หุ่นยนต์ทรงกลม
Sphere Robot

ธัญญา ศรีรัตนาวรรณ
นรินทร์เดช วรรณะเดชะ
รักชนก คำยนต์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

หุ่นยนต์ทรงกลม
Sphere Robot

ธัญญา ศรีรัตนาวรรณ
นรินทร์เดช วรรณะเดชะ
รักชนก คำยนต์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

Sphere Robot

TANYA SREERUTANAWAN
NARINDECH WANADECHA
RAKCHANOK KAMYON

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	โครงงานหุ่นยนต์ทรงกลม Sphere Robot		
นักศึกษาผู้จัดทำ	นางสาวธัญญา ศรีรัตนาวรรณ	**รหัสนักศึกษา**	54010613
	นายนรินทร์เดช วรรณะเดชะ	**รหัสนักศึกษา**	54010674
	นางสาวรักชนก คำยนต์	**รหัสนักศึกษา**	54011066
ปริญญา	วิศวกรรมศาสตรบัณฑิต		
สาขาวิชา	วิศวกรรมการวัดคุม		
ปีการศึกษา	2557		

อาจารย์ผู้ควบคุมปริญญานิพนธ์	ลายมือชื่อ
ผู้ช่วยศาสตราจารย์ ดร.พงษ์ชัย นิลาศ	29/05/58

หัวข้อปริญญานิพนธ์	หุ่นยนต์ทรงกลม	
	Sphere Robot	
นักศึกษาผู้จัดทำ	นายนรินทร์เดช วรรณะเดชะ	รหัสนักศึกษา 54010674
	นางสาวธัญญา ศรีรัตนาวรรณ	รหัสนักศึกษา 54010613
	นางสาวรักชนก คำยนต์	รหัสนักศึกษา 54011066
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตร์ตราจารย์ ดร. พงษ์ชัย นิลาศ	
ปีการศึกษา	2557	

บทคัดย่อ

จุดประสงค์หลักของโครงงานนี้คือสนใจที่คุณลักษณะของ หุ่นยนต์ทรงกลม (Sphere robot) และการทำในส่วนของฮาร์ดแวร์และซอฟแวร์ อีกทั้งได้ศึกษาและทดลองของแต่ละส่วนประกอบ ตัวอย่างเช่น ในส่วนแรกจะพูดถึงในส่วนของการสร้างแรงขับเคลื่อนให้แก่หุ่นยนต์ทรงกลม โดยในส่วนนี้จะมีไอซีขับเคลื่อนมอเตอร์ L293D และมอเตอร์กระแสตรงขนาดไม่เกิน 5 VDC เป็นตัวสร้างการเคลื่อนที่ให้กับหุ่นยนต์ ส่วนที่สองจะเป็นส่วนของการติดต่อสื่อสาร คือโมดูลบลูทูธ หมายเลข HC-05 ซึ่งเป็นเทคโนโลยีการสื่อสารแบบไร้สาย ใช้ในการรับส่งข้อมูลกันระหว่างส่วนไมโครคอนโทรลเลอร์และส่วนอุปกรณ์ติดต่อกับผู้ใช้งาน ส่วนที่สามเป็นส่วนของแอพพลิเค่ชั่น ซึ่งก็คือโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ ซึ่งเป็นหนึ่งในระบบปฏิบัติการที่นิยมมากที่สุดในโลก นำมาใช้ในส่วนของการแสดงผลการติดต่อระหว่างผู้ใช้งานกับหุ่นยนต์ และส่วนสุดท้ายเป็นส่วนของการควบคุม ในโครงงานนี้ได้ใช้บอร์ด อาดูนิโน ดูล (Arduino DUE) ทำหน้าที่เป็นไมโครคอนโทรลเลอร์สำหรับทดสอบการใช้งาน และ PIC ไมโครคอนโทรลเลอร์ ทำหน้าที่เป็นไมโครคอนโทรลเลอร์ที่ต่อร่วมเข้ากับวงจรควบคุมจริง ซึ่งทั้งสองมีบทบาทสำคัญต่อระบบในการควบคุมการทำงานต่างๆของหุ่นยนต์

Thesis Title	Sphere Robot
Authors	Mr.Narindech Wanadecha
	Miss.Tanya Sreeruttanawun
	Miss. Rakchanok Kamyon
Thesis Advisor	Asst. Prof. Dr. Phongchai Nilas
Year	2014

ABSTRACT

The objective of this project is focusing on Sphere robot's characteristic and performance in both hardware part and software part. Also, studying and testing on each Sphere robot's components. For example, first, propulsion part, L293D and DC motor are used in this project in case of creating force to drive the robot. Second, communication part, Bluetooth module is a wireless communication technology involved in this project. Third, application part, Android smartphone, one of the most popular operating system in the world, is used in this project as a Graphic User Interface (GUI) where we can interact with our robot. Finally, controller part, Arduino Due and PIC microcontroller are the important roles of this system which we use Arduino as a testing microcontroller and PIC microcontroller as a real one. Both of them are used to control robot's behavior.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ได้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับความเมตตาจากผู้ช่วยศาสตราจารย์ ดร.พงษ์ชัย นิลาศ ที่ได้ให้คำแนะนำและคำสอนแก่ผู้วิจัยปริญญานิพนธ์นี้ตลอดมา อีกทั้งยังเอื้อเฟื้อสถานที่และเครื่องมือต่างๆที่ใช้ในการทำปริญญานิพนธ์นี้ ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณผู้ช่วยศาสตราจารย์ เชื้อ นกอยู่ และอาจารย์ภาควิศวกรรมการวัดคุมทุกท่าน ที่ได้เอื้อเฟื้อเครื่องมือต่างๆและให้คำปรึกษาในการทำปริญญานิพนธ์ ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบคุณ พี่ปุ้งกี๋ นางสาว ธนานุช รัตนจรัสโรจน์ รุ่นพี่ปริญญาโท ภาควิชาวิศวกรรมการวัดคุม ที่คอยดูแล และเป็นที่ปรึกษาให้คำแนะนำที่ดีเสมอมา

ขอกราบขอบพระคุณ คุณพ่อ, คุณแม่ และเพื่อนๆภาควิชาวิศวกรรมระบบควบคุม และวิศวกรรมการวัดคุม อันเป็นที่รักทุกคน ที่คอยสนับสนุน และเป็นแรงบันดาลใจในการทำให้ปริญญานิพนธ์ฉบับนี้ ให้เสร็จลุล่วงไปได้

ผู้วิจัยขอให้ปริญญานิพนธ์ฉบับนี้ ได้ให้ความรู้ ประโยชน์ และคุณค่า กับผู้อ่านไม่มากก็น้อย

คณะผู้จัดทำ

สารบัญ

หน้า

บทคัดย่อภาษาไทย.....I
บทคัดย่อภาษาอังกฤษ.....II
กิตติกรรมประกาศ.....III
สารบัญ.....IV
สารบัญตาราง.....VI
สารบัญภาพ.....VII

บทที่ 1 บทนำ.....1
1.1 ความสำคัญของปริญญานิพนธ์.....1
1.2 วัตถุประสงค์ของปริญญานิพนธ์.....1
1.3 ขอบเขตของปริญญานิพนธ์.....1
1.4 ขั้นตอนการศึกษา.....2
1.5 ประโยชน์ที่คาดว่าจะได้รับ.....2

บทที่ 2 ทฤษฎี.....3
2.1 หลักการและทฤษฎีการเคลื่อนที่ของหุ่นยนต์ทรงกลม.....3
2.1.1 การเคลื่อนที่โดยอาศัยการวิ่งของหุ่นยนต์มีล้อภายในทรงกลม.....3
2.1.2 การเคลื่อนที่โดยอาศัยการแกว่งของตุ้มน้ำหนัก.....5
2.1.3 การเคลื่อนที่โดยอาศัยการเปลี่ยนรูปทรงของตัวหุ่นยนต์.....5
2.2 ปิก ไมโครคอนโทรลเลอร์.....7
2.2.1 ความเป็นมาของ ปิก ไมโครคอนโทรลเลอร์.....7
2.2.2.คุณสมบัติต่างๆของ PIC.....8
2.2.2.1 ความเร็วของ PIC.....9
2.2.2.2 หน่วยความจำของ PIC.....9
2.2.2.3 สถาปัตยกรรมของ PIC.....9
2.2.3 ลำดับขั้นตอนการใช้งาน PIC.....10
2.2.3.1 การเขียนโปรแกรมภาษาซี.....10
2.2.3.2 การจำลองการทำงาน.....11
2.2.3.3 การถ่ายโอนชุดคำสั่งจากคอมพิวเตอร์สู่ตัว PIC.....11
2.2.3.4 การนำ PIC ไมโครคอนโทรลเลอร์ไปต่อใช้งาน.....11
2.2.4 รายละเอียดคุณสมบัติของไมโครคอนโทรลเลอร์ PIC16F87XA.....11
2.2.5 โปรแกรม CCS 'C' คอมไพเลอร์.....13
2.2.6 เครื่องโปรแกรมเมอร์ PICKit2.....14
2.2.6.1 เครื่องโปรแกรมเมอร์ PX-700.....14
2.3 บอร์ดคอนโทรลเลอร์ อาร์ดูโน่ (Arduino).....15

สารบัญ (ต่อ)

หน้า

2.3.1 จุดเด่นที่ทำให้บอร์ด อาร์ดูโน่ เป็นที่นิยม....16
2.3.2 Layout & Pin out Arduino Board (Model: Arduino DUE)....18
2.3.3 รูปแบบเราเขียนโปรแกรมบนอาร์ดูโน่....19
2.4 เทคโนโลยีบลูทูธ (BLUETOOTH TECHNOLOGY)....21
2.4.1 ความหมายของบลูทูธ....21
2.4.2 ที่มาของเทคโนโลยีบลูทูธ....21
2.4.3 ลักษณะการทำงานขอบลูทูธ....22
2.4.4 โมดูลบลูทูธ รุ่น HC- 05....23
2.4.5 คำสั่ง เอที คอมมานด์ (AT command)....24
2.4.5.1 การใช้คำสั่ง เอที คอมมานด์....25
2.5 ระบบปฏิบัติการแอนดรอยด์....26
2.5.1 โครงสร้างของแอนดรอยด์....27
2.5.1.1 Applications....28
2.5.1.2 Application Framework....28
2.5.1.3 Libraries....29
2.5.1.4 Android Runtime....29
2.5.1.5 Linux Kernel....29
2.5.2 โปรแกรมและเครื่องมือในการพัฒนา....30
2.5.2.1 JDK (Java Development Kit)....30
2.5.2.2 Eclipse....30
2.5.2.3 Android SDK (Android Software Development Kit)...31
2.5.2.4 ADT(Android Development Tools)....32
2.5.3 การทดลองรันแอพพลิเคชั่น....32
2.5.3.1 รันบน อีมูเลเตอร์....32
2.5.3.2 รันบนอุปกรณ์จริง....33
2.5.4 โฟลเดอร์และไฟล์ในโปรเจคแอพ....33
2.5.4.1 src....33
2.5.4.2 gen....33
2.5.4.3 Android....33
2.5.4.4 assets....33
2.5.4.5 bin....33
2.5.4.6 red....34
2.5.4.7 AndroidManifest.xml....34
2.5.5 Activity และ Layout....34
2.5.5.1 activity....34

สารบัญ (ต่อ)

หน้า

2.5.5.2 Layout....34
2.5.6 การพัฒนาโปรแกรม....34
2.5.7 การทำงานของ Activity....35
2.5.8 การจัดเก็บข้อมูลใน Activities....36
2.5.9 การใช้งาน Threading....36
2.5.10 การใช้งาน Handler....36
2.6 ยางธรรมชาติ (Natural rubber)....37
2.6.1 โครงสร้างทางเคมี....37
2.6.2 สมบัติทั่วไปของยางธรรมชาติวัลคาไนซ์....38
2.6.2.1 ความแข็ง....38
2.6.2.2 ความต้านทานต่อแรงดึง....38
2.6.2.3 ความสามารถในการยืดจนขาด....38
2.6.2.4 ความต้านทานต่อการฉีก....38.
2.6.2.5 การใช้งานที่อุณหภูมิต่ำ....39
2.6.2.6 ความทนทานต่อการเสื่อมสภาพ....39
2.6.2.7 ความทนทานต่อความร้อน....39

บทที่ 3 วิธีการดำเนินงาน....40
3.1 กล่าวนำ....40
3.2 แนวทางการดำเนินงาน....40
3.2.1 การทำงานโดยสังเขปของหุ่นยนต์....40
3.2.2 ส่วนของซอฟท์แวร์ควบคุมการทำงานของหุ่นยนต์....41
3.2.2.1 โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง....41
3.2.2.2 โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง....48
3.2.3 ซอฟท์แวร์ส่วนหน้าต่างผู้ใช้งาน....50
3.2.3.1 แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง....51
3.2.3.2 แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง....63
3.3. วงจรไฟฟ้าควบคุมการทำงานของหุ่นยนต์....65
3.3.1 ขั้นตอนการทำวงจร....65
3.3.2 ส่วนของวงจรควบคุมความเร็วของมอเตอร์....78

สารบัญ (ต่อ)

หน้า

3.3.2.1 การใช้ซอฟท์แวร์ควบคุมความเร็ว....78
3.3.3 โครงสร้างของหุ่นยนต์และส่วนประกอบอื่นๆ....79
3.3.3.1 การหล่อแผ่นยางพารา....79
3.3.3.1.1 อุปกรณ์ที่ใช้....79
3.3.3.2 ขั้นตอนการหล่อแผ่นยางพารา....79
3.3.3.3 การทำล้อรถ....81
3.3.3.4 ผิวภายนอกของทรงกลม....81
3.3.3.5 โครงรถ....83
3.3.3.5.1 รถคันเล็ก....83
3.3.4.5.2 รถคันกลาง....84
3.3.4.5.3 รถคันใหญ่....85

บทที่ 4 ผลการทดลอง....86
4.1 การทดลองแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับแบบไม่ต่อเนื่อง....86
4.2 การทดลองแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับแบบต่อเนื่อง....89
4.3 คุณสมบัติของหุ่นยนต์ Sphere Robot ในรูปแบบต่างๆ....94
4.4 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดใหญ่....95
4.5 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดกลาง....97
4.6 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดเล็ก....98

บทที่ 5 สรุปผลการทดลองและข้อเสนอนะ....100
5.1 สรุปผล....100
5.2 ข้อเสนอแนะ....100

บรรณานุกรม....101

ภาคผนวก....102

สารบัญตาราง

ตารางที่ **หน้า**

2.1 ตารางเปรียบเทียบคุณสมบัติของบอร์ดอาร์ดูโน่ แต่ละรุ่น.....17
2.2 ชุดคำสั่ง เอที คอมมานด์ สำหรับควบคุมการทำงานของโมดูลบลูทูธ.....25
2.3 ประเภทการทำงานของแอปพลิเคชั่น.....34
4.1 ตารางแสดงผลการทดลองคุณลักษณะและรายละเอียดโดยรวม
ของหุ่นยนต์ Sphere Robot ในแต่ละรูปแบบ.....104

สารบัญรูป

รูปที่ หน้า

2.1 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยหุ่นยนต์มีล้อวิ่งอยู่ภายใน....4
2.2 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยหุ่นยนต์มีล้อวิ่งอยู่ภายใน มีสปริงดันผิวฝั่งตรงข้าม....4
2.3 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยการแกว่งของลูกตุ้มเพนดูลัม....5
2.4 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยการเปลี่ยนรูปร่างของหุ่น....6
2.5 ไดอะแกรมแสดงขาการทำงานต่างๆของ PIC เบอร์ 16F877A และ 16F876A....8
2.6 แสดงสถาปัตยกรรมภายในของ PIC เบอร์ 16F87XA....10
2.7 รูปภาพแสดงตัวไมโครคอนโทรลเลอร์ PIC เบอร์ 16F876A (รูปซ้าย)
และเบอร์ 16F877A (รูปขวา, กลาง)....13
2.8 รูปภาพแสดงหน้าต่างของโปรแกรม CCS 'C' compiler....13
2.9 รูปภาพแสดงหน้าต่างเครื่องโปรแกรมเบอร์ PX-700....14
2.10 รูปภาพแสดงบอร์ด อาร์ดูโน่ เชื่อมต่อกับอุปกรณ์ต่างๆ....16
2.11 แสดงส่วนประกอบในบอร์ด Arduino....18
2.12 รูปแบบการเขียนโปรแกรมบน Arduino....19
2.13 ภาพแสดงการเลือกรุ่นบอร์ด Arduino ที่ต้องการ Upload....19
2.14 ภาพแสดงการเลือกหมายเลข Com port ของบอร์ด....20
2.15 ภาพแสดงด้านซ้ายกดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ
Compile โค้ดโปรแกรม ภาพขวาแสดงการ Upload โค้ดโปรแกรม....20
2.16 โครงสร้างเครือข่ายเทคโนโลยีบลูทูธ....21
2.17 โมดูลบลูทูธรุ่น HC-05....24
2.18 แสดงหน้าต่างซีเรียลมอนิเตอร์....25
2.19 แสดงโค้ดโปรแกรมบนบอร์ดอาร์ดูโน่....26
2.20 โครงสร้างของระบบปฏิบัติการแอนดรอยด์....27
2.21 รูปภาพแสดงโปรแกรมพัฒนา Eclipse....31
2.22 รูปภาพแสดงหน้าต่างใช้งานสำหรับพัฒนาแอพพลิเคชั่นบนโปรแกรม Eclipse....31
2.23 รูปภาพแสดงหน้าต่างใช้งานสำหรับโปรแกรมจำลองโทรศัพท์มือถือ Genymotion....32
2.24 การทำงานของ Activity....35
2.45 แสดงโครงสร้างโมเลกุลของยางธรรมชาติ....37
3.1.1 บล็อกไดอะแกรมแสดงการทำงานของหุ่นยนต์....41
3.1.2 การทดสอบโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับบนบอร์ด Arduino....42
3.1.3 ตัวอย่างโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับบนบอร์ด Arduino....42
3.1.4 ขั้นตอนการสร้าง Source file ใหม่และตั้งชื่อ Source file....43
3.1.5 รูปภาพแสดงการเขียนโปรแกรมลงบน โปรแกรม CCS 'C' Compiler....44
3.1.6 ขั้นตอนการสร้าง ไฟล์ .HEX สำหรับโปรแกรมใส่ไมโครคอนโทรลเลอร์....44
3.1.7 ขั้นตอนเริ่มต้นการโปรแกรมไมโครคอนโทรลเลอร์ด้วยโปรแกรม PICKit2....45
3.1.8 ขั้นตอนการเรียกไฟล์ .HEX ของโค้ดที่ได้เขียนไว้มาโปรแกรมลงบนไมโครคอนโทรลเลอร์....45

สารบัญรูป(ต่อ)

3.1.9 ขั้นตอนการโปรแกรมลงบนไมโครคอนโทรลเลอร์ที่เสร็จสมบูรณ์แล้ว....46
3.1.10 การทดสอบการทำงานของโปรแกรมควบคุมบน PIC ไมโครคอนโทรลเลอร์....46
3.1.11 โฟลว์ชาร์ทการทำงานของโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง....48
3.1.12 โฟลว์ชาร์ทแสดงการทำงานของโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง....50
3.2.1 รูปภาพแสดงโปรแกรม Eclise Juno....51
3.2.2 รูปภาพแสดงขั้นตอนเลือก Workspace....51
3.2.3 รูปภาพแสดงขั้นตอนการสร้างโปรเจค....52
3.2.4 รูปภาพแสดงขั้นตอนการตั้งชื่อและเลือกเวอร์ชั่นระบบปฏิบัติการ....52
3.2.5 รูปภาพแสดงขั้นตอนการตั้งค่าโปรเจค....53
3.2.6 รูปภาพแสดงขั้นตอนการเลือก และปรับแต่งภาพที่จะใช้เป็นไอคอน....53
3.2.7 รูปภาพแสดงขั้นตอนการเลือก รูปแบบหน้าต่าง Layout....54
3.2.8 รูปภาพแสดงขั้นตอนการตั้งชื่อไฟล์ .java....54
3.2.9 รูปภาพแสดงหน้าต่างสำหรับพัฒนาแอพพลิเคชั่นบนโปรแกรม Eclipse....55
3.2.10 รูปภาพแสดงหน้าต่างหลักโปรแกรมอีมูเลเตอร์ Genymotion....56
3.2.11 รูปภาพแสดงหน้าต่างสร้าโทรศัพท์มือถือโปรแกรมอีมูเลเตอร์ Genymotion....56
3.2.12 รูปภาพแสดงรายละเอียดการสร้างโทรศัพท์มือถือบนโปรแกรม Genymotion....57
3.2.13 รูปภาพแสดงการสร้างโทรศัพท์มือถือเสร็จสมบูรณ์บนโปรแกรม Genymotion....57
3.2.14 รูปภาพแสดงการสร้างโทรศัพท์มือถือเสร็จสมบูรณ์บนโปรแกรม Genymotion....58
3.2.15 รูปภาพแสดงการรันโทรศัพท์มือถือจำลองบนโปรแกรม Genymotion....59
3.2.16 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์เวอร์ชั่น 2.x และ 3.x....59
3.2.17 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์เวอร์ชั่น 2.x และ 3.x....60
3.2.18 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์วเวอร์ชั่น 4.x....60
3.2.19 รูปภาพแสดงการติดตั้งไดร์ฟเวอร์ของโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์บนคอมพิวเตอร์....61
3.2.20 รูปภาพแสดงคำขออนุญาตเปิดใช้งาน USB debugging บนโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์....61
3.2.21 รูปภาพแสดงการเริ่มต้นการรันแอพพลิเคชั่นจากโปรแกรม Eclipse....62
3.2.22 รูปภาพแสดงการเลือกโทรศัพท์มือถือก่อนเริ่มการรันแอพพลิเคชั่น จากโปรแกรม Eclipse....62
3.2.23 รูปภาพแสดงโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสำหรับควบคุม การเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง....63

สารบัญรูป(ต่อ)

3.2.24 รูปภาพแสดงโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสำหรับควบคุม
การเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง....64
3.3.1 แสดงการวาด schematic ในโปรแกรม Altium....65
3.3.2 แสดงการแปลงจากไฟล์ schematic เป็น PCB....66
3.3.3 ภาพแสดงลาย PCB ที่ได้ทำการออกแบบไว้
ซึ่งเป็น PCB กัดปริ้น 2 ด้าน....66
3.3.4 ภาพแสดงการเปลี่ยนไฟล์ลายวงจรเป็นไฟล์ PDF เพื่อทำการกัดปริ้นต่อไป....67
3.3.5 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF....67
3.3.6 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(1)....68
3.3.7 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(2)....68
3.3.8 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(3)....69
3.3.9 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(4)....69
3.3.10 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(5)....69
3.3.11 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(6)....70
3.3.12 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(7)....70
3.3.13 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(8)....71
3.3.14 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB....72
3.3.15 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(1)....72
3.3.16 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(2)....72
3.3.17 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(3)....72
3.3.18 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(4)....73
3.3.19 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(5)....73
3.3.20 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(6)....73
3.3.21 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(7)....74
3.3.22 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(8)....74
3.3.23 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(9)....75
3.3.24 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(10)....75
3.3.25 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(11)....75
3.3.26 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(12)....76
3.3.27 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(13)....76
3.3.28 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(14)....76
3.3.29 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(15)....77
3.3.30 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(16)....77
3.3.31 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(17)....77
3.3.32 ภาพแสดงวงจรที่ได้ทำการกัดปริ้นมาของ Sphere robot
คันใหญ่(ซ้าย), คันกลาง(กลาง), คันขวา(ขวา)....78

สารบัญรูป(ต่อ)

3.3.34 ภาพแสดงวงจรควบคุมกระแสโดยใช้ Op-amp....79
3.3.35 ภาพแสดงวงจรควบคุมกระแสที่ได้ทำการต่อไว้....80
3.3.36 ภาพแสดงวงจรควบคุมความเร็วมอเตอร์แบบไอซี 555 กับ มอสเฟส....81
3.3.37 ภาพแสดงวงจรควบคุมความเร็วมอเตอร์แบบไอซี 555 กับ Power transistor....81
3.3.38 ภาพวงจรที่ได้ทำการต่อไว้....82
3.3.39 ภาพแสดงทิศทางการหมุนของมอเตอร์....82
3.3.40 ภาพแสดงวงจรขับมอเตอร์แบบกลับทางหมุน H –bridge แบบ NPN(ซ้าย) , PNP(ขวา)....83
3.3.41 ภาพวงจรแสดงการต่อวงจร H-Bridge แบบ NPN....83
3.3.42 ภาพแสดงวงจรกลับทางหมุนของมอเตอร์แบบ NPN....84
3.3.43 ภาพแสดงวงจรกลับทางหมุนของมอเตอร์แบบ PNP....84
3.3.44 ภาพแสดงการต่อวงจรรวมระหว่างวงจรความคุมความเร็วมอเตอร์กับวงจร H-bridge....85
3.3.45 ภาพแสดงวงจรที่ได้ทำการต่อไว้เพื่อทดสอบการทำงาน....86
3.3.46 ภาพแสดงการต่อวงจรควบคุมความเร็วกับวงจรกลับทางหมุน ของมอเตอร์ซึ่งปรับความเร็วโดย digital resister....86
3.3.47 ภาพแสดงการต่อวงจรเพื่อทดสอบการทำงานจริง....87
3.3.48 ภาพแสดงไอซีขับมอเตอร์ L293D....87
3.3.49 แสดงอุปกรณ์ที่ใช้หล่อแผ่นยางพารา....88
3.3.50 แสดงขั้นตอนการหล่อแผ่นยางพารา....88
3.3.51 แสดงขั้นตอนการหล่อแผ่นยางพารา(1)....88
3.3.52 แสดงขั้นตอนการหล่อแผ่นยางพารา(2)....89
3.3.53 แสดงขั้นตอนการหล่อแผ่นยางพารา(3)....89
3.3.54 แสดงขั้นตอนการหล่อแผ่นยางพารา(4)....90
3.3.55 แสดงการติดยางพาราที่ล้อรถ....90
3.3.56 แสดงผิวภายนอกของทรงกลมของรถคันใหญ่....91
3.3.57 แสดงผิวภายนอกของทรงกลมของรถคันกลาง....91
3.3.58 แสดงผิวภายนอกของทรงกลมของรถคันเล็ก....91
3.3.59 แสดงรถคันเล็ก....92
3.3.60 แสดงรถคันกลาง....93
3.3.61 แสดงรถคันใหญ่....94
4.1 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า)....95
4.2 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่เลี้ยวไปทางขวา)....96
4.3 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่เลี้ยวไปทางซ้าย)....96

สารบัญรูป(ต่อ)

4.4 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบไม่ต่อเนื่อง (สั่งเคลื่อนทีถอยหลัง).....97
4.5 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบไม่ต่อเนื่อง (สั่งหยุดการเคลื่อนที่).....98
4.6 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบไม่ต่อเนื่อง (หน้าต่างผู้ใช้งานแอพพลิเคชั่น).....98
4.7 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง.....98
4.8 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (หลังจากแอพพลิเคชั่นกับโมดูลบลูทูธได้เชื่อมต่อกันแล้ว).....99
4.9 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (สั่งหยุดการเคลื่อนที่).....99
4.10 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า).....100
4.11 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า และเพิ่มความเร็ว).....101
4.12 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (สั่งเคลื่อนที่ถอยหลัง และเพิ่มความเร็ว).....101
4.13 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (สั่งเคลื่อนที่เดินหน้า ในทิศทางเฉียงไปทางขวา และเพิ่มความเร็ว).....102
4.14 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ
แบบต่อเนื่อง (หน้าต่างผู้ใช้งานแอพพลิเคชั่น).....103
4.15 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง..103
4.16 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดใหญ่.....105
4.17 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดกลาง.....106
4.18 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดเล็ก.....108

บทที่ 1

บทนำ

1.1 ความสำคัญของปริญญานิพนธ์

ในโลกปัจจุบันเทคโนโลยีมีการพัฒนาไปอย่างรวดเร็ว ซึ่งสามารถสังเกตได้จากสิ่งรอบตัว ที่ไม่ว่าอะไรก็ล้วนแต่สะดวกสบายมากขึ้นด้วยศักยภาพของการพัฒนาของเทคโนโลยี การพัฒนาของเทคโนโลยีนั้นก็เกิดขึ้นได้ด้วยปัจจัยหลายอย่าง อาจจะเพื่อตอบสนองความต้องการทางด้านอุตสาหกรรม เพื่ออำนวยความสะดวกให้แก่ชีวิตของมนุษย์ เพื่อการประหยัดพลังงาน หรืออาจจะแค่เพื่อการสร้างสรรค์และตอบสนองแนวคิดส่วนบุคคล ซึ่งหากจะเปรียบเทียบแล้วหุ่นยนต์ก็คงเป็นเทคโนโลยีที่ถูกพัฒนาขึ้นมาจากแนวคิดอันสร้างสรรค์ของมนุษย์ที่ต้องการพัฒนาโครงสร้างให้มีกลไกการเคลื่อนที่อัตโนมัติที่ถูกควบคุมด้วยสมองกลหรือที่เรียกอีกอย่างหนึ่งว่าไมโครคอนโทรลเลอร์ (Microcontroller)

โดยปริญญานิพนธ์นี้เป็นการนำเสนอเทคโนโลยีหุ่นยนต์ในอีกรูปแบบหนึ่งโดยเป็นการศึกษาและทดลองจัดทำ Sphere Robot ซึ่งเป็นหุ่นยนต์ที่มีลักษณะเป็นทรงกลม ไม่มีแขนกล หรือ ขา ที่ใช้ในการเคลื่อนไหว ซึ่งการจะจัดทำได้นั้นจำเป็นต้องมีการศึกษาและค้นคว้าเป็นอย่างดี โดยเฉพาะในส่วนของการควบคุมการเคลื่อนที่ ต้องมีความเข้าใจในลักษณะการเคลื่อนไหวและ การ Orientation ของหุ่นยนต์ รวมถึงปัจจัยที่เกี่ยวของต่างๆ ดังนั้นโครงงานฉบับนี้จึงได้ถูกจัดทำขึ้นเพื่อเป็นแนวทางในการศึกษาและพัฒนาต่อไป

1.2 วัตถุประสงค์ของปริญญานิพนธ์

1. ได้ศึกษาคุณสมบัติ Characteristic และ Orientation ของหุ่นยนต์ทรงกลม
2. ได้ศึกษาหลักการเคลื่อนที่ของหุ่นยนต์ทรงกลมในหลายๆรูปแบบ
3. สามารถเลือกใช้อุปกรณ์อิเล็กทรอนิกส์ต่างๆเพื่อให้สามารถใช้งานกับตัวหุ่นยนต์ที่มีขนาดที่จำกัดได้อย่างเหมาะสมได้
4. ได้รับความรู้จากการศึกษาวิธีการควบคุมหุ่นยนต์ให้มีความเสถียรในการทำงาน

1.3 ขอบเขตของโครงงาน

1. ศึกษาคุณสมบัติและประสิทธิภาพการทำงานของรูปแบบการเคลื่อนที่ของหุ่นยนต์ทรงกลมในหลายๆรูปแบบ
2. ศึกษาวงจรไฟฟ้าสำหรับสร้างแรงขับเคลื่อนให้กับหุ่นยนต์ทรงกลม

3. ออกแบบและประกอบหุ่นยนต์ทรงกลม

1.4 ขั้นตอนการศึกษา

1. ศึกษาวิธีการออกแบบหุ่นยนต์ และนำข้อมูลมาวิเคราะห์เพื่อนำมาประกอบการตัดสินใจในการเลือกอุปกรณ์ที่มาใส่ในตัวหุ่นยนต์อย่างเหมาะสม ไม่ว่าจะเป็น โครงสร้างของหุ่นยนต์ ส่วนของแหล่งจ่ายไฟเลี้ยงวงจร ส่วนของการสื่อสาร ส่วนของการขับเคลื่อน และส่วนของการควบคุม รวมถึงการแสดงผล เป็นต้น
2. ศึกษาการใช้งานของอุปกรณ์ ไมโครคอนโทรลเลอร์ ที่ใช้ควบคุมหุ่นยนต์และอุปกรณ์ที่เกี่ยวข้อง ตลอดจนออกแบบและเขียนโปรแกรมควบคุมการทำงานให้ตรงตามเงื่อนไขที่ต้องการมากที่สุด
3. ศึกษาคุณสมบัติในส่วนของการขับเคลื่อนตัวหุ่นยนต์ รวมถึงสร้างและออกแบบวงจรไฟฟ้าที่ใช้สำหรับควบคุมการทำงานต่างๆ
4. ศึกษาในส่วนของการสื่อสารผ่านเครือข่ายไร้สาย การรับ-ส่งข้อมูล การคัดกรองข้อมูลเพื่อให้ได้ข้อมูลที่ถูกต้องสำหรับมาใช้ควบคุมการเคลื่อนที่ของหุ่นยนต์
5. ศึกษาในส่วนของการแสดงผลบนหน้าต่างอินเตอร์เฟส โดยเฉพาะอย่างยิ่ง การแสดงผลบนโทรศัพท์มือถือสมาร์ทโฟน รวมไปถึงการสร้างหน้าต่างควบคุมการทำงานของหุ่นยนต์ด้วย

1.5 ประโยชน์ที่คาดว่าจะได้รับ

การสร้างหุ่นยนต์เป็นการประยุกต์นำความรู้ที่ได้จากการศึกษาในด้านต่างๆ มาใช้ เพื่อทดสอบถึงปฏิภาณและความสามารถในการนำความรู้มาใช้ในการปฏิบัติจริง รู้จักการวิเคราะห์และนำข้อมูลความรู้ต่างๆ มาใช้ให้เกิดประโยชน์ ไม่ว่าจะด้วยการศึกษาด้วยตนเองหรือจากการถามผู้รู้ พร้อมทั้งมีไหวพริบในการปรับปรุงแก้ไขการทำงานเมื่อเกิดปัญหาขึ้น อีกทั้งยังเป็นการฝึกฝนตนเองให้รู้จักที่จะปฏิบัติงานร่วมกับผู้อื่น มีการรับฟังความคิดเห็นและรับฟังความคิดเห็นและรับผิดชอบในการปฏิบัติงานร่วมกันจนเกิดความสำเร็จ ซึ่งจะก่อให้เกิดเป็นประสบการณ์และความภาคภูมิใจในการที่จะปฏิบัติงานต่อไปในภายภาคหน้า

บทที่ 2

ทฤษฎี

2.1 หลักการและทฤษฎีการเคลื่อนที่ของหุ่นยนต์ทรงกลม

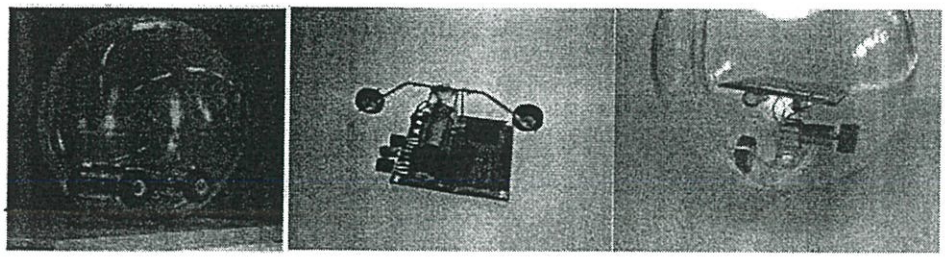
(Principles of Sphere Robot's Movement)

นักวิจัย และบริษัทผู้ผลิตหุ่นยนต์ทรงกลมนั้นต่างมีแนวคิดในการสร้างการเคลื่อนที่ให้กับหุ่นยนต์ที่หลากหลายแตกต่างกันไป ในรายงานฉบับนี้จึงขอยกตัวอย่างรูปแบบการสร้างการเคลื่อนที่ให้กับหุ่นยนต์มาพอสังเขป

