรูป Wireless

3) โซ่ปีกและข้อต่อ

ขนาดความยาว 1 เมตร 50 เซนติเมตร จำนวน 2 ชิ้น

รูปที่ 3.14 แสดงรูปโซ่ปีกและข้อต่อ

4) เฟืองโซ่ปีก

ขนาดจำนวน 18 ฟัน ขนาดเส้นผ่านศูนย์กลาง 80 มิลลิเมตร และ ขนาดเส้นผ่านศูนย์กลางรูเล็กขนาด 8 มิลลิเมตร จำนวนทั้งหมด 4 อัน

รูปที่ 3.15 แสดงรูปเฟืองโซ่ปีก

5) แบตเตอร์รี่ ยี่ห้อ SPA

- Cell Per Unit : 6
- Voltage Per Unit : 12
- น้ำหนัก : 0.9 กิโลกรัม
- ขนาด : 178*35*61*67 มิลลิเมตร
- Internal Resistance : Appox. 50 มิลลิโอห์ม

รูปที่ 3.16 แสดงรูปแบตเตอร์รี่

6) Charger

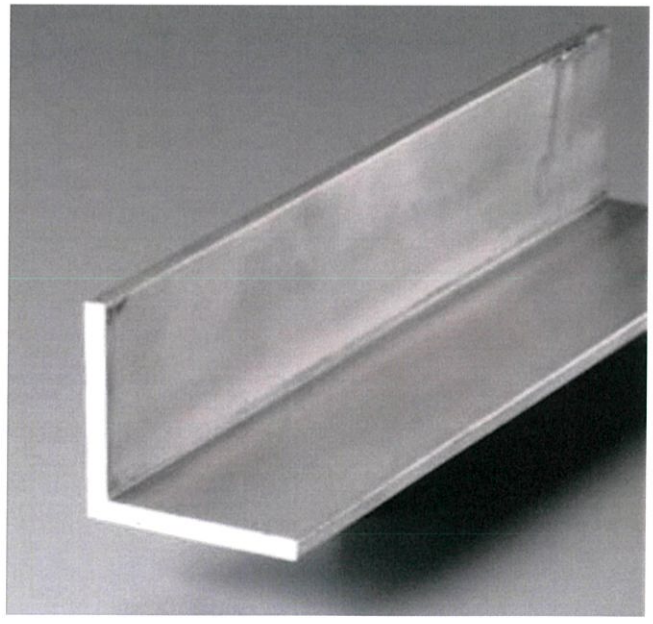
จ่ายไฟ 12v 2.3ah

รูปที่ 3.17 แสดงรูป Charger

7) ตลับลูกปืนอะลูมิเนียม
   - ขนาดรู 8 มิลลิเมตร
   - เจาะต๊าปเกลียว M.4

รูปที่ 3.18 แสดงรูปตลับลูกปืนอะลูมิเนียม

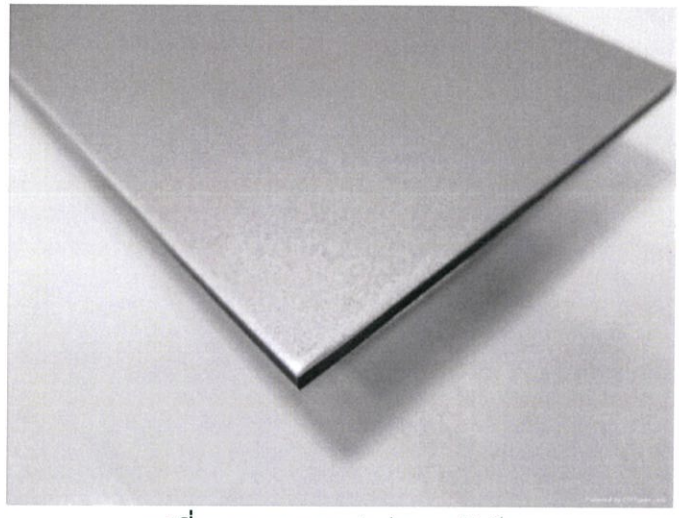
8) ฉากอะลูมิเนียม
   ความหนา 3 มิลลิเมตร และขนาดของหน้ากว้าง 3 นิ้ว



รูปที่ 3.19 แสดงรูปฉากอะลูมิเนียม

9) แผ่นอะลูมิเนียม

ขนาดความหนา 1.5 มิลลิเมตร



รูปที่ 3.20 แสดงรูปแผ่นอะลูมิเนียม

10) เพลา

ขนาด 8 มิลลิเมตร

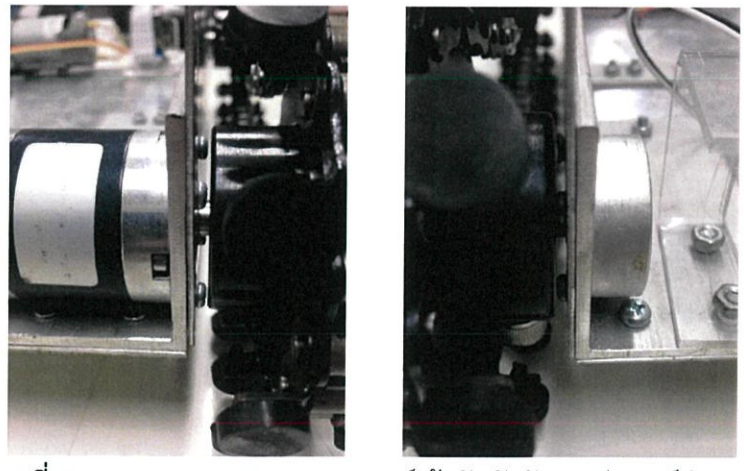
รูปที่ 3.21 แสดงรูปเพลา

3.5 การประกอบชิ้นส่วน

ขั้นตอนแรกเป็นการประกอบชิ้นส่วนของตัวโครงสร้างหุ่นยนต์ ซึ่งโครงสร้างหลักๆจะเป็นอลูมิเนียม ซึ่งการการตัดด้วยเครื่องตัดแผ่นเหล็ก ให้ได้ขนาดโครงสร้างรูปทรงตามที่ออกแบบไว้ และตัวโครงส่วนของด้วนบนซึ่งคลอบคลุมทั้งหมด ส่วนใหญ่จะใช้แผ่นอะคริลิคเป็นหลัก โดยผ่านการตัดและดัดงอให้ได้ตามขนาดที่ต้องการแล้ว โดยแต่ละส่วนจะมีการประกอบและยึดเข้ากันโดยใช้น๊อตขนาด 3 มิลลิเมตร และ 4 มิลลิเมตร

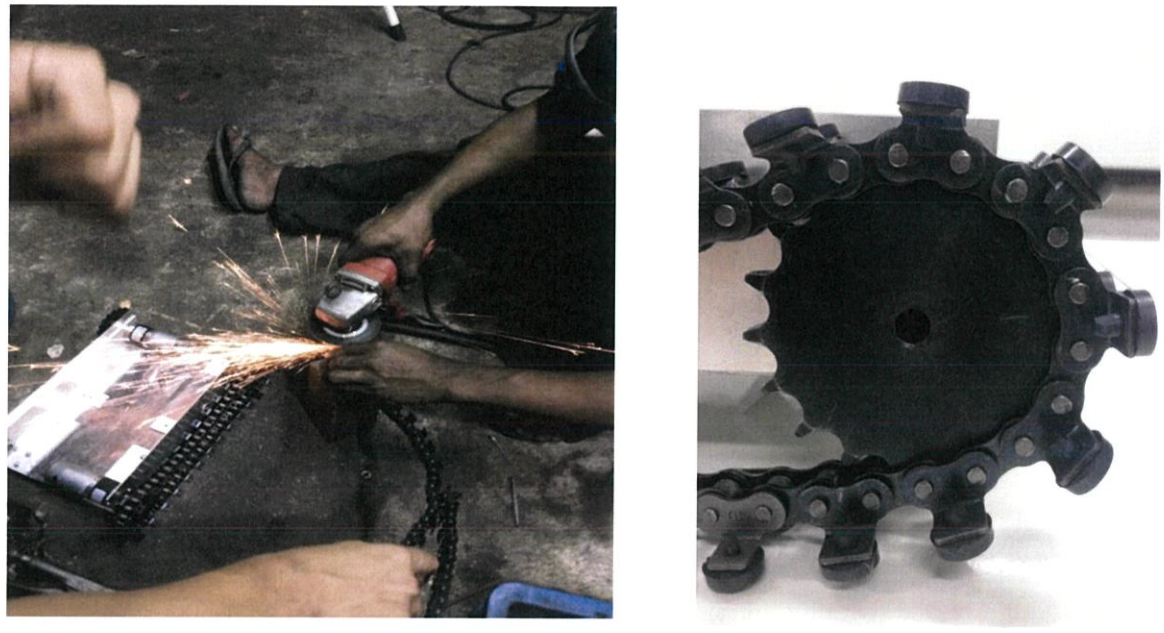
รูปที่ 3.22 แสดงการเจาะและการการตัดอุปกรณ์ชิ้นส่วนต่างๆ

ขั้นตอนสองเป็นการประกอบมอเตอร์เข้ากับตัวถังโดยใช้น๊อตขนาด 3 มิลลิเมตร ยึดตัวถังด้านข้างและฉาก เข้ากับมอเตอร์และลูกปืน ทั้งสองข้าง



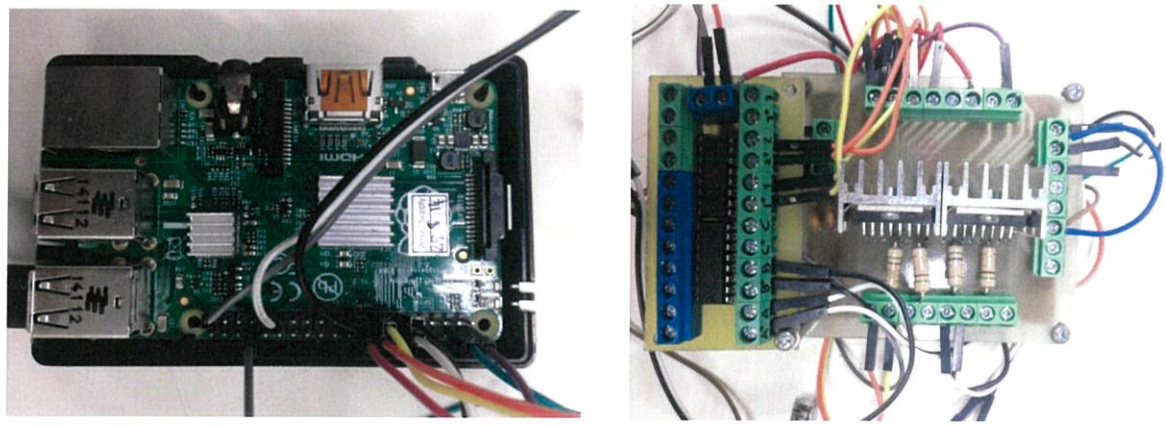
รูปที่ 3.23 แสดงการประกอบมอเตอร์เข้ากับตัวถังของหุ่นยนต์สำรวจ

ขั้นตอนที่สาม ตัดโซ่ให้ได้ความยาว ที่สัมพันธ์กับเฟืองและความยาวตัวถัง จากนั้นนำโซ่ที่ได้ไปขบกับเฟืองหน้าและเฟืองหลัง จากนั้นก็จะระบบขับเคลื่อนรูปแบบตีนตะขาบ

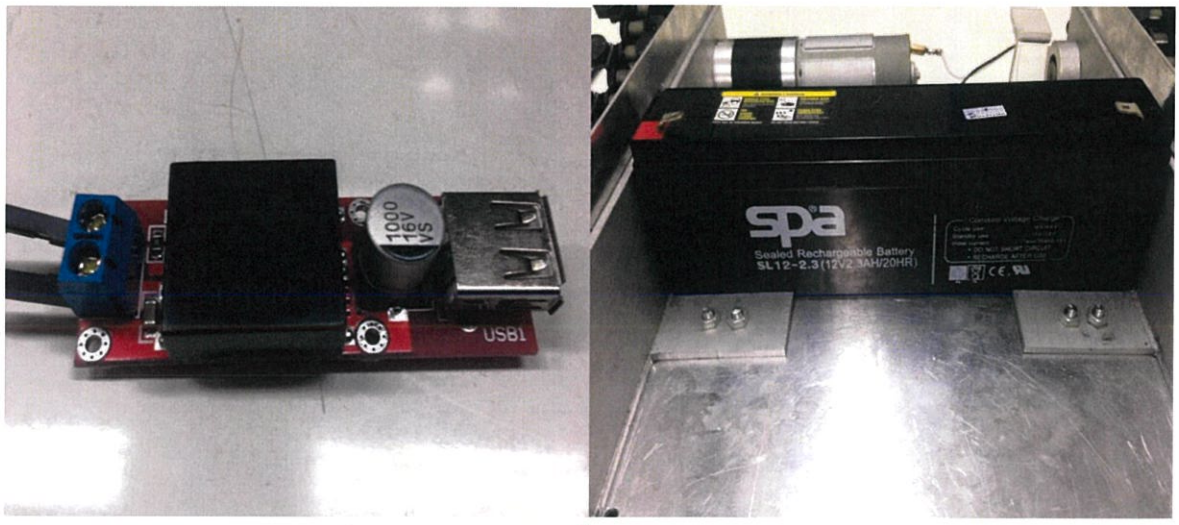


รูปที่ 3.24 แสดงการตัดโซ่ให้ได้ขนาดตามที่ต้องการ

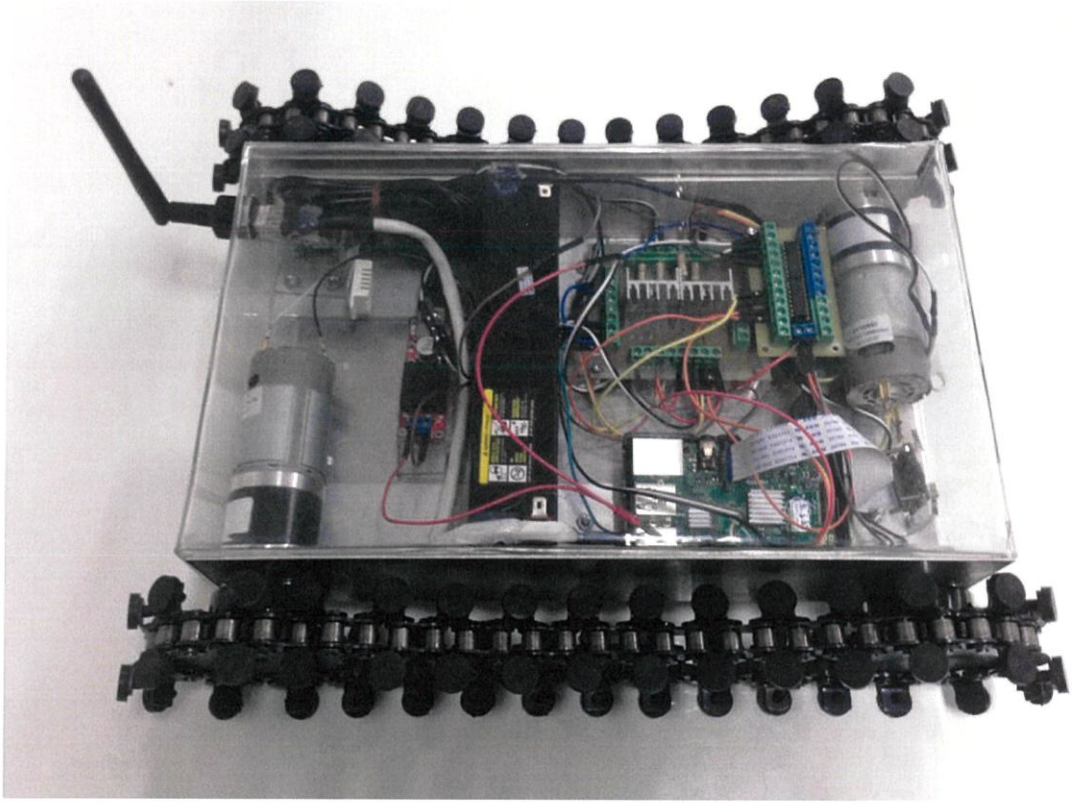
ขั้นตอนที่สี่ ทำการติดตั้งและเชื่อมต่อสายบอร์ด Raspberry Pi, มอเตอร์ คอนโทรลเลอร์, DC to DC stepdown converter, Battery, DHT22, กล้อง และ Buffer