1. การเคลื่อนที่โดยอาศัยการวิ่งของหุ่นยนต์มีล้อภายในทรงกลม
2. การเคลื่อนที่โดยอาศัยการแกว่งของตุ้มน้ำหนัก (Pendulum driven)
3. การเคลื่อนที่โดยอาศัยการเปลี่ยนรูปทรงของตัวหุ่นยนต์

2.1.1 การเคลื่อนที่โดยอาศัยการวิ่งของหุ่นยนต์มีล้อภายในทรงกลม (Internal car model)

หรืออีกชื่อหนึ่งคือ "Hamster ball" เป็นรูปแบบการเคลื่อนที่อย่างง่ายๆของหุ่นยนต์ทรงกลมที่อาศัยหุ่นยนต์ขนาดเล็กที่เคลื่อนที่ด้วยล้อวิ่งอยู่ภายในทรงกลม(คล้ายกับการวิ่งของหนูแฮมสเตอร์ในลูกบอล) สามารถเลือกใช้หุ่นยนต์แบบล้อเดี่ยว หรือ หลายล้อก็ได้ ล้อของหุ่นขนาดเล็กภายในจะสัมผัสกับพื้นผิวของหุ่นทรงกลมภายนอก ด้วยน้ำหนักของตัวหุ่นยนต์มีล้อบวกกับแรงขับเคลื่อนด้วยแล้ว มันจึงสามารถสร้างแรงขับหุ่นยนต์ทรงกลมภายนอกให้สามารถเคลื่อนที่ได้ และถ้าหากเราต้องการเปลี่ยนทิศทางการเคลื่อนที่ของหุ่นยนต์ทรงกลม เช่น เลี้ยวซ้าย เลี้ยวขวา เราเพียงแค่ทำการบังคับทิศทางการเคลื่อนที่ของหุ่นยนต์ล้อขนาดเล็ก ตัวอย่างหุ่นยนต์ล้อที่จะนำมาใส่ในหุ่นยนต์ทรงกลมอย่างง่ายๆก็คือ รถบังคับวิทยุ



รูปที่ 2.1 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยหุ่นยนต์มีล้อวิ่งอยู่ภายใน

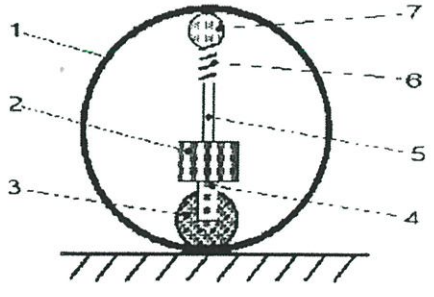
ข้อดี

การติดตั้งและการออกแบบสามารถทำได้โดยง่าย การควบคุมไม่มีความซับซ้อนมาก

ข้อเสีย

การควบคุมเป็นไปได้ยากถ้าหากต้องการความละเอียดและความแม่นยำในการเคลื่อนที่ อาจจะเกิดเหตุการณ์ลื่นไถลของหุ่นยนต์มีล้อได้หากว่าล้อไม่ได้สัมผัสกับผิวของหุ่นยนต์ทรงกลมอย่างเต็มที่

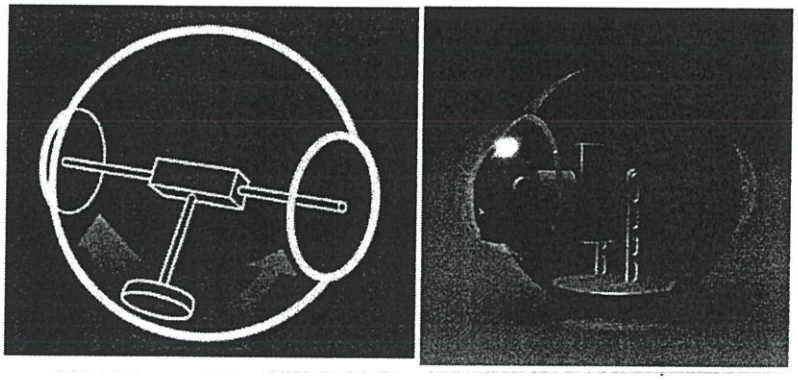
อีกรูปแบบหนึ่งของหุ่นยนต์มีล้อวิ่งอยู่ภายในหุ่นยนต์ทรงกลมก็คือ เป็นหุ่นยนต์มีล้อแบบล้อเดี่ยว ยึดเข้ากับแกนสปริงและปลายฝั่งตรงข้ามเป็นล้อเล็กๆหรือลูกกลิ้งเพื่อลดแรงเสียดทานที่กระทำต่อผิวของหุ่นยนต์ทรงกลม รูปแบบนี้จะช่วยให้ล้อหุ่นยนต์มีล้อขนาดเล็กสัมผัสกับผิวของหุ่นยนต์ทรงกลมได้อย่างเต็มที่ ลดโอกาสที่จะเกิดการลื่นไถลลงได้



รูปที่ 2.2 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยหุ่นยนต์มีล้อวิ่งอยู่ภายใน มีสปริงดันผิวฝั่งตรงข้าม โดยมีส่วนประกอบตามรูปคือ 1. ตัวหุ่นยนต์ทรงกลม 2. ส่วนควบคุมการสั่งการมอเตอร์ 3. ล้อขับเคลื่อน 4. แกนต่อล้อ 5. แกนต่อกับสปริง 6. สปริง 7. ลูกกลิ้ง

2.1.2 การเคลื่อนที่โดยอาศัยการแกว่งของตุ้มน้ำหนัก (pendulum driven)

รูปแบบนี้เป็นที่นิยมใช้มากในงานอุตสาหกรรม โดยโมเดลลูกตุ้มเพนดูลัม นี้จะประกอบไปด้วย แกนเพลาที่ถูกเจาะทะลุผ่านตัวหุ่นยนต์ทรงกลม ตัวเพนดูลัม และตุ้มบ็อบ ถ่วงน้ำหนักก็จะหมุนรอบกับแกนเพลาดังกล่าว ส่งผลให้จุดศูนย์กลางของมวลนั้นเลื่อนห่างออกไปจากจุดศูนย์กลาง และจะทำให้ตัวหุ่นยนต์ทรงกลมเริ่มหมุน อีกทั้งการเลื่อนของเพนดูลัม ไปทางซ้ายและขวาตามแนวแกนเพลา ก็จะส่งผลให้เกิดการเลื่อนของศูนย์กลางมวลไปทางซ้ายและขวาด้วยเช่นกัน หุ่นยนต์ทรงกลมก็จะหันไปตามทิศทางดังกล่าว



รูปที่ 2.3 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยการแกว่งของลูกตุ้มเพนดูลัม

ยิ่งน้ำหนักของลูกตุ้มบ็อบ เพิ่มขึ้น แรงบิดที่จะพาตัวหุ่นยนต์ทรงกลมให้เคลื่อนที่ไปได้ก็จะยิ่งสูงขึ้น แต่อย่างไรก็ตาม ยิ่งเพิ่มน้ำหนักลูกตุ้มบ็อบ ก็เท่ากับว่าหุ่นยนต์จะต้องหนักขึ้น ดังนั้น ข้อเสียส่วนมากของการเคลื่อนที่รูปแบบนี้คือ ถ้าหากต้องการให้หุ่นยนต์เคลื่อนที่บนพื้นผิวที่มีความชัน ก็จะทำให้เคลื่อนขึ้นยาก แต่อย่างไรก็ตาม หากมีการออกแบบที่ดีแล้วนั้น หุ่นยนต์ทรงกลมก็ยังสามารถที่จะเคลื่อนที่ขึ้นบนเนินที่มีความชันประมาณ 30 องศาได้

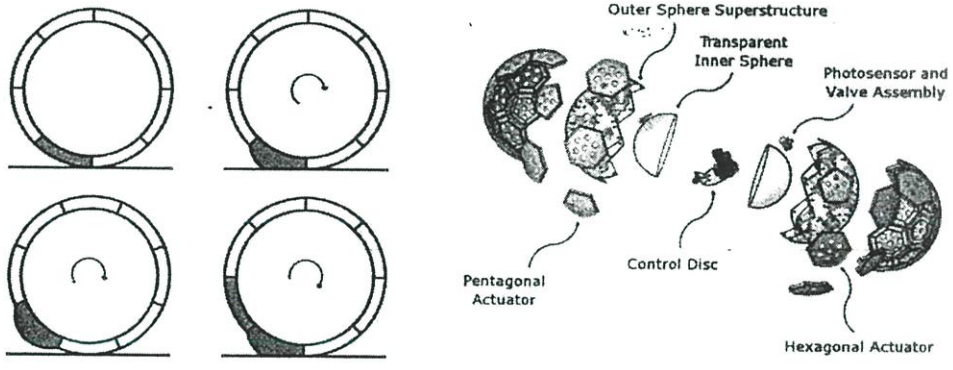
ตัวอย่างของหุ่นยนต์ทรงกลมที่เคลื่อนที่ด้วยหลักการลูกต้มเพนดูลัม นี้คือ หุ่น GroundBot ของค่าย Rotundus

2.1.3 การเคลื่อนที่โดยอาศัยการเปลี่ยนรูปทรงของตัวหุ่นยนต์

รูปแบบการเคลื่อนที่แบบนี้นั้นยังใหม่และมีแนวคิดที่น่าสนใจอยู่หลายประการ เพราะแทนที่จะใส่ระบบการเคลื่อนไหวด้วยชิ้นส่วนทางกลต่างๆเข้าไปในตัวหุ่นยนต์นั้น ตัวหุ่นยนต์เองจะทำการเปลี่ยนรูปทรงของตัวหุ่น ซึ่งทำได้โดยบีบอัดตัวเอง ทำให้ตัวหุ่นเปลี่ยนรูปร่างไป หรือจะใช้

สภาพแวดล้อมต่างๆเข้าช่วย เช่น มีแรงลม หรือแรงน้ำมากระทำบนตัวหุ่นยนต์ ถึงแม้ว่าการเคลื่อนที่ด้วยรูปแบบนี้จะดูหลากหลายและยืดหยุ่นกว่าการเคลื่อนไหวด้วยชิ้นส่วนทางกลก็ตาม แนวคิดนี้ก็ยังคงใหม่อยู่ถ้าเทียบกับแนวคิดก่อนๆ และประสิทธิภาพในการใช้งานของมันก็ยังคงเป็นเรื่องที่จะต้องค้นคว้าวิจัยต่อไป

ตัวอย่างการเปลี่ยนรูปทรงของตัวหุ่นยนต์นั้นคือ การอัดตัวของอากาศในถุงพักลม ซึ่งแนวคิดนี้ถูกเสนอโดย M.Artusi , Simon Fraser University Burnavy , California ตัวถังของหุ่นยนต์ทรงกลมนั้นจะประกอบไปด้วยตัว Dielectric elastomer actuators sections ทั้งหมด 4 ตัว ซึ่งมันจะสามารถเปลี่ยนรูปได้โดยการจ่ายสนามไฟฟ้าเข้าไป จากนั้นการเปลี่ยนรูปในแต่ละส่วนตามลำดับก็จะทำให้ตัวหุ่นยนต์ทรงกลมเกิดการกลิ้ง



รูปที่ 2.4 หุ่นยนต์ทรงกลมแบบเคลื่อนที่ด้วยการเปลี่ยนรูปร่างของหุ่น รูปซ้าย เป็นถุงพักลมอัดตัว แนวคิดของ M.Artusi รูปขวา เป็นถุงพักลมอัดตัวแบบหกเหลี่ยม แนวคิดของ K.Wait

อีกตัวอย่างหนึ่ง K.Wait ได้เสนอแนวคิดที่คลายคลึงกัน ซึ่งมีความก้าวหน้ากว่าการอัดตัวของถุงพักลมธรรมดา คือ ตัวหุ่นยนต์ทรงกลมนั้นจะมีลักษณะคล้ายกับลูกฟุตบอลที่แต่ละชิ้นส่วนหกเหลี่ยมของผิวทรงกลมนั้นสามารถพองตัวและยุบตัวได้ แต่ละส่วนของผิวทรงกลมนั้นจะเป็นถุงพักลมยืดหยุ่นได้ ซึ่งจะถูกอัดลมเข้าไป หุ่นยนต์ทรงกลมจะถูกผลักตามทิศทางที่แต่ละถุงอัดลมหกเหลี่ยมถูกอัดลมเข้าไป ยิ่งถุงพักลมมีจำนวนมากขึ้น ก็ยิ่งทำให้มีทิศทางการเคลื่อนที่ที่หลากหลายมากขึ้น

2.2 ปิก ไมโครคอนโทรลเลอร์ (PIC microcontroller)

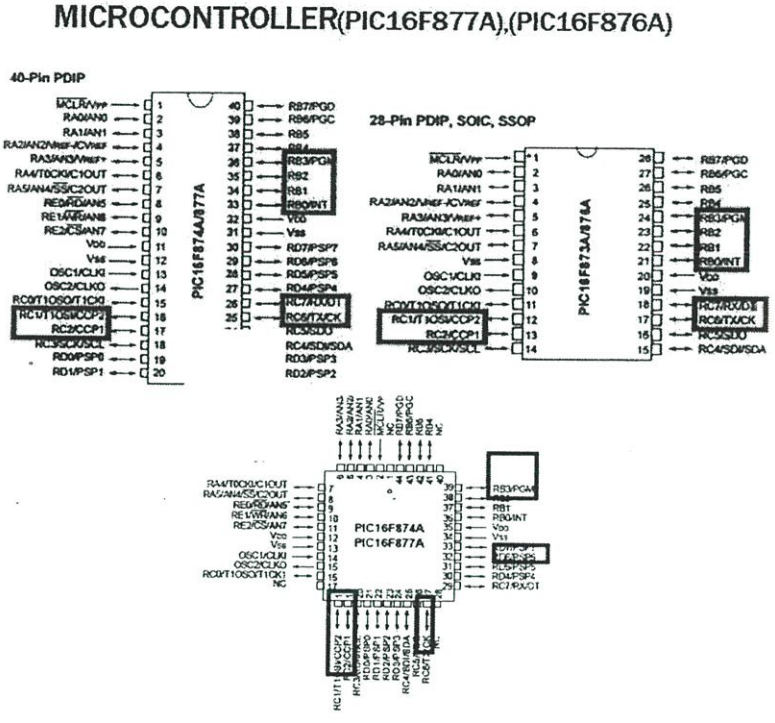
ไมโครคอนโทรลเลอร์เกิดจากคำสองคำที่นำมาผสมกัน นั่นคือคำว่า "ไมโคร" รวมกับคำว่า "คอนโทรลเลอร์" คำว่า ไมโคร คือ คำอุปสรรค (prefixes) เมื่อค่าในหน่วยฐานหรือหน่วยอนุพัทธ์น้อยหรือมากเกินไปเราอาจเขียนค่านั้นอยู่ในรูปตัวเลขคูณ ด้วย ตัวพหุคูณ · (ตัวพหุคูณ คือ เลขสิบยกกำลังบวกหรือลบ) ได้ ตัวอย่างเช่น ระยะทาง0.002 เมตร เขียนเป็น เมตร แทนด้วยคำอุปสรรคมิลลิ (m) ดังนั้นระยะทาง 0.002 เมตร อาจเขียนได้ว่า 2 มิลลิเมตร คำอุปสรรคที่ใช้แทนตัวพหุคูณและสัญลักษณ์ แสดงไว้ในตาราง SI (The International System of Units) ซึ่งในที่นี้ใช้คำว่าไมโครมีค่าเท่ากับ ซึ่งน้อยมาก ส่วนคำว่าคอนโทรลเลอร์ก็คือตัวควบคุม หากรวมกันแล้ว นั่นก็หมายความว่าตัวควบคุมที่มีขนาดน้อยมากหรือเล็กมาก ไมโครคอนโทรลเลอร์จะมีให้เลือกมากมายหลายตระกูลและหลายบริษัท เช่นไมโครคอนโทรลเลอร์ตระกูล MCS-51ของบริษัท Philips ไมโครคอนโทรลเลอร์ตระกูล AVR ของบริษัท ATMEL และไมโครคอนโทรลเลอร์ตระกูล PIC ที่บริษัท Micro chip เป็นผู้ผลิต

ไมโครคอนโทรลเลอร์ตระกูล PIC ได้ถูกพัฒนามาอย่างต่อเนื่อง จนสามารถนำมาใช้งานในการควบคุมได้อย่างหลากหลาย เนื่องจากมีคุณสมบัติที่ครบครัน อีกทั้งยังง่ายต่อผู้ที่เริ่มเรียน คือสามารถที่จะเรียนรู้ได้ง่ายและสามารถนำไปใช้งานได้จริง เนื่องจากมีประสิทธิภาพที่สูง ราคาถูก เป็นที่นิยม และมีให้เลือกมากมายหลายเบอร์

2.2.1 ความเป็นมาของ ปิก ไมโครคอนโทรลเลอร์ (PIC microcontroller)

PIC คือไมโครคอนโทรลลอร์ อีกตระกูลหนึ่ง ย่อมาจากคำว่า "Peripheral Interface Controller" ซึ่ง แนวคิดของไมโครคอนโทรลเลอร์ตระกูลนี้ก็คือ พยายามรวมเอาทุกอย่างเอาไว้ในตัวของมันไม่ว่าจะเป็นหน่วยความจำโปรแกรม(PROGRAM MEMORY),หน่วยความจำชั่วคราว(RAM), หน่วยความจำอ่านอย่างเดียว(EEPROM), พอร์ตสื่อสารอนุกรม(SERIAL), พอร์ตสื่อสารแบบไอสแควร์ซี(I2C), ขาสัญญาณ พัลส์วิดท์มอดูเลชั่น(PWM), โมดูลแปลงสัญญาณอนาล็อกเป็นดิจิตอล(A/D) ฯลฯ โดยไม่จำเป็นต้องต่ออุปกรณ์เสริมจากภายนอก ในตัวของ PIC จะมีฟังก์ชันที่ใช้ในการประมวลผล รวมทั้งหน่วยความจำ ซึ่งทำให้มันเหมือนกัน CPU ตัวหนึ่งเลยทีเดียว ผลที่ตามมาก็คือแผ่นวงจรจะมีขนาดเล็ก และอุปกรณ์ที่ใช้จะไม่มาก บางงานอาจจะใช้แค่ PIC เพียงตัวเดียวโดยไม่ต้องใช้ชิพ อื่นมาเพิ่มเติมเลย นี่คือคุณสมบัติพิเศษของ PIC ซึ่งปัจจุบันหลายบริษัทที่ผลิต microcontroller ก็เริ่มจะหัดมาเลียนแบบแนวทางนี้ แต่ทุกอย่างย่อมมีข้อเสีย เนื่องจากแนวคิดที่จะรวมทุกอย่างไว้ในชิพเดียว ทำให้หน่วยความจำโปรแกรม และหน่วยความจำข้อมูล ไม่สามารถขยายโดยใช้กับ หน่วยความจำภายนอกได้ในทางทฤษฎี PIC จึงเหมาะสำหรับงานเล็กๆ ไม่ใช่งานใหญ่ๆที่ต้องใช้กา คำนวณ หน่วยความจำมากๆ ความเร็วในการทำงานของ PIC ปัจจุบันสัญญาณนาฬิกาสูงสุดของ PIC มีค่าเท่ากับ 20 MHz ดังนั้นหนึ่งคำสั่งของ PIC ใช้เวลาเพียง 0.25 usec

ในปัจจุบันนี้ PIC ได้ถูกนำมาประยุกต์ใช้งานกันอย่างแพร่หลายในเครื่องมือต่างๆ ตัวอย่างเช่น การนำมาควบคุมการแสดงผลของจอแสดงผล,การนำมาส่งสัญญาณเพื่อสร้างคลื่นพาห์สำหรับการส่งอินฟราเรด, การนำมาควบคุมการปิด-เปิดสวิตช์ด้วยสัญญาณคลื่นวิทยุ, การนำมาควบคุมในเตาหุงต้มเหนี่ยวนำความร้อน, การนำมาควบคุมวงจรจุดชนวนอุปกรณ์อิเล็กทรอนิกส์กำลัง, การนำมาควบคุมหุ่นยนต์ เป็นต้น



รูปที่ 2.5 ไดอะแกรมแสดงขาการทำงานต่างๆของ PIC เบอร์ 16F877A และ 16F876A

2.2.2 คุณสมบัติต่างๆของ PIC

จากที่ได้กล่าวไปแล้วข้างไมโครคอนโทรลเลอร์ PIC นั้นมีคุณสมบัติต่างๆมากมาย ในส่วนต่อไปนี้ จึงจะกล่าวถึงรายละเอียดในด้านต่างๆของตัวไมโครคอนโทรลเลอร์ค่ายนี้โดยสังเขป

2.2.2.1 ความเร็วของ PIC

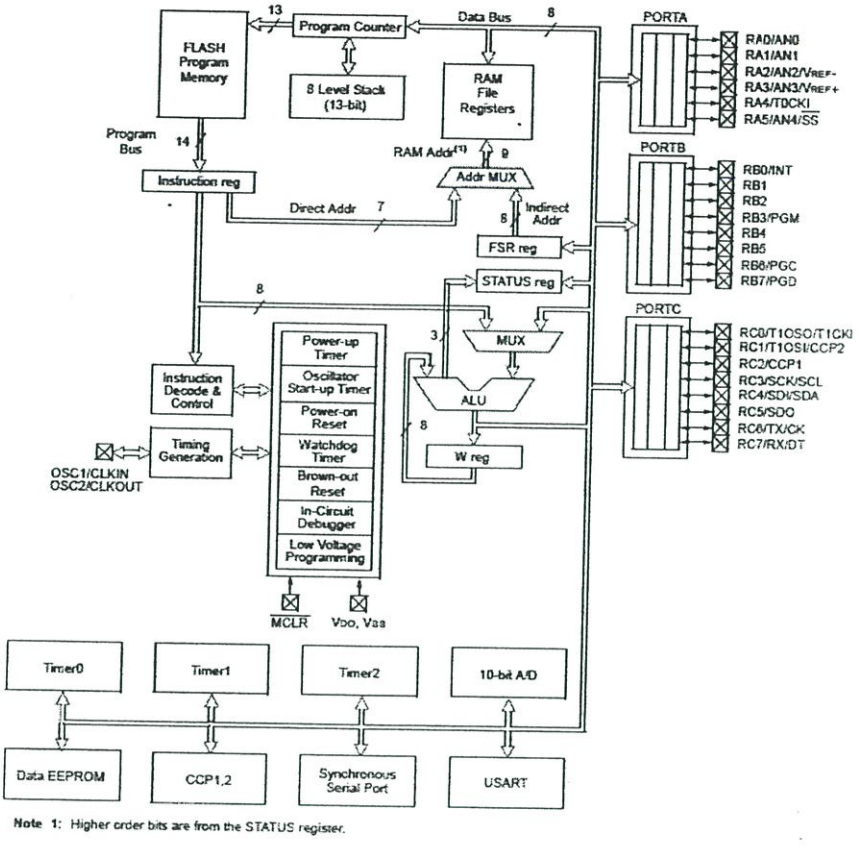
ภาคของความถี่สัญญาณนาฬิกา ปัจจุบันสามารถทำสัญญาณนาฬิกาได้ที่ 20 MHz ซึ่งทำให้หนึ่งคำสั่งของ PIC ใช้เวลาเพียง 0.25 μSec แต่อย่างไรก็ตามได้มีบริษัทอื่นได้ซื้อลิขสิทธิ์ PIC จาก microchipและได้สร้างชิพที่มีความเร็วได้มากกว่าเดิมขึ้นไปอีก

2.2.2.2 หน่วยความจำของ PIC

ในอดีตหน่วยความจำของ PIC จะค่อนข้างน้อย คืออยู่ระหว่าง 512 words ถึง 4K words แต่ในปัจจุบัน บริษัท microchip ซึ่งเป็นเจ้าของ PIC ได้พัฒนาจนทำให้หน่วยความจำ ของ PIC มีขนาดเป็นหลายสิบกิโลไบต์ และมีทีท่าว่าจะขยายได้ใหญ่ขึ้นเรื่อยๆ ในเรื่องของการนับขนาดของหน่วยความจำของ PIC จะนับไม่เหมือนปกติ โดยที่ หนึ่งคำสั่งของ PIC จะมีขนาด 14 bits ดังนั้นเราจะเรียกว่า 1 word ของ PIC จะมีขนาด 14 bits เช่น PIC16F84A ระบุว่ามีหน่วยความจำ 1 K (ซึ่งหมายถึง 1 Kword ถ้าคำนวณให้เป็นแบบ 1 byte = 8 bit จะได้ว่า 1 x 1,024 x 14 = 14,336 bits ดังนั้นก็คือ14,336/(8 x 1,024) = 1.75K bytes นั่นเอง

2.2.2.3 สถาปัตยกรรมของ PIC

ตอนนี้มี 3 สายหลักๆ สมัยก่อนมีแค่สอง คือขึ้นต้นด้วย 16xxx,17xxx และใหม่ล่าสุดคือ 18xxx ถ้าพูดถึง คุณสมบัติที่เหนือกว่าเรียงจากน้อยสุดไปมากสุดก็คือ 16 -> 17 -> 18 คำสั่ง assembly ของ 17และมี 18 จะมีมากกว่า 16 ทำให้เขียนโปรแกรมได้ง่ายกว่า ราคาก็จะสูงกว่าด้วย แต่ที่เป็นที่นิยมก็คือตระกูล 16xxx



รูปที่ 2.6 แสดงสถาปัตยกรรมภายในของ PIC เบอร์ 16F87XA

2.2.3 ลำดับขั้นตอนการใช้งาน PIC

2.2.3.1 การเขียนโปรแกรมภาษาซี

การเขียนโปรแกรมภาษาซีสำหรับ PIC จะมีหลากหลายโปรแกรมให้เลือกใช้ ตัวอย่างเช่นโปรแกรม “MikroC for PIC”,โปรแกรม “PIC Basic PRO”,โปรแกรม “C Compiler”และโปรแกรม “CCS ‘C’ Compiler” ซึ่งจะเป็นภาษาซีที่แตกต่างกันไม่มากนัก ส่วนในโปรเจคนี้จะใช้โปรแกรม “CCS ‘C’ Compiler” เนื่องจากโปรแกรมนี้ใช้ภาษาซีที่ผู้อ่านสามารถทำความเข้าใจได้ง่ายมากกว่าโปรแกรมอื่นๆ มีการพัฒนาโปรแกรมมาอย่างต่อเนื่อง มีฟังก์ชันต่างๆให้เลือกใช้งานมากมาย จึงเหมาะสมกับผู้เริ่มต้น เช่น ฟังก์ชันLCD.C ที่ผู้ออกแบบโปรแกรมได้เขียนเอาไว้ในไฟล์ไดร์ฟเวอร์ มาอ้างอิง เพื่อที่จะเป็นการอำนวยความสะดวกให้ผู้ศึกษาสามารถดึงไฟล์

ไดร์ฟเวอร์ ตัวดังกล่าวมาใช้ได้เลย อีกทั้งโปรแกรมนี้เป็นที่นิยมเป็นอย่างมาก จึงทำให้เกิดการแชร์ประสบการณ์ของนักเขียนโปรแกรมต่างๆ มีการแลกเปลี่ยนความคิด ร่วมกันปรึกษาปัญหาในการออกแบบและแก้ไขโปรแกรม และทำให้เกิดกระบวนการเรียนรู้ที่พัฒนามากยิ่งขึ้นไป

2.2.3.2 การจำลองการทำงาน

การจำลองการทำงานบนคอมพิวเตอร์ด้วยโปรแกรม PROTEUS เป็นโปรแกรมจำลองการทำงานของวงจรไฟฟ้าและสามารถออกแบบแผ่นปริ้นท์ได้อีกด้วย โปรแกรมนี้จะช่วยให้ผู้เขียนโปรแกรมสามารถดูผลการทำงานของโปรแกรม ก่อนที่จะทำการอัดโปรแกรมลงในตัวชิพ จึงช่วยให้ผู้เขียนโปรแกรมสามารถตรวจสอบข้อผิดพลาดของโปรแกรมได้ ในโปรแกรมจะประกอบไปด้วยอุปกรณ์ต่างๆ ให้เลือกใช้มากมายตัวอย่างเช่น หลอแอลอีดี สีต่างๆ สวิตช์ปุ่มกด ตัวต้านทาน ตัวเหนี่ยวนำ ตัวเก็บประจุ แบตเตอร์รี่ เซเว่นเซกเมนต์ จอแอลซีดี มอเตอร์ต่างๆ รวมไปถึงเครื่องมือวัดต่างๆ เช่น ออสซิลโลสโคป, โวลต์มิเตอร์, แอมป์มิเตอร์

2.2.3.3 การถ่ายโอนชุดคำสั่งจากคอมพิวเตอร์สู่ตัว PIC

การถ่ายโอนชุดคำสั่งหรือคอมไพเลอร์นั้น มีอยู่ 2 วิธีคือ 1)โปรแกรมผ่านพอร์ตอนุกรมและ 2)โปรแกรมผ่านพอร์ตยูเอสบี ซึ่งการเลือกใช้ว่าจะโปรแกรมด้วยวิธีไหนนั้นขึ้นอยู่กับความสะดวกของผู้ใช้ เนื่องจากคอมพิวเตอร์โน้ตบุ๊ค รุ่นใหม่ๆจะไม่มีพอร์ตอนุกรม การโปรแกรมผ่านพอร์ตยูเอสบี จึงง่ายกว่าและสะดวกกว่าสำหรับคอมพิวเตอร์โน้ตบุ๊ค การโปรแกรมผ่านพอร์ตยูเอสบี นั้น จะใช้บอร์ด Pic kit 2 และจะต้องใช้ควบคู่กันกับโปรแกรม Pic Kit 2

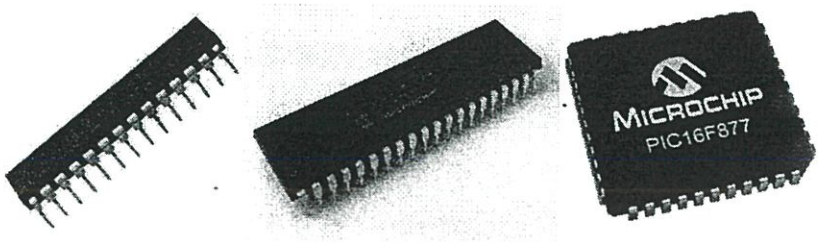
2.2.3.4 การนำ PIC ไมโครคอนโทรลเลอร์ไปต่อใช้งาน

การต่อใช้งานไมโครคอนโทรลเลอร์ PIC นั้น เป็นเรื่องง่ายมาก เนื่องจากทางบริษัทไมโครชิพมีแนวคิดคือการพยายามรวมเอาทุกอย่างไว้ในชิพตัวเดียวกัน การต่อเพิ่มเติมจึงมีไม่มาก เพียงจ่ายไฟเลี้ยงให้ ต่อวงจรสร้างสัญญาณนาฬิกา และวงจรรีเซ็ตเป็นวงจรพื้นฐานเท่านั้น ซึ่งในแต่ละการทดลองจะมีบอกไว้ อาจจะมีการเชื่อมต่อหน่วยความจำจากภายนอกก็ต่อเมื่อใช้งานมากขึ้นเท่านั้น เช่นการเชื่อมต่อกับไอซีที่บอก วัน/เดือน/ปี ในงานที่จำเป็นต้องเชื่อมต่อเช่น การสร้างนาฬิกาดิจิตอลเป็นต้น

2.2.4 รายละเอียดคุณสมบัติของไมโครคอนโทรลเลอร์ PIC16F87XA

คุณสมบัติต่างๆของไมโครคอนโทรลเลอร์ PIC เบอร์ 16F87XA นั้นมีดังต่อไปนี้

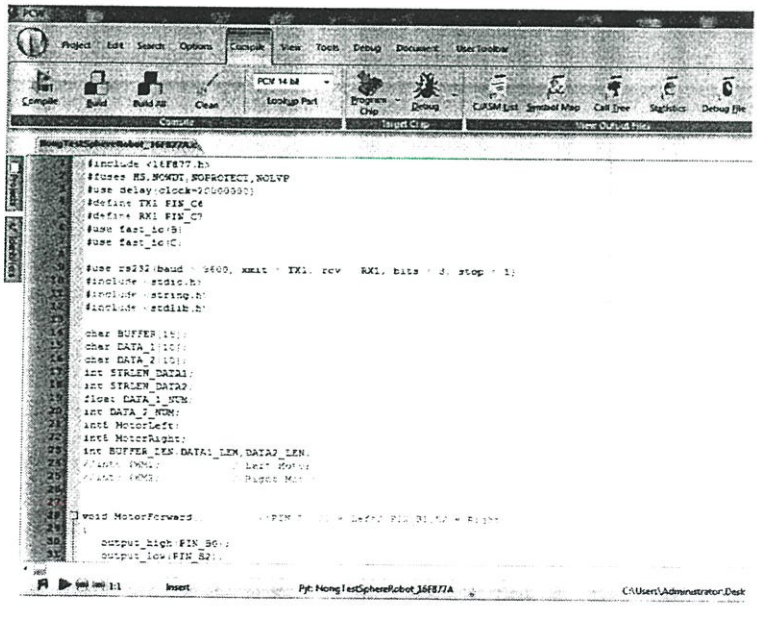
- มีคำสั่งในภาษาแอสเซ็มบลี้ 35 คำสั่ง
- ใช้ความถี่ออสซิเลเตอร์ได้สูงสุด 20MHz
- มีหน่วยความจำโปรแกรม Flash Memory ขนาด 8K word(14-bit words)
- มีหน่วยความจำข้อมูลแบบ RAM 368 Bytes
- มีหน่วยความจำข้อมูลแบบ EEPROM 256 Bytes
- มีการตอบสนองอินเตอร์รัพท์ทั้งหมด 14 แหล่ง
- สามารถเลือกระดับการป้องกันข้อมูล (Code Protection) ได้
- มีโหมดประหยัดพลังงาน (Sleep Mode
- สามารถเลือกแหล่งสัญญาณนาฬิกาได้หลายโหมด XT RC และออสซิลเลเตอร์พลังงานต่ำ
- มีฟังก์ชั่นการรักษาเสถียรภาพการทำงาน ได้แก่ POR,PWRT,OST,BOR และ WDT
- การโปรแกรมตัวชิพแบบ ICSP(In-Circuit Serial Programming)
- สามารถทำงานได้ที่ไฟเลี้ยงวงจรตั้งแต่ 2.0VDC ถึง 5.5VDC
- ขาพอร์ท I/O แต่ละขาสามารถรับและปล่อยกระแสได้สูงสุดที่ 25mA
- มีโมดูล Timer/Counter ใช้งานทั้งหมด 3 ตัว Timer 0 , Timer 1 และTimer 2
- มีโมดูล CCP (Compare/Capture/PWM) จำนวน 2 ชุด
- มีโมดูล Analog to Digital Converter ความละเอียดขนาด 8 บิต และ 10 บิต จำนวน 8 ช่องภายในตัวชิพ
- มีโมดูลสื่อสารอนุกรมแบบ USART (Universal Synchronous Asynchronous Receiver/Transmitte
- มีพอร์ท I/O จำนวน 5 พอร์ท A,B,C,D และ E มีขา I/O รวมกัน 33 ขา (สำหรับเบอร์ 16F876A จะมี 3 พอร์ท A,B และ C)



รูปที่ 2.7 รูปภาพแสดงตัวไมโครคอนโทรลเลอร์ PIC เบอร์ 16F876A (รูปซ้าย) และเบอร์ 16F877A (รูปขวา, กลาง)