รูปที่ 3.25 ทำการติดตั้งเชื่อมต่อสายไฟกับอุปกรณ์ควบคุมของหุ่นยนต์สำรวจ

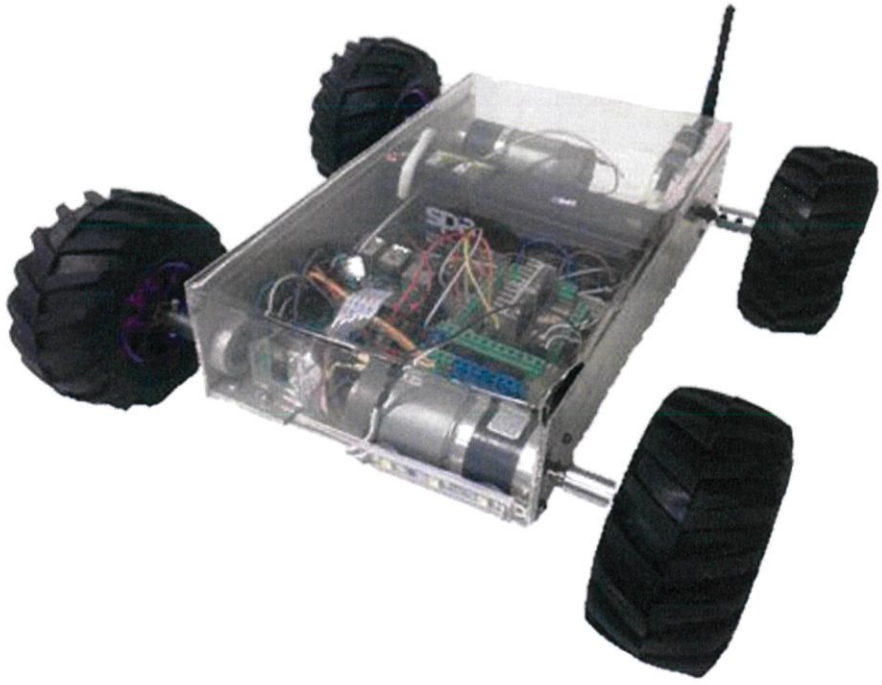


รูปที่ 3.26 แสดงการติดตั้งอุปกรณ์ต่างๆของหุ่นยนต์



รูปที่ 3.27 แสดงโครงสร้างของหุ่นยนต์สำรวจที่ประกอบเสร็จเรียบร้อย

รูปที่ 3.28 ล้อขนาด 4 นิ้ว สำหรับเปลี่ยนหุ่นยนต์เป็นรูปแบบใช้ล้อ



รูปที่ 3.29 หุ่นยนต์ในรูปแบบใช้ล้อ

3.6 การเขียนโปรแกรมข้อมูลบอร์ด Raspberry Pi โมเดล B+

ทางคณะผู้จัดทำได้ทำการเขียนข้อมูลในส่วนของซอฟแวร์ เพื่อให้หุ่นยนต์สามารถทำงานได้ตามเป้าหมาย โดยจะมีขั้นตอนการเขียนดังต่อไปนี้

3.6.1 ศึกษาและทำการติดตั้ง Node.js

Node.js คือ การเขียนโปรแกรมด้วย JavaScript ที่ฝั่ง server แทนที่ปกติแล้วจะเป็นฝั่ง client แต่จริงๆ แล้ว Node.js นั้นจะรวมไปถึง environment ต่างๆ ที่ทำขึ้นเพื่อให้เราเขียน JavaScript เอาไว้ที่ฝั่ง server ได้ด้วย (webserver, runtime และอื่นๆ) platform อย่างหนึ่ง ในโครงการนี้จะใช้เป็นเว็บไซต์ให้กับผู้ใช้งาน นอกจากนี้ Node.js ยังรับข้อความเกี่ยวกับการควบคุมและเปิดใช้งานพอร์ต GPIO บน Raspberry Pi ด้วย

ติดตั้ง Node.js บน Raspberry Pi

1) ดาวน์โหลดแพคเกต Node.js สำหรับ ARM

ขณะนี้ต้องคำนึงถึง Raspberry Pi ชิป ARM11 ดังนั้นแพคเกตที่เหมาะสำหรับ ARM จะมีให้ดาวน์โหลด ก็สามารถทำได้โดยการป้อนคำสั่ง

wget http://node-arm.herokuapp.com/node_latest_armhf.deb

เข้าไปใน terminal

2) ติดตั้งแพ็คเกต

เมื่อดาวน์โหลดเสร็จสมบูรณ์แล้ว จะต้องมีการติดตั้ง และสามารถทำได้โดยการใช้คำสั่งต่อไปนี้

sudo dpkg -i node_latest_armhf.deb

เมื่อทำเสร็จแล้ว จะมี Node.js และ NPM (Node Package Manager) ถูกติดตั้งอยู่บน Raspberry Pi

3.6.2 การติดตั้งโมดูล

1) Socket.io คือ Library ที่เขียนขึ้นโดย ภาษา node.js (java script) ทำงานแบบ real-time และรองรับการทำงานหลายๆ อย่างพร้อมกันในเวลาเดียว (Asynchronous) การทำงานของ socket.io คือทำการวนลูปตลอดเวลา เพื่อคอยรับ event เข้ามาแล้วนำไปทำงานต่อ สร้างมาเพื่อทำให้การรับ server และ client เป็นเรื่องที่ง่าย method ที่เรานิยมใช้งานกันจะมี 2 ตัวคือ

- socket.on (message, callback) ใช้สำหรับรับข้อความจาก server (receive message from server.)
- socket.emit (message, args) ใช้สำหรับส่งข้อความไปที่ server (send message to server.)

การติดตั้ง

npm install socket.io

การใช้งาน socket.html

Server (app.js)

```
var app = require('http').createServer(handler)
var io = require('socket.io')(app);
var fs = require('fs');

app.listen(80);

function handler (req, res) {
  fs.readFile(__dirname + '/index.html',
  function (err, data) {
    if (err) {
      res.writeHead(500);
      return res.end('Error loading index.html')
;
    }

    res.writeHead(200);
    res.end(data);
  });
}

io.on('connection', function (socket) {
  socket.emit('news', { hello: 'world' });
  socket.on('my other event', function (data) {
    console.log(data);
  });
});
```

Client (index.html)

```
<script src="/socket.io/socket.io.js"></script>
<script>
  var socket = io('http://localhost');
  socket.on('news', function (data) {
    console.log(data);
    socket.emit('my other event', { my: 'data' }
);
  });
</script>
```

รูปที่ 3.30 ตัวอย่างคำสั่งของ socket.io

1) Node static

Node static คือ การทำ GET และ HEAD request ซึ่งได้แนวคิดมาจากบางส่วนของโมดูล static-file อื่นๆ เช่น node-paperboy and antinode

การติดตั้ง

npm install static

การใช้งาน node-static

Synopsis

```
var static = require('node-static');

//
// Create a node-static server instance to serve the './public' folder
//
var file = new static.Server('./public');

require('http').createServer(function (request, response) {
  request.addListener('end', function () {
    //
    // Serve files!
    //
    file.serve(request, response);
  }).resume();
}).listen(8080);
```

รูปที่ 3.31 ตัวอย่างคำสั่ง node-static

2) Sleep

เป็นส่วนสำคัญในการแก้ไขจุดบกพร่อง การ Sleep จะหยุดการทำงานของ Java script ทั้งหมด ในหน้าต่างโมดูลให้ถูกถอยกลับไป ใน while loop ซึ่งจะใช้การทำงานของ CPU 100 เปอร์เซนต์

การติดตั้ง

npm install Sleep

การใช้งาน Sleep

```
var sleep = require('sleep');
```

- `sleep.sleep(n)`: sleep for n seconds
- `sleep.usleep(n)`: sleep for n microseconds (1 second is 1000000 microseconds)

รูปที่ 3.32 การใช้งานคำสั่ง Sleep

3) Optimist

คือ library ของ node.js สำหรับการทำ option parsing ที่ไม่ชอบใช้ option parsing npm install Optimist

3.6.3 การสร้างและใช้งาน MJPG-Streamer บน the Raspberry Pi

1) ติดตั้งสร้างการอ้างอิง

คำสั่งต่อไปนี้จะติดตั้ง 3 libraries ที่ใช้ใน MJPG-Streamer

```
$ sudo apt-get install libjpeg8-dev imagemagick libv4l-dev
```

2) เพิ่ม missing videodev.h

ไฟล์ header ของ videodev.h ต้องการถูกแทนที่ด้วย videodev2.h เพื่อให้ MJPG-Streamer เข้าถึงได้ง่ายขึ้น คุณจะต้องสร้างลิ้งค์

```
$ sudo ln -s /usr/include/linux/videodev2.h /usr/include/linux/videodev.h
```

3) ดาวน์โหลด MJPG-Streamer

ดาวน์โหลด MJPG-Streamer โดยใช้ wget

```
$ wget http://sourceforge.net/code-snapshots/svn/m/mj/mjpg-streamer/code/mjpg-streamer-code-182.zip
```

4) ทำการแตกไฟล์ MJPG-Streamer

```
$ unzip mjpg-streamer-code-182.zip
```

5) สร้าง MJPG-Streamer

ทำการติดตั้ง MJPG-Streamer โดยใช้คำสั่ง make

```
$ cd mjpg-streamer-code-182/mjpg-streamer
$ make mjpg_streamer input_file.so output_http.so
```

6) ติดตั้ง MJPG-Streamer

คำสั่งต่อไปนี้เป็นการ copy ไฟล์ที่จำเป็นไปไว้ที่ระบบ directories

```
$ sudo cp mjpg_streamer /usr/local/bin
$ sudo cp output_http.so input_file.so /usr/local/lib/
$ sudo cp -R www /usr/local/www
```

7) การเริ่มกล้อง

ตามคำสั่งต่อไปนี้

$ mkdir /tmp/stream

$ raspistill --nopreview -w 640 -h 480 -q 5 -o /tmp/stream/pic.jpg -tl 100 -t 9999999 -th 0:0:0 &

8) เริ่ม MJPG-Streamer

ขณะนี้กล้องอยู่ในขณะเขียนภาพ ดังนั้นสิ่งที่เหลืออยู่คือการเริ่มต้น MJPG-Streame โดยเริ่มต้นตามคำสั่งต่อไปนี้

LD_LIBRARY_PATH=/usr/local/lib mjpg_streamer -i "input_file.so -f /tmp/stream -n pic.jpg" -o "output_http.so –p 8091 -w /usr/local/www"

9) ดู stream

ตอนนี้สามารถเชื่อมต่อกับเว็บเบราว์เซอร์ และดูการถ่ายทอดสด stream ถ้าหากต้องการที่จะดูใน Raspberry Pi สามารถใส่ http://localhost:8091 ในแถบที่อยู่ของเบราว์เซอร์ แต่หากถ้าต้องการที่จะดูในคอมพิวเตอร์เครื่องอื่นในการใช้งานเครือข่ายของผู้ใช้คือ http://<IP-address>:8091

3.6.4 เรียนรู้และติดตั้ง Pi-Blaster โมดูล

Pi-Blaster ช่วยให้เซิร์ฟเวอร์ Node.js ส่งสัญญาณ PWM ไปที่ทุกๆขาของ GPIO บน Raspberry Pi ซึ่ง PWM มีข้อได้เปรียบกว่าสัญญาณสูงและต่ำ เพราะมันจะช่วยให้สามารถควบคุมความเร็วของแต่ละมอเตอร์ ให้สามารถเปลี่ยนเป็นแบบไดนามิก และการเปลี่ยนแปลงความเร็ว

การติดตั้ง

sudo apt-get install autoconf

จากนั้นทำการ unzip ไฟล์ Pi-blaster-master.zip และทำตามคำสั่งนี้

./autogen.sh

./configure && make

เริ่มต้น Pi-blaster และ ให้มันมีการเปิดใช้งานใหม่โดยอัตโนมัติ ในทุกๆการรีบูต

sudo make install

การเริ่มต้นใช้งานด้วยตนเอง ตามคำสั่ง

sudo ./pi-blaster

วิธีการใช้งาน

Pi-blaster จะสร้างไฟล์พิเศษ (FIFO) ใน /dev/pi-blaster แอพพลิเคชั่นบน Raspberry Pi ของผู้ใช้สามารถเขียนมันได้ (ซึ่งหมายความว่าเฉพาะ Pi-blaster เท่านั้น ที่ต้องการมีแหล่งแอพพลิเคชั่นของผู้ใช้สามรถทำงานเป็นปกติ) เมื่อใช้ Pi-blaster ขา GPIO ที่ผู้ใช้ส่งไปนั้น มันจะถูกกำหนดค่าเป็นเอาท์พุท ในการตั้งค่าของรหัส PIN ผู้ใช้งานจะต้องเขียนคำสั่งไปยัง /dev/pi-blaster ในรูปแบบ <GPIOPinName>=<value> where <value> จะต้องเป็นหมายเลขระหว่าง 0 และ 1

GPIO number	Pin in P1 header
4	P1-7
17	P1-11
18	P1-12
21	P1-13
22	P1-15
23	P1-16
24	P1-18
25	P1-22

รูปที่ 3.33 แสดงขา GPIO ที่ใช้กำหนดเป็นค่าเอาท์พุท

ตัวอย่างการเปิดขา PWM ON

1) ในการปิดอย่างสมบูรณ์ที่ GPIO 17

```
echo "17=0" > /dev/pi-blaster
```

2) ในการเปิดอย่างสมบูรณ์ที่ GPIO 17

```
echo "17=1" > /dev/pi-blaster
```

3) ในการตั้งค่า GPIO 17 ให้ PWM 20เปอร์เซนต์

```
echo "17=0.2" > /dev/pi-blaster
```

ตัวอย่างการเปิดขา PWM OFF (ปล่อยขาเพื่อที่จะสามารถนำมาใช้เป็น GPIO ดิจิตอล)
ในการยกเลิกการทำงานของ GPIO 17 ให้ใช้คำสั่ง

```
echo "release 17" > /dev/pi-blaster
```

3.6.5 เรียนรู้และทดสอบอุปกรณ์ตรวจรู้ (Sensor) DHT22

ในขั้นตอนแรกนี้ต้องการทดสอบอุปกรณ์ตรวจรู้(Sensor) เพราะต้องการที่จะสร้างโปรแกรมประยุกต์บนพื้นฐานของ Node.js จึงใช้โหนดเพื่อติดต่อกับอุปกรณ์ตรวจรู้(Sensor) DHT22 ต้องการทำเช่นนั้นเราจึงจะใช้โมดูลโหนดเฉพาะที่มีอยู่แล้ว เมื่อผู้ใช้ได้สร้างคำสั่ง เข้าในโฟลเดอร์แล้ว จึงทำตามขั้นตอนต่อไปนี้

1) ทำการจะทดสอบคำสั่ง หลังจากดาวน์โหลดคำสั่งจาก GitHub ไปที่โฟลเดอร์ sensors_test

```
sudo npm install node-dht-sensor
```

2) ให้เวลาสักครู่ หลังจากที่มันโหลด จึงทำการใส่คำสั่ง

```
sudo node sensors_test.js
```

3) จะเห็นค่าของอุณหภูมิและความชื้นใน terminal

```
Temperature: 21.00C, humidity: 35.00%
```

3.6.6 สร้าง nodeserver

มี 4 สคริปหลักในไดเร็กทอรี่ที่ใช้ในการเริ่มต้นซอฟต์แวร์ที่จำเป็นสำหรับการเริ่มต้นให้หุ่นยนต์ทำงาน มีดังนี้

1) Node.js
2) Socket.html
3) Jquery-2.0.3.min.js
4) Virtualjoystick.js

3.6.7 วิธีการทำงานของซอฟแวร์

หุ่นยนต์จะใช้ในส่วนของซอฟแวร์ ดังต่อไปนี้

1) Node.js ทำหน้าที่เป็นหน้าเว็บให้กับผู้ใช้ นอกจากนี้ยังตอบรับข้อความเกี่ยวกับการควบคุมและเปิดใช้งานพอร์ต GPIO บน Raspberry Pi (เข้าถึงเซิร์ฟเวอร์ผ่านพอร์ต 8090)

2) mjpg-streamer ทำหน้าที่ให้สตรีมมิ่งรูปถ่ายภาพที่ถูกฝังอยู่ในหน้าเว็บที่ให้บริการให้กับผู้ใช้ จะช่วยให้ผู้ใช้สามารถดูผ่านเว็บแคมที่ติดอยู่กับหุ่นยนต์ (สตรีมผ่านพอร์ต 8091)

3) pi-blaster ช่วยให้เซิร์ฟเวอร์ Node.js ส่งสัญญาณ PWM ไปที่ขา GPIO บน Raspberry Pi ซึ่ง PWM มีข้อได้เปรียบกว่าสัญญาณสูงและต่ำ เพราะมันจะช่วยให้สามรถควบคุมความเร็วของแต่ละมอเตอร์ ให้สามารถเปลี่ยนเป็นแบบไดนามิก และการเปลี่ยนแปลงความเร็ว

การติดตั้งทางกายภาพของหุ่นยนต์

หุ่นยนต์ใช้ GPIO 4 ขาสำหรับการส่งออกไปมอเตอร์ และพอร์ตกราวน์บน Raspberry ขา GPIO ที่ใช้มีดังนี้

มอเตอร์ขวา ใช้ขา 17,18
มอเตอร์ซ้าย ใช้ขา 22,23
อุปกรณ์ตรวจรู้(Sensor) DHT22 ใช้ขา 5 (data)
หลอดไฟ LED ใช้ขา 6

3.6.8 ขั้นตอนการเริ่มต้นหุ่นยนต์

เริ่มใช้งานกล้อง

1) mkdir /tmp/stream

2) raspistill --nopreview -w 640 -h 480 -q 5 -o /tmp/stream/pic.jpg -tl 100 -t 9999999 -th 0:0:0

เริ่มใช้งานกล้องเซิร์ฟเวอร์

3) LD_LIBRARY_PATH=/usr/local/lib mjpg_streamer -i "input_file.so -f /tmp/stream -n pic.jpg" -o "output_http.so -p 8091 -w /usr/local/www"

เริ่มใช้งาน Pi-blaster

4) cd /home/pi/Desktop/pi-blaster

5) sudo ./pi-blaster

เริ่มใช้งานโหนดเซิร์ฟเวอร์

cd /home/pi/Raspberry-Pi-Car/nodeserver

sudo node app.js

3.6.9 การกำหนด Hostapd

ขั้นตอนแรกต้องมีการกำหนดค่า โดยการสร้างไฟล์ต่อไปนี้

1) sudo nano /etc/hostapd/hostapd.conf

ซึ่งมีคำสั่ง ดังต่อไปนี้

interface=wlan0

ssid=RPIrobot

channel=1

wmm_enabled=1

wpa=1

wpa_passphrase=12345678910

wpa_key_mgmt=WPA-PSK

wpa_pairwise=TKIP

rsn_pairwise=CCMP

auth_algs=1

macaddr_acl=0

ขั้นตอนต่อไป ต้องการใช้เป็นการ์ดไร้สาย และ rPi ที่จะจัดการกับการกำหนดเส้นทางและDHCP ดังนั้นเราอาจจะไม่สามารถเชื่อมต่อกับจุดเชื่อมต่อ แต่จะไม่ได้รับ IP จึงจำเป็นต้องกำหนดเครือข่ายย่อยสำหรับการ์ดไร้สาย ตามคำสั่งดังนี้

2) sudo nano /etc/network/interfaces

และเพิ่มคำสั่งอีก 3 ขั้นตอนดังต่อไปนี้

iface wlan0 inet static

address 10.10.0.1

netmask 255.255.255.0

ขั้นตอนต่อไป ต้องเปิดใช้งาน DHCP บน Pi สำหรับเครือข่ายนี้เพื่อให้อุปกรณ์ใดๆที่เชื่อมต่อ จะได้รับ IP address ตามคำสั่งดังต่อไปนี้

3) sudo apt-get install isc-dhcp-server

4) sudo nano /etc/dhcp/dhcpd.conf

และให้แน่ใจว่ามีคำสั่งดังต่อไปนี้ด้วย

authoritative; #be careful with this setting

ddns-update-style none;

default-lease-time 600;

max-lease-time 7200;

log-facility local7;

#for the wireless network on wlan0

subnet 10.10.0.0 netmask 255.255.255.0 {

range 10.10.0.25 10.10.0.50;

option domain-name-servers 8.8.8.8, 8.8.4.4;

option routers 10.10.0.1;

interface wlan0;

ต่อไปจึงเริ่มต้นใหม่ในการทดสอบ HostAPD และ DHCP โดนรันดังคำสั่งต่อไปนี้

5) sudo reboot

หลังจาก Raspberry Pi กลับมา เริ่มต้น hostapd ในพื้นหลัง ต่อไปก็รีบูตเครื่องเซิฟเวอร์ DHCP โดยรันตามคำสั่งต่อไปนี้

6) sudo hostapd -B /etc/hostapd/hostapd.conf

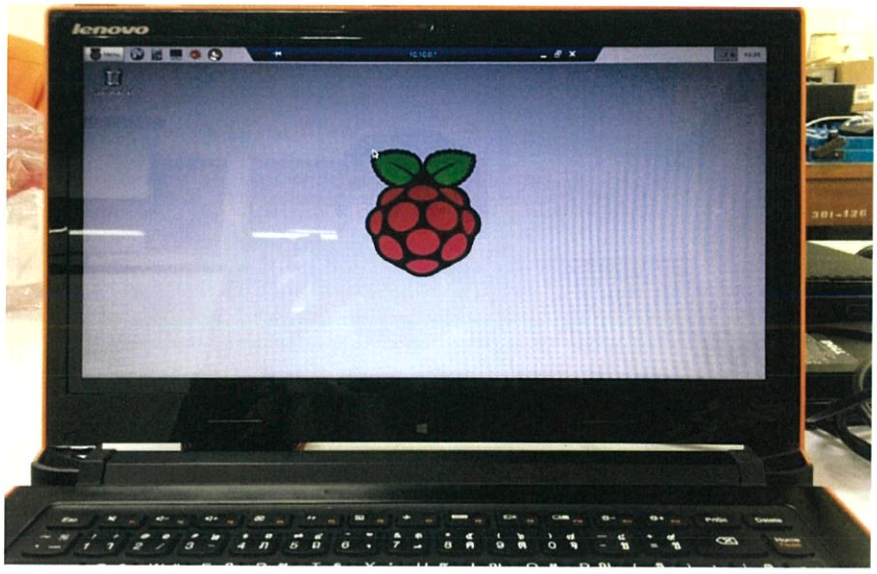
7) sudo /etc/init.d/isc-dhcp-server restart

เมื่อทำการเชื่อมต่อแบบไร้สายได้แล้ว เราต้องเชื่อมต่อไร้สายนี้โดยการบูต ที่มีคำสั่งต่อไปนี้

8) sudo nano /etc/default/hostapd

ต่อไปทำการอัพเดท โดยคำสั่งต่อไปนี้

DAEMON_CONF="/etc/hostapd/hostapd.conf"



รูปที่ 3.34 แสดงการเปิดเครื่อง หลังจากกำหนดค่า Hostapd

บทที่ 4
การทดลอง

4.1 กล่าวนำ

การทดลองควบคุมหุ่นยนต์ สมาร์ทดีไวซ์ ผ่านระบบการเชื่อมต่อแบบไร้สายนั้น จะประกอบไปด้วย การทดลองการเดิน ,การเลี้ยวซ้าย ,การเลี้ยวขวา ,การเคลื่อนที่ในสภาพแวดล้อมต่างๆที่แตกต่างกัน , การทดลองตรวจจับอุณหภูมิ และความชื้น โดยจะแสดงในลักษณะของค่าจริงในสภาพแวดล้อมนั้นๆ ,การแสดงภาพการเคลื่อนไหวในรูปแบบลักษณะ ทันที ซึ่งค่าต่างๆที่ได้จะออกมาในรูปแบบที่แสดงบน สมาร์ทดีไวซ์

4.2 ขั้นตอนการทดลอง

4.2.1 เปิดโปรแกรมและป้อนคำสั่ง

ขั้นตอนแรกทำการสร้างไฟล์โดยใช้คำสั่ง sudo nano /home/pi/Raspberry-pi-car/start1.sh และใส่คำสั่งต่อไปนี้ เพื่อทำการสร้างไฟล์ start1.sh ซึ่งเป็นคำสั่งการเริ่มต้นการทำงานของกล้อง โดยมีคำสั่งการใช้งานในไฟล์ start1.sh ดังนี้

1) mkdir /tmp/stream

2) raspistill --nopreview -w 640 -h 480 -q 5 -o /tmp/stream/pic.jpg -tl 100 -t 9999999 -th 0:0:0 &

ขั้นตอนสองทำการสร้างไฟล์โดยใช้คำสั่ง sudo nano /home/pi/Raspberry-pi-car/start/2.shและใส่คำสั่งต่อไปนี้ ทำการสร้างไฟล์ที่สองเป็นไฟล์ star2.sh ซึ่งเป็นคำสั่งการเริ่มการใช้งานเซิฟเวอร์ของวิดีโอ โดยมีคำสั่งการใช้งานในไฟล์ start2.sh ดังนี้

3) LD_LIBRARY_PATH=/usr/local/lib mjpg_streamer -i "input_file.so -f /tmp/stream -n pic.jpg" -o "output_http.so -p 8091 -w /usr/local/www"

ขั้นตอนสามทำการสร้างไฟล์โดยใช้คำสั่ง sudo nano /home/pi/Raspberry-pi-car/start/3.shและใส่คำสั่งต่อไปนี้ ทำการสร้างไฟล์ที่สามเป็นไฟล์ start3.sh ซึ่งเป็นคำสั่งเริ่มต้นการใช้งาน PI-BLASTER และการเริ่มการใช้งานเซฟเวอร์ของ node โดยมีคำสั่งการใช้งานในไฟล์ start3.sh ดังนี้

4) cd /home/pi/Desktop/pi-blaster

5) sudo ./pi-blaster

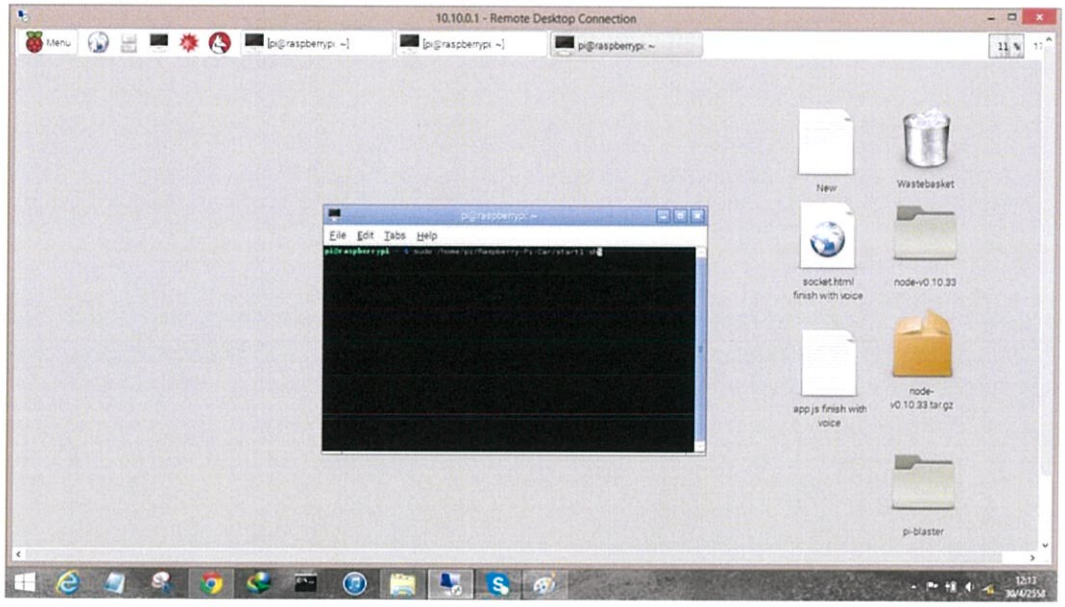
6) cd /home/pi/Raspberry-Pi-Car/nodeserver

7) sudo node app.js

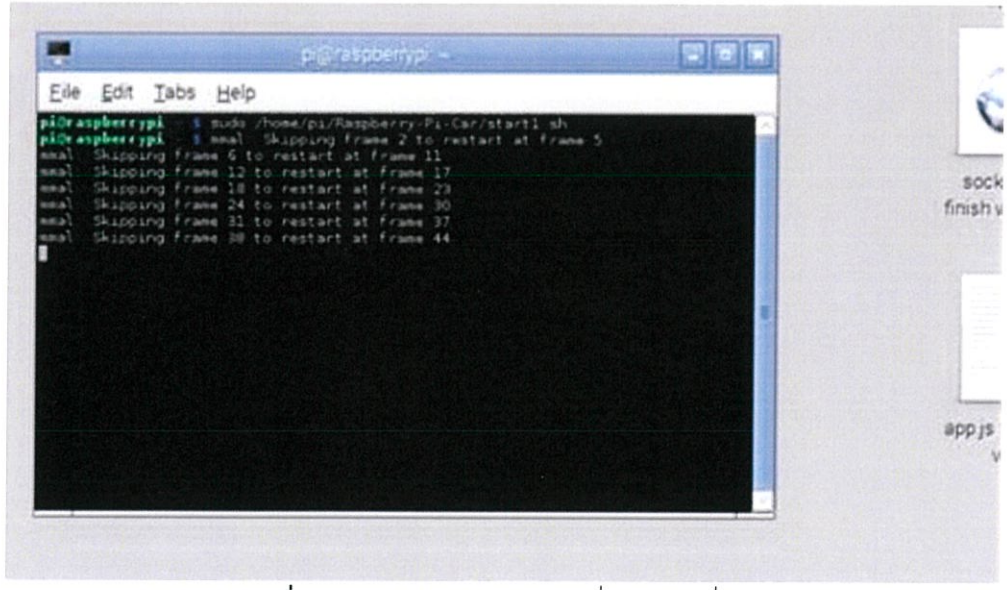
จากนั้นขั้นตอนต่อมาทำการป้อนคำสั่ง sudo chmod +x /home/pi/Raspberry-pi-car/start1.sh ทำการป้อนคำสั่งเพื่อให้สามารถรันอัตโนมัติได้ทั้ง ทำแบบนี้ทั้งคำสั่งของ start1.sh, start2.sh และ start3.sh

จากนั้นขั้นตอนถัดมาทำการป้อนคำสั่ง sudo /home/pi/Raspberry-pi-car/start1.sh เพื่อสามารถรัน start1.sh, start2.sh และ start3.sh แยกคนละหน้า คอมมานไลน์

ขั้นตอนที่ห้า ทำการเปิดหน้าต่างเทอมินอล และป้อนคำสั่ง sudo /home/pi/Raspberry-Pi-Car/start1.sh แล้วรันโปรแกรมต่อไป