2.2.5 โปรแกรม CCS 'C' คอมไพเลอร์

CCS 'C' เป็นซอฟต์แวร์สำหรับใช้ในการเขียนโค้ดและคอมไพล์โปรแกรมภาษา C สำหรับไมโครคอนโทรลเลอร์ PIC โดยแบ่งออกเป็น 2 เวอร์ชัน คือ เวอร์ชันคอมไพล์โปรแกรมผ่าน คอมมานด์ ไลน์ (ในโหมดดอสพร้อม) และเวอร์ชั่นคอมไพล์โปรแกรมบนไมโครซอฟต์วินโดวส์ ซึ่งทำงานในรูปแบบ IDE หรือ Integrated Development Environment (เขียนแล้วคอมไพล์โค้ดในตัว) สำหรับโปรเจคนี้จะใช้งานเฉพาะ เวอร์ชั่นที่คอมไพล์โปรแกรมบนวินโดวส์



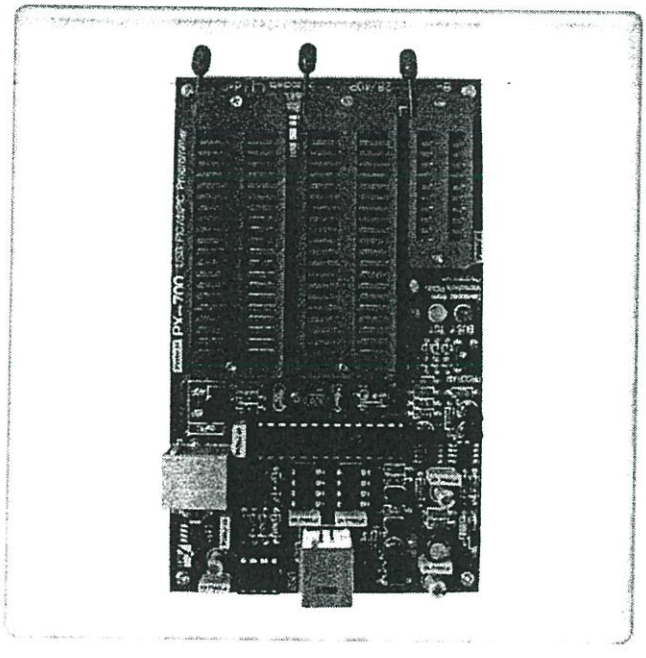
รูปที่ 2.8 รูปภาพแสดงหน้าต่างของโปรแกรม CCS 'C' compiler

2.2.6 เครื่องโปรแกรมเมอร์ PICkit2

PICKit2 นั้นเป็นเครื่องโปรแกรมเมอร์และดีบักเกอร์ ที่มีราคาถูก ใช้งานง่าย เหมาะสำหรับเป็นเครื่องมือช่วยในงานพัฒนาด้านต่างๆ เพราะสามารถใช้งานได้ดีกับไมโครคอนโทรลเลอร์ตระกูลไมโครชิพ (Microchip) อีกทั้งยังเป็นตัวโปรแกรมเมอร์ที่อินเตอร์เฟสได้ดีระหว่างวินโดว์ กับอุปกรณ์ไมโครชิพรุ่นเบสไลน์ (baseline) (PIC10F, PIC12F5xx, PIC16F5xx), รุ่นมิดเรนจ์ (Midrange) (PIC12F6xx, PIC16F) รวมไปถึงรุ่น PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 ทั้งตระกูล 8-bit, 16-bit และ 32-bit

ยิ่งไปกว่านั้น PICkit2 สามารถทำงานในโหมดดีบักแบบ อินเซอร์กิตดีบักกิง บนตัวไมโครคอนโทรลเลอร์ได้อีกด้วย ซึ่งมันคือการตรวจเช็คความถูกต้องของโปรแกรมที่ได้เขียนลงไป ผ่านการรันเป็นสเต็ปๆ โดยที่ไม่จำเป็นต้องถอดตัวไมโครคอนโทรลเลอร์ออกมาจากแผงวงจรหรือชิ้นงาน และยังสามารถปรับปรุงเปลี่ยนแปลงโปรแกรมที่เขียนได้อีกด้วย

2.2.6.1 เครื่องโปรแกรมเมอร์ PX-700



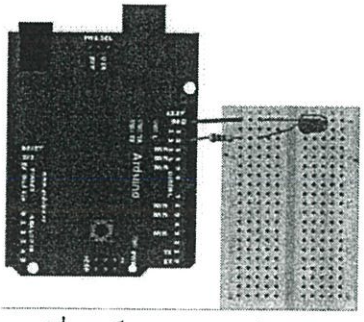
รูปที่ 2.9 รูปภาพแสดงหน้าต่างเครื่องโปรแกรมเบอร์ PX-700

เป็นเครื่องโปรแกรมเมอร์สำหรับไมโครคอนโทรลเลอร์ PIC ซึ่งสามารถใช้งานได้เทียบเท่ากับเครื่องโปรแกรมเมอร์ PICkit2 ซึ่งใช้ในโปรเจคนี้สำหรับโปรแกรมตัวไมโครคอนโทรลเลอร์เบอร์ 16F877A และ 16F876A โดยจะมีคุณสมบัติดังต่อไปนี้

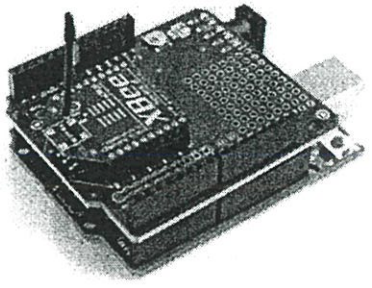
• เชื่อมต่อคอมพิวเตอร์ผ่านพอร์ต USB
• ใช้ไฟเลี้ยงจากพอร์ต USB
• มี LED แสดงสถานะการทำงาน 3 ดวง (POWER, TARGET และ BUSY)
• สามารถอัพเกรดเฟิร์มแวร์ได้โดยผ่านทางซอฟต์แวร์
• บนบอร์ดมีซ็อกเก็ต ZIF 3 ชุดสำหรับโปรแกรม
ไมโครคอนโทรลเลอร์ PIC 8-20 ขา PIC12Fxxx,PIC16Fxxx
ไมโครคอนโทรลเลอร์ PIC 28/40 ขา PIC16F/PIC18F
ไมโครคอนโทรลเลอร์ dsPIC 28/40 dsPIC30F
• การดีบักขึ้นอยู่กับรุ่นของเฟิร์มแวร์ โดยต้องใช้ PX-700 ร่วมกับโปรแกรม
MPLAB V8 ขึ้นไป โดยเลือก Debug Tools เป็น Pickit2
• สามารถโปรแกรมไมโครคอนโทรลเลอร์ PIC แบบแฟลชได้ทุกเบอร์ ดูข้อมูลเพิ่มเติม และดาวน์โหลดโปรแกรมรวมทั้งเฟิร์มแวร์ล่าสุดได้ที่ www.microchip.com ค้นหาหน้า PICkit2
• มีแผ่นพลาสติกรองเพื่อป้องกันแผงวงจร

2.3 บอร์ดคอนโทรลเลอร์ อาร์ดูโน่ (Arduino)

Arduino อ่านว่า (อา-ดู-อิ-โน่ หรือ อาดุยโน่) เป็นบอร์ดไมโครคอนโทรเลอร์ตระกูลเอวีอาร์ (AVR) ที่มีการพัฒนาแบบโอเพ่นซอร์ส(Open Source) คือมีการเปิดเผยข้อมูลทั้งด้านฮาร์ดแวร์ (Hardware) และซอฟท์แวร์(Software) ตัวบอร์ดอาร์ดูโน่ ถูกออกแบบมาให้ใช้งานได้ง่าย ดังนั้นจึงเหมาะสำหรับผู้เริ่มต้นศึกษา ทั้งนี้ผู้ใช้งานยังสามารถดัดแปลง เพิ่มเติม พัฒนาต่อยอดทั้งตัวบอร์ด หรือโปรแกรมต่อได้อีกด้วย ความง่ายของบอร์ดอาร์ดูโน่ในการต่ออุปกรณ์เสริมต่างๆ คือผู้ใช้งานสามารถต่อวงจรอิเล็กทรอนิคส์จากภายนอกแล้วเชื่อมต่อเข้ามาที่ขาอินพุท/เอาท์พุท(I/O) ของบอร์ด (ดูตัวอย่างรูปที่ 1) หรือเพื่อความสะดวกสามารถเลือกต่อกับบอร์ดเสริมอาร์ดูโน่ ชีลด์ (Arduino Shield) ประเภทต่างๆ (ดูตัวอย่างรูปที่ 2) เช่นอาร์ดูโน่เอ็กซ์บีชีลด์ (Arduino XBee Shield), อาร์ดูโน่มิวสิคชีลด์(Arduino Music Shield), อาร์ดูโน่รีเลย์ชีลด์(Arduino Relay Shield), อาร์ดูโน่วายเลสชีลด์(Arduino Wireless Shield), อาร์ดูโน่จีพีอาร์เอสชีลด์(Arduino GPRS Shield) เป็นต้น มาเสียบกับบอร์ดบนบอร์ดอาร์ดูโน่แล้วเขียนโปรแกรมพัฒนาต่อได้เลย



รูปที่1 บอร์ด Arduino ต่อกับ LED



รูปที่2 บอร์ด Arduino ต่อกับบอร์ด XBee Shield

รูปที่ 2.10 รูปภาพแสดงบอร์ด อาร์ดูโน่ เชื่อมต่อกับอุปกรณ์ต่างๆ

2.3.1 จุดเด่นที่ทำให้บอร์ด อาร์ดูโน่ เป็นที่นิยม

- ง่ายต่อการพัฒนา มีรูปแบบคำสั่งพื้นฐาน ไม่ซับซ้อนเหมาะสำหรับผู้เริ่มต้น
- มีอาร์ดูโน่ คอมมิวนิตี้(Arduino Community) กลุ่มคนที่ร่วมกันพัฒนาที่แข็งแรง
- โอเพ่น ฮาร์ดแวร์ (Open Hardware) ทำให้ผู้ใช้สามารถนำบอร์ดไปต่อยอดใช้งานได้หลายด้าน
- ราคาไม่แพง
- ครอส แพลทฟอร์ม (Cross Platform) สามารถพัฒนาโปรแกรมบนระบบปฏิบัติการ(OS) ใดก็ได้

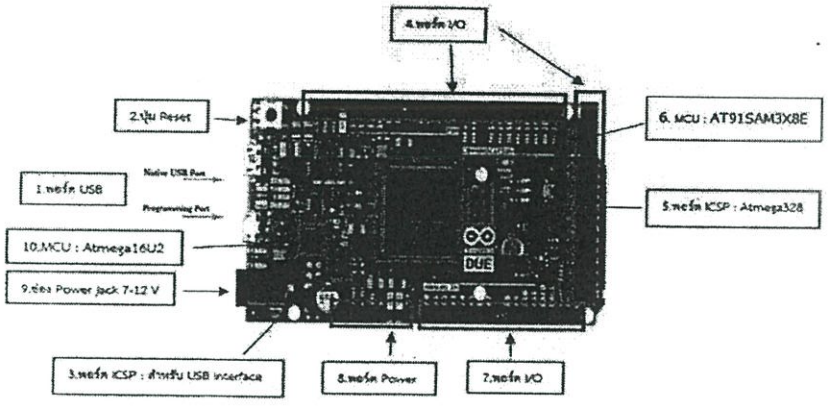
ตารางที่ 2.1 ตารางเปรียบเทียบคุณสมบัติของบอร์ดอาร์ดูโน่ แต่ละรุ่น

	Processor					Input / Output							Power				Connectivity				
	Family	SRAM	FLASH	EEPROM	Clock	Digital I/O	Analog In	ADC Bits	PWM	UART	Analog Out	DAC Bits	VCC	Vin Range	5V	3V3	USB-Serial	I2C	Ethernet	USB-Host	SD Card
Arduino UNO R3	ATmega328	2k	32k	1k	16MHz	14	6	10	6	1	N/A	N/A	5V	7-12V	Yes	Yes	ATmega16U2	1	No	No	No
Arduino UNO SMD	ATmega328	2k	32k	1k	16MHz	14	6	10	6	1	N/A	N/A	5V	7-12V	Yes	Yes	ATmega16U2	1	No	No	No
Arduino Mega 2560 R3	ATmega2560	8k	256k	4k	16MHz	54	16	10	14	4	N/A	N/A	5V	7-18V	Yes	Yes	ATmega16U2	1	No	No	No
Arduino Mega ADK	ATmega2560	8k	256k	4k	16MHz	54	16	10	14	4	N/A	N/A	5V	7-18V	Yes	Yes	ATmega16U2	1	MAX3421E	No	No
Arduino Leonardo	ATmega32U4	2.5k	32k	1k	16MHz	25	12	10	7	1	N/A	N/A	5V	7-12V	Yes	Yes	Built-in	1	No	No	No
Arduino Mini 05	ATmega328	2k	32k	1k	16MHz	14	8	10	6	1	N/A	N/A	5V	7V-9V	Yes	No	N/A	1	No	No	No
Arduino Pro Mini 328 - 3.3V	ATmega328	2k	32k	1k	8MHz	14	6	10	6	1	N/A	N/A	3.3V	5V-12V	No	Yes	N/A	1	No	No	No
Arduino Pro Mini 328 – 5V	ATmega328	2k	32k	1k	16MHz	14	6	10	6	1	N/A	N/A	5V	7V-12V	Yes	No	N/A	1	No	No	No
Arduino Ethernet with PoE module	ATmega328	2k	32k	1k	16MHz	9	6	10	4	1	N/A	N/A	5V	6-18V	Yes	Yes	N/A	1	No	No	No
Arduino Ethernet without PoE module	ATmega328	2k	32k	1k	16MHz	9	6	10	4	1	N/A	N/A	5V	6-18V	Yes	Yes	N/A	1	No	No	No
Arduino DUE	SAM3X8E	96kb	512k	N/A	84MHz	70	12	12	12	4	2	12	3.3V	7-12V	No	VCC	Built-in	2	No	Yes	No

จะเห็นได้ว่าบอร์ดอาร์ดูโน่ แต่ละรุ่นนั้นต่างมีคุณสมบัติที่แตกต่างกัน จึงต้องเลือกให้ให้เหมาะสมกับงานที่ทำ ในโปรเจ็คนี้เลือกใช้บอดร์อาร์ดูโน่ดิว (Arduino DUE) ซึ่ง เป็นบอร์ดอาร์ดูโน่ ที่เปลี่ยนชิพไมโครคอนโทรลเลอร์(MCU) ใหม่ จากเดิมเป็นตระกูลเอวีอาร์(AVR) เปลี่ยนเป็นเบอร์ “AT91SAM3X8E”(ตระกูล ARM Cortex-M3) แทน ทำให้การประมวลผลเร็วขึ้น แต่ยังคงรูปแบบโค้ดโปรแกรมของ อาร์ดูโน่ ที่ง่ายอยู่

ข้อควรระวัง: เนื่องจากชิพ MCU เป็นคนละเบอร์กับบอร์ดอาร์ดูโน่ อูโน่ อาร์ 3 (Arduino Uno R3) อาจจะทำให้บอร์ดชีลด์ บางตัวหรือไลบรารี่(Library) ใช้ร่วมกันกับบอร์ดอาร์ดูโน่ ลีโอนาโด(Arduino Leonardo) ไม่ได้

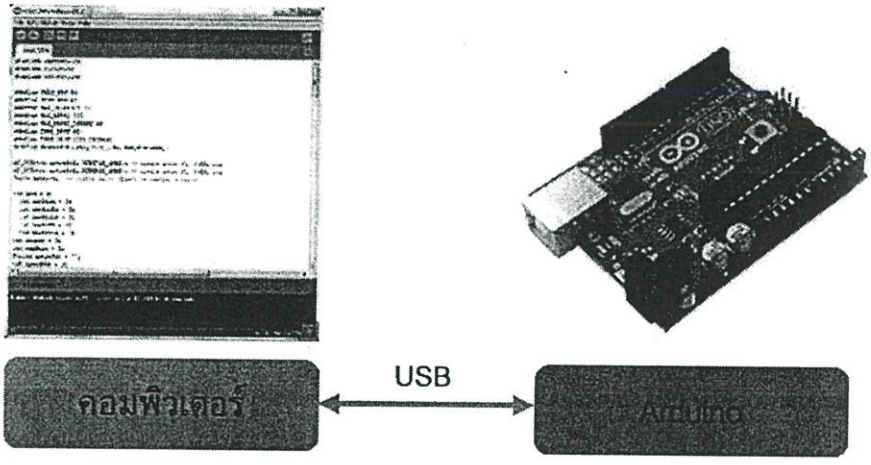
2.3.2 Layout & Pin out Arduino Board (Model: Arduino DUE)



รูปที่ 2.11 แสดงส่วนประกอบในบอร์ด Arduino

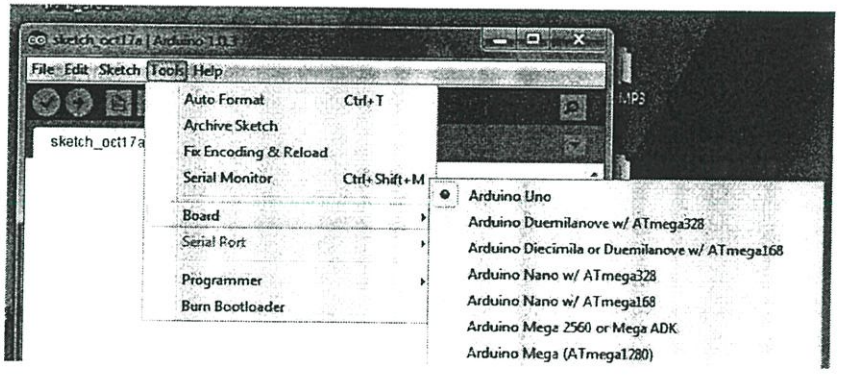
1. USB Port: ใช้สำหรับต่อกับ Computer เพื่ออับโหลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด
2. Reset Button: เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3. ICSP Port ของ Atmega16U2 เป็นพอร์ตที่ใช้โปรแกรม Visual Com - port บน Atmega16U2
4. I/O Port: Digital I/O ตั้งแต่ขา D0, A1, A22 ถึง D53 นอกจากนี้ บาง Pin จะทำหน้าที่อื่นๆ เพิ่มเติมด้วยเช่น Pin0,1,14,15,16,17,18,19 เป็นขา Tx,Rx,Serial,Pin2,3,4,5,6,7,8,9,10,11,12 และ 13 เป็นขา PWM ขาการสื่อสาร เช่น SCL1,SDA1,SDA20,SCL21 เป็นต้น
5. ICSP Port: Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Boot loader
6. MCU: At91SAM3X8E เป็น MCU ที่ใช้บนบอร์ด Arduino
7. I/O Port: นอกจากจะเป็น Digital I/O แล้ว ยังเปลี่ยนเป็น ช่องรับสัญญาณอนาล็อก ตั้งแต่ขา A0-A11
8. Power Port: ไฟเลี้ยงของบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ประกอบด้วยขาไฟเลี้ยง +3.3 V, +5V, GND, Vin
9. Power Jack: รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU ของ Atmega16U2 เป็น MCU ที่ทำหน้าที่เป็น USB to Serial โดย Atmega328 จะ ติดต่อกับ Computer ผ่านAtmega16U2

2.3.3 รูปแบบการเขียนโปรแกรมบนอาร์ดูโน่

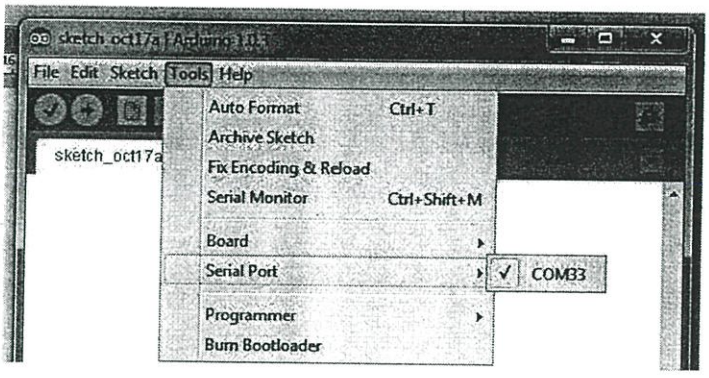


รูปที่ 2.12 รูปแบบการเขียนโปรแกรมบน Arduino

1. เขียนโปรแกรมบนคอมพิวเตอร์ ผ่านทางโปรแกรม Arduino IDE ซึ่งสามารถดาวน์โหลดได้ จาก Arduino.cc/en/main/software
2. หลังจากที่เขียนโค้ดโปรแกรมเรียบร้อยแล้ว ให้ผู้ใช้งานเลือกรุ่นบอร์ด Arduino ที่ใช้และหมายเลข Comport

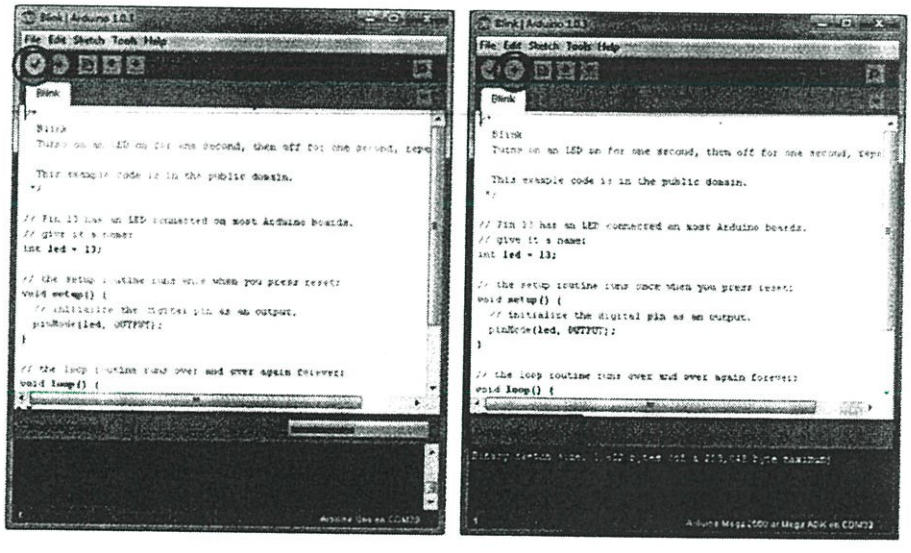


รูปที่ 2.13 ภาพแสดงการเลือกรุ่นบอร์ด Arduino ที่ต้องการ upload



รูปที่ 2.14 ภาพแสดงการเลือกหมายเลข Comport ของบอร์ด

3. กดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ Compile โค้ดโปรแกรม จากนั้นกดปุ่ม Upload โค้ด โปรแกรมไปยังบอร์ดอาร์ดูโน่ ผ่านทางสาย USB เมื่ออัปโหลดเรียบร้อยแล้ว จะแสดงข้อความแถบ ข้างล่าง “Done uploading” และบอร์ดจะเริ่มทำงานตามที่เขียนโปรแกรมไว้ได้ทันที

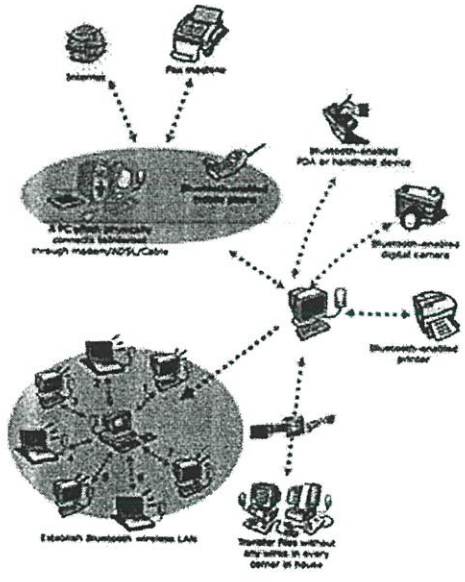


รูปที่ 2.15 ภาพแสดงด้านซ้ายกดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ Compile โค้ดโปรแกรม ภาพขวาแสดงการ Upload โค้ดโปรแกรม

2.3 เทคโนโลยีบลูทูธ (BLUETOOTH TECHNOLOGY)

2.4.1 ความหมายของบลูทูธ

บลูทูธ คือ ระบบสื่อสารของอุปกรณ์อิเล็คโทรนิคแบบสองทาง ด้วยคลื่นวิทยุระยะสั้น (Short-Range Radio Links) โดยปราศจากการใช้สายเคเบิ้ล หรือ สายสัญญาณเชื่อมต่อ และไม่จำเป็นจะต้องใช้การเดินทางแบบเส้นตรงเหมือนกับอินฟราเรด ซึ่งถือว่าเพิ่มความสะดวกมากกว่าการเชื่อมต่อแบบอินฟราเรด ที่ใช้ในการเชื่อมต่อระหว่างโทรศัพท์มือถือ กับอุปกรณ์ ในโทรศัพท์เคลื่อนที่รุ่นก่อนๆ และในการวิจัย ไม่ได้มุ่งเฉพาะการส่งข้อมูลเพียงอย่างเดียว แต่ยังศึกษาถึงการส่งข้อมูลที่เป็นเสียง เพื่อใช้สำหรับ หูฟัง บนโทรศัพท์มือถือด้วย



รูปที่ 2.16 โครงสร้างเครือข่ายเทคโนโลยีบลูทูธ

2.4.2 ที่มาของเทคโนโลยีบลูทูธ

คำว่า Bluetooth หรือ ฟันสีฟ้า ความจริงแล้วเป็นนามของกษัตริย์ประเทศเดนมาร์ก ที่มีชื่อว่า "Harald Bluetooth" (ภาษาเดนมาร์ก Harald BlÅtand) ในช่วงปี ค.ศ. 940-981 หรือประมาณ 1,000 กว่าปีก่อนหน้า กษัตริย์องค์นี้ได้ปกครองประเทศเดนมาร์กและนอร์เวย์ในยุคของไวกิ้งค์ และต้องการรวมประเทศให้เป็นหนึ่งเดียว นอกจากนั้น ยังทรงเป็นผู้นำเอาศาสนาคริสต์เข้าสู่ประเทศเดนมาร์กอีกด้วย และเพื่อเป็นการรำลึกถึงกษัตริย์ผู้ปกครองประเทศกลุ่มสแกนดิเนเวีย ซึ่งใน

ปัจจุบันเป็นกลุ่มผู้นำในด้านการผลิตโทรศัพท์มือถือป้อนสู่ตลาดโลก และระบบบลูทูธนี้ ก็ถูกสร้างขึ้นมาเพื่อใช้กับโทรศัพท์มือถือ และเริ่มต้นจากประเทศในแถบนี้ด้วยเช่นกัน

ปี 1994 บริษัท อีริคสัน โมบาย คอมมูนิเคชั่น เริ่มต้นที่จะค้นคว้าวิจัยความเป็นไปได้ในการนำคลื่นสัญญาณวิทยุ มาใช้ระหว่างโทรศัพท์มือถือและอุปกรณ์ต่างๆ และเป็นผู้นำชื่อ บลูทูธ มาใช้

ปี 1998 กลุ่มผู้พัฒนาวิจัยระบบบลูทูธ ได้ถูกก่อตั้งขึ้น โดยเกิดจากการรวมตัวของบริษัทยักษ์ใหญ่อย่าง Ericsson, Nokia, IBM, Toshiba และ Intel ในกลุ่มที่ใช้ชื่อว่า Special Interest Group (SIG) ซึ่งในกลุ่มจะประกอบด้วย กลุ่มผู้นำทางด้านโทรศัพท์มือถือ, คอมพิวเตอร์ ฯลฯ ซึ่งกลุ่มเหล่านี้ได้ประเมินว่า ภายในปี 2002 ในอุปกรณ์การสื่อสาร, เครื่องใช้, คอมพิวเตอร์ จะถูกติดตั้งบลูทูธ ที่จะใช้เชื่อมต่อระหว่างอุปกรณ์ต่างๆ อย่างแพร่หลาย โดยในปีเดียวกัน บริษัทเหล่านี้ได้ประกาศ การรวมตัวกัน และเชิญชวนบริษัทอื่นๆ ให้เข้าร่วม ในลักษณะของการนำเทคโนโลยีนี้ไปใช้ โดยในปี 1999 ได้ทำการเผยแพร่ Bluetooth specification Version 1.0 และได้สมาชิกเพิ่มขึ้นดังนี้ Microsoft, Lucent, 3Com, Motorola

2.4.3 ลักษณะการทำงานของบลูทูธ

บลูทูธจะใช้สัญญาณวิทยุความถี่สูง 2.4 GHz. (กิ๊กกะเฮิร์ซ) แต่จะแยกย่อยออกไปตามแต่ละประเทศ อย่างในแถบยุโรปและอเมริกา จะใช้ช่วง 2.400 ถึง 2.4835 GHz. แบ่งออกเป็น 79 ช่องสัญญาณ และจะใช้ช่องสัญญาณที่แบ่งนี้ เพื่อส่งข้อมูลสลับช่องไปมา 1,600 ครั้งต่อ 1 วินาที ส่วนที่ญี่ปุ่นจะใช้ความถี่ 2.402 ถึง 2.480 GHz. แบ่งออกเป็น 23 ช่อง ระยะทำการของบลูทูธจะอยู่ที่ 5-10 เมตร โดยมีระบบป้องกันโดยใช้การป้อนรหัสก่อนการเชื่อมต่อ และ ป้องกันการดักสัญญาณระหว่างสื่อสาร โดยระบบจะสลับช่องสัญญาณไปมา จะมีความสามารถในการเลือกเปลี่ยนความถี่ที่ใช้ในการติดต่อเองอัตโนมัติ โดยที่ไม่จำเป็นต้องเรียงตามหมายเลขช่อง ทำให้การดักฟังหรือลักลอบขโมยข้อมูลทำได้ยากขึ้น โดยหลักของบลูทูธจะถูกออกแบบมาเพื่อใช้กับอุปกรณ์ที่มีขนาดเล็ก เนื่องจากใช้การขนส่งข้อมูลในจำนวนที่ไม่มาก อย่างเช่น ไฟล์ภาพ, เสียง, แอพพลิเคชั่นต่างๆ และสามารถเคลื่อนย้ายได้ง่าย ขอให้อยู่ในระยะที่กำหนดไว้เท่านั้น (ประมาณ 5-10 เมตร) นอกจากนี้ยังใช้พลังงานต่ำ กินไฟน้อย และสามารถใช้งานได้นาน โดยไม่ต้องนำไปชาร์จไฟบ่อยๆ ด้วย

ส่วนความสามารถการส่งถ่ายข้อมูลของ Bluetooth จะอยู่ที่ 1 Mbps (1 เมกกะบิตต่อวินาที) และคงจะไม่มีปัญหาอะไรมากกับขนาดของไฟล์ที่ใช้กันบนโทรศัพท์มือถือ หรือการใช้งานแบบทั่วไป แต่ถ้าเป็นข้อมูลที่มีขนาดใหญ่อาจจะช้าเกินไป และถ้าถูกนำไปเปรียบกับระบบแลนไร้สาย Wireless LAN (WLAN) แล้ว ความสามารถของบลูทูธ คงจะมีความแตกต่างกันอยู่มาก ซึ่งในส่วนของ WLAN ก็ยังมีระยะการรับ-ส่งที่ไกลกว่า แต่ขอได้เปรียบของบลูทูธจะอยู่ที่ขนาดที่เล็ก

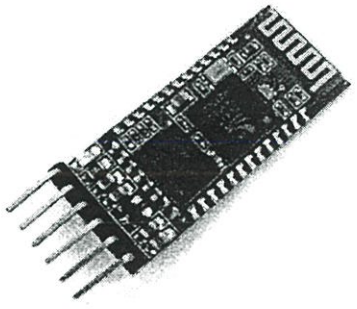
กว่า การติดตั้งทำได้ง่ายกว่า และที่สำคัญ การใช้พลังงานก็น้อยกว่ามาก อยู่ที่ 0.1 วัตต์ หากเทียบกับคลื่นของโทรศัพท์มือถือ

2.4.4 โมดูลบลูทูธ รุ่น HC-05

โมดูลบลูทูธ HC-05 มีการเชื่อมต่อในรูปแบบพอร์ทสื่อสารอนุกรม(Serial UART TTL) โดยมีโปรโตคอลในการสื่อสารแบบบลูทูธเอสพีพี (BluetoothSPP (Serial Port Protocol)) อีกทั้งยังมีการม็อดดูเลทสัญญาณตามมาตรฐาน "Bluetooth V2.0+EDR (Enhanced Data Rate) ที่ 3Mbps พร้อมทั้งติดตั้งบนโมดูลสำเร็จรูปขนาดเล็ก ทำให้สามารถเชื่อมต่อและใช้งานง่าย ผู้ใช้สามารถนำไปพัฒนาให้อุปกรณ์หรือชิ้นงานต่างๆ ทำงานในรูปแบบไร้สายได้อีกด้วย

คุณลักษณะพิเศษ มีดังต่อไปนี้โดยสังเขป

1) โปรโตคอลสื่อสารของบลูทูธ : Bluetooth Specification V2.0+EDR
2) ย่านความถี่ที่ใช้ : 2.4GHz ISM
3) การม็อดดูเลทของสัญญาณ : GFSK (Gaussian Frequency Shift Keying)
4) ความเร็วการรับส่งข้อมูล
   a. แบบ อะซิงโครนัส : 2.1Mbps(Max)/160 kbps
   b. แบบ ซิงโครนัส : 1Mbps/1Mbps
5) ระบบความปลอดภัย : มีระบบ Authentication และ Encryption
6) โปรไฟล์ : เป็นบลูทูธแบบพอร์ทอนุกรม
7) แหล่งจ่ายไฟที่ใช้งาน : +3.3v ถึง 6v DC, 50mA
8) อุณหภูมิใช้งาน : -20 ถึง +75 เซนติเกรด
9) สามารถทำงานได้ในโหมด Master และ Slave
10) สั่งการโดยคำสั่ง AT command



รูปที่ 2.17 โมดูลบลูทูธ รุ่น HC-05

2.4.5 คำสั่ง เอที คอมมานด์ (AT command)

เอที คอมมานด์ (AT-COMMAND) คือ ชุดคำสั่งมาตรฐาน ที่สามารถใช้ติดต่อสื่อสารกับอุปกรณ์สื่อสารต่างๆ เช่น โมเด็ม หรือ อุปกรณ์ DTE (Data Terminal Equipment) เพื่อโต้ตอบตั้งค่าหรือสั่งงานอุปกรณ์เหล่านั้น ให้ทำงานตามที่ต้องการ และสำหรับการติดต่อกับโทรศัพท์มือถือจะใช้ชุดคำสั่งที่เรียกว่า GSM AT COMMAND ตัวอย่างคำสั่ง เอที คอมมานด์ ที่ใช้กับโมดูลบลูทูธ จะแสดงดังตารางที่ 2.2

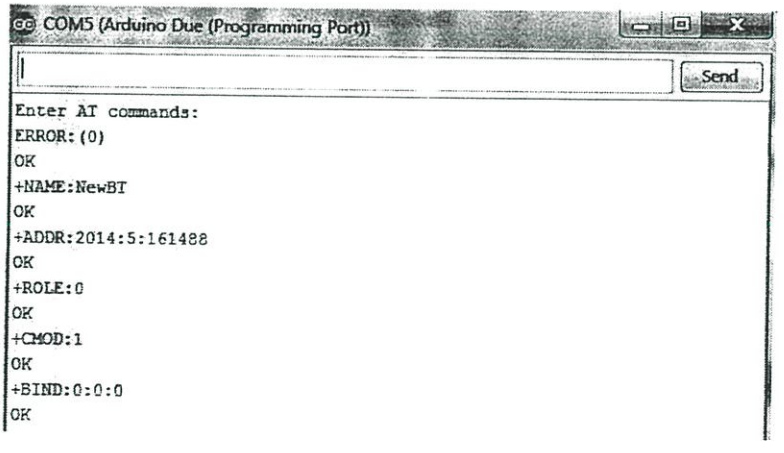
คำสั่ง	**ผลตอบสนอง**	**ความหมายของคำสั่ง**
AT	OK	ทดสอบการตอบสนองของโมดูล
AT+RESET	OK	รีเซ็ตโมดูล
AT+VERSION?	+VERSION<param>: OK	สอบถามเวอร์ชั่นของเฟิร์มแวร์
AT+ADDR?	+ADDR:<param> OK	สอบถาม IP address ของโมดูล
AT+NAME=<param>	OK	ตั้งชื่อโมดูล
AT+ROLE=<param>	OK	กำหนดโหมดหน้าที่การทำงานของโมดูลว่าเป็น Master/Slave
AT+CMODE=<param>	OK	กำหนดโหมดการเชื่อมต่อ : เชื่อมต่อกับ IP address ที่ตั้ง