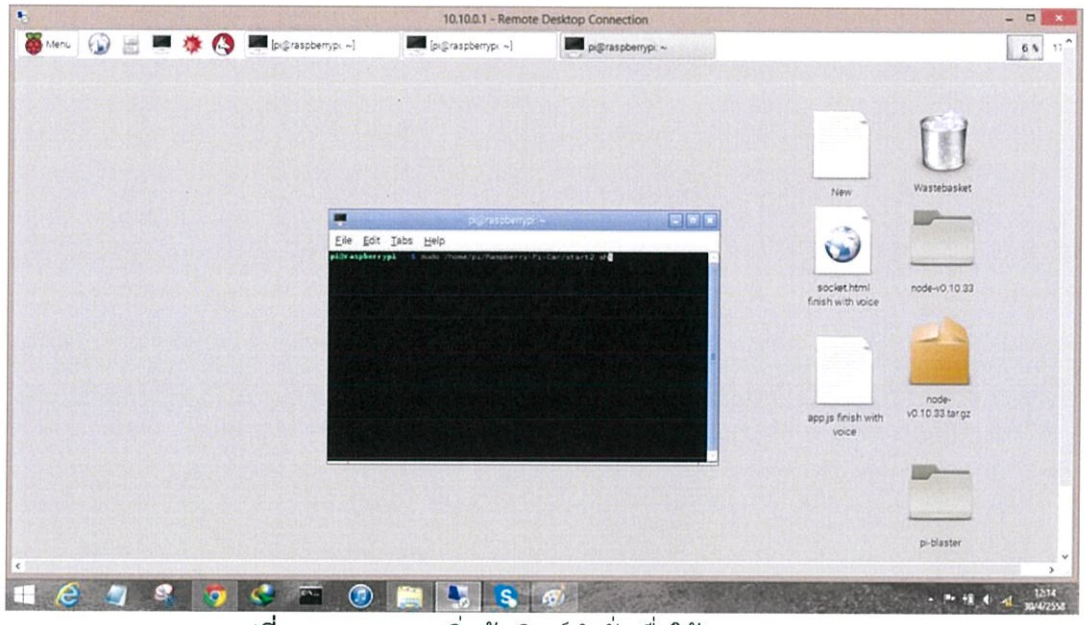


รูปที่ 4.1 แสดงการเริ่มต้นพิมพ์คำสั่งเพื่อใช้งาน start1

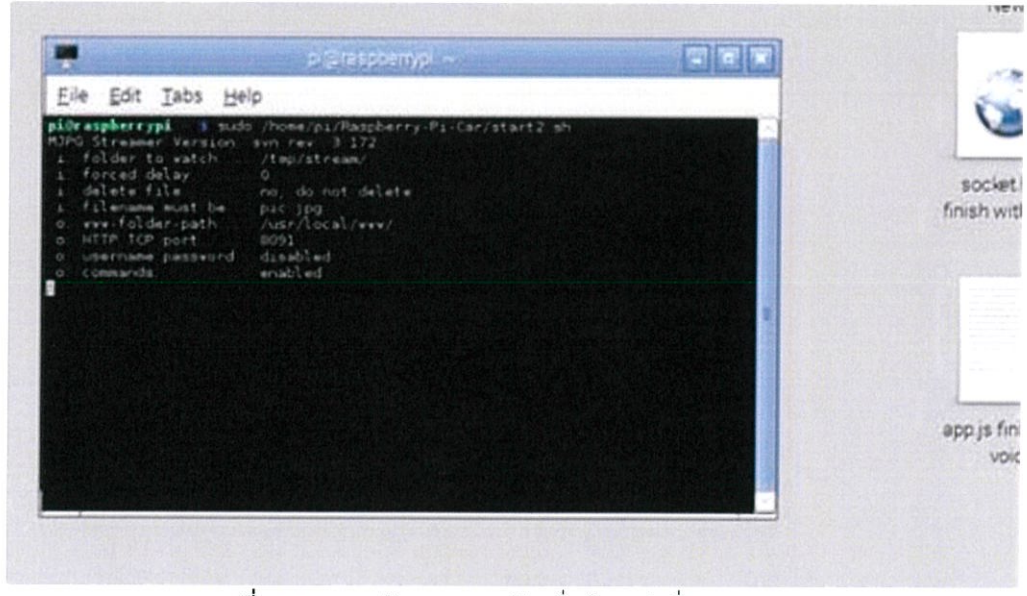


รูปที่ 4.2 แสดงลักษณะการรันเมื่อป้อนคำสั่งของ start1

ขั้นตอนที่หก ทำการเปิดหน้าต่างเทอมินอล และป้อนคำสั่ง sudo /home/pi/Raspberry-Pi-Car/start2.sh แล้วรันโปรแกรมต่อไป

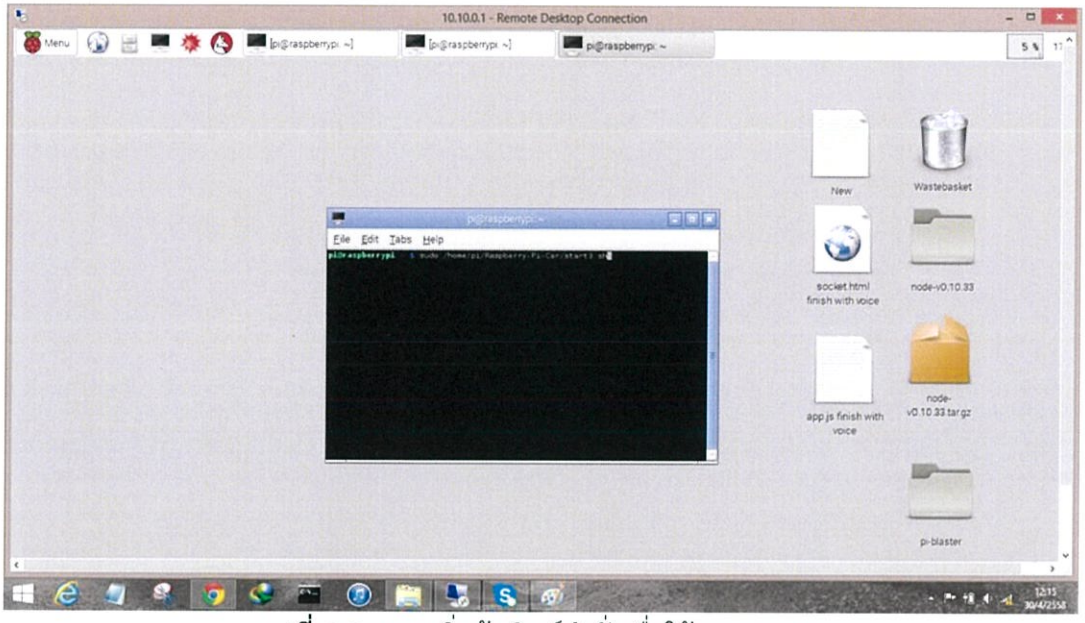


รูปที่ 4.3 แสดงการเริ่มต้นพิมพ์คำสั่งเพื่อใช้งาน start2

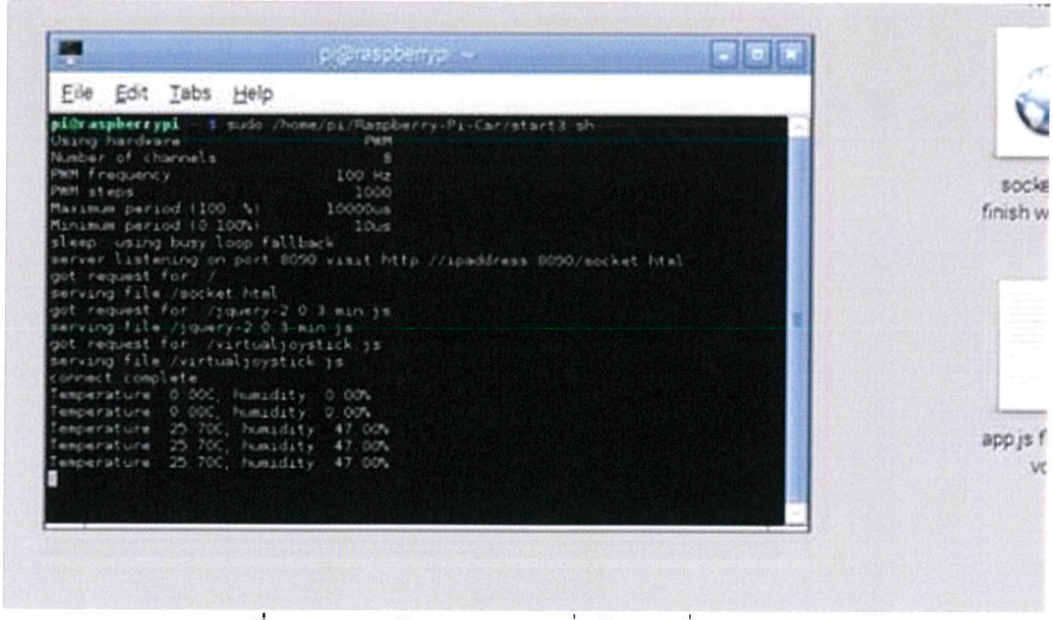


รูปที่ 4.4 แสดงลักษณะการรันเมื่อป้อนคำสั่งของ start2

ขั้นตอนที่เจ็ด ทำการเปิดหน้าต่างเทอมินอล และป้อนคำสั่ง sudo /home/pi/Raspberry-Pi-Car/start3.sh แล้วรันโปรแกรมต่อไป

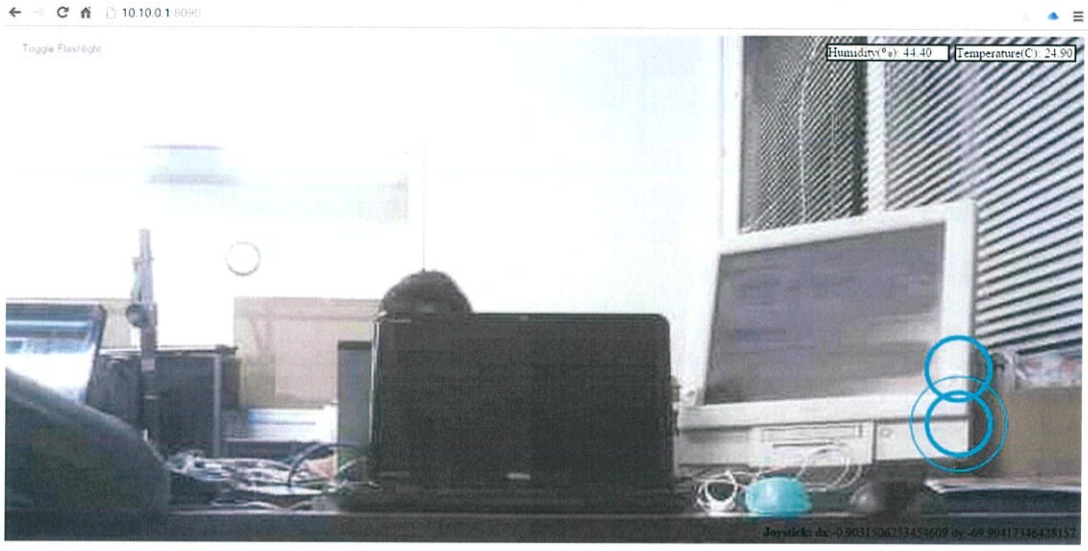


รูปที่ 4.5 แสดงเริ่มต้นพิมพ์คำสั่งเพื่อใช้งาน start3



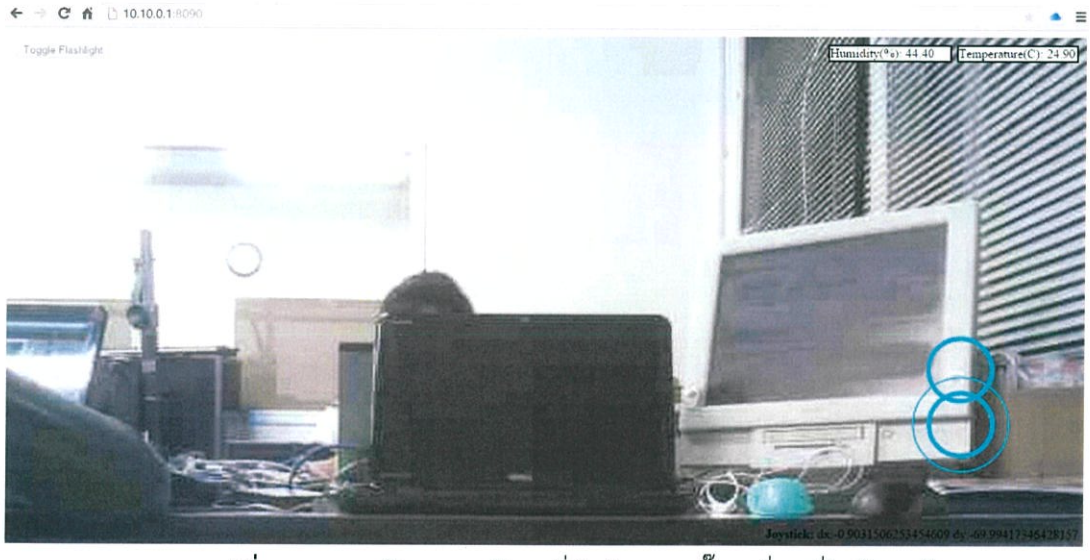
รูปที่ 4.6 แสดงลักษณะการรันเมื่อป้อนคำสั่งของ start3

จากนั้น เมื่อรันครบทุกคำสั่งและ ขั้นตอนต่อมาทำการเปิดหน้าเว็บบราวเซอร์ และพิมพ์เป็น http://10.10.0.1.8090 จากนั้นหน้าคอนโทรลหุ่นยนต์สำรวจจะปรากฏขึ้นมา