		ค่าไว้ หรือเชื่อมต่อกับ IP address ใดๆก็ได้

ตารางที่ 2.2 ชุดคำสั่ง เอที คอมมานด์ สำหรับควบคุมการทำงานของโมดูลบลูทูธ

2.4.5.1 การใช้คำสั่ง เอที คอมมานด์

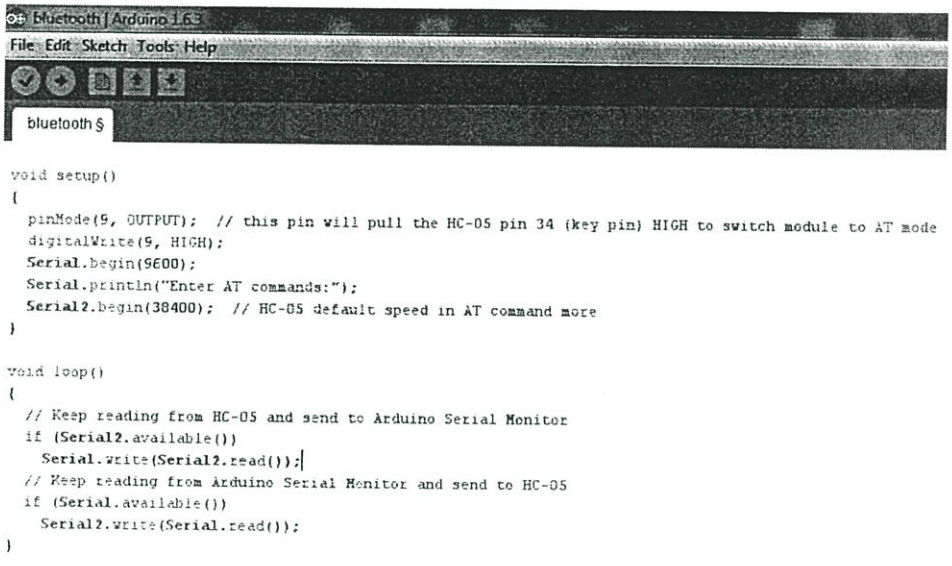
เนื่องจากตัวโมดูลบลูทูธรุ่น HC-05 นี้ ทำงานโดยใช้การสื่อสารแบบอนุกรม หรือ UART (Universal Asynchronous Receiver/Transmitter : Tx, Rx) ซึ่งเป็นรูปแบบที่ง่ายต่อการรับส่งข้อมูล เราจึงสามารถป้อนคำสั่ง เอที คอมมานด์ทางพอร์ตนี้ ไปยังโมดูลบลูทูธ โดยผ่านโปรแกรมเทอร์มินอลทั่วไปได้ ในโปรเจคนี้ โปรแกรมที่ใช้นั้นคือ โปรแกรม อาร์ดูโน่ ไอดีอี 1.6.3 (Arduino IDE 1.6.3) เป็นโปรแกรมที่ใช้งานร่วมกับบอร์ดอาดูโน่ ภายในโปรแกรมนั้นจะมีส่วนเทอร์มินอลโปรแกรมที่เรียกว่า ซีเรียลมอนิเตอร์ (Serial monitor) ใช้สำหรับ ป้อนข้อมูลหรือแสดงข้อมูลผ่านพอร์ตอนุกรมใดๆ ซึ่งส่วนนี้เองทำให้เราสามารถป้อนคำสั่ง เอที คอมมานด์ พร้อมกับดูผลตอบกลับของคำสั่งได้ ในคราวเดียวกัน แสดงดังรูปที่ 2.18



รูปที่ 2.18 หน้าต่างซีเรียลมอนิเตอร์ แสดงช่องป้อนคำสั่งและผลการตอบกลับคำสั่งจากโมดูบลูทูธ

อย่างไรก็ตาม ก่อนที่เราจะสามารถทำการป้อนคำสั่งและดูผลการตอบสนองกลับมาของโมดูลได้นั้น เราต้องทำการตั้งค่า เพื่อเปิดการใช้งานโหมด เอที คอมมานด์ของตัวบลูทูธเสียก่อน โดยเพียงให้ค่าลอจิกของขา EN (สำหรับโมดูลบางรุ่นจะใช้ชื่อว่า KEY (ขา EN หรือ KEY บนตัวโมดูลสำเร็จรูปนั้นจะต่อกับ ขาที่ 34 ของตัวชิพบลูทูธ)) เป็น HIGH (จ่ายไฟ 5VDC) แล้ว

เปิดการทำงานของบลูทูธใหม่อีกครั้ง ก็จะสามารถเข้าสู่โหมด เอที คอมมานด์ได้เรียบร้อย จากนั้นทำการเขียนโปรแกรมลงบนบอร์ดอาร์ดูโน่ เพื่อให้บอร์ดอาร์ดูโน่ตอบผลการตอบสนองกลับมาแสดงบนหน้าจอซีเรียลมอนิเตอร์ ดังรูปที่ 2.19



รูปที่ 2.19 โค้ดโปรแกรมบนอาร์ดูโน่ สำหรับป้อนคำสั่งและแสดงผลการตอบสนอง

หลังจากที่เราได้ป้อนคำสั่ง เอที คอมมานด์ และดูผลการตอบสนองแล้ว ให้เราป้อนลอจิกของขา EN (KEY) เป็น LOW และทำการเปิดการทำงานของโมดูลอีกครั้ง โมดูลบลูทูธก็จะถูกตั้งค่านั้นๆ ตามคำสั่ง เอที คอมมานด์ที่ได้ป้อนเข้าไป เช่น ชื่อโมดูล โหมดการทำงาน เป็นต้น ซึ่งหลังจากนี้ เราก็สามารถนำโมดูลบลูทูธไปใช้ได้ตามปกติ ตามรูปแบบการทำงานที่ได้ถูกตั้งค่าไว้แล้ว

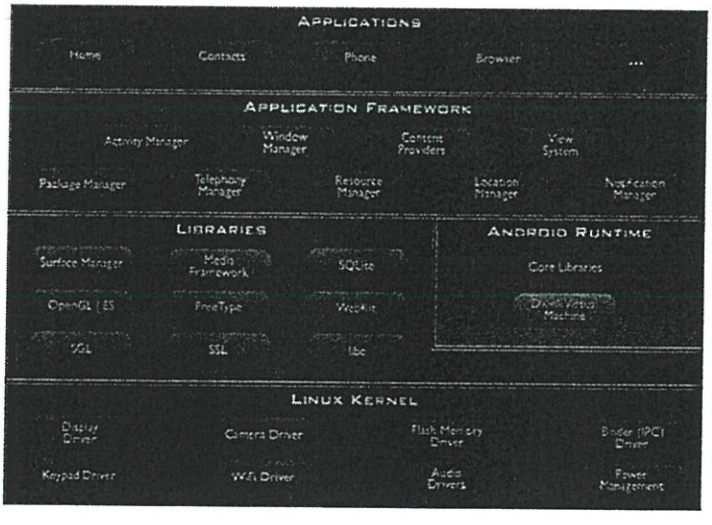
2.5 ระบบปฏิบัติการแอนดรอยด์

ระบบปฏิบัติการแอนดรอยด์ คือระบบปฏิบัติการแบบเปิดเผยซอร์ฟแวร์ต้นฉบับ (Open Source) โดยบริษัท กูเกิ้ล (Google Inc.) ที่ได้รับความนิยมเป็นอย่างสูง เนื่องจากอุปกรณ์ที่ใช้ระบบปฏิบัติการแอนดรอยด์ มีจำนวนมาก อุปกรณ์มีหลากหลายระดับ หลายราคา รวมทั้งสามารถทำงานบนอุปกรณ์ที่มีขนาดหน้าจอ และความละเอียดแตกต่างกันได้ ทำให้ผู้บริโภคสามารถเลือกได้ตามต้องการ และหากมองในทิศทางสำหรับนักพัฒนาโปรแกรม (Programmer) แล้วนั้น การพัฒนาโปรแกรมเพื่อใช้งานบนระบบปฏิบัติการแอนดรอยด์ ไม่ใช่เรื่องที่ยาก เพราะมีข้อมูลในการพัฒนา

รวมทั้ง Android SDK (Software Development Kit) เตรียมไว้ให้กับนักพัฒนาได้เรียนรู้ และเมื่อนักพัฒนาต้องการจะเผยแพร่หรือจำหน่ายโปรแกรมที่พัฒนาเสร็จแล้ว แอนดรอยด์ยังมีตลาดในการเผยแพร่โปรแกรมผ่าน ตลาดแอนดรอยด์ (Android Market) ส่วนโครงสร้างภาษาที่ใช้ในการพัฒนานั้น สำหรับ Android SDK จะยึดโครงสร้างของภาษาจาวา (Java language) ในการเขียนโปรแกรม เพราะโปรแกรมที่พัฒนามาได้จะต้องทำงานอยู่ภายใต้ Dalvik Virtual Machine เช่นเดียวกับโปรแกรมจาวา ที่ต้องทำงานอยู่ภายใต้ Java Virtual Machine (Virtual Machine เปรียบได้กับสภาพแวดล้อมที่โปรแกรมทำงานอยู่)นอกจากนั้นแล้ว แอนดรอยด์ ยังมีโปรแกรมแกรมที่เปิดเผยซอร์ฟแวร์ต้นฉบับ (Open Source) เป็นจำนวนมาก ทำให้นักพัฒนาที่สนใจ สามารถนำซอร์ฟแวร์ต้นฉบับ มาศึกษาได้อย่างไม่ยาก ประกอบกับความนิยมของแอนดรอยด์ได้เพิ่มขึ้นอย่างมาก

2.5.1 โครงสร้างของแอนดรอยด์

การทำความเข้าใจโครงสร้างของระบบปฏิบัติการแอนดรอยด์ ถือว่าเป็นสิ่งสำคัญ เพราะถ้านักพัฒนาโปรแกรม สามารถมองภาพโดยรวมของระบบได้ทั้งหมด จะให้สามารถเข้าใจถึงกระบวนการทำงานได้ดียิ่งขึ้น และสามารถนำไปช่วยในการออกแบบโปรแกรมที่ต้องการพัฒนาเพื่อให้เกิดประสิทธิภาพในการทำงานทั้งนี้แสดงโครงสร้างของแอนดรอยด์ดังรูปที่ 2.20



รูปที่ 2.20 โครงสร้างของระบบปฏิบัติการแอนดรอยด์

จากรูปที่ 2.20 เป็นโครงสร้างของระบบปฏิบัติการแอนดรอยด์ จะสังเกตได้ว่า มีการแบ่งออกมาเป็นส่วนๆ ที่มีความเกี่ยวเนื่องกัน โดยส่วนบนสุดจะเป็นส่วนที่ผู้ใช้งานทำการติดต่อโดยตรงซึ่งก็คือส่วนของแอปพลิเคชั่นจากนั้นก็จะลำดับลงมาเป็นองค์ประกอบอื่นๆตามลำดับ และ

สุดท้ายจะเป็นส่วนที่ติดต่อกับอุปกรณ์โดยผ่านทาง Linux Kernel โครงสร้างของแอนดรอยด์ พอที่จะอธิบายเป็นส่วนๆได้ดังนี้

2.5.1.1 Applications

เป็นส่วนของโปรแกรมที่มีมากับระบบปฏิบัติการ หรือเป็นกลุ่มของโปรแกรมที่ผู้ใช้งานได้ทำการติดตั้งไว้ โดยผู้ใช้งานสามารถเรียกใช้โปรแกรมต่างๆได้โดยตรง ซึ่งการทำงานของแต่ละโปรแกรมจะเป็นไปตามที่ผู้พัฒนาโปรแกรมได้ออกแบบและเขียนโค้ดโปรแกรมเอาไว้

2.5.1.2 Application Framework

เป็นส่วนที่มีการพัฒนาขึ้นเพื่อให้นักพัฒนาสามารถพัฒนาโปรแกรมได้สะดวก และมีประสิทธิภาพมากยิ่งขึ้น โดยนักพัฒนาไม่จำเป็นต้องพัฒนาในส่วนที่มีความยุ่งยากมากๆ เพียงแค่ทำการศึกษาถึงวิธีการเรียกใช้งาน Application Framework ในส่วนที่ต้องการใช้งาน แล้วนำมาใช้งาน ซึ่งมีหลายกลุ่มด้วยกัน ตัวอย่างเช่น

1) Activities Manager เป็นกลุ่มของชุดคำสั่งที่จัดการเกี่ยวกับวงจรการทำงานของหน้าต่างโปรแกรม (Activity)

2) Content Providers เป็นกลุ่มของชุดคำสั่ง ที่ใช้ในการเข้าถึงข้อมูลของโปรแกรมอื่น และสามารถแบ่งปันข้อมูลให้โปรแกรมอื่นเข้าถึงได้

3) View System เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับการจัดการโครงสร้างของหน้าจอที่แสดงผลในส่วนที่ติดต่อกับผู้ใช้งาน (User Interface)

4) Telephony Manager เป็นกลุ่มของชุดคำสั่งที่ใช้ในการเข้าถึงข้อมูลด้านโทรศัพท์ เช่นหมายเลขโทรศัพท์ เป็นต้น

5) Resource Manager เป็นกลุ่มของชุดคำสั่งในการเข้าถึงข้อมูลที่เป็นข้อความ, รูปภาพ

6) Location Manager เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับตำแหน่งทางภูมิศาสตร์ ที่ระบบปฏิบัติการได้รับค่าจากอุปกรณ์

7) Notification Manager เป็นกลุ่มของชุดคำสั่งที่จะถูกเรียกใช้เมื่อโปรแกรม ต้องการแสดงผลให้กับผู้ใช้งาน ผ่านทางแถบสถานะ (Status Bar) ของหน้าจอ

2.5.1.3 Libraries

เป็นส่วนของชุดคำสั่งที่พัฒนาด้วย C/C++ โดยแบ่งชุดคำสั่งออกเป็นกลุ่มตามวัตถุประสงค์ของการใช้งาน เช่น จัดการเกี่ยวกับการแสดงผล (Surface Manage)จัดการเกี่ยวกับการการแสดงภาพและเสียง (Media Framework), จัดการเกี่ยวกับภาพ 3 มิติ และ 2มิติ (Open GL | ES และ SGL), จัดการเกี่ยวกับระบบฐานข้อมูล (SQLite) เป็นต้น

2.5.1.4 Android Runtime

จะมี Dalrvik Virtual Machine ที่ถูกออกแบบมา เพื่อให้ทำงานบนอุปกรณ์ที่มี หน่วยความจำ หน่วยประมวลผลกลางและพลังงาน(Battery)ที่จำกัด ซึ่งการทำงานของ Darvik Virtual Machine จะทำการแปลงไฟล์ที่ต้องการทำงาน ไปเป็นไฟล์ .DEX ก่อนการทำงาน เหตุผลก็เพื่อให้มีประสิทธิภาพเพิ่มขึ้นเมื่อใช้งานกับ หน่วยประมวลผลกลางที่มีความเร็วไม่มาก ส่วนต่อมาคือ Core Libraries ที่เป็นส่วนรวบรวมคำสั่งและชุดคำสั่งสำคัญ โดยถูกเขียนด้วยภาษาจาวา (Java Language)

2.5.1.5 Linux Kernel

เป็นส่วนที่ทำหน้าที่หัวใจสำคัญ ในจัดการกับบริการหลักของระบบปฏิบัติการ เช่น เรื่องหน่วยความจำ พลังงาน ติดต่อกับอุปกรณ์ต่างๆ ความปลอดภัย เครือข่าย โดยแอนดรอยด์ได้นำเอาส่วนนี้มาจากระบบปฏิบัติการลินุกซ์ รุ่น 2.6 (Linux 26. Kernel) ซึ่งได้มีการออกแบบมาเป็นอย่างดี

2.5.2 โปรแกรมและเครื่องมือในการพัฒนา

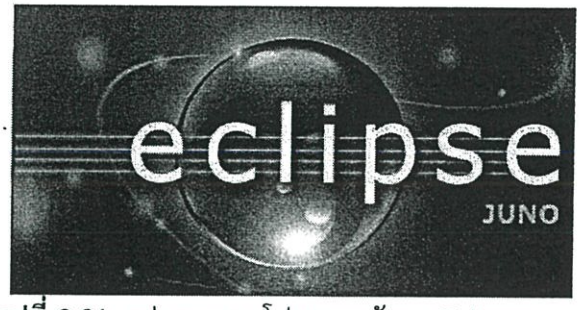
โปรแกรมหรือเครื่องมือที่ใช้พัฒนา แอปพลิเคชั่น แอนดรอยด์ นั้นมีให้เลือกใช้หลายค่ายหลายทางเลือกด้วยกัน แต่สำหรับโครงงานชิ้นนี้จะใช้โปรแกรมที่ Google แนะนำ ซึ่งฟรีและเป็นที่นิยมใช้กันอย่างแพร่หลายสำหรับนักพัฒนาทั่วไป ซึ่งโปรแกรมหรือเครื่องมือที่ต้องติดตั้ง มีดังนี้

2.5.2.1 JDK (Java Development Kit)

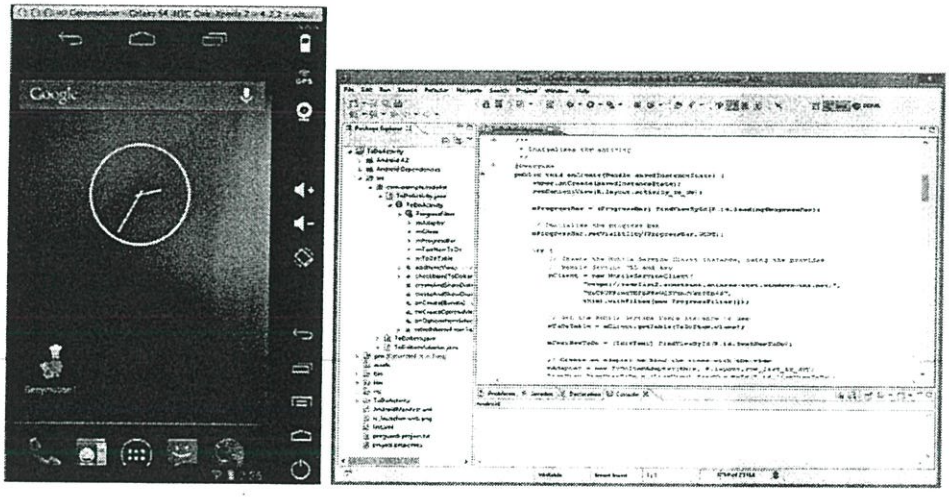
ใช้ในการพัฒนาโปรแกรมจาวาของบริษัทซันไมโครซิสเต็มส์ ซึ่งใครก็ตามที่ต้องการจะพัฒนาโปรแกรมโดยใช้ภาษาจาวา

2.5.2.2 Eclipse

คือโปรแกรมที่ใช้สำหรับพัฒนาภาษา Java ซึ่งโปรแกรม Eclipse เป็นโปรแกรมหนึ่งที่ใช้ในการพัฒนาแอปพลิเคชั่นเซิฟเวอร์ได้อย่างมีประสิทธิภาพ และเนื่องจาก Eclipse เป็นซอฟต์แวร์เปิดเผย ซอร์ฟแวร์ต้นฉบับที่พัฒนาขึ้นเพื่อใช้โดยนักพัฒนาเอง ทำให้ความก้าวหน้าในการพัฒนาของ Eclipse เป็นไปอย่างต่อเนื่องและรวดเร็ว โดย Eclipse มีองค์ประกอบหลักที่เรียกว่า Eclipse Platform ซึ่งให้บริการพื้นฐานหลักสำหรับรวบรวมเครื่องมือต่างๆ จากภายนอกให้สามารถเข้ามาทำงานร่วมกันในสภาพแวดล้อมเดียวกัน และมีองค์ประกอบที่เรียกว่า Plug-in Development Environment (PDE) ซึ่งใช้ในการเพิ่มความสามารถในการพัฒนาซอฟต์แวร์มากขึ้น เครื่องมือภายนอกจะถูกพัฒนาในรูปแบบที่เรียกว่า Eclipse plug-ins ดังนั้นหากต้องการให้ Eclipse ทำงานใดเพิ่มเติม ก็เพียงแต่พัฒนาปลั๊กอิน (plugin) สำหรับงานนั้นขึ้นมา และนำ Plug-in นั้นมาติดตั้งเพิ่มเติมให้กับ Eclipse ที่มีอยู่เท่านั้น Eclipse Plug-in ที่มีมาพร้อมกับ Eclipse เมื่อเราดาวโหลดมาครั้งแรกก็คือองค์ประกอบที่เรียกว่า Java Development Toolkit (JDT) ซึ่งเป็นเครื่องมือในการเขียนและดีบัคโปรแกรมภาษาจาวา ทั้งนี้ ข้อดีของโปรแกรม Eclipse คือ ติดตั้งง่าย สามารถใช้ได้กับ J2SDK (Java Software Development Kit) ได้ทุกเวอร์ชั่น รองรับภาษาต่างประเทศอีกหลายภาษามีปลั๊กอินที่ใช้เสริมประสิทธิภาพของโปรแกรม สามารถทำงานได้กับไฟล์หลายชนิด เช่น HTML, Java, C, JSP, EJB, XML และ GIF



รูปที่ 2.21 รูปภาพแสดงโปรแกรมพัฒนา Eclipse



รูปที่ 2.22 รูปภาพแสดงหน้าต่างใช้งานสำหรับพัฒนาแอพพลิเคชั่นบนโปรแกรม Eclipse

2.5.2.3 Android SDK (Android Software Development Kit)

เป็นชุดโปรแกรมที่ทาง Google พัฒนาออกมาเพื่อแจกจ่ายให้นักพัฒนาแอพพลิเคชั่น หรือผู้สนใจทั่วไปดาวน์โหลดไปใช้กันโดยไม่มีค่าใช้จ่ายซึ่งในชุด SDK นั้นจะมีโปรแกรมและไลบรารี่ต่างๆ ที่จำเป็นต่อการพัฒนาแอพพลิเคชั่นบนแอนดรอยด์ อย่างเช่นอีมูเลเตอร์ (Emulator) ซึ่งทำให้ผู้ใช้สามารถสร้างแอพพลิเคชั่นและนำมาทดลองรันบนตัวอีมูเลเตอร์ ก่อน โดยมีสภาวะแวดล้อมเหมือนมือถือที่รันระบบปฏิบัติการแอนดรอยด์จริงๆ

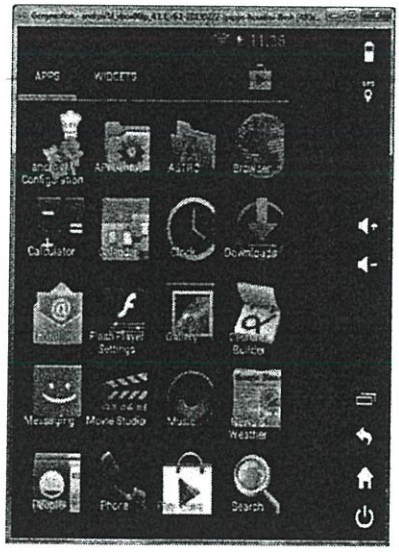
2.5.2.4 ADT (Android Development Tools)

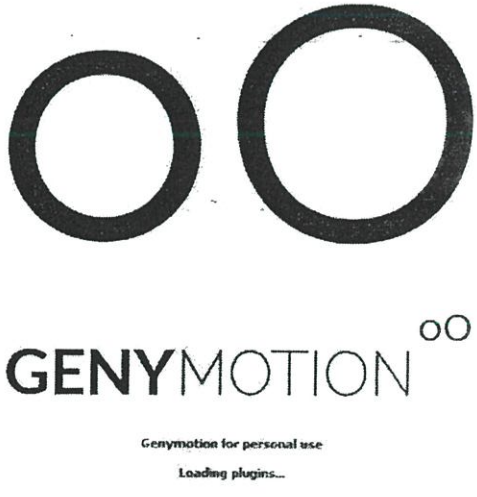
คือ เครื่องมือที่ใช้พัฒนาแอนดรอยด์ ในการพัฒนา Application บนระบบ Android OS จะใช้ภาษา Java โดยต้องติดตั้งส่วนเสริม ซึ่งก็คือเจ้า ADT หรือ Android Development Tools ซึ่งเป็นส่วนเสริมของ IDE หรือที่หลายคนมักเรียกว่าเป็น ปลั๊กอินของโปรแกรม Eclipse ซึ่งใช้ในการเขียนโปรแกรม และ ADT นี้ก็รวมอยู่เป็นส่วนหนึ่งของ Android SDK

2.5.3 การทดลองรันแอปพลิเคชั่นสามารถทำได้ 2 วิธีคือ

2.5.3.1 รันบน อีมูเลเตอร์

คือการจำลองเครื่องคอมพิวเตอร์ให้เป็นสมาร์ทโฟน หรือแท็บเล็ต (Tablet) ซึ่งตัวจำลองนี้เรียกว่า ADV (Android Virtual Device) ซึ่งในที่นี้ตัวอีมูเลเตอร์ที่นำมาใช้กับโปรเจค จะเป็นโปรแกรมที่มีชื่อว่า Genymotion





รูปที่ 2.23 รูปภาพแสดงหน้าต่างใช้งานสำหรับโปรแกรมจำลองโทรศัพท์มือถือ Genymotion

2.5.3.2 **รันบนอุปกรณ์จริง**

โดยส่งไฟล์แอปพลิเคชั่น ไปรันบนสมาร์ทโฟน แท็บเล็ต โดยการติดตั้งไดร์เวอร์ (Driver) สำหรับอุปกรณ์นั้นๆ

2.5.4 โฟลเดอร์และไฟล์ในโปรเจ็คแอพ

โปรเจ็คแอพ (Project App) ประกอบไปด้วย

2.5.4.1 src

ประกอบด้วยไฟล์ซอสโค้ด java (Source code)

2.5.4.2 gen

ประกอบด้วยไฟล์ที่ระบบสร้างให้อัตโนมัติ เช่น R.java ใช้สำหรับการอ้างอิง(Reference) รีซอสหรือทรัพยากรต่างๆที่ใช้ในแอปพลิเคชั่น ซึ่งไฟล์นี้ ระบบสร้างให้อัตโนมัติ เราไม่ต้องยุ่งหรือแก้ไข (สามารถเปิดดูข้อมูลภายในเพื่อศึกษาได้)

2.5.4.3 Android

เก็บไฟล์ไลบรารี่ที่ใช้งานสำหรับแอปพลิเคชั่น

2.5.4.4 assets

โฟลเดอร์สำหรับเก็บไฟล์ต่างๆที่ใช้งานในแอปพลิเคชั่น เช่น ไฟล์ฐานข้อมูล ภาพ ฯลฯ

2.5.4.5 bin

เก็บไฟล์ไบนารี่ที่คอลไพล์หรือบิ้วท์ (Built) เช่น ไฟล์ .apk (Android Package) ซึ่งไฟล์นี้เป็นไฟล์แพคเกจ แอปพลิเคชั่นสำเร็จที่นำไปติดตั้งและรันบนเครื่องแอนดรอยด์

2.5.4.6 red

โฟลเดอร์เก็บทรัพยากรที่ใช้กับแอปพลิเคชั่นได้แก่

1) drawable... เก็บไฟล์ภาพสำหรับเครื่องแอนดรอยด์หน้าจอขนาดต่างๆ

2) layout เก็บไฟล์ xml ที่ใช้เป็นเลย์เอ้าท์ (Layout ควบคุมรูปแบบการแสดงออกจอภาพ

3) values เก็บไฟล์ข้อมูลต่างๆ เช่น ข้อความ (Strings), สไตล์ควบคุมรูปแบบ (Styles)

2.5.4.7 AndroidManifest.xml

เป็นไฟล์ควบคุมระบบที่สำคัญมากๆ กำหนดการใช้งานในส่วนต่างๆ เช่น จะให้ .java ใดรันเป็นอันดับแรก (กรณีมีหลายไฟล์) , กำหนดเปิดความยินยอม (permission) ต่างๆ เช่นให้แอปพลิเคชั่นเชื่อมต่ออินเทอร์เน็ตได้หรือไม่ บันทึกข้อมูลลงในการ์ดความจำได้หรือไม่, ใช้งานกล้องได้หรือไม่ ฯลฯ เก็บเลขเวอร์ชั่นของแอปพลิเคชั่นและเก็บการตั้งค่าอื่นๆ

2.5.5 Activity และ Layout

2.5.5.1 activity

เป็นเหมือนแกนกลไกหลักที่รับข้อมูลจากส่วนต่างๆมา เช่น Values, Style, Res ได้แก่ภาพ ข้อความ ข้อมูลต่างๆ เมื่อประมวลผลเสร็จแล้วก็ส่งต่อให้แก่ Layout

2.5.5.2 Layout

รับข้อมูลจาก activity แล้วแสดงผลออกที่จอแสดงผล

2.5.6 การพัฒนาโปรแกรม

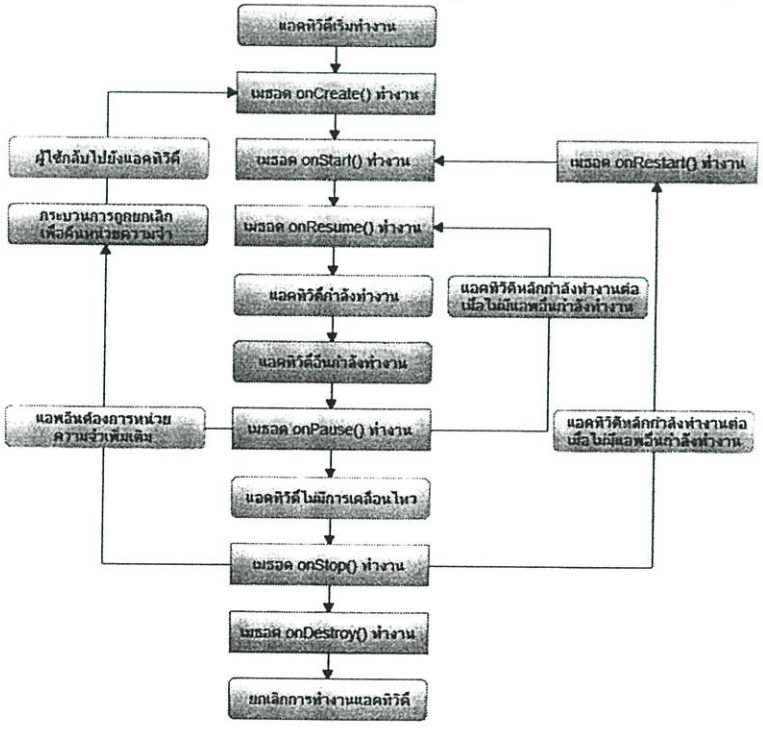
การพัฒนาโปรแกรมมีลักษณะที่สำคัญ ดังตารางที่ 2.3

ตารางที่ 2.3 แสดง ประเภทการทำงานของแอปพลิเคชั่น

ลักษณะการทำงาน	Java Base Class	ตัวอย่างการทำงาน
การทำงานของผู้ใช้งาน	Activity	แก้ไขไฟล์ข้อมูล หรือเล่นเกม
ทำงานเบื้องหลัง	Service	เล่นไฟล์เพลง หรือ แสดงข้อมูลสภาพอากาศ
รับแจ้งเหตุ	BroadcastReceiver	นาฬิกาปลุก หรือแจ้งเตือน
การเก็บและเรียก	ContentProvider	การเปิดสมุดโทรศัพท์ ใช้ข้อมูล

2.5.7 การทำงานของ Activity

การทำงาน ของ Activity คือ มีลักษณะดังรูปที่ 2.24



รูปที่ 2.24 การทำงานของ Activity

1) onCreate( ) คือ การเริ่มต้นทำงานของโปรแกรม

2) onStart( ) คือ ทำงานเมื่อแสดงหน้าจอหลักของ โปรแกรม

3) onResume( ) คือ ทำงานเมื่อมีการใช้งาน กับผู้ใช้

4) onPause( ) คือ ทำงานเมื่อต้องการหยุดโปรแกรม

5) onStop( ) คือ ทำงานเมื่อโปรแกรม ไม่ได้แสดงผล หน้าจอ

6) onDestroy( ) คือ ทำงานก่อนที่โปรแกรมจะถูกปิด

7) onRestart( ) คือ ทำงานใหม่อีกครั้ง หลังจากถูก onStop()

2.5.8 การจัดเก็บข้อมูลใน Activities

แอปพลิเคชั่นของแอนดรอยด์ จะมีการทำงานร่วมกับข้อมูลต่างๆ อยู่เสมอ ในระบบแอนดรอยด์มีไลบารี่ ที่เกี่ยวข้องอยู่หลายตัว เช่น ข้อมูลประเภท Shared Preferences เหมาะสำหรับข้อมูลที่มีขนาดไม่ใหญ่ เช่น ค่าเซ็ทอัพต่างๆ ฐานข้อมูล SQLite เหมาะกับข้อมูล ประเภททะเบียนที่รายการจำนวนมากๆ ไฟล์ข้อมูลประเภทสตรีมโดยใช้คำสั่ง InputFileStream และ OutputFileStream ในการอ่านเขียนข้อมูล

2.5.9 การใช้งาน Threading

ในกรณีที่เราต้องการให้มีการกระทำใดๆ หลายๆ อย่างพร้อมๆกัน หรือการกระทำนั้นๆ เกิดขึ้นอัตโนมัติโดยที่เราไม่ได้ใช้งานใดๆ ผ่าน User Interface จะต้องให้งาน Thread เพื่อให้ได้ผลลัพธ์ดังกล่าว โดยส่วนใหญ่ในการใช้งาน Thread มักจะมีการหน่วงเวลาเพื่อให้เกิดผลการกระทำนั้นๆ ตามเวลา หรือช่วงเวลาที่ต้องการ

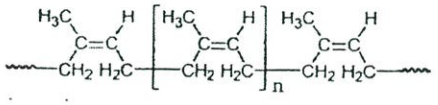
2.5.10 การใช้งาน Handler

โดยปกติแล้ว เมื่อต้องการจะเปลี่ยนแปลงค่าของวัตถุต่างๆ เช่น เปลี่ยนข้อความใน TextView เปลี่ยนรูปภาพใน ImageView เรามักจะกระทำโดยผ่าน Interact กับวัตถุนั้นๆ หรือวัตถุๆ ข้างเคียง เช่น กดปุ่มแล้วทำให้ข้อความเปลี่ยนแปลงแต่หากมีความต้องการที่จะเปลี่ยนแปลงค่าของวัตถุต่างๆ โดยที่เราไม่ได้ทำการ Interact กับวัตถุใดๆ เลย เช่น การเปลี่ยนแปลงค่าโดยผ่าน Thread หรือเปลี่ยนแปลงค่าเพราะทำการโหลดข้อมูลเข้ามาจากอินเตอร์เน็ต เป็นต้น การกระทำเหล่านี้จะต้องใช้ Handler เข้ามาช่วย

2.6 ยางธรรมชาติ (Natural rubber)