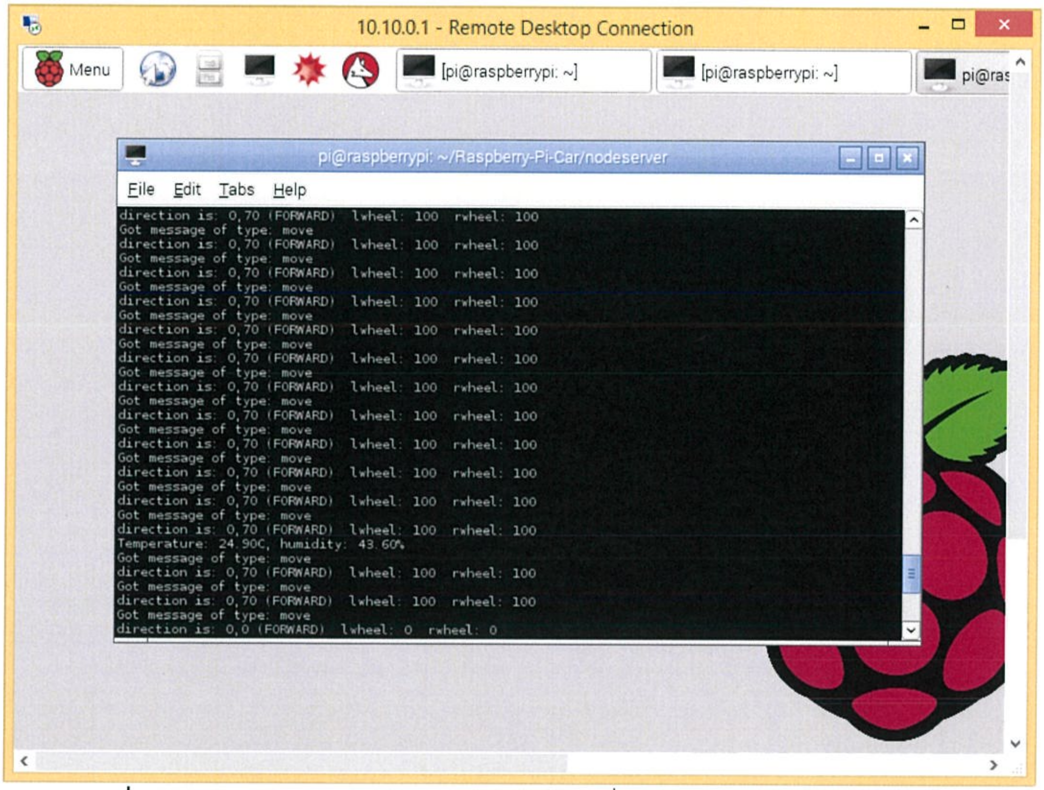


รูปที่ 4.7 แสดงลักษณะหน้าจอคอนโทรลหุ่นยนต์

4.2.2 สั่งให้เดินไปข้างหน้า

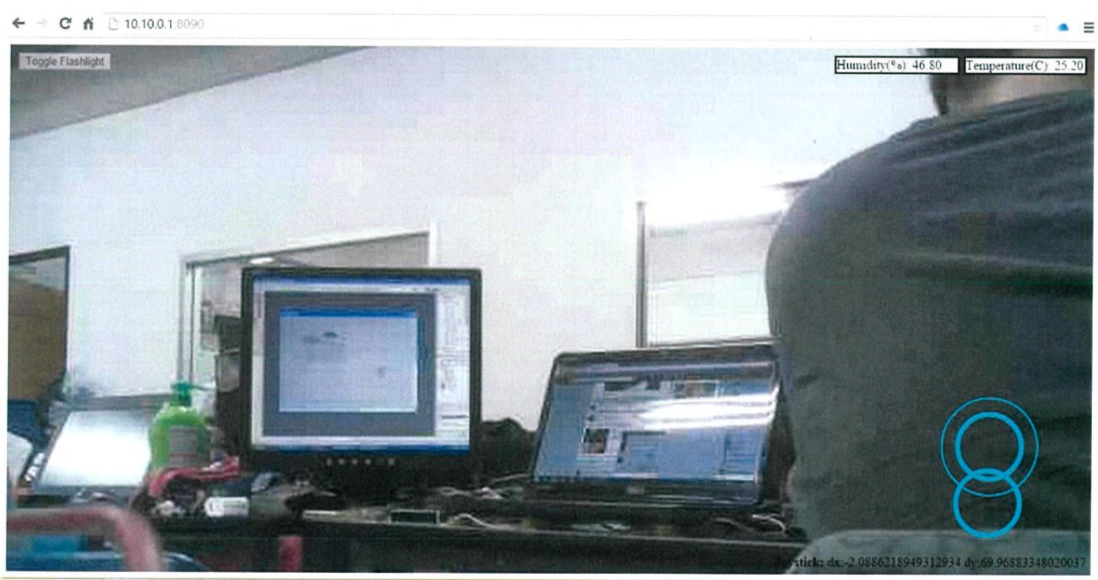


รูปที่ 4.8 แสดงลักษณะหน้าจอที่บังคับจอยสติ๊กเคลื่อนที่ไปข้างหน้า

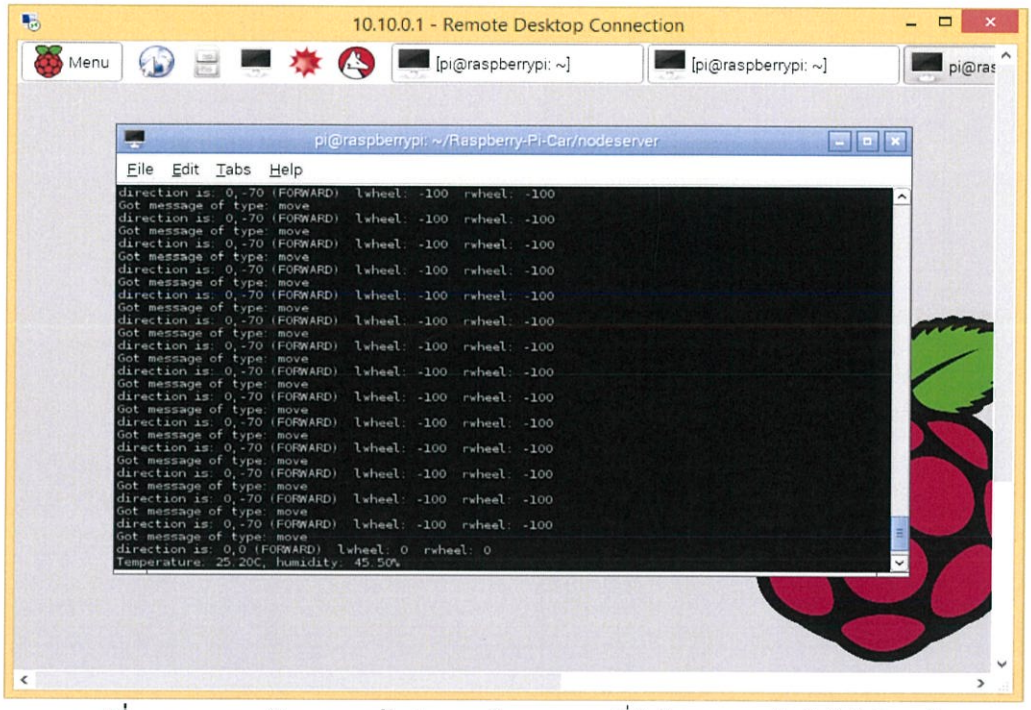


รูปที่ 4.9 แสดงลักษณะหน้าต่างเทอมินอลขณะที่รันโปรแกรมบังคับให้เดินหน้า

4.2.3 การเดินถอยหลัง

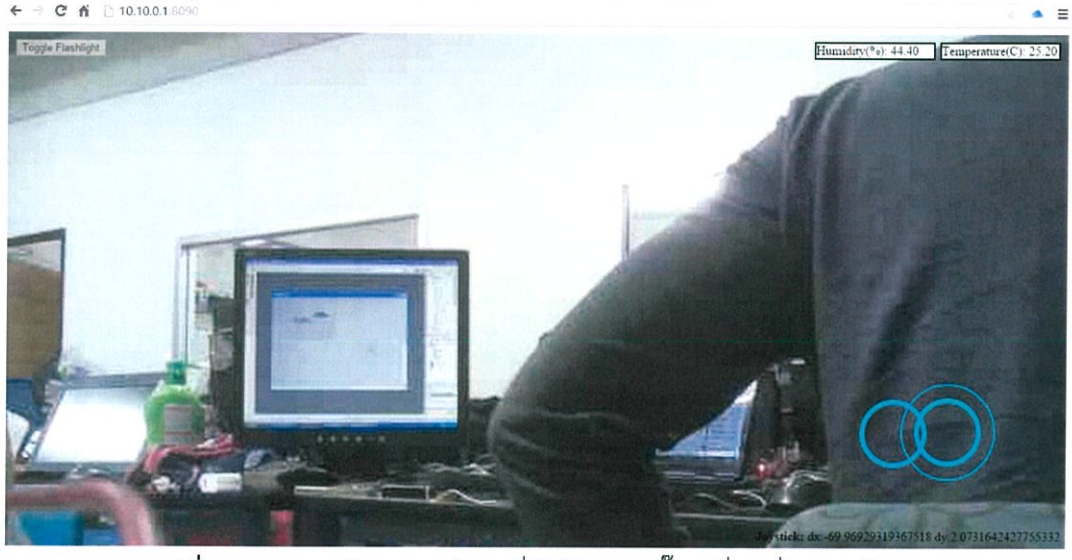


รูปที่ 4.10 แสดงลักษณะหน้าจอที่บังคับจอยสติ๊กเคลื่อนที่ไปข้างหลัง

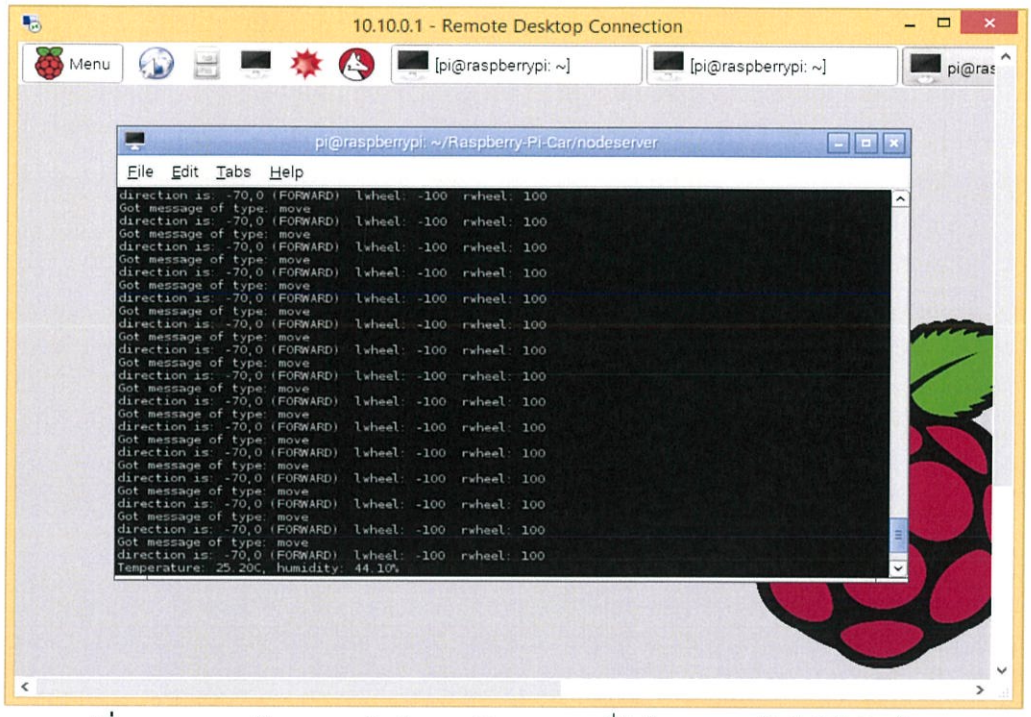


รูปที่ 4.11 แสดงลักษณะหน้าต่างเทอมินอลขณะที่รันโปรแกรมบังคับให้เดินหลัง

4.2.4 การหมุนตัวไปทางซ้าย

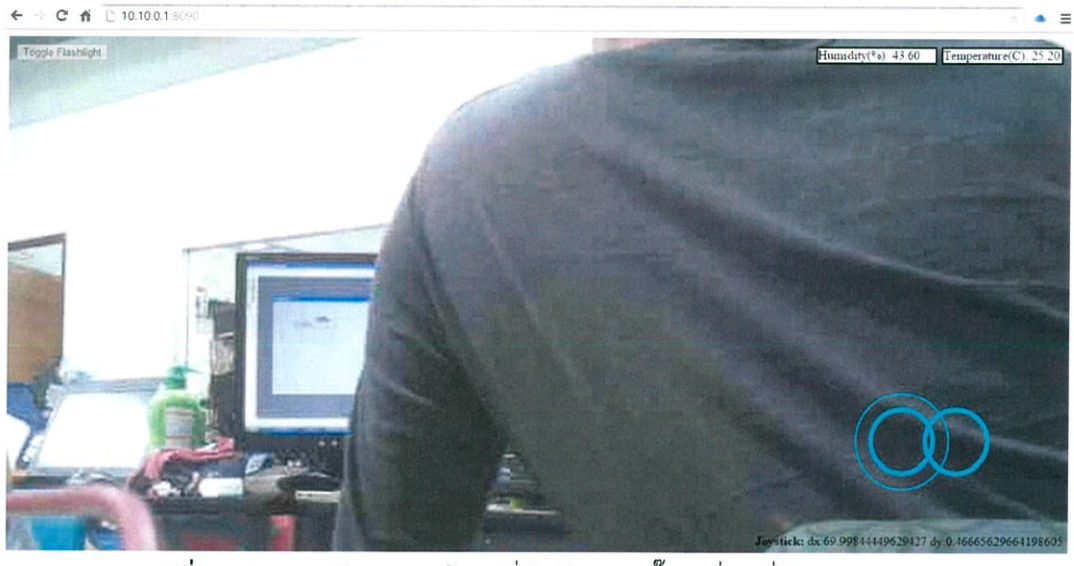


รูปที่ 4.12 แสดงลักษณะหน้าจอที่บังคับจอยสติ๊กเคลื่อนที่ไปทางซ้าย

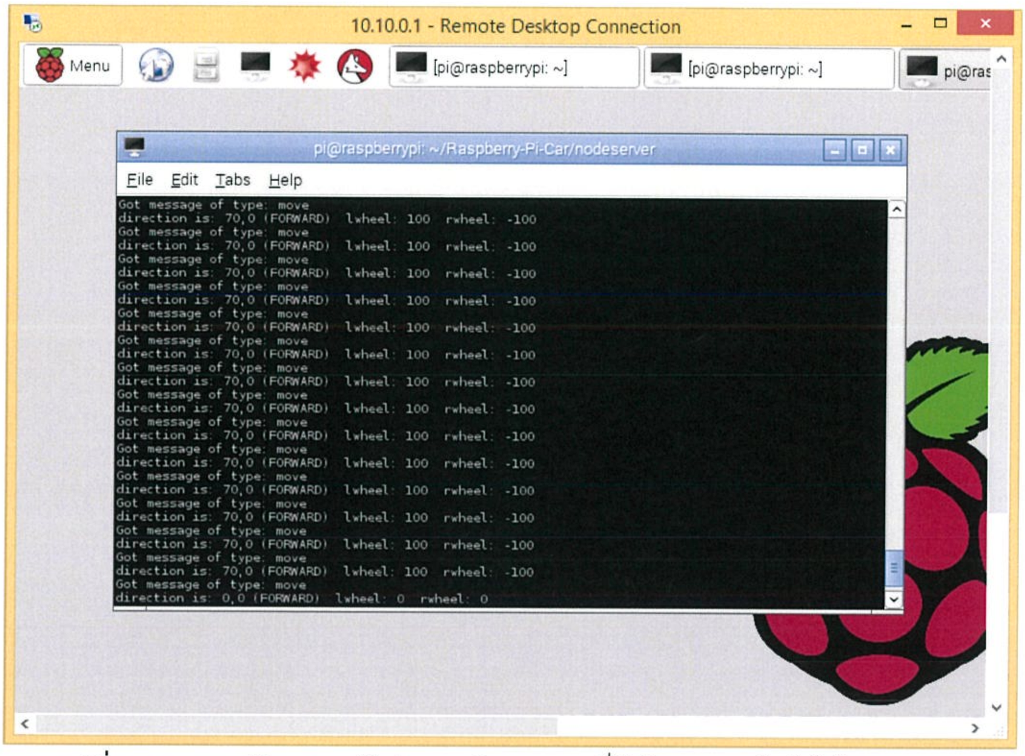


รูปที่ 4.13 แสดงลักษณะหน้าต่างเทอมินอลขณะที่รันโปรแกรมบังคับให้เดินซ้าย

4.2.5 การหมุนตัวไปทางขวา

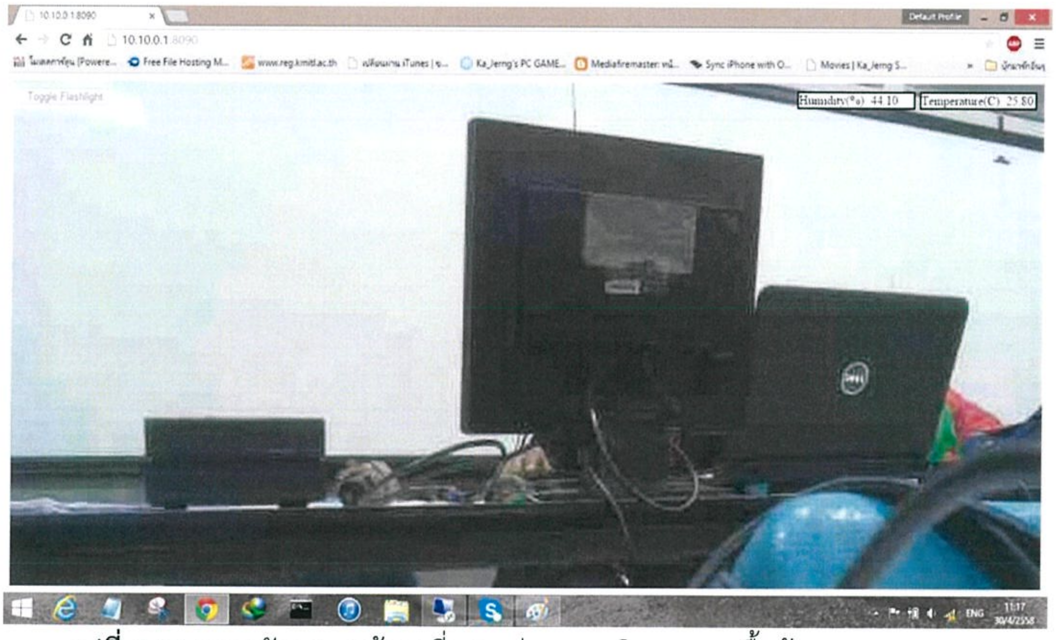


รูปที่ 4.14 แสดงลักษณะหน้าจอที่บังคับจอยสติ๊กเคลื่อนที่ไปทางขวา

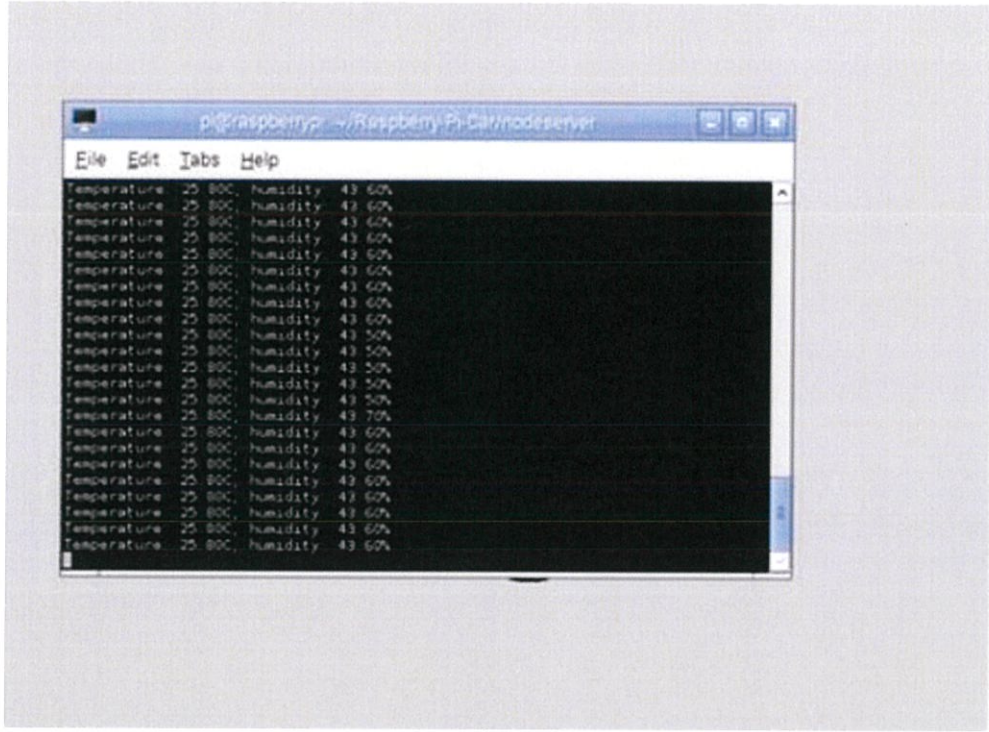


รูปที่ 4.15 แสดงลักษณะหน้าต่างเทอมินอลขณะที่รันโปรแกรมบังคับให้เดินขวา

4.2.6 การทดสอบค่าอุณหภูมิ และความชื้น



รูปที่ 4.16 แสดงลักษณะหน้าจอที่แสดงค่าอุณหภูมิและความชื้นด้านมุมขวาของจอ



รูปที่ 4.17 แสดงลักษณะหน้าต่างเทอมินอลขณะที่รันโปรแกรมแสดงค่าอุณหภูมิและอุปกรณ์ตรวจรู้

บทที่ 5
สรุปผลการทดลอง

5.1 สรุปผลการทดลอง

จากการทดลองหุ่นยนต์สำรวจควบคุมด้วยสมาร์ทดีไวซ์ ผ่านระบบการเชื่อมต่อแบบไร้สาย (Wireless Explorer Robot) ซึงใช้มอเตอร์ จำนวน 2 ตัว ยึดเข้ากับโครงสร้างทั้งสองด้านของตัวหุ่นยนต์ และใช้บอร์ด Raspberry Pi ,มอเตอร์ คอนโทรลเลอร์ ,DC to DC stepdown converter และ buffer เป็นตัวควบคุม จึงได้ทำการสรุปผลการทดลองดังนี้

ในส่วนของฮาร์ดแวร์หุ่นยนต์สำรวจ ซึ่งสามารถเคลื่อนที่ได้ในทิศทางเดินหน้าถอยหลัง และตรวจจับอุณหภูมิ และความชื้นได้อย่างปกติและมีประสิทธิภาพ แต่ในการเคลื่อนที่ในทิศทางเลี้ยวซ้ายและเลี้ยวขวา ยังไม่ได้ประสิทธิภาพเท่าที่ควร เนื่องจากอาจพบอุปสรรคทางด้านน้ำหนักของฮาร์ดแวร์ แต่กรณีการศึกษาในอนาคตทางผู้จัดทำ ยังสามารถลดขนาดทางด้านฮาร์ดแวร์ได้อีก โดยลดขนาดของแบตเตอร์รี่ อีกทั้งเรายังสามารถเพิ่มอุปกรณ์ตรวจรู้ชนิดต่างๆเข้าไปได้อีก ทำให้หุ่นยนต์สำรวจนี้มีประสิทธิภาพที่เพิ่มมากขึ้นมากกว่าเดิม

ส่วนในส่วนของซอฟท์แวร์นั้น ทางผู้จัดทำได้ทำการทดสอบแล้ว พบว่าในส่วนซอฟท์แวร์นี้ไม่พบอุปสรรคใดๆและสามารถควบได้อย่างมีประสิทธิภาพ และเป็นไปตามเป้าหมาย ทั้งทางด้าน รูปแบบการเดิน ,รูปแบบการเดินหน้า ,รูปแบบการถอยหลัง ,รูปแบบการเลี้ยวซ้าย ,รูปแบบการเลี้ยวขวา ,รูปแบบการแสดงภาพแบบ ทันที บนหน้าจอ ,รูปแบบการตรวจจับค่าความร้อนและความชื้น

5.2 ข้อเสนอแนะ

จากการศึกษา และได้ทำการทดลองในรูปแบบต่างของหุ่นยนต์ ทำให้พบปัญหาต่างๆที่เกิดขึ้น ได้แก่ ประสิทธิภาพในการเคลื่อนที่ของหุ่นยนต์สำรวจนั้นยังไม่เสถียรเท่าที่ควร เนื่องจากพบปัญหาทางด้านฮาร์ดแวร์ซึ่งมีน้ำหนักมากเกินไป สามารถแก้ไขโดยอาจทำให้ในส่วนนี้มีน้ำหนักเบาลง โดยลดขนาดน้ำหนักของแบตเตอรี่ที่ใช้งานจริงอยู่ในปัจจุบัน และพบปัญหาทางด้านการจัดระเบียบสายไฟ เนื่องจากสายไฟที่ใช้ค่อยข้างเยอะและหยุ่งเหยิงทำให้ดูไม่เป็นระเบียบ จึงควรจัดเรียงให้เป็นระเบียบ เพื่อสะดวกและง่ายต่อการแก้ไขปัญหาที่เกิดขึ้น

บรรณานุกรม

[1] Rodrigo Ventura, Pedro U. Lima, "Search and Rescue Robots: the Civil Protection Teams of the Future" Institute for Systems and Robotics. Instituto Superior Tecnico, Technical University of Lisbon ´Av. Rovisco Pais, 1, 1049-001 Lisboa, Portugal. 978-0-7695-4791-6/12 © 2012 IEEE DOI 10.1109/EST.2012.40

[2] Matthew DeVries, Matt Johnson, Paul Lyzenga, Karl Stough, "Project Final Report Team 8: R.E.S.C.U.E: Robot for Extraction of Survivors Confined in Unreachable Environments" May2012Calvin College, ENGR 339/340 © 2012, Matthew DeVries, Matt Johnson, Paul Lyzenga, Karl Stough and Calvin College

[3] AutoCAD 2011 Manual Book by I.T. Solution Computer Thailand Co., LTD. 937 Srinakharin Road, Suanluang Dristricts of Bangkok 10520

[4] AutoCAD® 2012 Preview Guide by cadstudio www.autodesk.com/autocad

[5] SolidWorks Manual Book by D.Eng.(Agricultural Engineering) Jaturong Lungkapin, Rajamangala University of Technology Thanyaburi, 39 Moo 1, Rangsit-Nakhonnayok Road, Thanyaburi, Pathum Thani 12110

[6] Computer for Industrials Book Unit1 Introduction by Thanakon Yaowaphan

[7] Altium Design (Basic) Document by P'Nutt&P'Pass – 47C, Tutor 's Team – 49C. Department of Electronic Engineering, King Mongkut's Institute of Technology Ladkrabang

[8] Arduino and Motor Control : Part 1 on http://www.arduitronics.com/article/arduino-and-motor-control-part-1

[9] pi-blaster by mvitousek on https://github.com/mvitousek/pi-blaster/

[10] pi-blaster by sarfata on https://github.com/sarfata/pi-blaster/

[11] Software PWM on a Raspberry Pi on http://ozzmaker.com/2013/09/23/software-pwm-on-a-raspberry-pi/

[12] Using your Raspberry Pi as a Wireless Router and Web Server on http://www.daveconroy.com/using-your-raspberry-pi-as-a-wireless-router-and-web-server/

[13] Stream Video from the Raspberry Pi Camera to Web Browsers, Even on iOS and Android on http://blog.miguelgrinberg.com/post/stream-video-from-the-raspberry-pi-camera-to-web-browsers-even-on-ios-and-android

[14] node-dht-sensor by momenso on https://github.com/momenso/node-dht-sensor/

[15] Monitor Your Home With the Raspberry Pi B+ Created by Marc-Olivier Schwartz on https://learn.adafruit.com/downloads/pdf/monitor-your-home-with-the-raspberry-pi-b-plus.pdf

[16] picar by shaunuk on https://github.com/shaunuk/picar

[17] Raspberry-Pi-Car by RetroMelon https://github.com/RetroMelon/Raspberry-Pi-Car/

[18] Node.js https://nodejs.org/

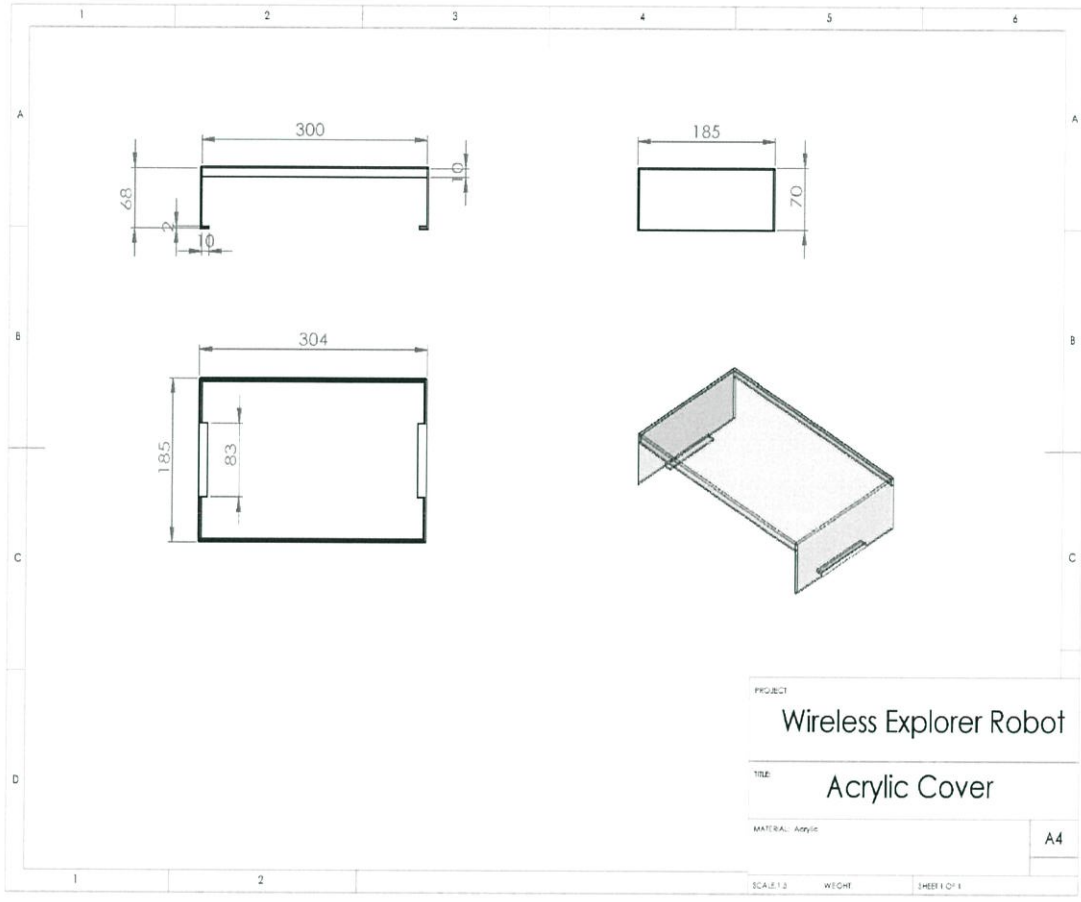
[19] [Raspberry Pi] Installation NodeJS http://doc.inex.co.th/r-pi-nodejs-installation/

ภาคผนวก

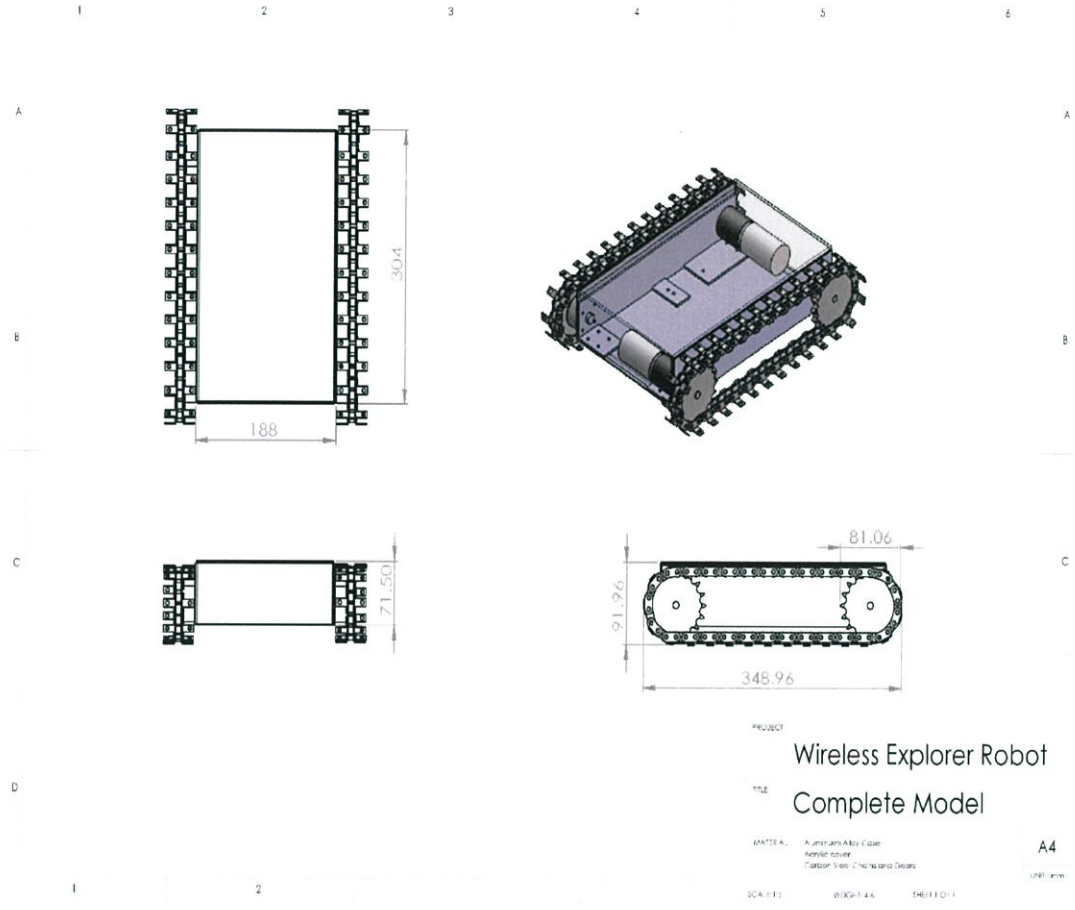
ภาคผนวก ก.