2.6.1 โครงสร้างทางเคมี

ยางธรรมชาติที่ได้จากยางพันธุ์ฮีเวียบราซิเลียนซิส (Hevea brasiliensis) เป็นพอลิเมอร์ที่มีไฮโดรคาร์บอนเป็นองค์ประกอบหลักร้อยละ 94 โดยน้ำหนัก ส่วนประกอบที่เหลือเป็นโปรตีน ไขมันและเกลืออินทรีย์อื่นๆ (เช่น ขี้เถ้า และอนุมูลของโลหะ เป็นต้น) มีหน่วยที่ซ้ำๆ กันเป็นไอโซพรีน (C_5H_8) มาต่อกันเป็นโมเลกุลยาวเรียกว่า พอลิโซพรีน $(C_5H_8)_n$ โดย n มีค่าตั้งแต่ 5,000 – 15,000 ยางธรรมชาติประกอบด้วยโมเลกุลที่มีน้ำหนักโมเลกุลตั้งแต่ 50,000 – 3,000,000 และประมาณร้อยละ 60 ของโมเลกุลเหล่านี้ มีน้ำหนักโมเลกุลสูงกว่า 1,300,000 โครงสร้างโมเลกุลของยางธรรมชาติมีลักษณะเป็นไอโซเมอร์ชนิด cis-Isomer ดังนั้นจึงมักจะเรียกยางธรรมชาติว่า cis-1,4-Polyisoprene ซึ่งมีโครงสร้างแสดงดังรูป



รูปที่ 2.45 แสดงโครงสร้างโมเลกุลของยางธรรมชาติ

โครงสร้างหลักที่มีผลกระทบต่อสมบัติของยางธรรมชาติ คือ

1. การมีองค์ประกอบที่เป็นคาร์บอนและไฮโดรเจนเป็นส่วนใหญ่ ทำให้ยางธรรมชาติมีสมบัติไม่ทนน้ำมันแต่เป็นฉนวนไฟฟ้าที่ดี
2. มีพันธะคู่ที่ว่องไวต่อปฏิกิริยา ทำให้เกิดปฏิกิริยาวัลคาไนซ์ด้วยกำมะถันได้ดี แต่ทำให้เกิดปฏิกิริยาได้ง่ายกับออกซิเจนและโอโซนจึงเป็นสาเหตุการเสื่อมของยางธรรมชาติ
3. มีสายโซ่โมเลกุลที่เคลื่อนไหวหกงอไปมาและยืดหยุ่นได้ง่าย ทำให้ยางธรรมชาติคงสภาพความยืดหยุ่น (Elasticity) ได้ดีและสามารถใช้งานที่อุณหภูมิต่ำได้
4. โครงสร้างโมเลกุลที่สม่ำเสมอทำให้ยางธรรมชาติสามารถเกิดผลึกได้เมื่อยืด จึงมีความต้านทานต่อแรงดึงสูง และทำให้มีความแข็งแรงขณะไม่วัลคาไนซ์ (Green strength) สูง และความเหนียวติดกัน (Tack) ดี
5. มีน้ำหนักโมเลกุลสูง ทำยางแข็งเกินไปที่จะนำไปแปรรูปโดยตรงจะต้องนำยางไปบดเพื่อโมเลกุลเล็กลงก่อนนำไปใช้งาน

2.6.2 สมบัติทั่วไปของยางธรรมชาติวัลคาไนซ์

2.6.2.1 ความแข็ง (Hardness)

ยางธรรมชาติสามารถวัลคาไนซ์ให้มีความแข็งต่างๆ กันได้ ตั้งแต่นิ่มมาก ไปจนถึงความแข็งของยางอีโบไนด์ การปรับความแข็งของยางทำได้โดยการปรบเปลี่ยนปริมาณสารตัวเติมหรือการเปลี่ยนปริมาณกำมะถัน

2.6.2.2 ความต้านทาต่อแรงดึง (Tensile strength)

ยางธรรมชาติมีโครงสร้างโมเลกุลที่สม่ำเสมอ ทำให้สามารถตกผลึกได้เมื่อดึง ดังนั้นจึงทำให้ยางธรรมชาติมีความต้านทานต่อแรงดึงและความต้านทานต่อแรงฉีกขาดสูง ยางธรรมชาติที่ไม่มีสารตัวเติมสามารถคอมปาวด์ให้มีความแข็งแรงสูงถึงประมาณ 30 MPa ซึ่งสมบัตินี้ทำให้ยางธรรมชาติมีความเหมมาะสมในการทำผลิตภัณฑ์ที่มีความบาง นิ่ม และแข็งแรง

เนื่องจากยางธรรมชาติมีความแข็งแรงในตัวเองสูง จึงทำให้สามารถใช้สารตัวเติมราคาถูก เช่น แคลเซียมคาร์บอเนตและดินขาว สำหรับในผลิตภัณฑ์ที่ไม่ต้องการความแข็งแรงเป็นสมบัติลัก ซึ่งจากการที่ยางธรรมชาติสามารถใช้สารตัวเติมราคาถูกได้ ทำให้ยางธรรมชาติได้เปรียบกว่ายางอื่นๆ ในด้านต้นทุนการผลิต การใช้สารตัวเติมชนิดเสริมแรงที่มีราคาแพง เช่น เขม่าดำและซิลิกา จะใช้ก็ต่อเมื่อต้องการความแข็งแรงสูงเท่านั้น

2.6.2.3 ความสามารถในการยืดจนขาด (Elongation at break)

ยางธรรมชาติที่ไม่ผสมสารตัวเติมสามารถยืดได้สูงถึง 100% ความสามารถในการยืดของยางธรรมชาติลดลงตามการเพิ่มปริมาณของสารตัวเติม และตามระดับของพันธะวัลคาไนซ์ การมีสมบัติด้านการยืดสูงทำให้สามารถใช้ยางธรรมชาติในการทำผลิตภัณฑ์ที่ต้องการวามยืดสูง

2.6.2.4 ความต้านทานต่อการฉีกขาด (Tear resistance)

ยางธรรมชาติมีความสามารถในการตกผลึกได้ดีขณะถูกดึงจึงทำให้มีความต้านทานต่อการฉีกขาดสูงกว่ายางสังเคราะห์อื่นๆ ยกเว้นยางพอลิยูรีเทนที่เชื่อมโยงพันธะด้วยไอโซไซยาเนตและความต้านทานต่อการฉีกขาดจะเพิ่มขึ้นเมื่อใช้สารตัวเติมเสริมประสิทธิภาพร่วมด้วย

2.6.2.5 การใช้งานที่อุณหภูมิต่ำ (Low temperature flexibility)

การที่มีความสามารถในการยืดหยุ่นได้ที่อุณหภูมิต่ำ โดอุณหภูมิกลาสทรานซิชั่น มีค่าประมาณ -72 °C ดังนั้นทำให้มีความเป็นยางได้แม้อุณหภูมิต่ำ การใช้งานยางที่อุณหภูมิต่ำจะใช้ได้ในช่วงเวลาสั้นๆ แต่ถ้าเป็นเวลานานโมเลกุลยางจะเรียงตัวกันเกิดผลึกได้ โดยอัตราการเกิดผลึกมีค่าสูงสุดที่อุณหภูมิ -26 °C นอกจากนั้นการผสมน้ำมันลงในยางธรรมชาติจะทำให้สามารถใช้งานที่อุณหภูมิต่ำกว่าเดิม

2.6.2.6 ความทนทานต่อการเสื่อมสภาพ (Aging resistance)

พันธะคู่ที่ว่องไวต่อปฏิกิริยาในโมเลกุลยางธรรมชาติทำให้เกิดการวัลคาไนซ์ได้อย่างรวดเร็วเมื่อใช้กำมะถันเป็นสารวัลคาไนซ์ แต่พันธะคู่ก็เป็นจุดอ่อนที่ทำให้เกิดปฏิกิริยากับออกซิเจนและโอโซน ทำให้เกิดพันธะที่มีพลังงานพันธะต่ำ (C-o และ O-O) ทำให้เกิดการขาดของโมเลกุลได้ง่าย ทำให้สมบัติเชิงกลของยางด้อยลง ดังนั้นยางธรรมชาติจึงไม่เหมาะกับงานหลายชนิดที่ต้องการความต้านทานต่อการเสื่อมสภาพ โดยเฉพาะอย่างยิ่งที่อุณหภูมิสูงและมีโอโซนอยู่ด้วยเพื่อจะให้ได้ยางธรรมชาติวัลคาไนซ์มีความต้านทานต่อการเสื่อมีขึ้นจึงผสมสารต้านออกซิเดชันและสารตัวเร่งประเภทไธอาโซลให้เกิดการวัลคาไนซ์โดยใช้ระยะเวลาสั้นที่อุณหภูมิไม่สูงเกินไป

2.6.2.7 ความทนทานต่อความร้อน (Heat resistance)

การใช้งานของยางธรรมชาติค่อนข้างจำกัดการใช้งานในสภาวะที่รุนแรงที่มีผลต่อการเสื่อมสภาพของยาง ยางธรรมชาติจึงไม่เหมาะกับการใช้งานที่อุณหภูมิสูงกว่า 70 °C เป็นระยะเวลานาน โดยเฉพาะอย่างยิ่งถ้าเป็นยางบาง ซึ่งมีพื้นที่ผิวสัมผัสมากและจะไม่ใช้ยางธรรมชาติในกรณีที่ต้องยืดยางในบรรยากาศที่มีโอโซนอยู่ด้วย แต่ถ้าจำเป็นต้องใช้จะมีการผสมสารต้านออกซิเดชันและสารต้านโอโซนซึ่งเป็นสารช่วยป้องกันการเสื่อมสภาพ

บทที่ 3

ขั้นตอนการดำเนินงาน

3.1 กล่าวนำ

การทำโครงงานนั้นเมื่อเราได้ทำการศึกษาในรายละเอียดที่เกี่ยวข้องกับโครงงานไม่ว่าจะเป็นในส่วนของทฤษฎีต่างๆ เช่น ภาคสื่อสารรับส่งข้อมูล, ภาควงจรการควบคุม หรือภาควงจรขับเคลื่อน เป็นต้น ในขั้นต่อมาก็จะทำการนำทฤษฎีเหล่านั้นมาประยุกต์ใช้กับโครงงานของเรา เพื่อให้เกิดผลลัพธ์ตามจุดประสงค์ที่ได้วางไว้ เช่น การสร้างการเคลื่อนที่ของหุ่นยนต์ เป็นต้น

3.2 แนวทางการดำเนินงาน

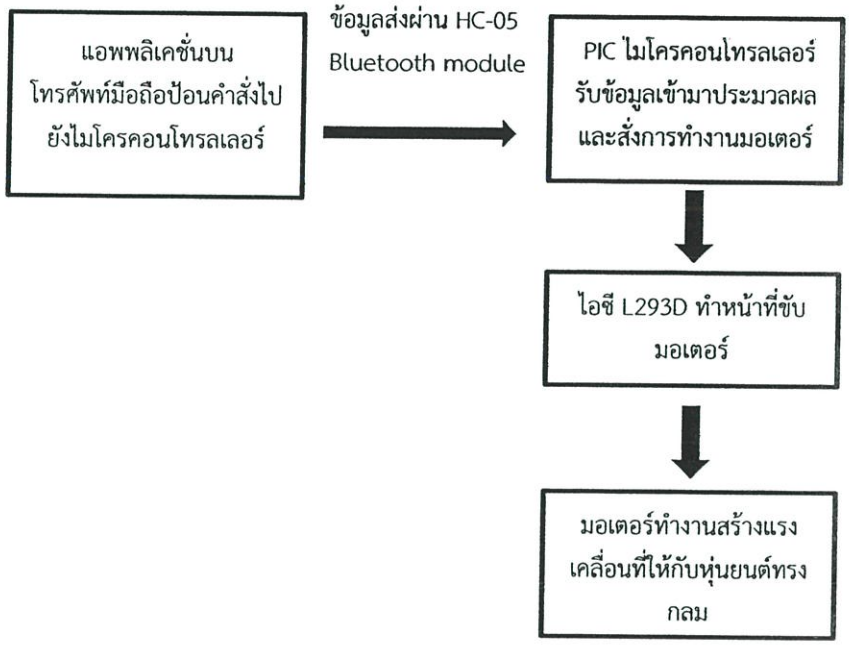
จากเนื้อหาส่วนหลักการและทฤษฎีการเคลื่อนที่ของหุ่นยนต์ตามที่กล่าวไว้แล้วในบทที่ 2 โครงงานนี้ได้เลือกใช้วิธีการสร้างแรงขับเคลื่อนหุ่นยนต์โดยอาศัยการเคลื่อนที่ของหุ่นยนต์มีล้อวิ่งอยู่ภายในทรงกลม เพราะฉะนั้น โครงงานนี้จึงมุ่งเน้นไปที่การพัฒนาโครงสร้างและการควบคุมการทำงานของรถบังคับที่อยู่ภายในหุ่นยนต์ทรงกลมเป็นหลัก จากนั้นจึงพัฒนาในส่วนโครงสร้างของทรงกลม

ในส่วนของการดำเนินงานนั้น สามารถแบ่งส่วนการทำงานออกได้เป็น 2 ส่วนใหญ่ๆ ด้วยกันดังนี้

1. ส่วนซอฟท์แวร์
   a. ซอฟท์แวร์ควบคุมการทำงานของหุ่นยนต์
   b. ซอฟท์แวร์ส่วนหน้าต่างผู้ใช้งาน
2. ส่วนฮาร์ดแวร์
   a. วงจรไฟฟ้าควบคุมการทำงานของหุ่นยนต์
   b. โครงสร้างของหุ่นยนต์และส่วนประกอบอื่นๆ

3.2.1 การทำงานโดยสังเขปของหุ่นยนต์

การทำงานเริ่มต้นจากสั่งการผ่านแอพพลิเคชั่นบนโทรศัพท์มือถือ โดยคำสั่งที่ส่งมาจะเป็นค่ามุมองศา (Angle) เพื่อใช้บอกทิศทางการเคลื่อนที่ของรถบังคับ ว่าควรจะเดินหน้า, ถอยหลัง หรือ หันเลี้ยวไปไหนทิศทางใด และอีกค่าหนึ่งคือค่าความเร็ว (Speed) เพื่อสั่งการเคลื่อนที่ของรถบังคับด้วยความเร็วต่างๆ ซึ่งค่าทั้งสองค่านี้จะถูกส่งจากโทรศัพท์มือถือผ่านเทคโนโลยีบลูทูธ มายังตัวโมดูลบลูทูธฝั่งที่อยู่กับวงจรไมโครคอนโทรลเลอร์บนรถบังคับจากนั้น ไมโครคอนโทรลเลอร์ก็จะทำการรับข้อมูลมาประมวลผล แล้วสั่งการไปยังไอซีขับมอเตอร์ L293D เพื่อทำการขับมอเตอร์ให้หมุนไปข้างหน้าหรือหลังตามความเร็วต่างๆ โดยรูปที่ 3.1.1 นี้จะเป็นรูปภาพแสดงบล็อกไดอะแกรม การทำงานของหุ่นยนต์โดยสังเขป



รูปที่ 3.1.1 บล็อกไดอะแกรมแสดงการทำงานของหุ่นยนต์

3.2.2 ส่วนซอฟท์แวร์ควบคุมการทำงานของหุ่นยนต์

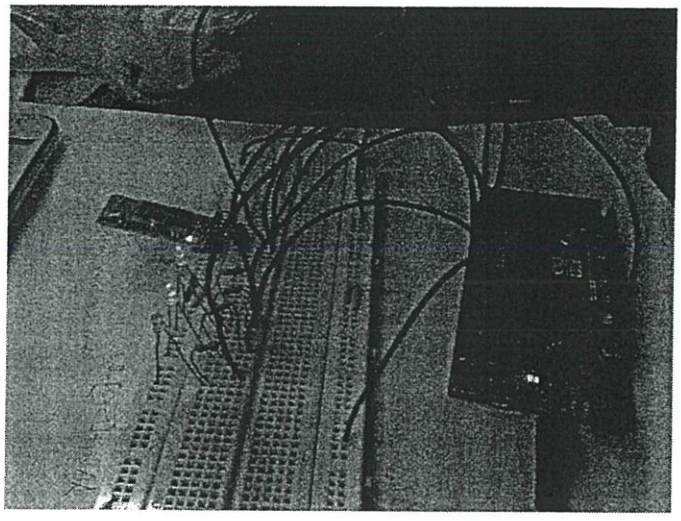
ซอฟท์แวร์สำหรับควบคุมการทำงานของหุ่นยนต์ในโครงงานนี้ ได้มีการออกแบบไว้อยู่ 2 ประเภทด้วยกัน คือ

1. โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control)
2. โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (Continuous Control)

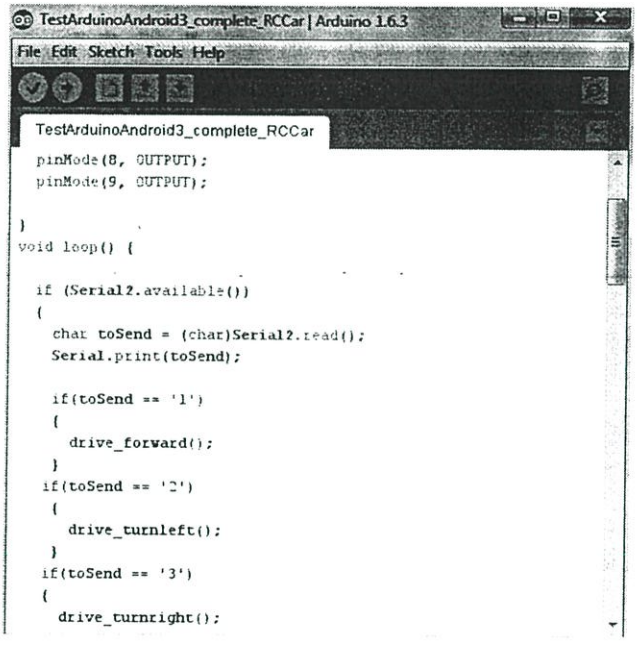
3.2.2.1 โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control)

เป็นโปรแกรมบนไมโครคอนโทรลเลอร์สำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง โดยตัวโปรแกรมที่เขียนลงไมโครคอนโทรลเลอร์นั้น จะสามารถสั่งการรถบังคับได้เพียงแค่ เดินหน้า ถอยหลัง เลี้ยวซ้าย เลี้ยวขวา และ หยุด เท่านั้น ไม่ได้มีการเคลื่อนที่ในทิศทางอื่นๆเพิ่มเติม และไม่ได้มีการควบคุมความเร็วในการหมุนของมอเตอร์แต่อย่างใด เพราะฉะนั้น รถบังคับจึงเคลื่อนที่ได้แค่ทิศทางตามที่กำหนดเท่านั้น

ส่วนของไมโครคอนโทรลเลอร์ ในตอนต้นเลือกใช้งานกับบอร์ด Arduino DUE เพื่อทดสอบการทำงานของโปรแกรมก่อน เหตุผลที่เลือกทดสอบกับบอร์ด Arduino เนื่องจากว่า มีความสามารถในการต่อวงจรที่ง่าย อีกทั้งการเขียนและแก้ไขโปรแกรมก็สามารถทำได้ง่ายอีกด้วย



รูปที่ 3.1.2 การทดสอบโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับบนบอร์ด Arduino



รูปที่ 3.1.3 ตัวอย่างโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับบนบอร์ด Arduino

ในการทดสอบนั้น เราจะให้บอร์ด Arduino DUE สั่งการหลอดไฟ LED ทั้ง 4 หลอด โดยสมมติให้หลอด LED เป็นขาของมอเตอร์ทั้ง 2 ตัว (มอเตอร์ซ้าย และ มอเตอร์ขวา) โดยสั่งการผ่านแอพพลิเคชั่นบนโทรศัพท์มือถือ และมีโมดูลบลูทูธ HC-05 ทำหน้าที่เชื่อมต่อสื่อสารระหว่างโทรศัพท์มือถือกับบอร์ด Arduino DUE

หลังจากทดสอบการทำงานของโปรแกรมควบคุมบนบอร์ด Arduino DUE เสร็จเรียบร้อยแล้วก็จะทำการเขียนโปรแกรมลงบน PIC ไมโครคอนโทรลเลอร์ ซึ่งเป็น

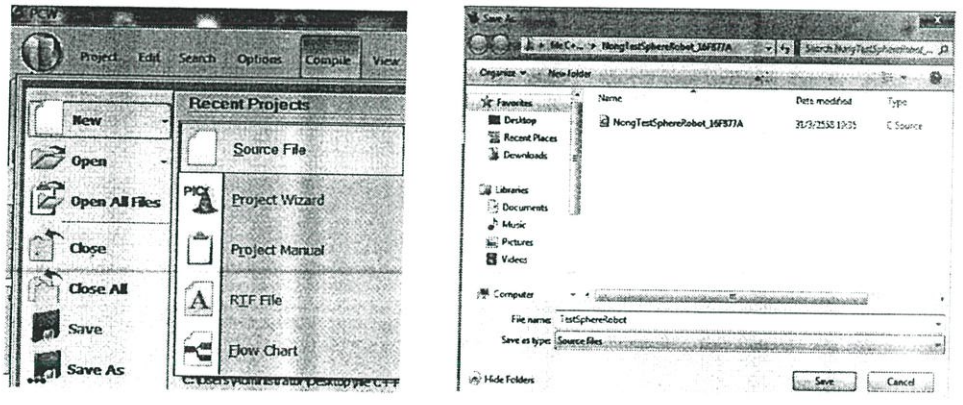
ไมโครคอนโทรลเลอร์ที่เราต้องการใช้งานจริง เหตุผลที่เลือกใช้ PIC ไมโครคอนโทรลเลอร์แทนบอร์ด Arduino DUE เนื่องจากว่า

1.ขนาดของไมโครคอนโทรลเลอร์ : บอร์ด Arduino DUE มีขนาดใหญ่เกินไปสำหรับโครงทรงกลมที่ใช้ใส่รถบังคับไว้ข้างในเพื่อสร้างการเคลื่อนที่ให้กับหุ่นยนต์ จึงทำให้จำเป็นต้องเลือกใช้ไมโครคอนโทรลเลอร์ที่มีขนาดเล็กกว่า

2.จำนวน I/O : เนื่องจากขาอินพุทเอาท์พุทที่ใช้ในการขับมอเตอร์นั้น ตามจริงแล้วใช้เพียงไม่กี่ขา ทำให้บอร์ด Arduino DUE นั้นไม่เหมาะกับการใช้งาน เนื่องจากเป็นการใช้ขา I/O ที่สิ้นเปลืองเกินความจำเป็น

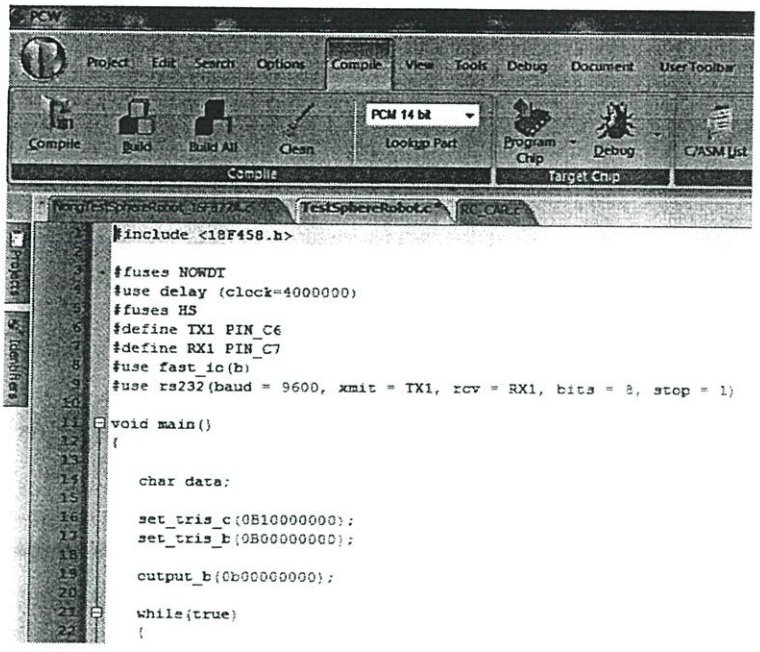
3. ราคา : เนื่องจากว่าบอร์ด Arduino DUE นั้นมีราคาแพง ถ้าเทียบกับ PIC ไมโครคอนโทรลเลอร์ หากเกิดความเสียหายขึ้นกับตัวรถบังคับ อาจจะเกิดความเสียหายต่อบอร์ดด้วย

ในส่วนต่อไปเป็นขั้นตอนการเขียนโปรแกรมลงบน PIC ไมโครคอนโทรลเลอร์เบอร์ 16F977A ซึ่งเริ่มตั้งแต่การสร้าง Source file โดยใช้โปรแกรม CCS 'C' Compiler และเครื่องโปรแกรมเมอร์ PX-700 ที่มีประสิทธิภาพเทียบเท่าเครื่องโปรแกรมเมอร์ PICkit2

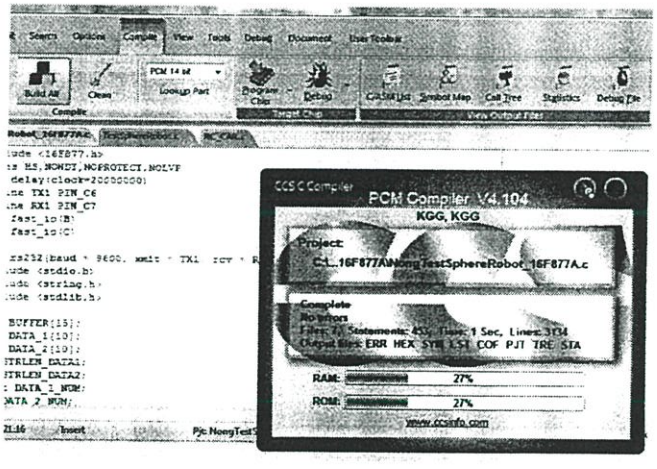


รูปที่ 3.1.4 ขั้นตอนการสร้าง Source file ใหม่ และตั้งชื่อ Source file

เริ่มต้นจากการสร้าง Source file และการตั้งชื่อ Source file ตามรูปที่ 3.1.4 ซึ่งจะได้ไฟล์ .C ขึ้นมาเก็บไว้ยังไดเรกทอรี่ที่เราเลือก

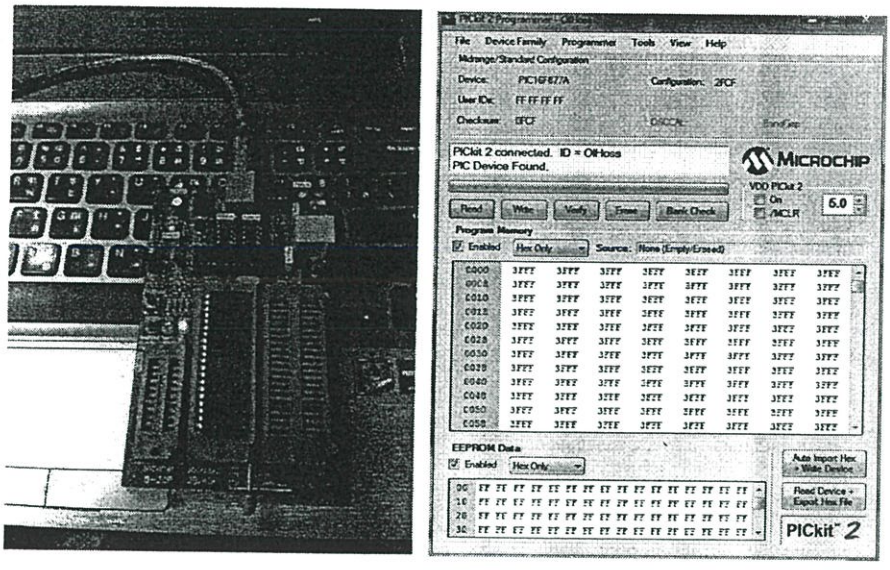


รูปที่ 3.1.5 รูปภาพแสดงการเขียนโปรแกรมลงบน โปรแกรม CCS 'C' Compiler

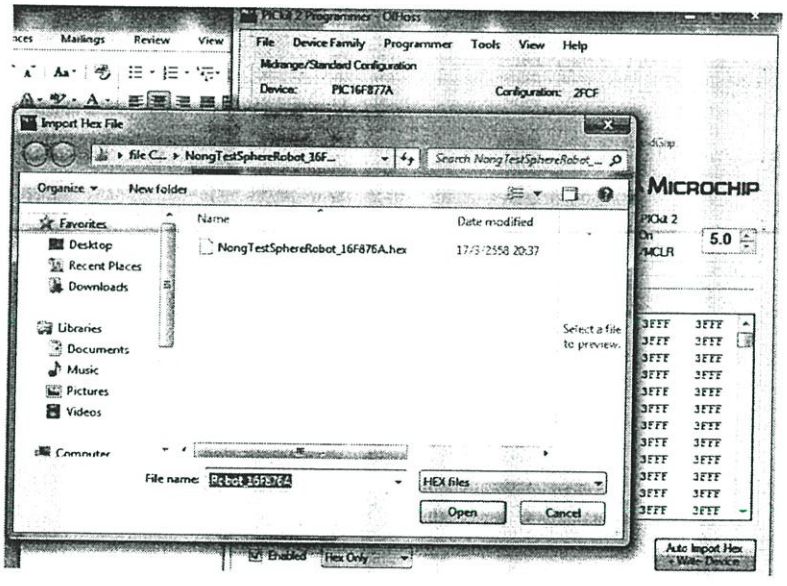


รูปที่ 3.1.6 ขั้นตอนการสร้าง ไฟล์ .HEX สำหรับโปรแกรมใส่ไมโครคอนโทรลเลอร์

จากนั้น ให้ทำการเขียนโค้ดโปรแกรมควบคุมการทำงานของรถบังคับตามเงื่อนไขต่างๆที่ต้องการ และทำการสร้างไฟล์ .HEX เพื่อใช้ในการโปรแกรมลงบนไมโครคอนโทรลเลอร์ตามรูปที่ 3.1.5 และ 3.1.6

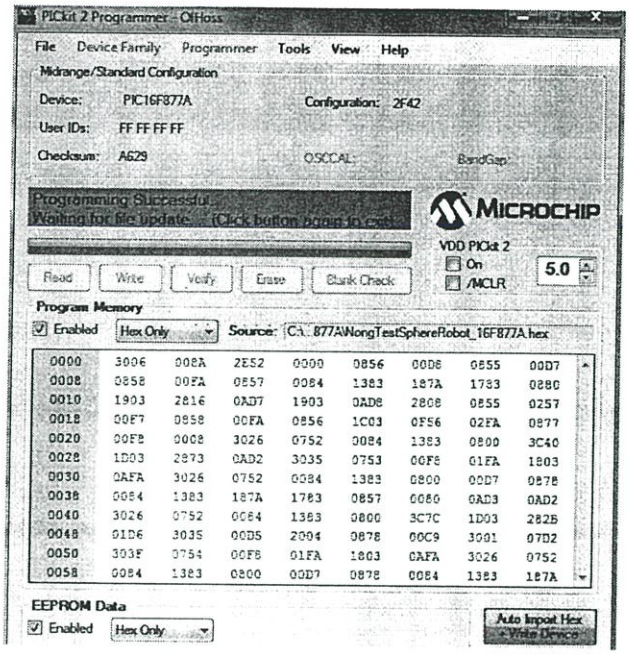


รูปที่ 3.1.7 ขั้นตอนเริ่มต้นการโปรแกรมไมโครคอนโทรลเลอร์ด้วยโปรแกรม PICkit2



รูปที่ 3.1.8 ขั้นตอนการเรียกไฟล์ .HEX ของโค้ดที่ได้เขียนไว้มาโปรแกรมลงบนไมโครคอนโทรลเลอร์

จากนั้นนำไมโครคอนโทรลเลอร์ที่ต้องการใช้ใส่บนเครื่องโปรแกรม PX-700 และเข้าโปรแกรม PICkit2 โปรแกรมจะทำการค้นหาเบอร์ของไมโครคอนโทรลเลอร์โดยอัตโนมัติและแสดงผลการค้นหา ถ้าหากค้นหาเบอร์ได้ตรงกันกับตัวไมโครคอนโทรลเลอร์จริง ตัวอุปกรณ์ก็พร้อมที่จะทำการโปรแกรมไฟล์ .HEX ลงไมโครคอนโทรลเลอร์แล้ว โดยการกดเลือก Auto Import HEX ดังแสดงในรูปที่ 3.1.7 และ 3.1.8



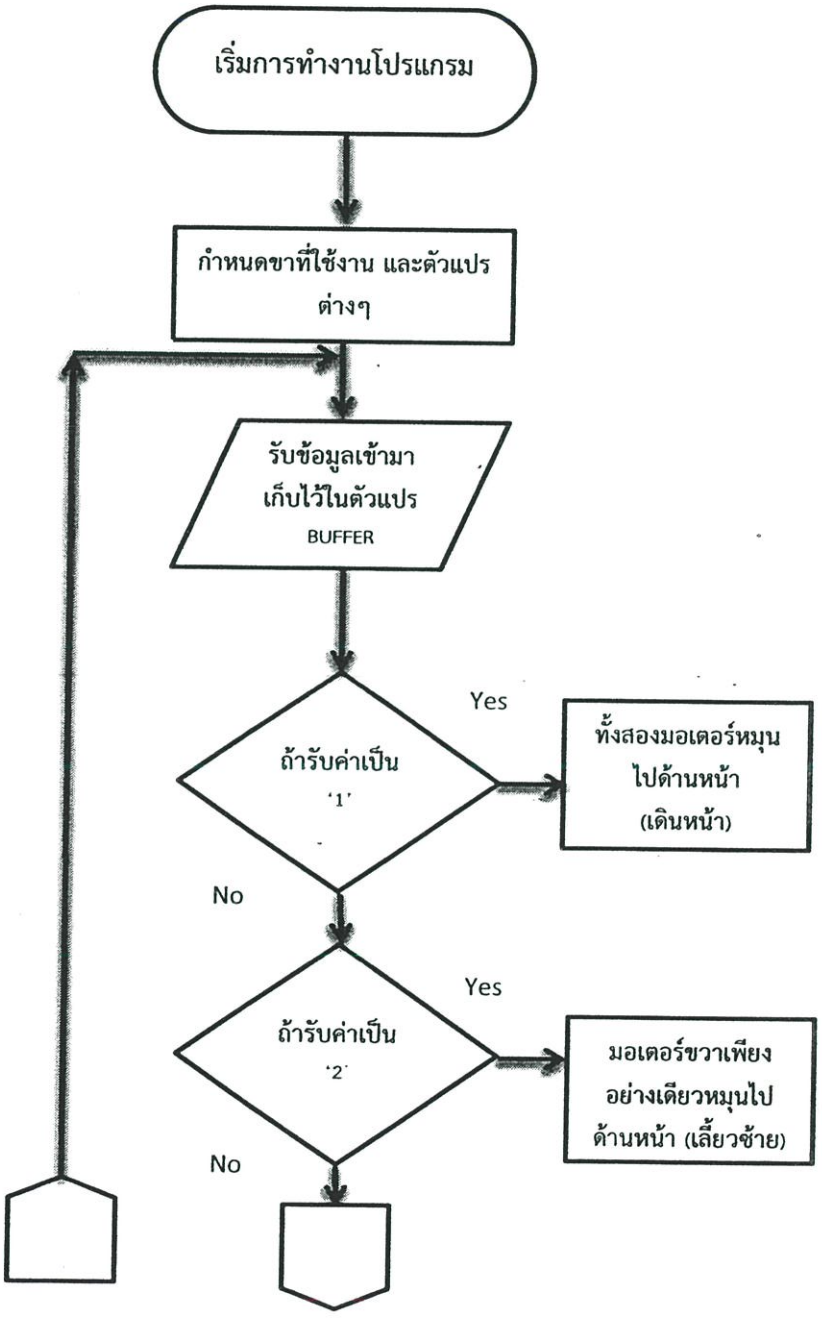
รูปที่ 3.1.9 ขั้นตอนการโปรแกรมลงบนไมโครคอนโทรลเลอร์ที่เสร็จสมบูรณ์แล้ว