แบบร่างชิ้นส่วนของโครง Wireless Explorer Robot

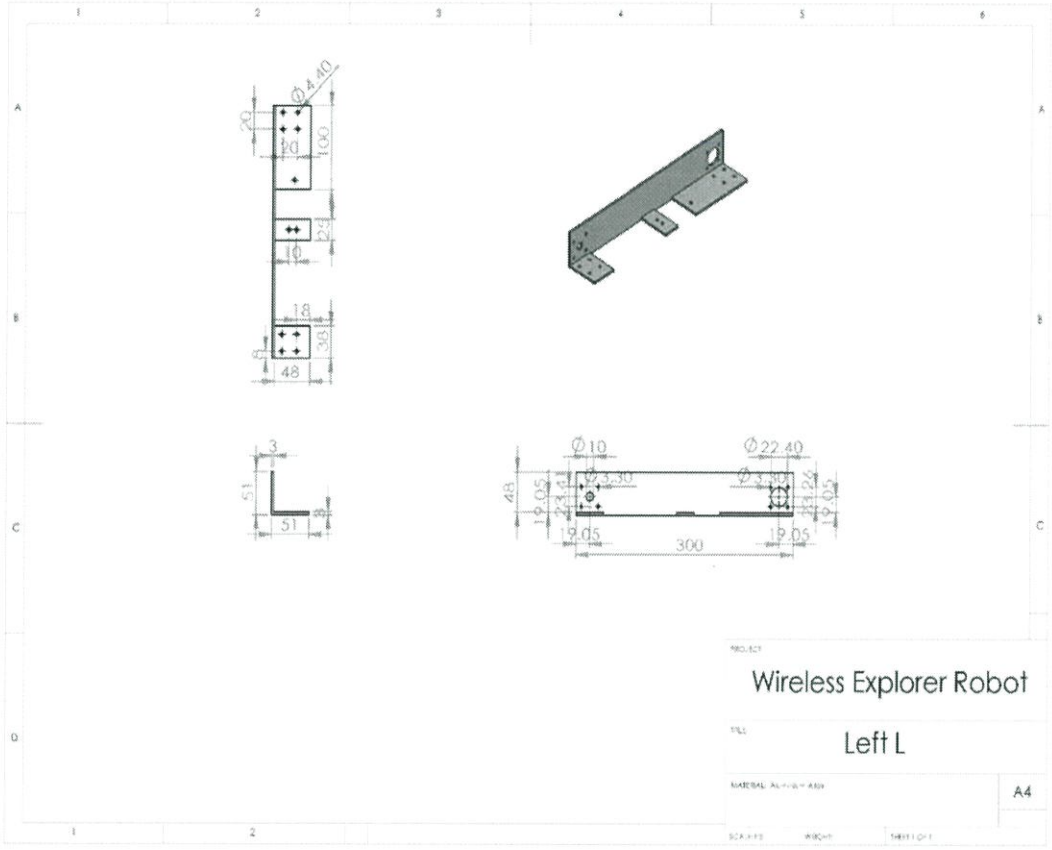
แบบร่างชิ้นส่วนโครงสร้างอะคริลิค



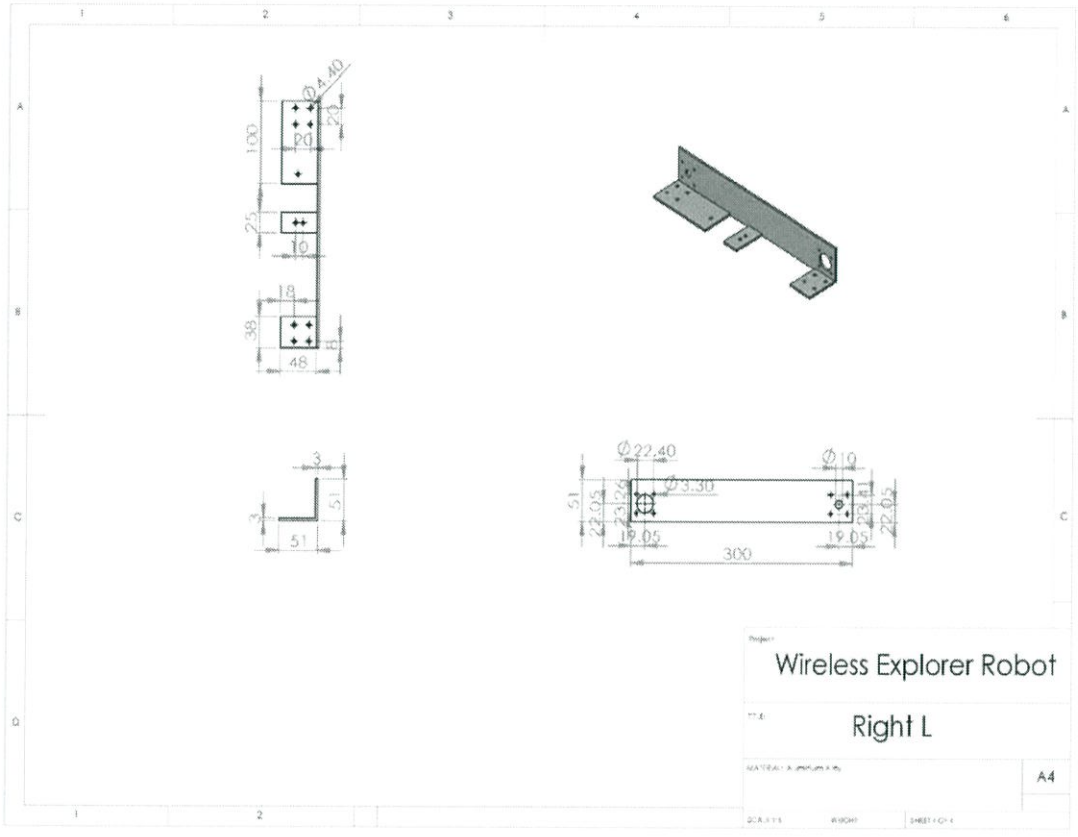
แบบร่างชิ้นส่วนขนาดของล้อโซ่ปีก



แบบร่างชิ้นส่วนของฉากอะลูมิเนียมด้านซ้าย



แบบร่างชิ้นส่วนของฉากอะลูมิเนียมด้านขวา



ภาคผนวก ข.

โปรแกรม

โปรแกรมส่วนของ Test Sensor

```
// Make measurements from sensors
var dht_sensor = {
    initialize: function () {
        return sensorLib.initialize(22, 4);
    },
    read: function () {
        var readout = sensorLib.read();

        console.log('Temperature: ' + readout.temperature.toFixed(2) + 'C, ' + 'humidity: ' +
readout.humidity.toFixed(2) + '%');
        setTimeout(function () {
            dht_sensor.read();
        }, 2000);
    }
};
if (dht_sensor.initialize()) {
    dht_sensor.read();
} else {
    console.warn('Failed to initialize sensor');
}
    });
```

โปรแกรมส่วนของ App.js

```
//declare required modules
var app = require('http').createServer(handler)
  , io = require('socket.io').listen(app)
  , fs = require('fs')
  , static = require('node-static')
  , sys = require('sys')
  , sleep = require('sleep')
  , blaster = require('pi-blaster.js')
  , shellexecute = require('child_process').exec
  , argv = require('optimist').argv
  , sensorLib = require('node-dht-sensor')
  app.listen(8090);


var file = new(static.Server)();
function handler(request, response) {
  console.log("got request for: "+request.url);
  if(request.url == "/"){
    request.url = "/socket.html";
  }
  console.log('serving file',request.url)
  file.serve(request, response);
};

console.log('server listening on port 8090 visit http://ipaddress:8090/socket.html');
var child = shellexecute('echo "server. listening. on. port. 80. 90" | festival --tts',
function(error, stdin, stdout){});

function setgpio(pin, value){ //value should be between 0 and 1
        blaster.setPwm(pin, value);
}

var rightwheel = [1, 2];
```

```
var leftwheel = [4, 5]; //22, 27
var flash = 6

var leftwheelpowergraph = [[0, 100], [90, 100], [120, 0], [180, -100], [270, -100], [300, 0],
[360, 100]];
var rightwheelpowergraph = [[0, 100], [60, 0], [90, -100], [180, -100], [240, 0], [270, 100],
[360, 100]];


//power should be a number between -1 and 1
function setwheelpower(wheel, power){
  if(power>0){
    setgpio(wheel[0], power);
    setgpio(wheel[1], 0);
  }
else{
    setgpio(wheel[1], Math.abs(power));
    setgpio(wheel[0], 0);
  }
}

function setFlash(value){
  if(value){
    setgpio(flash, 1);
  }else{
    setgpio(flash, 0);
  }
}


//converts radians to degrees
function toDegrees (angle) {
  return angle * (180 / Math.PI);
}
```

```
//calculates the angle the joystick is pointed at
function calculateangle(x, y){
  return (360+toDegrees(Math.atan2(x, y)))%360;
}

//calculates the y value at point x on the line between the pair of points
function calculatelocation(xlocation, pairofpoints){
  p1 = pairofpoints[0];
  p2 = pairofpoints[1];
  gradient = (p2[1]-p1[1])/(p2[0]-p1[0]);
  return p1[1] + (xlocation-p1[0])*gradient;
}

//looks up the value of y at the point x for the graph described by "pointlist"
function lookupvalue(x, pointlist){
  for(i=0; i < pointlist.length; i++){
    if (x<pointlist[i][0]){
      return calculatelocation(x,[pointlist[i-1], pointlist[i]]);
    }
  }
}

function getdirectiontext(x, y){
  if(Math.sqrt(Math.pow(x, 2)+Math.pow(y, 2)) < 20){
    return "CENTRE";
  }
if(x==0){
    x=0.00001 //to avoid dividing by zero error
  }

  var sidesratio=y/x;

  if(Math.abs(sidesratio) < 0.75){ //if the gradient of the line is <0.75, we must be pointing
in either the left or right direction
    if(x>0){
```

```
      return "RIGHT";
    }else{
      return "LEFT";
    }
  }else{//if the sides ratio is over a certain amount then we must be either going forwrd
or backward
    if(y>0){
      return "BACKWARD";
    }else{
      return "FORWARD";
    }
  }

}

//calculates a number between 0 and 1, dependent on how actuated the joystick is,
compared to its maximum value.
var joystickradius = 70.0;
function calculatepowercoefficient(direction){
    var x = direction[0];
    var y = direction[1];
    var hypot = Math.sqrt(x*x + y*y);

    return hypot / joystickradius;
}

//goes in the direction specified by a list of form [x component, y component]
function goindirection(direction){
  if(direction[0] > joystickradius){direction[0] = joystickradius;}
  if(direction[1] > joystickradius){direction[1] = joystickradius;}
  if(direction[0] < -joystickradius){direction[0] = -joystickradius;}
  if(direction[1] < -joystickradius){direction[1] = -joystickradius;}

  var joystickangle = calculateangle(direction[0], direction[1]);
  var leftwheelpower = lookupvalue(joystickangle, leftwheelpowergraph);
```

```
  var rightwheelpower = lookupvalue(joystickangle, rightwheelpowergraph);
  var power = calculatepowercoefficient(direction);
if(direction[0]==0 && direction[1]==0){
    leftwheelpower = rightwheelpower = 0;
  }

  setwheelpower(leftwheel, (leftwheelpower/100) * power);
  setwheelpower(rightwheel, (rightwheelpower/100) * power);

  console.log("direction is: " + direction.toString() +
              " (" + getdirectiontext(direction) + ")"+
              "  lwheel: " + leftwheelpower.toString()+
              "  rwheel: " + rightwheelpower.toString());
}


var d = new Date();
var lastcommandtime = d.getTime();

setInterval(function(){
        if (d.getTime() - lastcommandtime > 1000){
            //SWITCH OFF THE มอเตอร์S
            console.log("Have not recieved command in over a second, so stopping bot.");
        }
}, 500); //every second check if we haven't had a new command in a while. if we
haven't, stop the robot. NOTE: I HAVE A FEELING THIS PIECE OF CODE DOESN'T WORK.

//fire up a web socket server
io.sockets.on('connection', function (socket) {
  socket.on('fromclient', function (data) {
    console.log("Got message of type: "+data.commandtype);
    if(data.commandtype=="move"){
        lastcommandtime = d.getTime();
        var x = data.joystickX;
        var y = -data.joystickY; //inverting the y value
```

```
        goindirection([x, y]);
    }

    else if(data.commandtype=="flashlight"){
        setFlash(data.value);
    }
  });
});

io.sockets.on('connection',function(socket){
console.log('connect complete');
// Make measurements from sensors
var dht_sensor = {
    initialize: function () {
        return sensorLib.initialize(22, 4);
    },
    read: function () {
        var readout = sensorLib.read();

        socket.emit('temp',{'temp':readout.temperature.toFixed(2)});
        socket.emit('humid',{'humid':readout.humidity.toFixed(2)});

        console.log('Temperature: ' + readout.temperature.toFixed(2) + 'C, ' +
            'humidity: ' + readout.humidity.toFixed(2) + '%');
        setTimeout(function () {
            dht_sensor.read();
        }, 2000);
    }
};
if (dht_sensor.initialize()) {
    dht_sensor.read();
} else {
    console.warn('Failed to initialize sensor');
}
```

```
});


process.on('SIGINT', function() {
  gpio.destroy();
  console.log("\nGracefully shutting down from SIGINT (Ctrl-C)");
  return process.exit();
});
```

โปรแกรมในส่วนของ Socket.html

```
<html>
<head>
<script src="/jquery-2.0.3.min.js" language="javascript"></script>
<script src="/socket.io/socket.io.js" language="javascript"></script>
<script src="/virtualjoystick.js" language="javascript"></script>
<meta name="viewport" content="user-scalable=no" />
<meta name="mobile-web-app-capable" content="yes">
<meta name="apple-mobile-web-app-capable" content="yes">

<style type="text/css">

html, body{
  overflow: hidden;
}

.maindiv{
  position:relative;
  max-height:100%;
  min-height:100%;
}

.mainimg{
  position:relative;
  width:100%;
  max-height:100%;
  min-height:100%;
  pointer-events: none;

}

#text1{
  position:absolute;
```

```
  bottom:0;
  right:0;
  text-size:20px;
  margin:6px;
  pointer-events: none;
  -webkit-touch-callout: none;
  -webkit-user-select: none;
  -khtml-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  user-select: none;
}


#flashlightbutton{
  opacity: 0.5;
}

.buttonsdiv{
  position:absolute;
  top:10px;
  left:10px;
}

.maincanvas{
  width:100%;
  height:100%;
  position:absolute;
  top:10px;
  left:10px;
}

</style>

<script>
```

```
    var socket;

    var flashlight = false;

    function switchFlash(){
        flashlight = !flashlight;
        socket.emit('fromclient', {commandtype:"flashlight", value:flashlight});
    }

    //the function to set the img's src to the correct location
    function setimgsrc(){
        document.getElementById("mainimg").src = "http://10.10.0.1:8091/?action=stream";
    }

    //the code that runs once the page is finished loading
    $(function(){
        setimgsrc();

        //creating a socket to send data back to the node.js server
        socket = io.connect('http://10.10.0.1:8090');

    socket.on('temp', function(data){$('#texttemp').text('Temperature(C): '+data.temp);});
    socket.on('humid', function(data){$('#texthumid').text('Humidity(%): '+data.humid);});

    });




</script>
</head>
<body>

  <div class="mainDiv" id="mainDiv">
    <script>
```

```
//creating a joystick
var joystick = new VirtualJoystick({
  container : document.getElementById('container'),
  mouseSupport    : true,
  limitStickTravel : true,
  stickRadius : 70,
});

/*----code for logging touch events
joystick.addEventListener('touchStart', function(){
  console.log('down')
});
joystick.addEventListener('touchEnd', function(){
  console.log('up')
});
*/

//updating the bottom text that tells us what position the joystick is in.
setInterval(function(){
  var outputEl       = document.getElementById('text1');
  if(joystick.deltaX() != 0 || joystick.deltaY() != 0){
    outputEl.innerHTML = '<b>Joystick:</b> '
        + ' dx:'+joystick.deltaX()
        + ' dy:'+joystick.deltaY();
  }else{
    outputEl.innerHTML = ' ';
  }
}, 1/30 * 1000);


var uppressed = downpressed = leftpressed = rightpressed = keyunpressed = false;

function keydownfunction(e){
    switch(e.keyCode){
```

```
        case 37:
            leftpressed=true;
            break;
        case 38:
            uppressed=true;
            break;
        case 39:
            rightpressed=true;
            break;
        case 40:
            downpressed=true;
            break;
    }
    keyunpressed=false;
}

function keyupfunction(e){
    switch(e.keyCode){
        case 37:
            leftpressed=false;
            break;
        case 38:
            uppressed=false;
            break;
        case 39:
            rightpressed=false;
            break;
        case 40:
            downpressed=false;
            break;
    }
    keyunpressed=true;
}

window.addEventListener("keydown", keydownfunction, false);
```

```
    window.addEventListener("keyup", keyupfunction, false);


    var joystickX = 0;
    var joystickY = 0;

    setInterval(function(){
    //if a key is currently pressed, send a command for it instead of for the joystick.
    if(keyunpressed){
        keyunpressed=false;
        socket.emit('fromclient', {commandtype:"move", joystickX:0, joystickY:0});
    }
    if(uppressed || downpressed || leftpressed || rightpressed){
        if(rightpressed){
            socket.emit('fromclient', {commandtype:"move", joystickX:100, joystickY:0});
        }
        else if(leftpressed){
            socket.emit('fromclient', {commandtype:"move", joystickX:-100, joystickY:0});
        }
        else if(uppressed){
            socket.emit('fromclient', {commandtype:"move", joystickX:0, joystickY:-100});
        }
        else if(downpressed){
            socket.emit('fromclient', {commandtype:"move", joystickX:0, joystickY:100});
        }
    }
    else if(joystick.deltaX()!=joystickX || joystick.deltaY()!=joystickY){
        joystickX = joystick.deltaX();
        joystickY = joystick.deltaY();
        socket.emit('fromclient', {commandtype:"move", joystickX:Math.round(joystickX),
joystickY:Math.round(joystickY)} );
      }
    }, 50);
```

```
    </script>
    <img src="http://1.2.3.4:8091/?action=stream" class="mainimg" id="mainimg">
    <p value="?" id="text1" class="text1">?</p>
  </div>
  <div class="buttonsdiv">
  <input type="button" value="Toggle Flashlight" id="flashlightbutton"
onclick="switchFlash()">
  <div id="texttemp">
  </div>
  <div id="texthumid">
  </div>



</body>
</head>
</html>
```

โปรแกรมในส่วนของ virtualjoystick.js

```
var VirtualJoystick      = function(opts)
{
        opts                    = opts                  || {};
        this._container         = opts.container        || document.body;
        this._strokeStyle       = opts.strokeStyle      || 'cyan';
        this._stickEl           = opts.stickElement     || this._buildJoystickStick();
        this._baseEl            = opts.baseElement      || this._buildJoystickBase();
        this._mouseSupport      = opts.mouseSupport !== undefined ? opts.mouseSupport
: false;
        this._stationaryBase    = opts.stationaryBase || false;
        this._baseX             = this._stickX = opts.baseX || 0
        this._baseY             = this._stickY = opts.baseY || 0
        this._limitStickTravel  = opts.limitStickTravel || false
        this._stickRadius       = opts.stickRadius !== undefined ? opts.stickRadius : 100
        this._useCssTransform = opts.useCssTransform !== undefined ?
opts.useCssTransform : false

        this._container.style.position  = "relative"

        this._container.appendChild(this._baseEl)
        this._baseEl.style.position     = "absolute"
        this._baseEl.style.display      = "none"
        this._container.appendChild(this._stickEl)
        this._stickEl.style.position    = "absolute"
        this._stickEl.style.display     = "none"

        this._pressed   = false;
        this._touchIdx  = null;

        if(this._stationaryBase === true){
                this._baseEl.style.display      = "";
                this._baseEl.style.left         = (this._baseX - this._baseEl.width /2)+"px";
                this._baseEl.style.top          = (this._baseY - this._baseEl.height/2)+"px";
```

```
        }

        this._transform= this._useCssTransform ? this._getTransformProperty() : false;
        this._has3d     = this._check3D();

        var __bind      = function(fn, me){ return function(){ return fn.apply(me,
arguments); }; };
        this._$onTouchStart   = __bind(this._onTouchStart  , this);
        this._$onTouchEnd     = __bind(this._onTouchEnd    , this);
        this._$onTouchMove    = __bind(this._onTouchMove   , this);
        this._container.addEventListener( 'touchstart', this._$onTouchStart  , false );
        this._container.addEventListener( 'touchend' , this._$onTouchEnd    , false );
        this._container.addEventListener( 'touchmove'         , this._$onTouchMove , false
);
        if( this._mouseSupport ){
                this._$onMouseDown  = __bind(this._onMouseDown , this);
                this._$onMouseUp    = __bind(this._onMouseUp   , this);
                this._$onMouseMove  = __bind(this._onMouseMove , this);
                this._container.addEventListener( 'mousedown'       ,
this._$onMouseDown  , false );
                this._container.addEventListener( 'mouseup' , this._$onMouseUp    , false
);
                this._container.addEventListener( 'mousemove'       , this._$onMouseMove
        , false );
        }
}

VirtualJoystick.prototype.destroy       = function()
{
        this._container.removeChild(this._baseEl);
        this._container.removeChild(this._stickEl);

        this._container.removeEventListener( 'touchstart'     , this._$onTouchStart  , false
);
```

```
		this._container.removeEventListener( 'touchend'		, this._$onTouchEnd
		, false );
		this._container.removeEventListener( 'touchmove'	, this._$onTouchMove, false
);
		if( this._mouseSupport ){
			this._container.removeEventListener( 'mouseup'		,
this._$onMouseUp	, false );
			this._container.removeEventListener( 'mousedown'	,
this._$onMouseDown	, false );
			this._container.removeEventListener( 'mousemove'	, this._$onMouseMove
		, false );
		}
}

/**
 * @returns {Boolean} true if touchscreen is currently available, false otherwise
*/
VirtualJoystick.touchScreenAvailable	= function()
{
		return 'createTouch' in document ? true : false;
}

/**
 * microevents.js - https://github.com/jeromeetienne/microevent.js
*/
;(function(destObj){
		destObj.addEventListener	= function(event, fct){
			if(this._events === undefined)		this._events	= {};
			this._events[event] = this._events[event]	|| [];
			this._events[event].push(fct);
			return fct;
		};
		destObj.removeEventListener = function(event, fct){
			if(this._events === undefined)		this._events	= {};
			if( event in this._events === false  )	return;
```

```
                this._events[event].splice(this._events[event].indexOf(fct), 1);
        };
        destObj.dispatchEvent          = function(event /* , args... */){
                if(this._events === undefined)          this._events    = {};
                if( this._events[event] === undefined )         return;
                var tmpArray    = this._events[event].slice();
                for(var i = 0; i < tmpArray.length; i++){
                        var result      = tmpArray[i].apply(this,
Array.prototype.slice.call(arguments, 1))
                        if( result !== undefined )      return result;
                }
                return undefined
        };
})(VirtualJoystick.prototype);

//////////////////////////////////////////////////////////////////////////////
//                                                                          //
//////////////////////////////////////////////////////////////////////////////

VirtualJoystick.prototype.deltaX        = function(){ return this._stickX - this._baseX;}
VirtualJoystick.prototype.deltaY        = function(){ return this._stickY - this._baseY; }

VirtualJoystick.prototype.up    = function(){
        if( this._pressed === false )   return false;
        var deltaX      = this.deltaX();
        var deltaY      = this.deltaY();
        if( deltaY >= 0 )                               return false;
        if( Math.abs(deltaX) > 2*Math.abs(deltaY) )     return false;
        return true;
}
VirtualJoystick.prototype.down          = function(){
        if( this._pressed === false )   return false;
        var deltaX      = this.deltaX();
        var deltaY      = this.deltaY();
        if( deltaY <= 0 )                               return false;
```

```
        if( Math.abs(deltaX) > 2*Math.abs(deltaY) )    return false;
        return true;
}
VirtualJoystick.prototype.right = function(){
        if( this._pressed === false )    return false;
        var deltaX      = this.deltaX();
        var deltaY      = this.deltaY();
        if( deltaX <= 0 )                                return false;
        if( Math.abs(deltaY) > 2*Math.abs(deltaX) )    return false;
        return true;
}
VirtualJoystick.prototype.left  = function(){
        if( this._pressed === false )    return false;
        var deltaX      = this.deltaX();
        var deltaY      = this.deltaY();
        if( deltaX >= 0 )                                return false;
        if( Math.abs(deltaY) > 2*Math.abs(deltaX) )    return false;
        return true;
}

//////////////////////////////////////////////////////////////////////////////////
//                                                                              //
//////////////////////////////////////////////////////////////////////////////////

VirtualJoystick.prototype._onUp       = function()
{
        this._pressed   = false;
        this._stickEl.style.display       = "none";

        if(this._stationaryBase == false){
                this._baseEl.style.display       = "none";

                this._baseX     = this._baseY   = 0;
                this._stickX    = this._stickY  = 0;
        }
```

```
}

VirtualJoystick.prototype._onDown	= function(x, y)
{
	this._pressed	= true;
	if(this._stationaryBase == false){
		this._baseX	= x;
		this._baseY	= y;
		this._baseEl.style.display	= "";
		this._move(this._baseEl.style, (this._baseX - this._baseEl.width /2),
(this._baseY - this._baseEl.height/2));
	}

	this._stickX	= x;
	this._stickY	= y;

	if(this._limitStickTravel === true){
		var deltaX	= this.deltaX();
		var deltaY	= this.deltaY();
		var stickDistance = Math.sqrt( (deltaX * deltaX) + (deltaY * deltaY) );
		if(stickDistance > this._stickRadius){
			var stickNormalizedX = deltaX / stickDistance;
			var stickNormalizedY = deltaY / stickDistance;

			this._stickX = stickNormalizedX * this._stickRadius + this._baseX;
			this._stickY = stickNormalizedY * this._stickRadius + this._baseY;
		}
	}

	this._stickEl.style.display	= "";
	this._move(this._stickEl.style, (this._stickX - this._stickEl.width /2), (this._stickY -
this._stickEl.height/2));
}

VirtualJoystick.prototype._onMove	= function(x, y)
```

```
{
	if( this._pressed === true ){
		this._stickX	= x;
		this._stickY	= y;

		if(this._limitStickTravel === true){
			var deltaX	= this.deltaX();
			var deltaY	= this.deltaY();
			var stickDistance = Math.sqrt( (deltaX * deltaX) + (deltaY * deltaY)
);
			if(stickDistance > this._stickRadius){
				var stickNormalizedX = deltaX / stickDistance;
				var stickNormalizedY = deltaY / stickDistance;

				this._stickX = stickNormalizedX * this._stickRadius +
this._baseX;
				this._stickY = stickNormalizedY * this._stickRadius +
this._baseY;
			}
		}

	this._move(this._stickEl.style, (this._stickX - this._stickEl.width /2), (this._stickY -
this._stickEl.height/2));
	}
}


////////////////////////////////////////////////////////////////////////////
//		bind touch events (and mouse events for debug)		//
////////////////////////////////////////////////////////////////////////////

VirtualJoystick.prototype._onMouseUp	= function(event)
{
	return this._onUp();
}
```

```
VirtualJoystick.prototype._onMouseDown	= function(event)
{
	event.preventDefault();
	var x	= event.clientX;
	var y	= event.clientY;
	return this._onDown(x, y);
}

VirtualJoystick.prototype._onMouseMove	= function(event)
{
	var x	= event.clientX;
	var y	= event.clientY;
	return this._onMove(x, y);
}

//////////////////////////////////////////////////////////////////////////////
//		comment
//////////////////////////////////////////////////////////////////////////////

VirtualJoystick.prototype._onTouchStart	= function(event)
{
	// if there is already a touch inprogress do nothing
	if( this._touchIdx !== null )	return;

	// notify event for validation
	var isValid	= this.dispatchEvent('touchStartValidation', event);
	if( isValid === false )	return;

	// dispatch touchStart
	this.dispatchEvent('touchStart', event);

	event.preventDefault();
	// get the first who changed
	var touch	= event.changedTouches[0];
```

```
		// set the touchIdx of this joystick
		this._touchIdx	= touch.identifier;

		// forward the action
		var x		= touch.pageX;
		var y		= touch.pageY;
		return this._onDown(x, y)
}

VirtualJoystick.prototype._onTouchEnd	= function(event)
{
		// if there is no touch in progress, do nothing
		if( this._touchIdx === null )	return;

		// dispatch touchEnd
		this.dispatchEvent('touchEnd', event);

		// try to find our touch event
		var touchList	= event.changedTouches;
		for(var i = 0; i < touchList.length && touchList[i].identifier !== this._touchIdx; i++);
		// if touch event isnt found,
		if( i === touchList.length)	return;

		// reset touchIdx - mark it as no-touch-in-progress
		this._touchIdx	= null;

//??????
// no preventDefault to get click event on ios
event.preventDefault();

		return this._onUp()
}

VirtualJoystick.prototype._onTouchMove	= function(event)
{
```

```
        // if there is no touch in progress, do nothing
        if( this._touchIdx === null )   return;

        // try to find our touch event
        var touchList   = event.changedTouches;
        for(var i = 0; i < touchList.length && touchList[i].identifier !== this._touchIdx; i++ );
        // if touch event with the proper identifier isnt found, do nothing
        if( i === touchList.length)     return;
        var touch       = touchList[i];

        event.preventDefault();

        var x           = touch.pageX;
        var y           = touch.pageY;
        return this._onMove(x, y)
}


////////////////////////////////////////////////////////////////////////////////
//                build default stickEl and baseEl                            //
////////////////////////////////////////////////////////////////////////////////

/**
 * build the canvas for joystick base
 */
VirtualJoystick.prototype._buildJoystickBase  = function()
{
        var canvas      = document.createElement( 'canvas' );
        canvas.width    = 126;
        canvas.height   = 126;

        var ctx         = canvas.getContext('2d');
        ctx.beginPath();
        ctx.strokeStyle = this._strokeStyle;
        ctx.lineWidth   = 6;
```

```
        ctx.arc( canvas.width/2, canvas.width/2, 40, 0, Math.PI*2, true);
        ctx.stroke();

        ctx.beginPath();
        ctx.strokeStyle = this._strokeStyle;
        ctx.lineWidth   = 2;
        ctx.arc( canvas.width/2, canvas.width/2, 60, 0, Math.PI*2, true);
        ctx.stroke();

        return canvas;
}

/**
 * build the canvas for joystick stick
 */
VirtualJoystick.prototype._buildJoystickStick  = function()
{
        var canvas      = document.createElement( 'canvas' );
        canvas.width    = 86;
        canvas.height   = 86;
        var ctx         = canvas.getContext('2d');
        ctx.beginPath();
        ctx.strokeStyle = this._strokeStyle;
        ctx.lineWidth   = 6;
        ctx.arc( canvas.width/2, canvas.width/2, 40, 0, Math.PI*2, true);
        ctx.stroke();
        return canvas;
}

////////////////////////////////////////////////////////////////////////////
//                move using translate3d method with fallback to translate > 'top' and 'left'

//      modified from https://github.com/component/translate and dependents
////////////////////////////////////////////////////////////////////////////
```

```
VirtualJoystick.prototype._move = function(style, x, y)
{
	if (this._transform) {
		if (this._has3d) {
			style[this._transform] = 'translate3d(' + x + 'px,' + y + 'px, 0)';
		} else {
			style[this._transform] = 'translate(' + x + 'px,' + y + 'px)';
		}
	} else {
		style.left = x + 'px';
		style.top = y + 'px';
	}
}

VirtualJoystick.prototype._getTransformProperty = function()
{
	var styles = [
		'webkitTransform',
		'MozTransform',
		'msTransform',
		'OTransform',
		'transform'
	];

	var el = document.createElement('p');
	var style;

	for (var i = 0; i < styles.length; i++) {
		style = styles[i];
		if (null != el.style[style]) {
			return style;
		}
	}
}
```

```
VirtualJoystick.prototype._check3D = function()
{
	var prop = this._getTransformProperty();
	// IE8<= doesn't have `getComputedStyle`
	if (!prop || !window.getComputedStyle) return module.exports = false;

	var map = {
		webkitTransform: '-webkit-transform',
		OTransform: '-o-transform',
		msTransform: '-ms-transform',
		MozTransform: '-moz-transform',
		transform: 'transform'
	};

	// from: https://gist.github.com/lorenzopolidori/3794226
	var el = document.createElement('div');
	el.style[prop] = 'translate3d(1px,1px,1px)';
	document.body.insertBefore(el, null);
	var val = getComputedStyle(el).getPropertyValue(map[prop]);
	document.body.removeChild(el);
	var exports = null != val && val.length && 'none' != val;
	return exports;
}
```