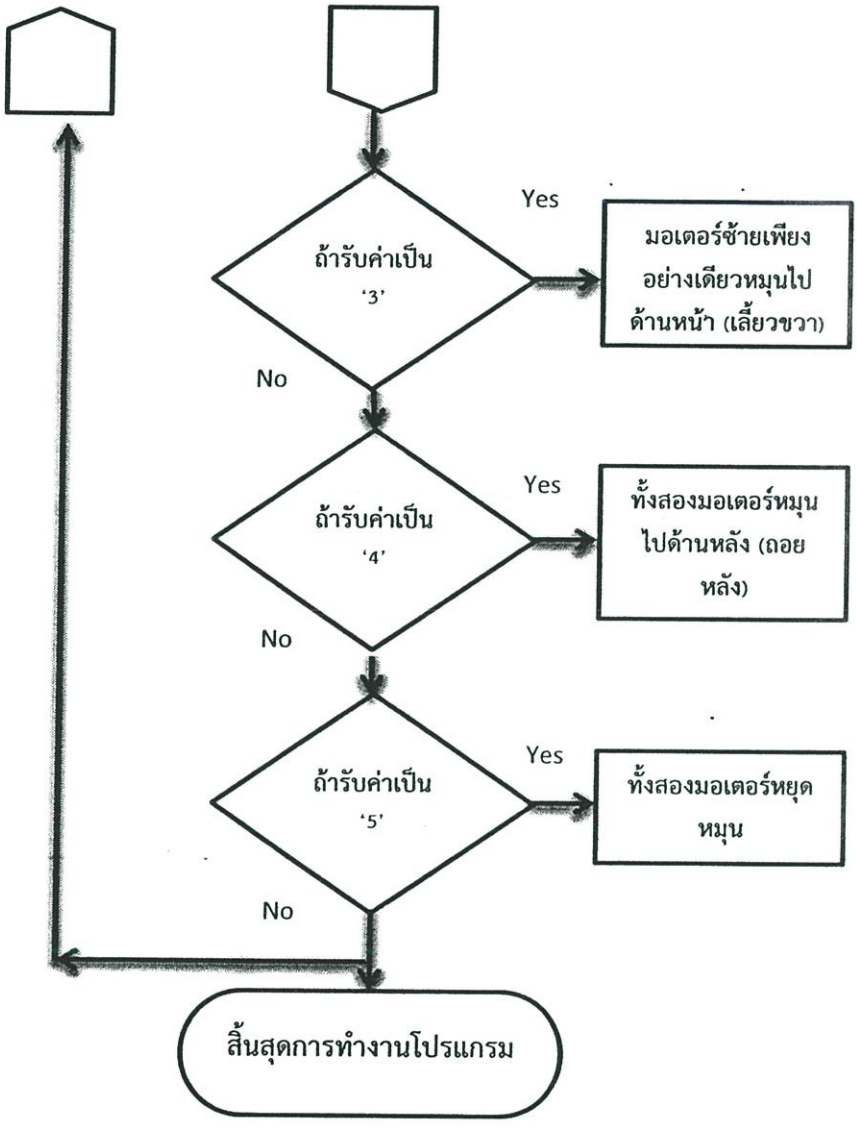
หลังจากที่เรียกไฟล์ .HEX ของโค้ดควบคุมการทำงานที่ได้สร้างไว้เรียบร้อยแล้ว โปรแกรม PICkit2 จะทำการโปรแกรมโค้ดลงไมโครคอนโทรลเลอร์โดยอัตโนมัติ ถ้าหากโปรแกรมโค้ดลงไปสำเร็จ ตัวหน้าต่างโปรแกรม PICkit2 จะแสดงข้อความว่า Programming Successful จากนั้นให้กดปุ่ม Auto Import HEX อีกครั้ง เป็นอันเสร็จสมบูรณ์

รูปที่ 3.1.10 การทดสอบการทำงานของโปรแกรมควบคุมบน PIC ไมโครคอนโทรลเลอร์

จากนั้นทำการต่อวงจร สร้างสัญญาณนาฬิกา ต่อหลอดไฟ LED (แทนการทำงานของมอเตอร์)และต่อช่องทางสื่อสารอนุกรมระหว่าง PIC ไมโครคอนโทรลเลอร์ กับ โมดูลบลูทูธ แล้วทำการสั่งค่ามาควบคุมการทำงานจากโทรศัพท์มือถือ ดังแสดงในรูปที่ 3.1.10

ขั้นตอนการทำงานของโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง สามารถแสดงได้ในโฟลว์ชาร์ทดังต่อไปนี้





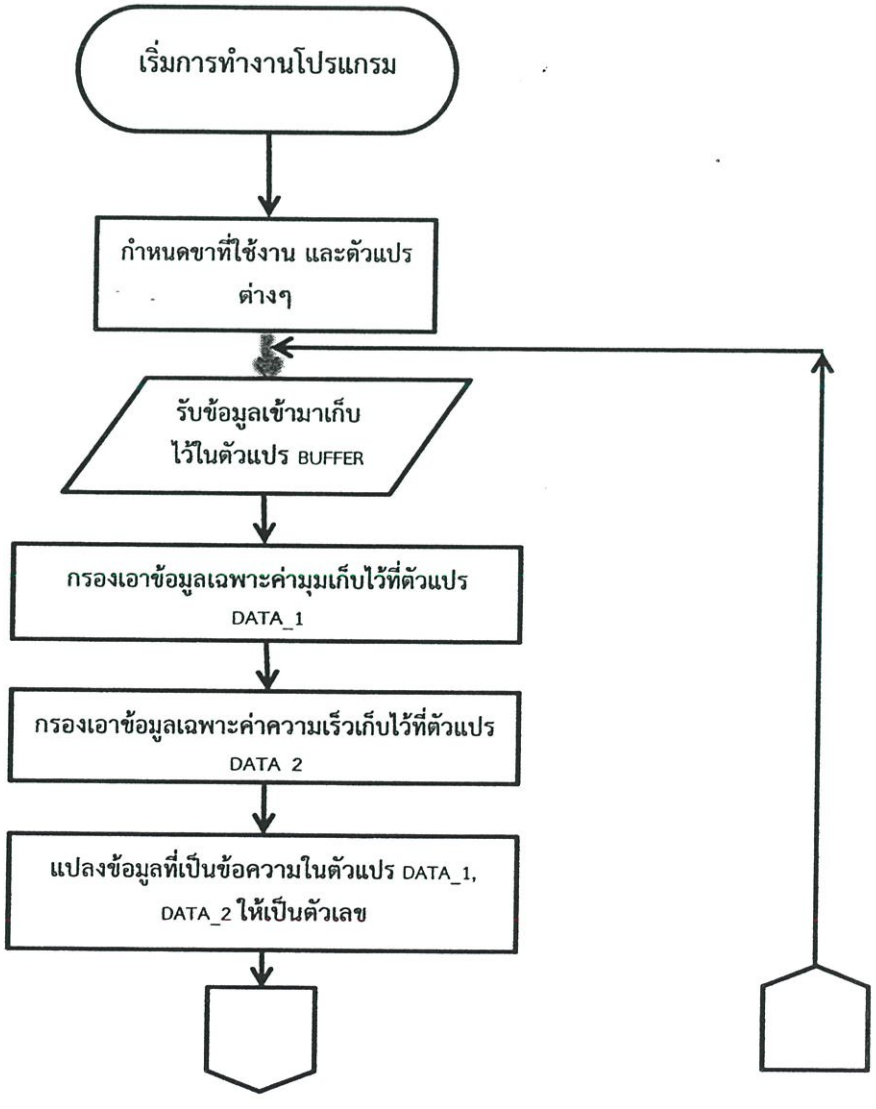
รูปที่ 3.1.11 โฟลว์ชาร์ทแสดงการทำงานของโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง

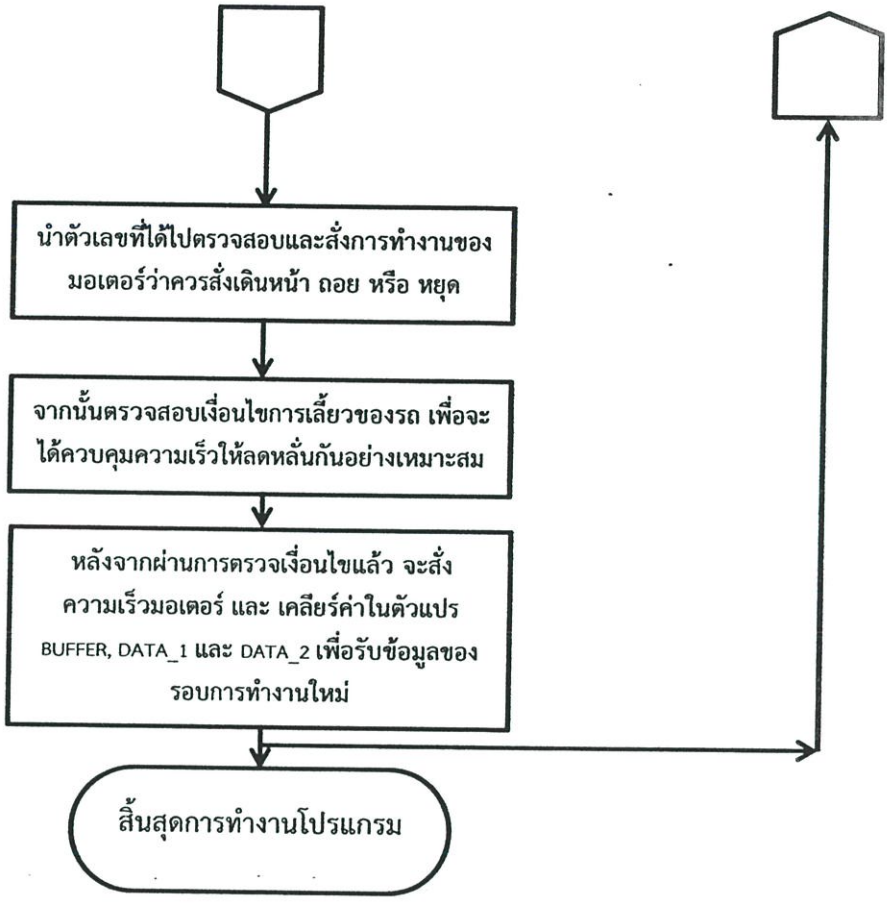
3.2.2.2 โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (Continuous Control)

เป็นโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ ที่สามารถควบคุมทิศทางการเคลื่อนที่ได้อย่างละเอียด และสามารถควบคุมความเร็วในการหมุนของมอเตอร์ได้ด้วย ซึ่งเป็นปัจจัยสำคัญที่สร้างการเลี้ยวของรถบังคับ ยิ่งเราสามารถควบคุมความเร็วได้ละเอียดมากเท่าใด เราก็สามารถสร้างการเคลื่อนที่ของรถบังคับได้ละเอียดมากขึ้นเท่านั้น

หลักการควบคุมความเร็วในที่นี้เราจะใช้ PWM (Pulse Width Modulation) ในการควบคุมมอเตอร์ ยิ่งเราให้ค่า PWM มากขึ้นเท่าใด มอเตอร์ก็มีกำลังในการขับเคลื่อนมากขึ้นเท่านั้น ซึ่งย่านของค่า PWM นั้นจะอยู่ที่ 0-255 (เป็นค่าอนาล็อก)

อย่างไรก็ตาม ขณะที่โปรแกรมนำค่าความเร็ว (Speed) มาตรวจสอบเพื่อสร้าง PWM นั้น โปรแกรมอีกส่วนหนึ่งก็นำค่ามุมองศา (Angle) มาตรวจสอบเพื่อกำหนดทิศทางการเคลื่อนที่ของรถเช่นเดียวกัน และจะสัมพันธ์กันกับค่าความเร็วด้วย ยกตัวอย่างเช่น หากค่ามุมองศาตรวจจับได้มีค่าเป็น 90 องศา เท่ากับว่ารถมีทิศทางเดินหน้าเต็มตัว เพราะฉะนั้นความเร็วของมอเตอร์ทั้งสองข้างก็จะขึ้นอยู่กับค่า Speed ที่ถูกส่งมา แต่ถ้าหากว่า ขณะนั้นมุมองศาถูกตรวจจับได้ว่ามีค่าเป็น 145 องศา เท่ากับว่ารถมีทิศทางเดินหน้าแบบเฉียงไปทางขวาด้วย เพราะฉะนั้นความเร็วของมอเตอร์ทั้งสองข้างย่อมแน่นอนว่าต้องไม่เท่ากันแน่นอน แม้ว่าจะได้รับค่าความเร็วมาเท่าใดก็ตาม ย่อมลดหลั่นกันไปตามมุมองศาที่ตรวจเจอ





รูปที่ 3.1.12 โฟลว์ชาร์ทแสดงการทำงานของโปรแกรมควบคุมการเคลื่อนที่ของรถบังคับแบบต่อเนื่อง

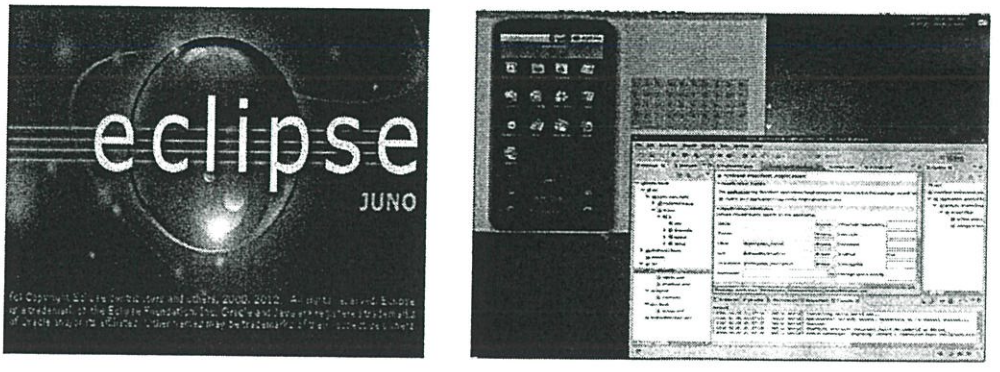
หลังจากที่เราทำการเขียนโปรแกรมควบคุมการทำงานของรถบังคับ และทำการทดสอบการทำงานต่างๆได้ตรงตามเงื่อนไขที่สั่งการมาแล้ว เราก็สามารถต่ออุปกรณ์ PIC ไมโครคอนโทรลเลอร์เข้ากับส่วนวงจรควบคุมการขับมอเตอร์ได้ในตอนต่อๆไป

3.2.3 ซอฟท์แวร์ส่วนหน้าต่างผู้ใช้งาน

ในส่วนหน้าต่างผู้ใช้งานของโครงงานนี้นั้นถูกออกแบบมาเพื่อใช้งานบนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ โดยโปรแกรมพัฒนาแอพพลิเคชั่นชื่อว่า “Eclipse” เวอร์ชั่น 4.2 “Juno” ซึ่งเป็นแพลทฟอร์มเวอร์ชั่นเก่า แต่ยังคงมีการรองรับระบบปฏิบัติการแอนดรอยด์รุ่นใหม่ๆอยู่

อย่างไรก็ตาม ในปัจจุบัน โปรแกรมพัฒนาแอพพลิเคชั่นของระบบปฏิบัติการแอนดรอยด์ส่วนมากนั้นจะถูกสนับสนุนให้ใช้โปรแกรม Android Studio มากกว่า ทั้งนี้ เพราะว่า Android Studio ถูกออกแบบมาให้ใช้งานง่ายกว่า โดยยังคงรูปแบบเดิมของโปรแกรม Eclipse เอาไว้ อีกทั้งโปรแกรม Eclipse นั้นได้สิ้นสุดการพัฒนาไปแล้ว

โปรแกรม Eclipse นั้นเป็นโปรแกรมพัฒนาแอพพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ที่มีผู้ใช้เป็นจำนวนมาก มีตัวอย่างให้ศึกษา และมีไลบรารี่เพื่อเรียกใช้ฟังก์ชั่นการทำงานเสริมต่างๆมากมาย ด้วยเหตุนี้เอง จึงทำให้โครงงานนี้เลือกใช้โปรแกรม Eclipse เป็นแพลทฟอร์มสำหรับการพัฒนาแอพพลิเคชั่นที่ใช้ควบคุมการเคลื่อนที่ของรถบังคับในหุ่นยนต์ทรงกลม



รูปที่ 3.2.1 รูปภาพแสดงโปรแกรม Eclipse Juno

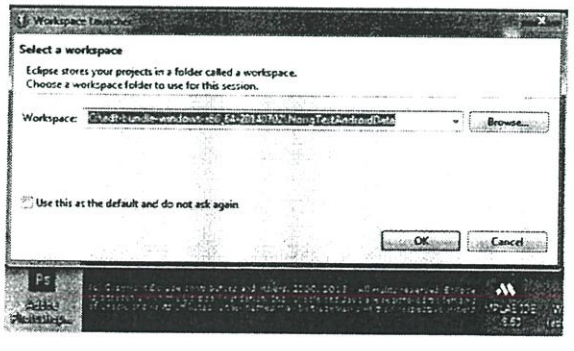
ในส่วนของแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับในหุ่นยนต์ทรงกลมนั้น สามารถแบ่งออกได้เป็น 2 แอพพลิเคชั่นด้วยกันคือ

1.แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง

2.แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง

3.2.3.1 แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control Sphere Robot Application)

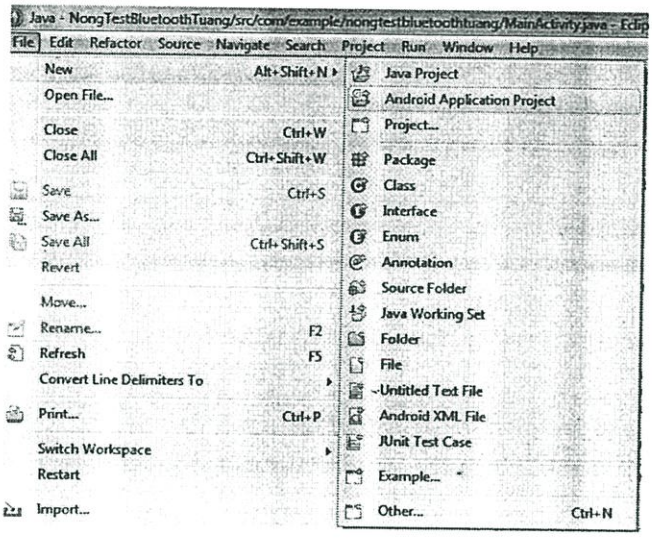
แอพพลิเคชั่นนี้ ถูกออกแบบมาเพื่อใช้งานร่วมกับโปรแกรมการควบคุมการเคลื่อนของรถบังคับบนไมโครคอนโทรลเลอร์แบบ ไม่ต่อเนื่อง ซึ่งก่อนที่จะเริ่มทำการเขียนโปรแกรมนั้น เราจะต้องทำการเริ่มสร้างโปรเจคบนโปรแกรม Eclipse เสียก่อน



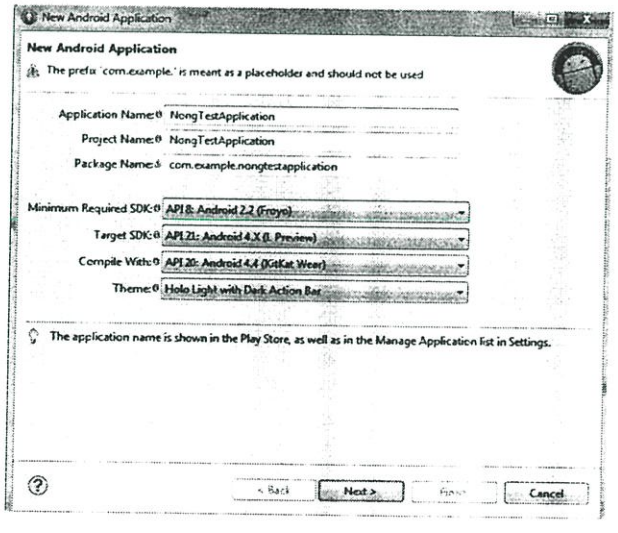
รูปที่ 3.2.2 รูปภาพแสดงขั้นตอนเลือก Workspace

จากรูปที่ 3.2.2 แสดงให้เห็นว่าก่อนที่เราจะทำการสร้างโปรเจคใดๆนั้น เราจำเป็นต้องเลือก Workspace หรือไดเรกทอรี่สำหรับเก็บโปรเจคที่เราสร้างก่อน จึงจะสามารถเข้าไปยังหน้าต่างสำหรับพัฒนาแอพพลิเคชั่นได้

จากนั้น ทำการสร้างโปรเจคใหม่ โดยเลือก Android Application Project แล้วทำการตั้งชื่อ แอพพลิเคชั่น ชื่อโปรเจค เลือกรุ่นของระบบปฏิบัติการแอนดรอยด์ขั้นต่ำสุดและสูงสุดที่ตัวแอพพลิเคชั่นสามารถรันได้ ตามรูปที่ 3.2.3 และ 3.2.4



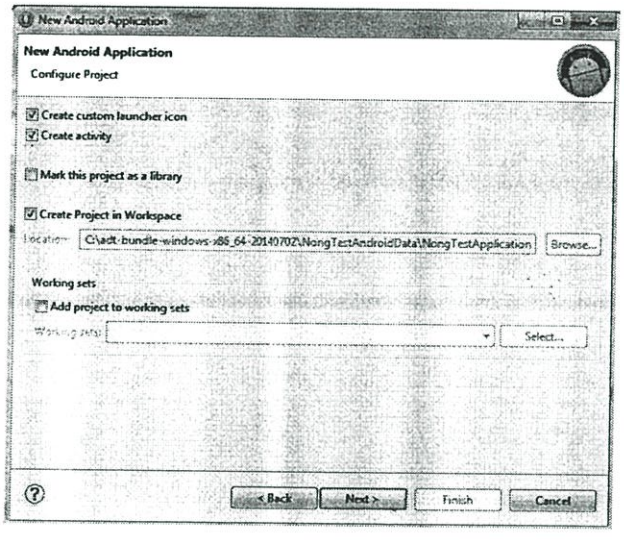
รูปที่ 3.2.3 รูปภาพแสดงขั้นตอนการสร้างโปรเจค



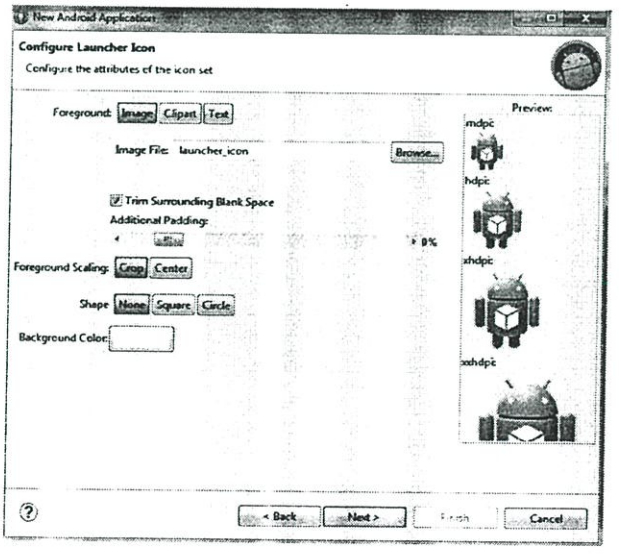
รูปที่ 3.2.4 รูปภาพแสดงขั้นตอนการตั้งชื่อและเลือกเวอร์ชั่นระบบปฏิบัติการ

เมื่อทำการตั้งชื่อและเลือกเวอร์ชั่นระบบปฏิบัติการที่ต้องการแล้ว ให้ทำการตั้งค่าโปรเจค เช่น ตั้งค่าว่าจะให้สร้างแอคทิวิตี้ขึ้นมาให้อัตโนมัติเลยหรือไม่ สร้างโปรเจคโดยใช้รูป

ไอคอนที่ปรับแต่งได้เองหรือไม่ หรือโปรเจคที่สร้างนั้นจะให้เก็บข้อมูลไว้ใน Workspace ที่ตั้งไว้หรือไม่ ซึ่งการตั้งค่าดังกล่าวนี้ เราไม่จำเป็นต้องปรับแก้ใดๆ เนื่องจากระบบตั้งค่าเหล่านี้มาเป็นดีฟอล์ทอยู่แล้ว จึงสามารถกดตกลงและไปยังหน้าเลือกภาพที่จะนำมาใช้เป็นไอคอนของแอพพลิเคชั่น เราสามารถปรับขนาด ปรับรูปทรงไอคอน พื้นหลังของไอคอน หรือแม้แต่เรียกรูปภาพอื่นๆ มาใช้เป็นไอคอนก็ได้ ดังแสดงในรูปที่ 3.2.5 และ 3.2.6



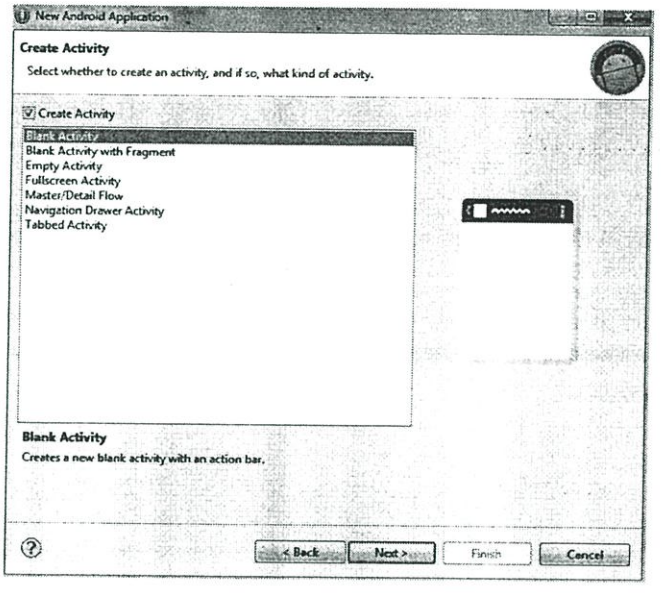
รูปที่ 3.2.5 รูปภาพแสดงขั้นตอนการตั้งค่าโปรเจค



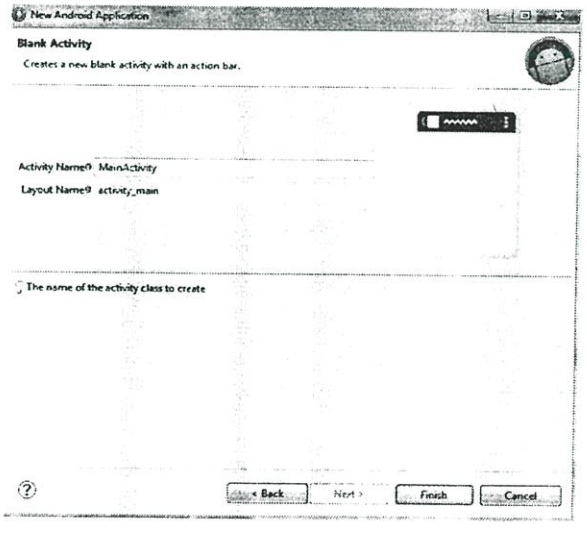
รูปที่ 3.2.6 รูปภาพแสดงขั้นตอนการเลือก และปรับแต่งภาพที่จะใช้เป็นไอคอนของแอพพลิเคชั่น

จากนั้นจะเรียกไปยังหน้าการเลือกรูปแบบ layout ของแอพพลิเคชั่น เช่น การสร้างหน้าเปล่า การสร้างหน้าต่างแบบมีแถบแสดงชื่อด้านข้าง การสร้างหน้าต่างแบบมีแถบแสดง

ชื่อด้านบน เป็นต้น โดยในที่นี้เราจะเลือกเป็น การสร้างหน้าต่างแบบมีแถบแสดงชื่อแอพพลิเคชั่นด้านบน ซึ่งเป็นค่าที่ถูกตั้งเป็นดีฟอล์ทไว้อยู่แล้ว เมื่อเลือกรูปแบบ layout เสร็จแล้ว จะต้องทำการตั้งชื่อไฟล์ .java และ .xml ที่ใช้เป็นไฟล์ควบคุมการทำงานของแอพพลิเคชั่น และควบคุมการออกแบบหน้าต่าง layout เป็นหลัก ดังรูปที่ 3.2.7 และ 3.2.8

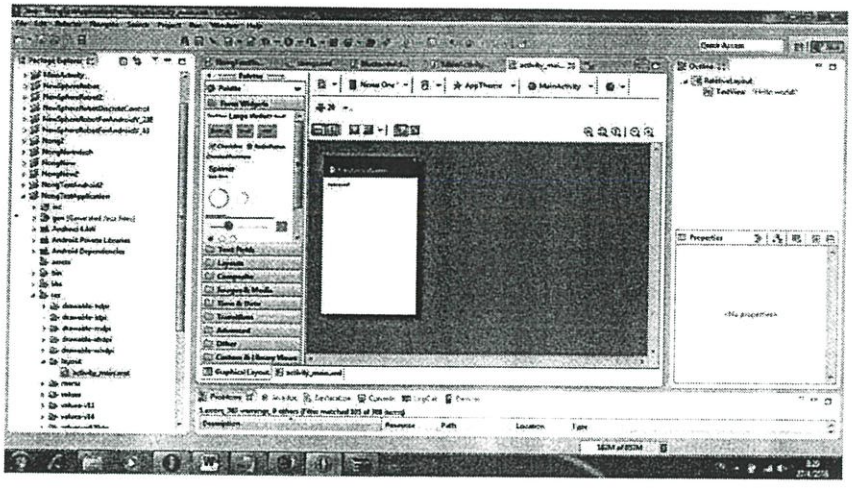


รูปที่ 3.2.7 รูปภาพแสดงขั้นตอนการเลือก รูปแบบหน้าต่าง layout



รูปที่ 3.2.8 รูปภาพแสดงขั้นตอนการตั้งชื่อไฟล์ .java ที่ใช้เป็นแอคทิวิตี้และไฟล์ .xml ที่เป็นหน้า layout ของแอพพลิเคชั่น

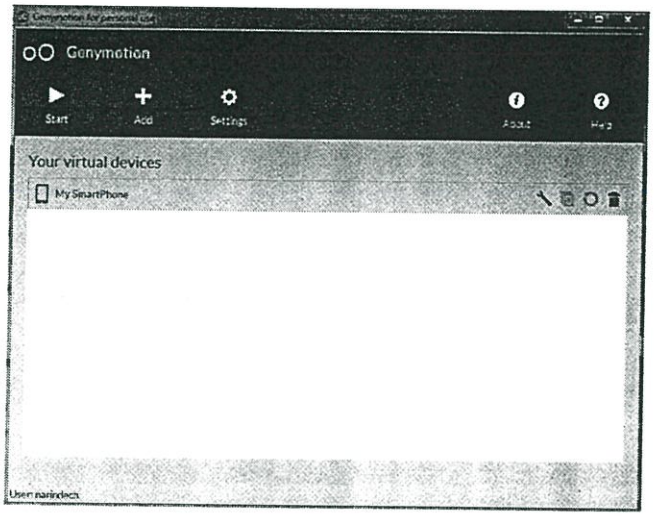
เมื่อทำการตั้งค่าต่างๆเสร็จเรียบร้อยแล้ว ถ้าหากไม่มีข้อผิดพลาดใดๆ ก็จะได้โปรเจคเปล่าสำหรับพัฒนาแอพพลิเคชั่นหน้าตาดังแสดงในรูปที่ 3.2.9



รูปที่ 3.2.9 รูปภาพแสดงหน้าต่างสำหรับพัฒนาแอพพลิเคชั่นบนโปรแกรม Eclipse

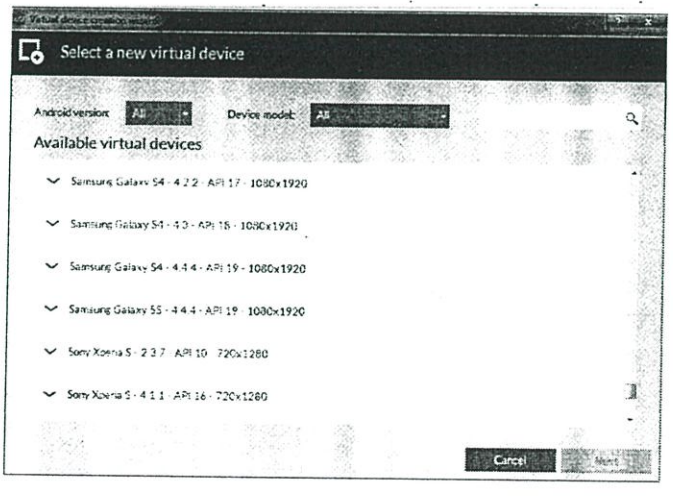
หลังจากสร้างโปรเจคเปล่าขึ้นมาเรียบร้อยแล้ว เราก็สามารถทำการเขียนแอพพลิเคชั่นขึ้นมาได้ โดยโปรแกรมส่วนควบคุมการทำงานของแอพพลิเคชั่นทั้งหมดนั้นจะอยู่ในส่วนไฟล์ MainActivity.java ส่วนโปรแกรมโครงสร้าง layout ของหน้าต่างผู้ใช้งานนั้นจะอยู่ในส่วนไฟล์ layout_activity.xml ส่วนการตั้งค่าการอนุญาตใช้ฟังก์ชั่นของโทรศัพท์มือถือต่างๆ จะอยู่ในไฟล์ AndroidManifest.xml และส่วนการอนุญาตการใช้งานไลบรารี่ต่างๆ จะอยู่ในส่วน project.properties

ถ้าหากเราต้องการทดสอบการทำงานของแอพพลิเคชั่นที่เราได้เขียนขึ้นนั้น เราจำเป็นต้องรันผ่านโทรศัพท์มือถือแอนดรอยด์จริงๆ หรืออาจจะใช้โปรแกรม อีมูเลเตอร์ จำลองโทรศัพท์มือถือแอนดรอยด์ขึ้นมาก็ได้ ซึ่งโปรแกรมอีมูเลเตอร์ที่ใช้ในโครงงานนี้ก็คือโปรแกรม Genymotion

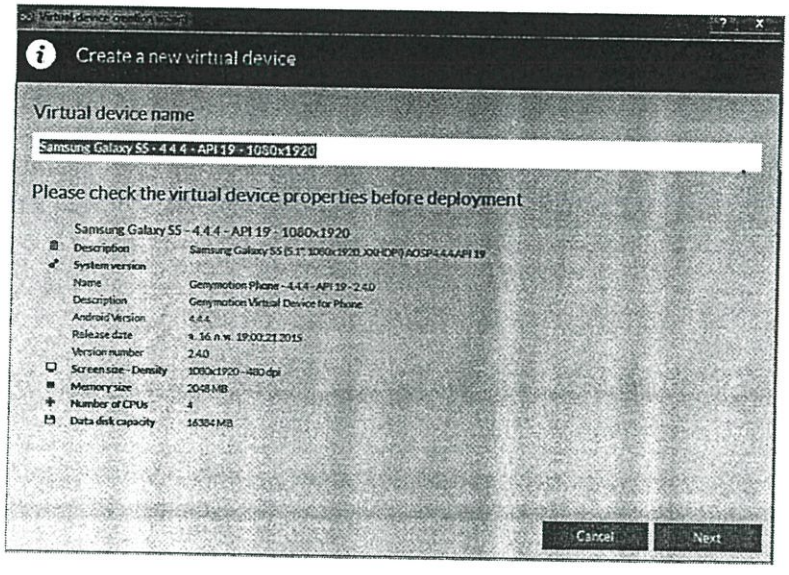


รูปที่ 3.2.10 รูปภาพแสดงหน้าต่างหลักโปรแกรมอีมูเลเตอร์ Genymotion

ก่อนที่เราจะทำการเปิดใช้งานโทรศัพท์มือถือจำลองนั้น เราจำเป็นต้องสร้างโทรศัพท์มือถือขึ้นมาก่อนโดย กดปุ่ม Add เพื่อเป็นการเพิ่มโทรศัพท์มือถือเครื่องใหม่เข้ามา จากนั้นเลือกรุ่น ยี่ห้อ และเวอร์ชั่นของระบบปฏิบัติการที่ต้องการ เมื่อเลือกเสร็จแล้วตัวโปรแกรมจะแสดงรายละเอียดของโทรศัพท์มือถือที่สร้างและให้เราตั้งชื่อโทรศัพท์พร้อมทั้งยืนยันการสร้างโทรศัพท์ด้วย ตามรูปที่ 3.2.11 และ 3.2.12

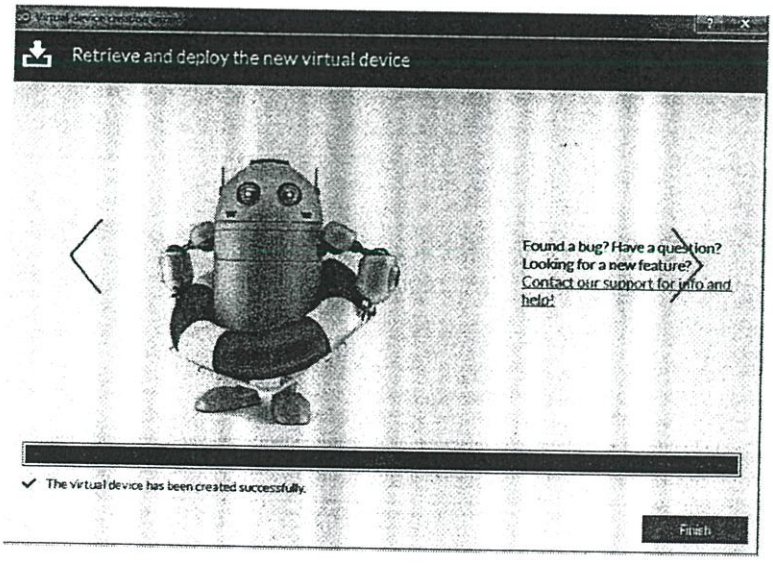


รูปที่ 3.2.11 รูปภาพแสดงหน้าต่างสร้างโทรศัพท์มือถือโปรแกรมอีมูเลเตอร์ Genymotion

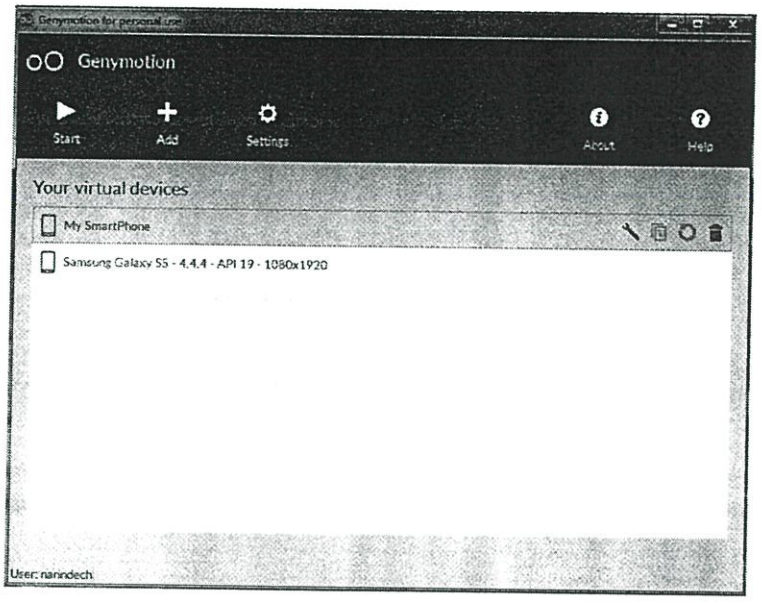


รูปที่ 3.2.12 รูปภาพแสดงรายละเอียดการสร้างโทรศัพท์มือถือโปรแกรมอีมูเลเตอร์ Genymotion

ในรายละเอียดจะบอกถึง เวอร์ชั่นของระบบปฏิบัติการ ขนาดพื้นที่หน่วยความจำที่กำหนด ขนาดของหน้าจอ เป็นต้น เมื่อตั้งชื่อและยืนยันการสร้างแล้ว โปรแกรมก็จะทำการสร้างมือถือขึ้นมา ซึ่งใช้ระยะเวลาพอสมควรนั้นการติดตั้ง หลังจากติดตั้งเสร็จเรียบร้อยแล้ว เราก็จะได้โทรศัพท์มือถือเครื่องใหม่ขึ้นมาปรากฏอยู่บนหน้าต่างหลักของโปรแกรม ดังรูปที่ 3.2.13 และ 3.2.14



รูปที่ 3.2.13 รูปภาพแสดงการสร้างโทรศัพท์มือถือเสร็จสมบูรณ์บนโปรแกรมอีมูเลเตอร์ Genymotion



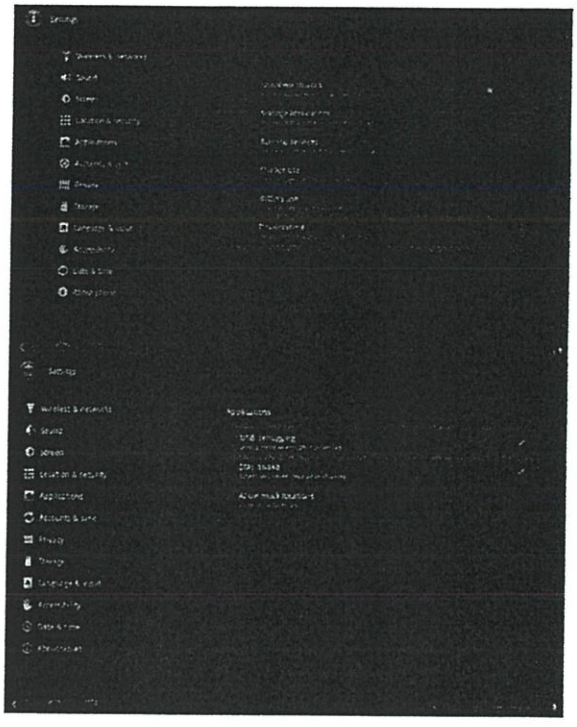
รูปที่ 3.2.14 รูปภาพแสดงการสร้างโทรศัพท์มือถือเสร็จสมบูรณ์บนโปรแกรมอีมูเลเตอร์ Genymotion

หลังจากนั้นทุกๆครั้งที่เราต้องการรันแอพพลิเคชั่นบนโทรศัพท์มือถือจำลองในโปรแกรม Genymotion นี้ ให้เราเลือกโทรศัพท์มือถือที่ต้องการ ในรายการโทรศัพท์มือถือที่มีการสร้างขึ้นมาในโปรแกรม จากนั้นกดปุ่มเริ่มการทำงาน (Start) ตัวโปรแกรมจะประมวลผลอยู่พักหนึ่ง จากนั้นโทรศัพท์มือถือจำลองก็จะปรากฏขึ้นบนหน้าจอคอมพิวเตอร์ ดังแสดงในรูปที่ 3.2.15

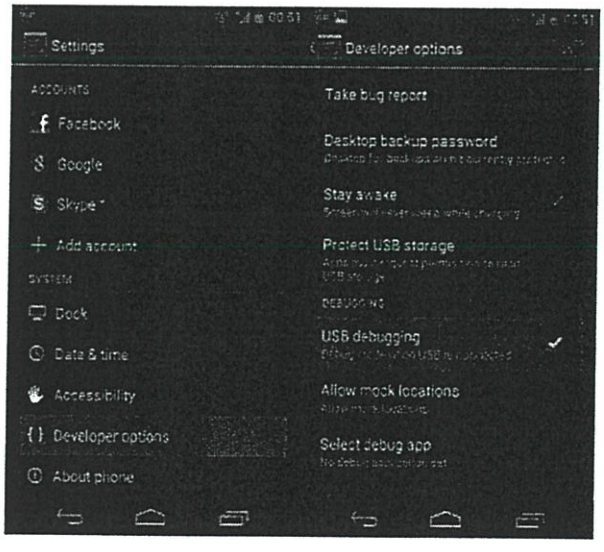
รูปที่ 3.2.15 รูปภาพแสดงการรันโทรศัพท์มือถือจำลองบนโปรแกรมอีมูเลเตอร์ Genymotion

อย่างไรก็ตาม ถ้าหากต้องการที่จะทดสอบแอพพลิเคชั่นกับโทรศัพท์มือถือจริง ให้เราทำการเปิดการใช้งาน USB debugging บนโทรศัพท์มือถือ เนื่องจากการติดตั้งแอพพลิเคชั่นบนโทรศัพท์มือถือนั้นทำได้โดย การส่งข้อมูลแอพพลิเคชั่นผ่านสาย USB ซึ่งขั้นตอนการเปิดการใช้งาน USB debugging นั้นมีหลายรูปแบบด้วยกัน ขึ้นอยู่กับเวอร์ชั่นของแต่ละระบบปฏิบัติการแอนดรอยด์ แสดงในรูปที่ 3.2.16, 3.2.17 และ 3.2.18

รูปที่ 3.2.16 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์เวอร์ชั่น 2.x และ 3.x

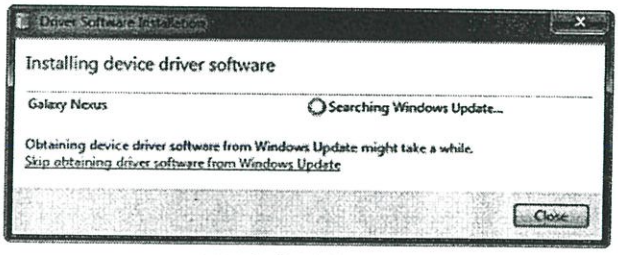


รูปที่ 3.2.17 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์เวอร์ชั่น 2.x และ 3.x



รูปที่ 3.2.18 รูปภาพแสดงการเปิดการใช้งาน USB debugging ของโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์เวอร์ชั่น 4.x

จากนั้นเสียบโทรศัพท์มือถือเข้ากับคอมพิวเตอร์ด้วยสาย USB ตัวคอมพิวเตอร์จะทำการติดตั้งไดร์ฟเวอร์ของโทรศัพท์มือถือนั้นๆ ดังแสดงในรูปที่ 3.2.19

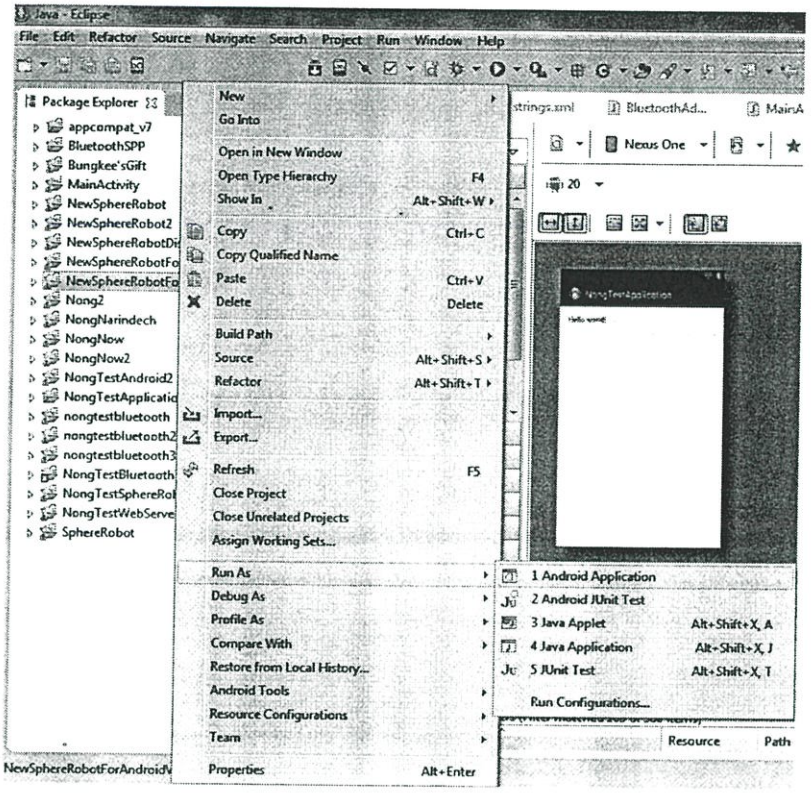


รูปที่ 3.2.19 รูปภาพแสดงการติดตั้งไดร์ฟเวอร์ของโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์บนคอมพิวเตอร์

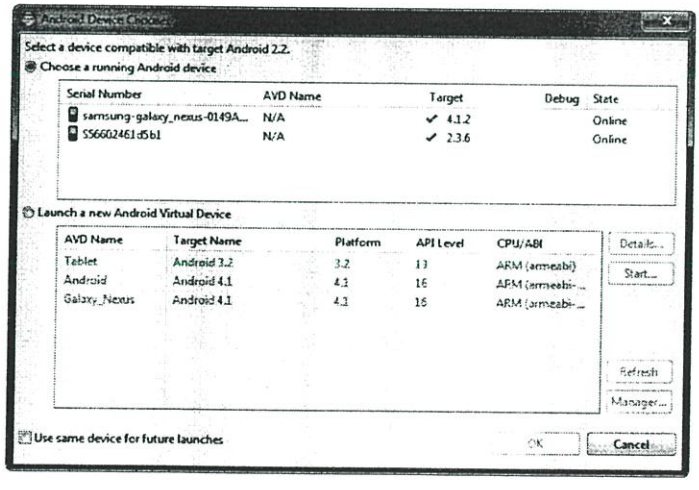
สำหรับโทรศัพท์มือถือรุ่นใหม่ บางเครื่องจะมีระบบป้องกันเผื่อว่าผู้ใช้งานบังเอิญเปิดการใช้งาน USB debugging อย่างไม่ได้ตั้งใจ ซึ่งจะแสดงข้อความคำขออนุญาตขึ้นมา ให้เราอนุญาตเปิดการใช้งานได้เลย ตามรูปที่ 3.2.20

รูปที่ 3.2.20 รูปภาพแสดงคำขออนุญาตเปิดใช้งาน USB debugging บนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

เมื่อการติดตั้งเสร็จสิ้น เราก็จะสามารถติดตั้งแอพพลิเคชั่นผ่านโทรศัพท์มือถือจริงได้ ซึ่งในส่วนของการทดสอบการทำงานของแอพพลิเคชั่น ไม่ว่าเราจะใช้โทรศัพท์จำลองหรือโทรศัพท์จริงก็ตาม ให้เราไปที่ตัวโปรแกรม Eclipse คลิกขวาที่โปรเจคที่เราต้องการจะทดสอบ จากนั้นเลือก Run as Android Application หากมีโทรศัพท์มือถือจำลอง หรือโทรศัพท์มือถือจริง พร้อมใช้งานอยู่ ตัวโปรแกรมจะแสดงโทรศัพท์มือถือเหล่านั้นขึ้นมา ให้เรากดเลือกโทรศัพท์มือถือที่ต้องการ ถ้าการติดตั้งเสร็จสมบูรณ์ และแอพพลิเคชั่นที่เราสร้างไม่มีข้อบกพร่องใดๆ โทรศัพท์มือถือก็จะรันแอพพลิเคชั่นที่เราสร้างได้อย่างเสร็จสมบูรณ์

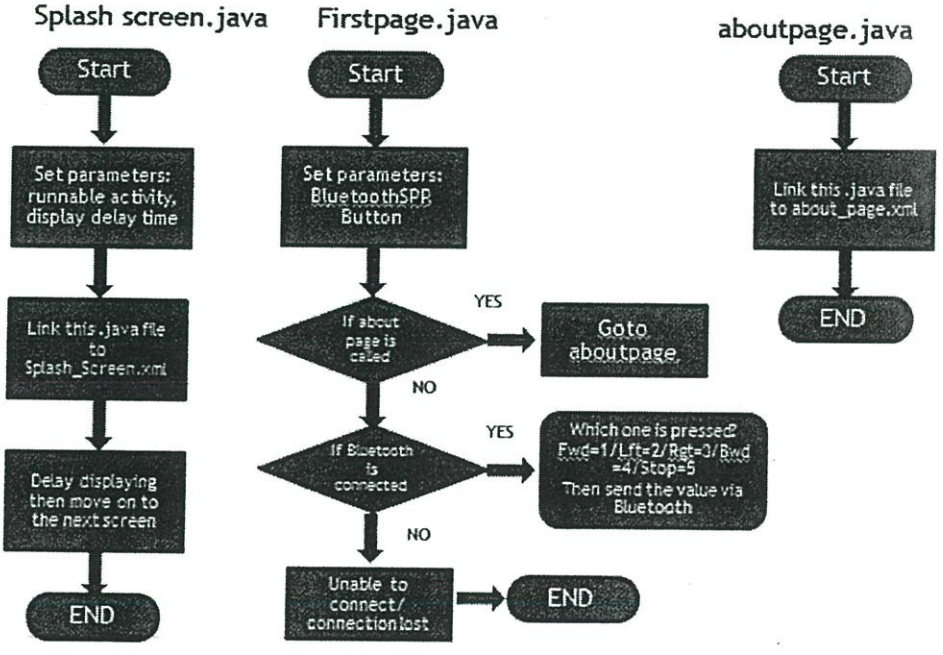


รูปที่ 3.2.21 รูปภาพแสดงการเริ่มต้นการรันแอพพลิเคชั่นจากโปรแกรม Eclipse



รูปที่ 3.2.22 รูปภาพแสดงการเลือกโทรศัพท์มือถือก่อนเริ่มการรันแอพพลิเคชั่นจากโปรแกรม Eclipse

สำหรับขั้นตอนการทำงานของแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่องนั้น จะขออธิบายเป็นการทำงานของแต่ละ Activity หรือแต่ละหน้าของแอพพลิเคชั่นไป



รูปที่ 3.2.23 รูปภาพแสดงโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง

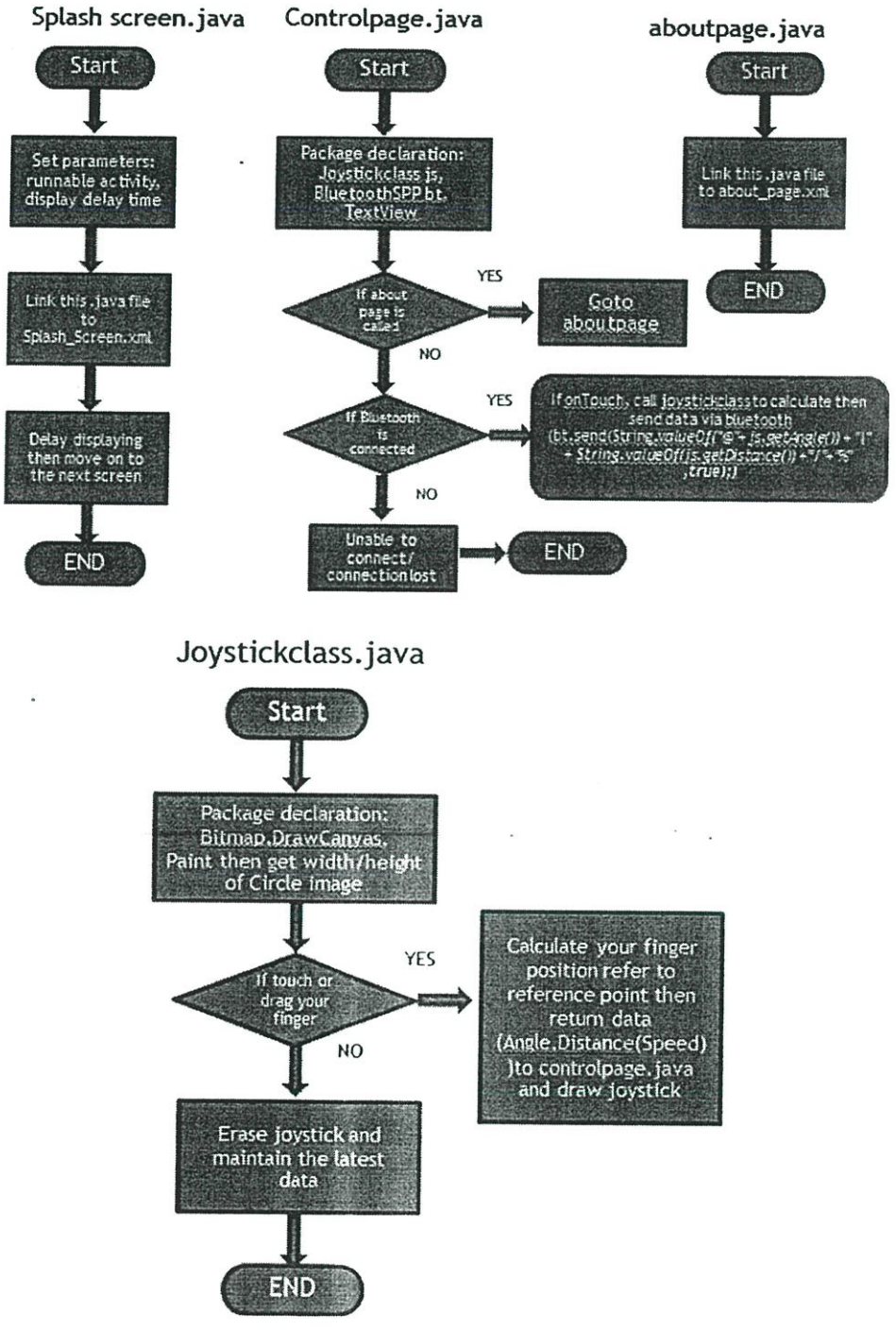
จากโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสามารถอธิบายได้ว่า การทำงานเริ่มต้นจาก Activity ชื่อ Splash screen ทำหน้าที่แสดงหน้าแรกของแอพพลิเคชั่น

จากนั้น Activity ชื่อ Firstpage จะทำงานต่อ ซึ่งทำหน้าที่เปิดการใช้งานเทคโนโลยีบลูทูธบนโทรศัพท์มือถือ ทำหน้าที่ตรวจสอบเงื่อนไขการกดปุ่มสั่งการ เป็นต้น

และสุดท้าย Activity ชื่อ aboutpage จะทำงานก็ต่อเมื่อมีการกดปุ่ม about page จากหน้า layout ชื่อ Firstpage ซึ่งถ้าหากมีการทำงานแล้ว ก็จะแสดงผลข้อมูลเกี่ยวกับแอพพลิเคชั่นต่างๆโดยสังเขป

3.2.3.2 แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (Continuous Control Sphere Robot Application)

แอพพลิเคชั่นนี้ ถูกออกแบบมาเพื่อใช้งานร่วมกับโปรแกรมการควบคุมการเคลื่อนของรถบังคับบนไมโครคอนโทรลเลอร์แบบ ต่อเนื่อง ในส่วนของการสร้างโปรเจคบนโปรแกรม Eclipse และการรันทดสอบแอพพลิเคชั่นที่ได้เขียนขึ้นนั้นจะเหมือนกันทุกประการ กับหัวข้อก่อนหน้านี้ ดังนั้น รูปที่ 3.2.24 จะอธิบายโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง



รูปที่ 3.2.24 รูปภาพแสดงโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง

จากโฟลว์ชาร์ตการทำงานของแอพพลิเคชั่นสามารถอธิบายได้ว่า การทำงานเริ่มต้นจาก Activity ชื่อ Splash screen ทำหน้าที่แสดงหน้าแรกของแอพพลิเคชั่น

จากนั้น Activity ชื่อ Controlpage จะทำงานต่อ ซึ่งทำหน้าที่เปิดการใช้งานเทคโนโลยีบลูทูธบนโทรศัพท์มือถือ เรียกใช้งาน คลาสของ Activity ที่ชื่อว่า Joystickclass เข้า

มาใช้งานร่วม ทำหน้าที่ตรวจสอบเงื่อนไขการกดปุ่มสั่งการ เช่น ถ้าหากผู้ใช้สัมผัสบนหน้าจอส่วนที่เป็นจอยสติกสำหรับควบคุมการเคลื่อนที่ จะให้มีการส่งค่ามุมและค่าความเร็วผ่านบลูทูธออกไป เป็นต้น ซึ่งค่ามุมและความเร็วนั้นสัมพันธ์กับตำแหน่งที่นิ้วมือของผู้ใช้สัมผัสลงบนจอยสติกด้วย

ในส่วน Activity ชื่อ aboutpage จะทำงานก็ต่อเมื่อมีการกดปุ่ม about page จากหน้า layout ชื่อ Firstpage ซึ่งถ้าหากมีการทำงานแล้ว ก็จะแสดงผลข้อมูลเกี่ยวกับแอพพลิเคชั่นต่างๆโดยสังเขป

และสุดท้าย Activity หรือ คลาส ชื่อ Joystickclass นี้เป็นคลาสส่วนที่ทำหน้าที่ประมวลผลทุกอย่างที่เกี่ยวข้องกับการสัมผัสจอยสติกของผู้ใช้งาน และส่งผลให้เกิดค่ามุมและค่าความเร็ว รีเทิร์นค่ากลับไปยัง Activity ชื่อ Controlpage ได้ เพื่อที่ Activity นั้นจะได้ส่งค่าออกผ่านฟังก์ชั่นการใช้งานบลูทูธของโทรศัพท์มือถือได้นั่นเอง

3.3 วงจรไฟฟ้าควบคุมการทำงานของหุ่นยนต์

ในส่วนนี้คือวงจรสำหรับควบคุมการทำงานของหุ่นยนต์ได้แบ่งออกเป็น 2 ส่วนคือ

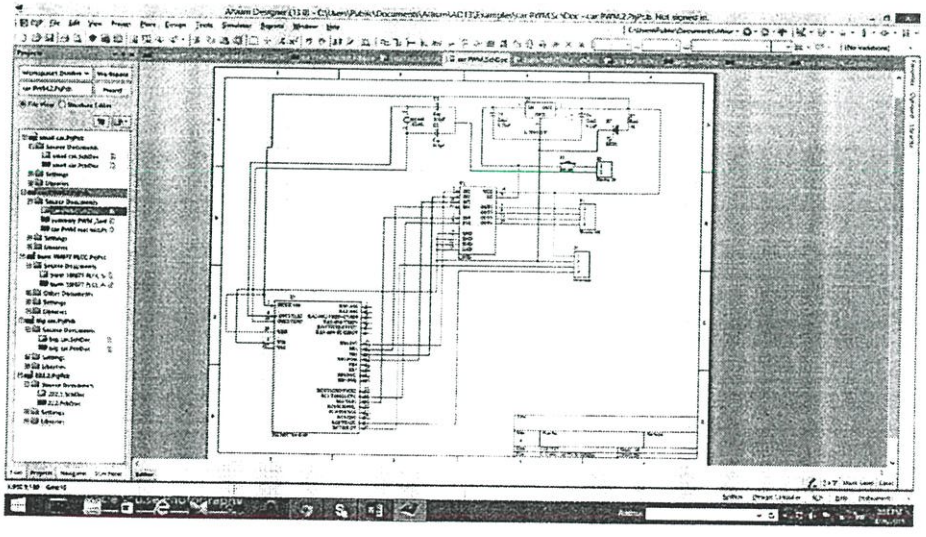
3.3.1.ขั้นตอนการทำวงจร

3.3.2.ส่วนของวงจรควบคุมความเร็วของมอเตอร์

3.3.1. ขั้นตอนการทำวงจร

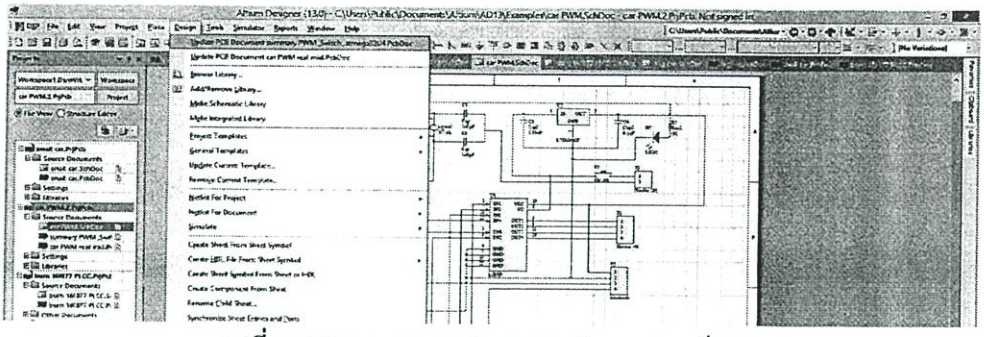
เมื่อเราได้ลองต่อวงจร sphere robot มาแล้ว ก็ทำการการปริ้นแผ่นวงจรหรือที่เรียกว่า PCB จะมีขั้นตอนดังนี้

1. วาด schematic ลงบนโปรแกรม altium จะได้ดังนี้ (วงจร small sphere robot)



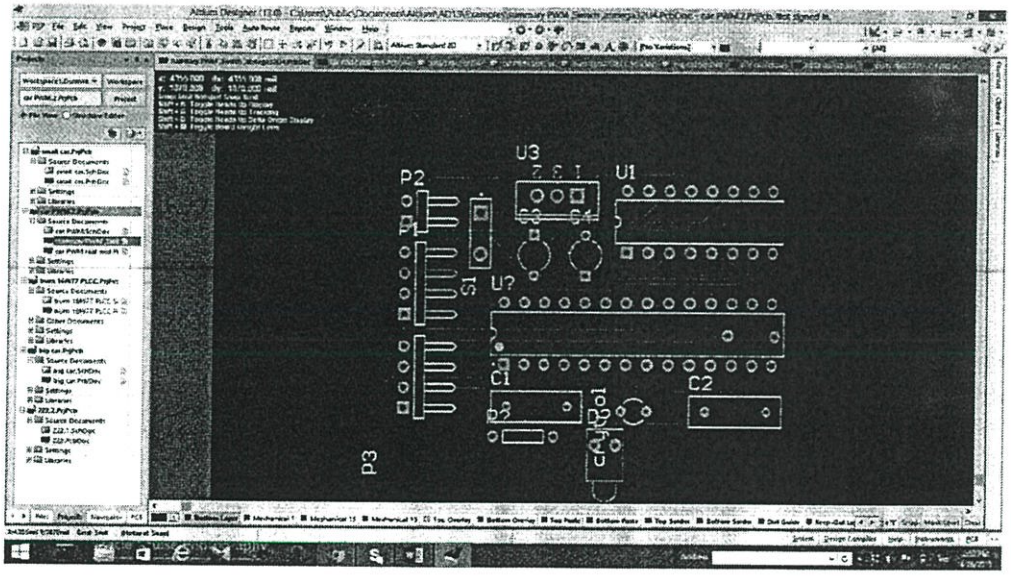
รูปที่ 3.3.1 แสดงการวาด schematic ในโปรแกรม altium

2. จากนั้นก็ไปที่ design แล้วเลือก update PCB document **(ชื่อไฟล์ที่** save สำหรับเก็บไฟล์ PCB ไว้)



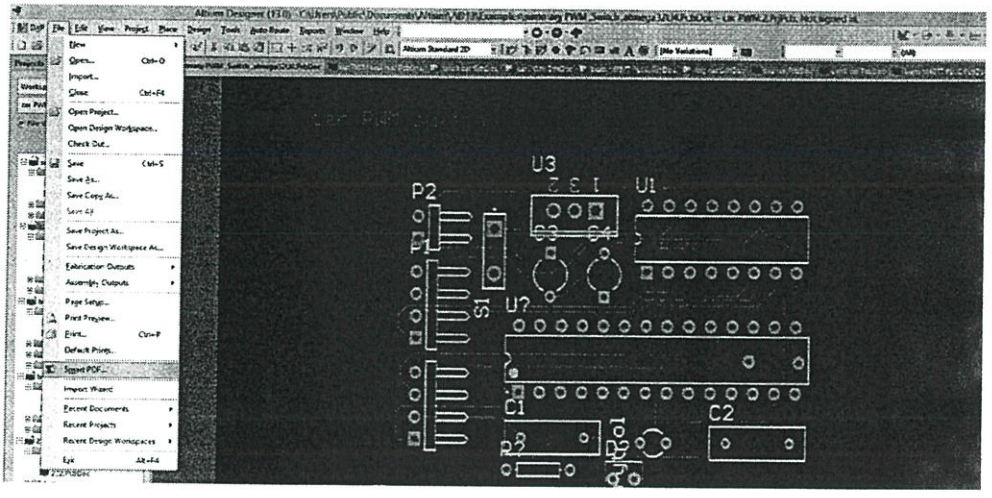
รูปที่ 3.3.2 แสดงการแปลงจาก schematic เป็น PCB

3. จากนั้นก็จัดเรียงอุปกรณ์ต่างๆ และวาดลายวงจรตามที่เราต้องการ



รูปที่ 3.3.3 ภาพแสดงลาย PCB ที่ได้ทำการออกแบบไว้ ซึ่งเป็น PCB กัดปริ้น 2 ด้าน

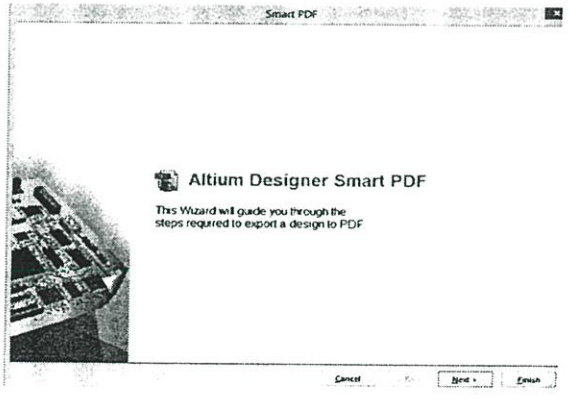
4. ทำการปริ้นลาย PCB เป็นไฟล์ PDF เพื่อทำการกัดปริ้นต่อไป โดนมีขั้นตอนดังนี้ คือไป ที่ menu File ⟶ Smart PDF



รูปที่ 3.3.4 ภาพแสดงการเปลี่ยนไฟล์ลายวงจรเป็นไฟล์ PDF เพื่อทำการกัดปริ้นต่อไป

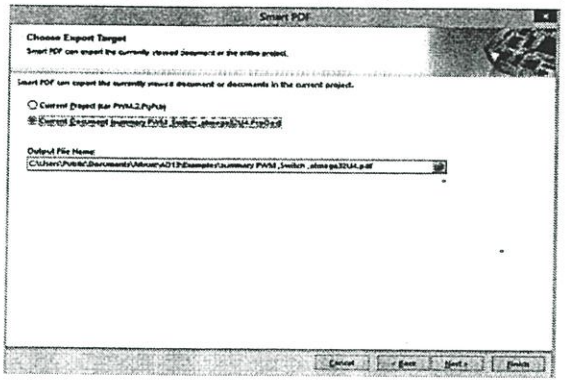
5. ขั้นตอนจากการแปลง PCB จาก altrium เป็น PDF เพื่อทำการกัดปริ้น

5.1 กด NEXT



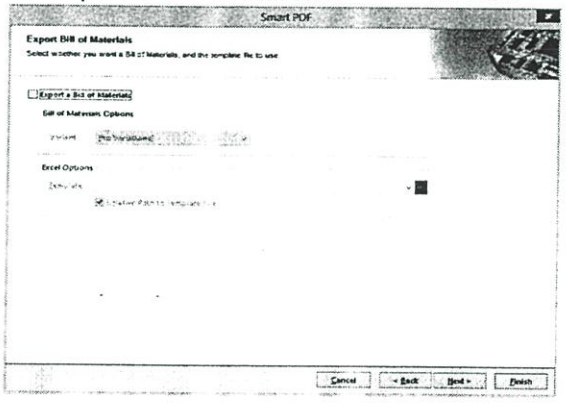
รูปที่ 3.3.5 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF

5.2 เลือก current document ⟶ กด next ⟶ กด OK



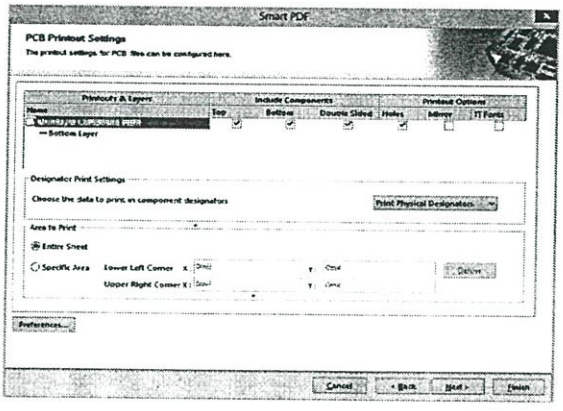
รูปที่ 3.3.6 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(1)

5.3 คลิ๊กออกจาก Export a bill of material ⟶ กด next



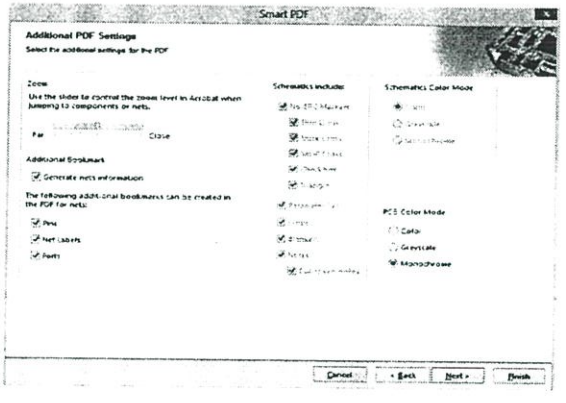
รูปที่ 3.3.7 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(2)

5.4 เลือก layer ที่จะปริ้นออกมา โดยหากเป็นการกัดปริ้น 2 layer ก็ต้องปริ้นทีละ layer ซึ่งถ้าเป็น bottom layer ไม่ต้องเลือกที่ ช่อง mirror แต่หากเป็นการปริ้น Top layer ก็ต้องเลือกแบบ mirror



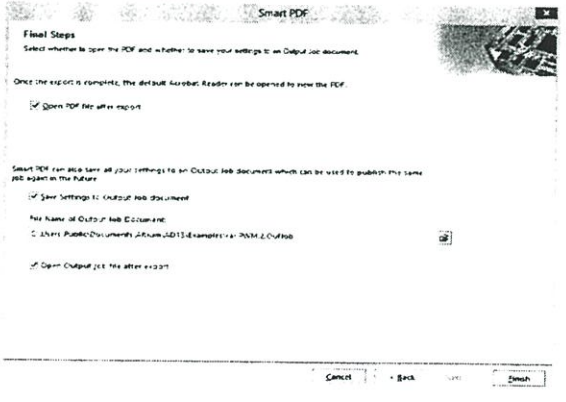
รูปที่ 3.3.8 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(3)

5.5 เลือก monochrome ⟶ next



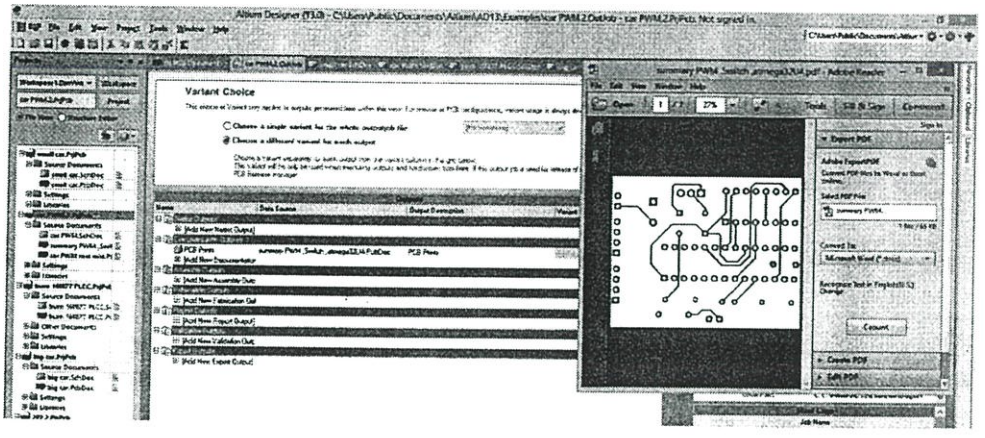
รูปที่ 3.3.9 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(4)

5.6 เลือก next ⟶ Finish



รูปที่ 3.3.10 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(5)

5.7 จะได้ลายวงจรออกมาในโปรแกรม PDF ⟶ ให้ไปคลิ๊กขวงและเลือกที่ page setup

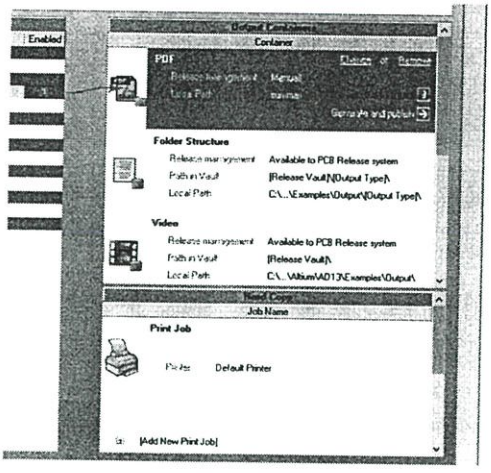


รูปที่ 3.3.11 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(6)

5.8 ตรงช่อง scale mode ให้เลือกที่ scaled print⟶ และตั้ง scale เป็น 1 ทุกช่อง สุดท้ายเลือกcolor set เป็น mono ⟶ กด close

รูปที่ 3.3.12 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(7)

5.9 คลิ๊กตรง generate content เพื่อเปิด PCB เป็นไฟล์ PDF ขึ้นมาแล้วนำไปทำการกัดปริ้นตามขั้นตอนต่อไปได้เลย จะทำการ save แบบนี้ทั้ง 2 top layer และ bottom layer

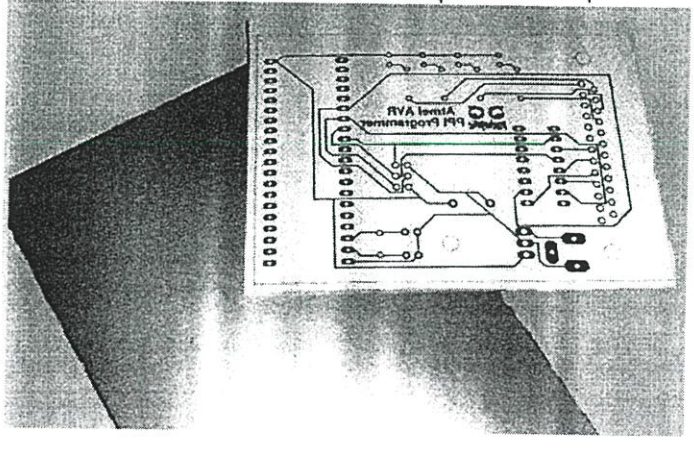


รูปที่3.3.13 ภาพแสดงขั้นตอนการแปลงไฟล์ PCB เป็น PDF(8)

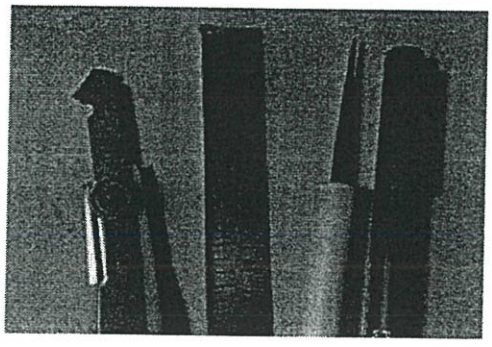
6. ขั้นตอนการกัดปริ้น หรือ ทำ PCB

6.1 อุปกณ์ที่ใช้ในการทำ PCB

- กระดาษโฟโต้ที่มีลายวงจร
- แผ่นทองแดง หากเป็นการกัดปริ้น 2 หน้าให้ใช้แผ่นทองแดงที่มีทองแดงทั้ง 2 ด้าน
- คัทเตอร์ ไว้ตัด PCB ขูด
- ไม่บรรทัดโลหะ (คัทเตอร์จะได้ไม่กัดไม้บรรทัดแหว่ง)
- ปากกา Permanente หรือ ปากกาเขียน CD
- เหล็กแหลม ไว้ตกแต่งลายปริ้น หรือวัสดุอะไรที่แหลมๆ ก็ได้

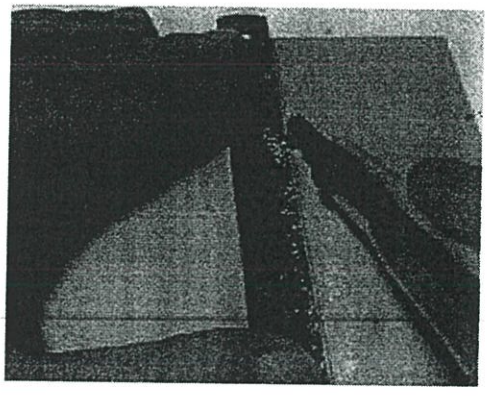


รูปที่ 3.3.14 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB

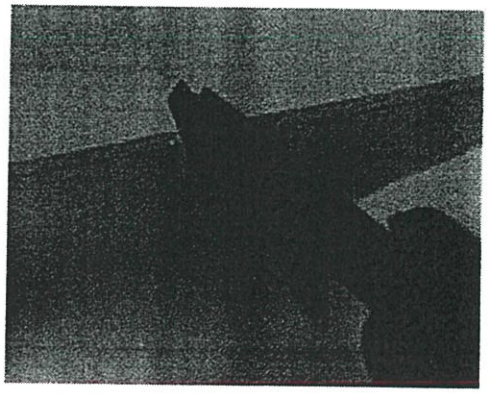


รูปที่ 3.3.15 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(1)

6.2 วัดขนาด PCB ที่ต้องการตัด แล้วใช้ปากกาขีดเส้นตามขนาดลงบนแผ่น PCB

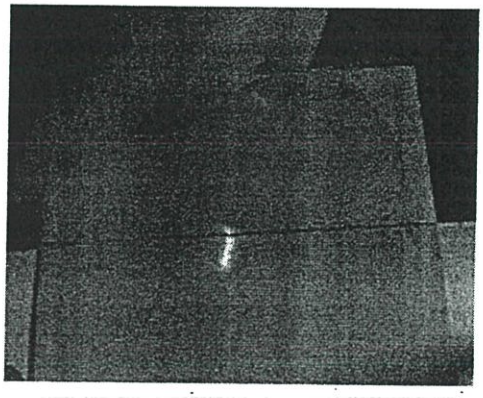


รูปที่ 3.3.16 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(2)

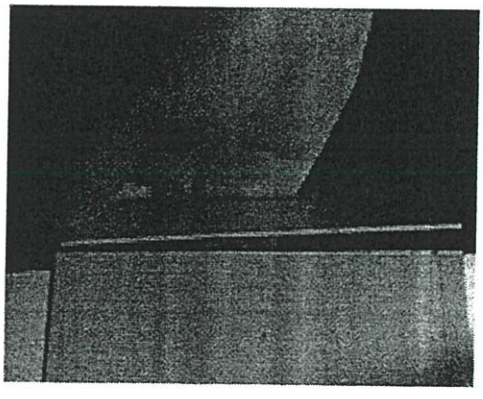


รูปที่ 3.3.17 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(3)

รูปที่ 3.3.18 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(4)

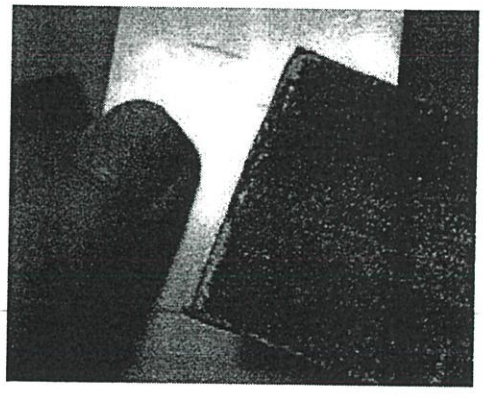


รูปที่ 3.3.19 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(5)



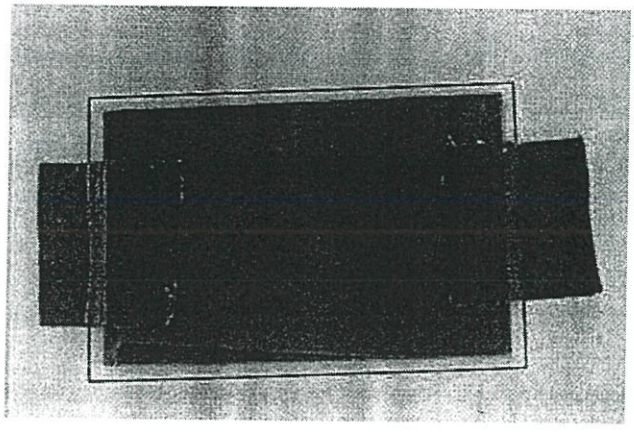
รูปที่ 3.3.20 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(6)

รูปที่ 3.3.21 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(7)

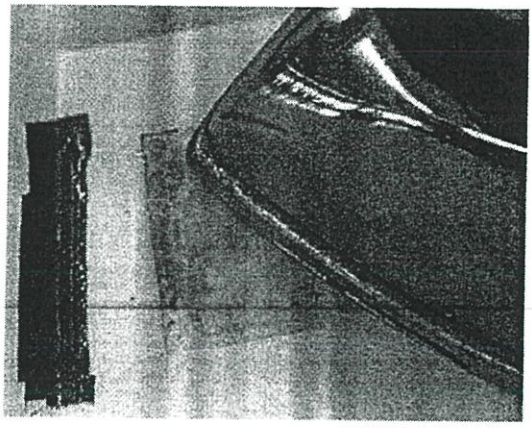


รูปที่ 3.3.22 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(8)

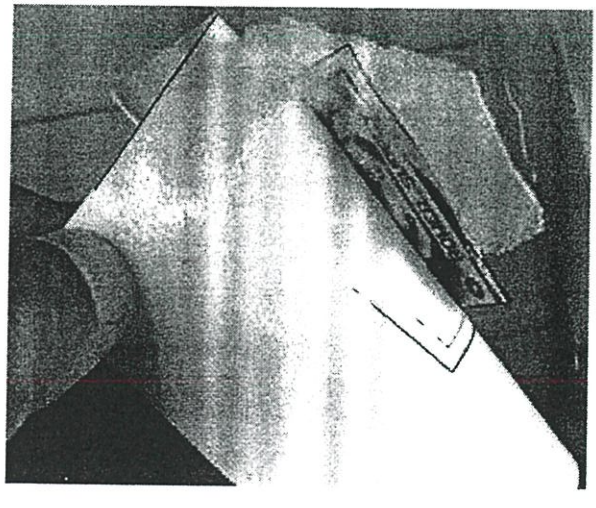
ล้างมือให้สะอาด แล้วเช็ดให้แห้ง พยายามอย่าให้มือถูกแผ่นทองแดงอีกเพราะจะทำให้ กรดกัดไม่ออก เมื่อมีน้ำมันจากมือ ไปถูกแผ่นทองแดง ปกติแล้ว กรดที่ใช้สามารถใช้ได้หลายครั้ง แต่ถ้าแผ่น PCB ไม่สะอาดหรือมีน้ำมันจากมือไปติด นานๆไปกรดจะเป็นฝ้า หากฝ้ามาติด แผ่นทองแดง จะทำให้กรดไม่กัดบริเวณนั้น จะต้องใช้ผ้ากรองน้ำกรดให้ฝ้าเหล่านั้นหมดไปก่อน



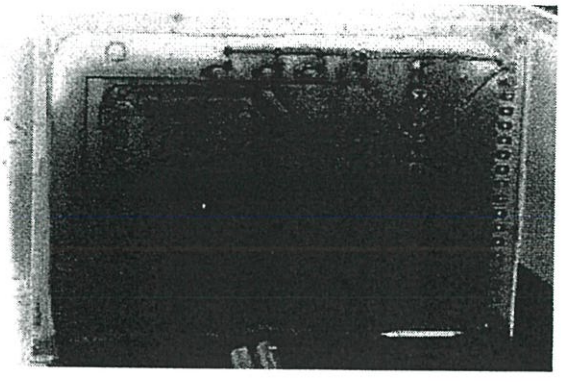
รูปที่ 3.3.23 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(9)



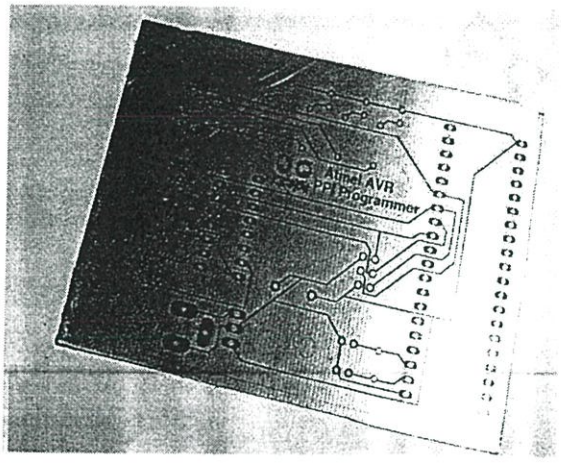
รูปที่ 3.3.24 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(10)



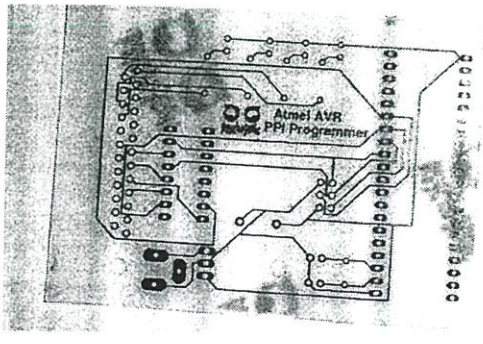
รูปที่ 3.3.25 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(11)



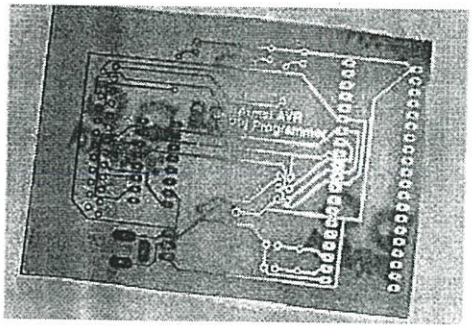
รูปที่ 3.3.26 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(12)



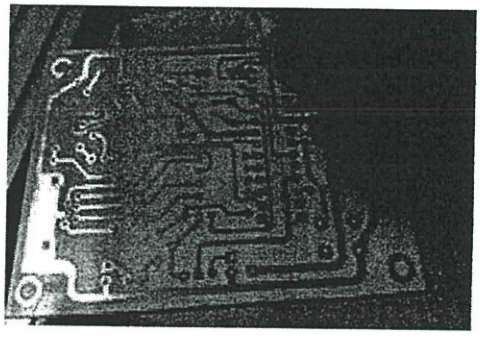
รูปที่ 3.3.27 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(13)



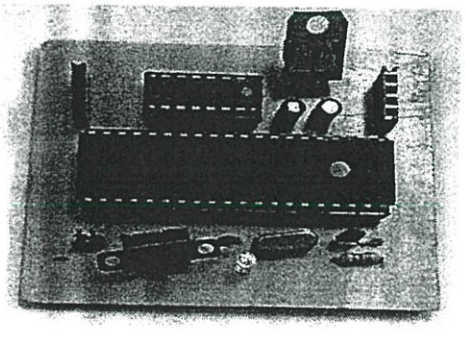
รูปที่ 3.3.28 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(14)



รูปที่ 3.3.29 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(15)



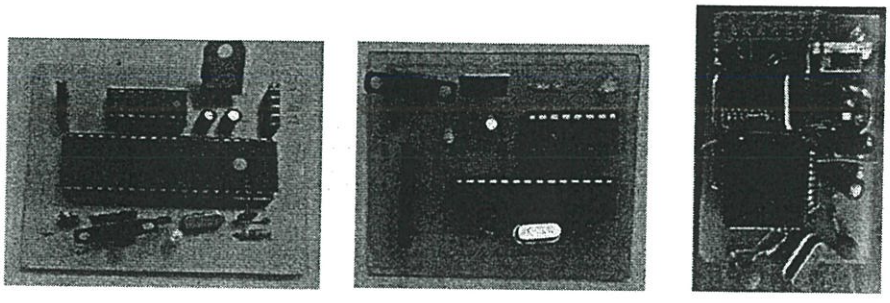
รูปที่ 3.3.30 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(16)



รูปที่ 3.3.31 ภาพแสดงขั้นตอนการกัดปริ้นแผ่น PCB(17)

โดยจะมีขั้นตอนการทำแผ่น PCB เป็นแบบนี้ทั้ง 3 วงจร คือ small Sphere robot, medium Sphere robot and big Sphere robot แต่จะแตกต่างกันในเรื่องของการใช้ ไมโครคอนโทลเลอร์ที่มีขนาดและเบอร์แตกต่างกัน เนื่องมาจากความต้องการให้ Sphere robot มีขนาดเล็กลงเรื่อยๆ ดังนั้นต้องหาเบอร์ไมโครคอลโทลเลอร์ที่มีขนาดเล็กลงตามมาด้วย โดยใน big Sphere robot ใช้ไมโครคอลโทลเลอร์เบอร์ 16f877a I/SP มีขนาดของไมโครคอลโทลเลอร์ที่ใหญ่ที่สุดคือ 15.11 x 51.94 mm. ตามด้วย medium Sphere robot ใช้เบอร์ไมโคลคอลโทลเลอร์เบอร์ 16f867a I/SP มีขนาด

ของไมโคลคอลโทลเลอร์คือ 6.99 x 34.16 mm. และ small Sphere robot ใช้ไมโครเบอร์ 16f877a I/SP ชนิด PLCC มีขนาดของไมโคลคอลโทลเลอร์ที่เล็กที่สุดคือ 16.51 x 17.40 mm.

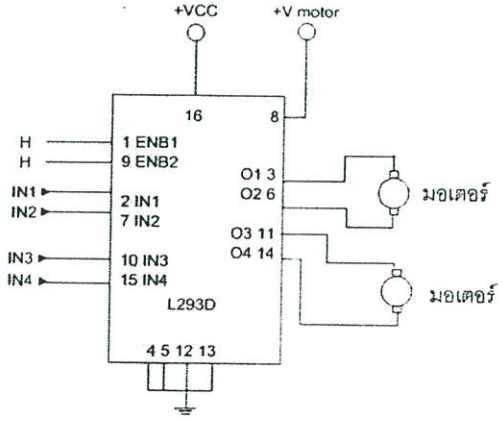


รูปที่ 3.3.32 ภาพแสดงวงจรที่ได้ทำการกัดปริ้นมาของ Sphere robot คันใหญ่(ซ้าย), คันกลาง (กลาง), คันขวา(ขวา)

3.3.2 ส่วนของวงจรควบคุมความเร็วของมอเตอร์

3.3.2.1 การใช้ซอฟแวร์ควบคุมความเร็วมอเตอร์โดยผ่านไอซีขับมอเตอร์ L293D

โดยนำขา Enable 1 กับ 2 ของไอซีขับมอเตอร์ต่อเข้ากับไมโครคอนโทลเลอร์ เพื่อที่จะควบคุมความเร็วของมอเตอร์ โดย 1 ขา ควบคุมมอเตอร์ 1 ตัว ซึ่งสามารถควบคุมได้ 255 ระดับเป็นการควบคุม ความกว้างของสัญญาณพัลน์ในการจ่ายโวลท์ให้กับไอซีขับมอเตอร์ก็จะทำให้ปรับความเร็วของมอเตอร์ได้นั่นเอง



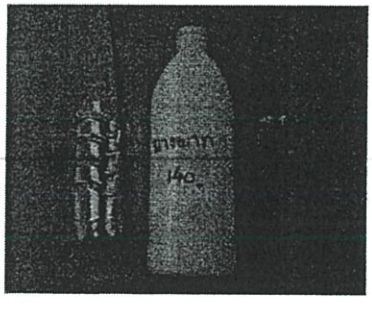
รูปที่ 3.3.48 ภาพแสดงไอซีขับมอเตอร์ L293D

3.3.3 โครงสร้างของหุ่นยนต์และส่วนประกอบอื่นๆ

ในเรื่องนี้จะพูดถึงโครงสร้างและองค์ประกอบของหุ่นยนต์ว่าจะประกอบไปด้วยอะไรบ้าง และพูดถึงการหล่อยางที่ต้องนำมาใช้ในการทำส่วนที่ห้อหุ้มลูกบอลชั้นนอกสุดเพื่อให้เกิดพื้นสัมผัสที่ดีขึ้นและมาทำยางรถเพื่อให้รถที่วิ่งยึดเกาะกับลูกบอลข้างในส่งผลให้ลูกบอลเครื่อนที่ได้ดีขึ้นไม่หมุนรอบตัวเอง

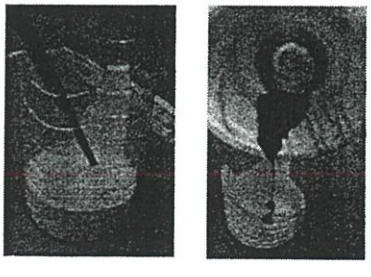
3.3.3.1 อุปกรณ์ที่ใช้ในการหล่อแผ่นยางพารา

1.ยางพารา
2.ภาชนะบรรจุ เช่น แก้ว
3.ไม้หรือแปรง
4.ฟิวเจอร์บอร์ด
5.ผงสีคาร์บอน



รูปที่ 3.3.49 แสดงอุปกรณ์ที่ใช้หล่อแผ่นยางพารา

ขั้นตอนการหล่อแผ่นยางพารา



รูปที่ 3.3.50 แสดงขั้นตอนการหล่อแผ่นยางพารา

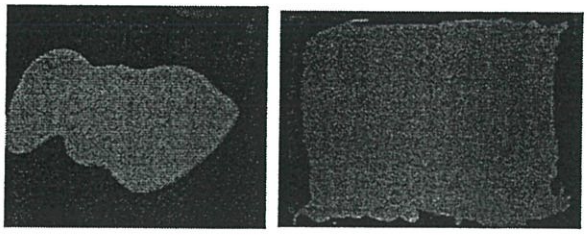
1.เทยางพาราลงในแก้วที่จัดเตรียมไว้ปริมาณตามต้องการ หากต้องการให้งานมีสี ให้นำผงสีคาร์บอนมาละลายน้ำแล้วใส่ลงไปในแก้วที่มียางพารา

รูปที่ 3.3.51 แสดงขั้นตอนการหล่อแผ่นยางพารา(1)

2.ใช้ไม้คนเบาๆ ไม่ให้เกิดฟองอากาศ แล้วทิ้งไว้ประมาณ 20 นาที

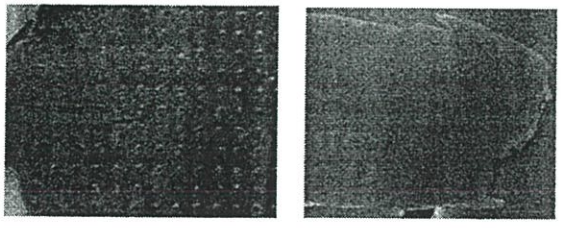
รูปที่ 3.3.52 แสดงขั้นตอนการหล่อแผ่นยางพารา(2)

3.จากนั้นนำมาเทลงบนฟิวเจอร์บอร์ดที่เตรียมไว้ (หากต้องการให้งานที่ทำออกมามีปุ่มเราต้องทำการเจาะรูฟิวเจอร์บอร์ก่อน)



รูปที่ 3.3.53 แสดงขั้นตอนการหล่อแผ่นยางพารา(3)

4.จากนั้นเอียงแผ่นฟิวเจอร์บอร์ดเพื่อให้ยางพาราไหลจนเต็มแผ่นฟิวเจอร์บอร์ด

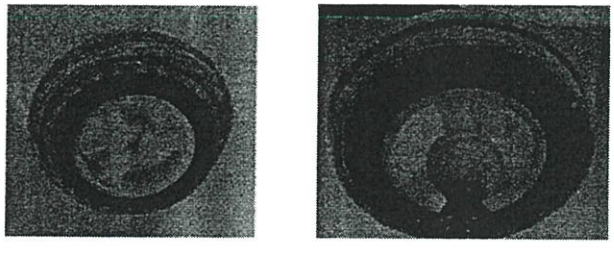


รูปที่ 3.3.54 แสดงขั้นตอนการหล่อแผ่นยางพารา(4)

5.นำไปวางตากแดดจนแห้ง จากนั้นทำการลอกออกเพื่อนำไปใช้งาน (หากมีเป็นแบบปุ่มเวลาลอกระวังปุ่มขาด)

การทำล้อรถ

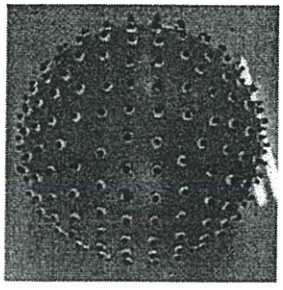
เนื่องจากเวลารถวิ่งภายในทรงกลมล้อรถจะสัมผัสผิวภายในของทรงกลมเพียงบางส่วน เราจึงได้ทำการหล่อแผ่นยางพารามาติดรอบล้อรถเป็นลักษณะคล้ายสามเหลี่ยม(ดังรูป) เพื่อเพิ่มพื้นที่ผิวสัมผัสระหว่างล้อรถและผิวภายในของทรงกลม และยังเป็นการเพิ่มแรงเสียดทานให้มากขึ้นด้วย



รูปที่ 3.3.55 แสดงการติดยางพาราที่ล้อรถ

ผิวภายนอกของทรงกลม

เนื่องจากผิวภายนอกของทรงกลมมีความลื่น เราจึงได้ทำการเพิ่มผิวสัมผัสระหว่างพื้นกับผิวภายนอกของทรงกลม โดยมีลักษณะดังนี้

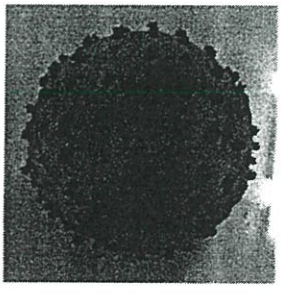


รูปที่ 3.3.56 แสดงผิวภายนอกของทรงกลมของรถคันใหญ่

- รถคันใหญ่เราได้ทำการติดเม็ดพลาสติกที่มีลักษณะเป็นทรงกระบอกและตุ่มยางพาราที่ผิวภายนอกของทรงกลม

รูปที่ 3.3.57 แสดงผิวภายนอกของทรงกลมของรถคันกลาง

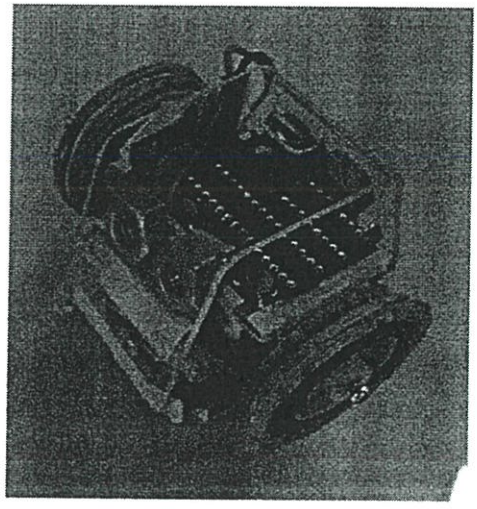
- รถคันกลางเราใช้ยางพาราแบบแผ่นเรียบในการหุ้มผิวภายนอกของทรงกลม



รูปที่ 3.3.58 แสดงผิวภายนอกของทรงกลมของรถคันเล็ก

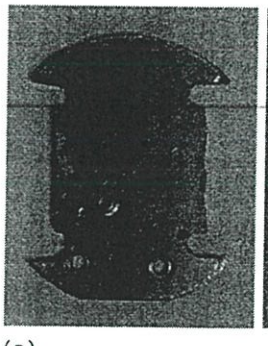
- รถคันเล็ก เราใช้ยางพาราแบบแผ่นที่มีตุ่มด้วยในการหุ้มผิวของทรงกลม

3.3.3.2 โครงรถ

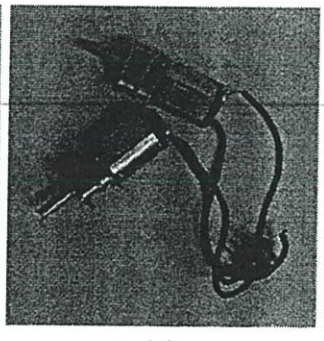


รูปที่ 3.3.59 แสดงรถคันเล็ก

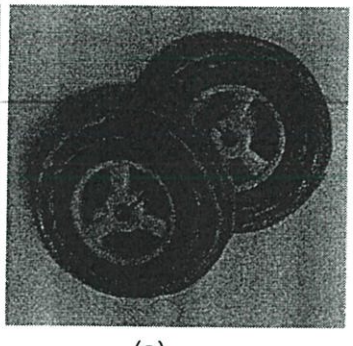
รถคันเล็กประกอบด้วยอุปกรณ์ดังนี้



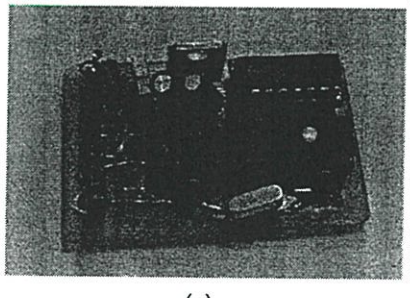
(ก)



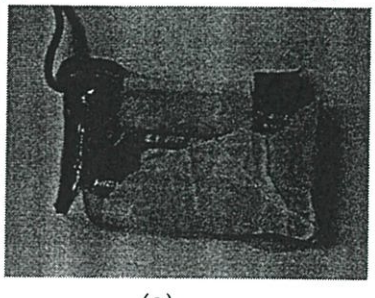
(ข)



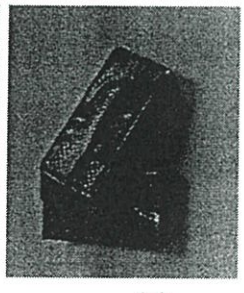
(ค)



(ง)



(จ)

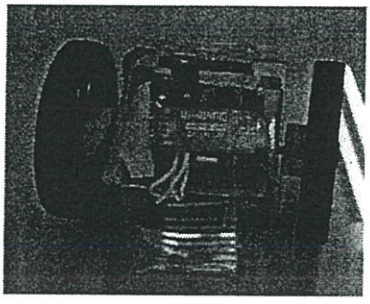


(ฉ)

รูปที่ แสดงอุปกรณ์ของรถคันเล็กโดยประกอบด้วย

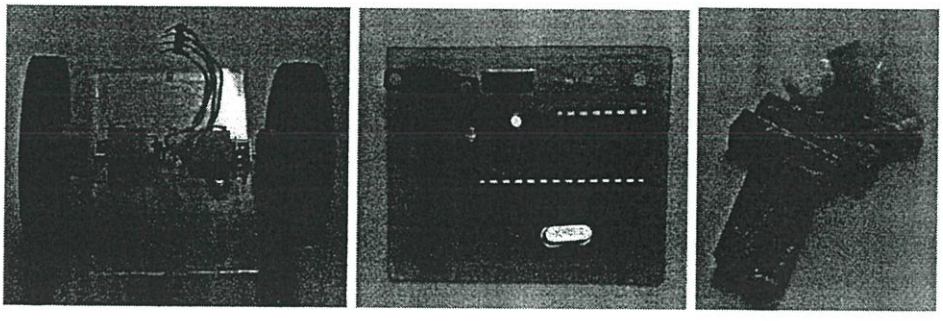
(ก) ฐานแผ่นเหล็ก (ข) มอเตอร์ (ค) ล้อ

(ง) วงจร (จ) แบตเตอรี่ (ฉ) ตุ้มถ่วงน้ำหนัก

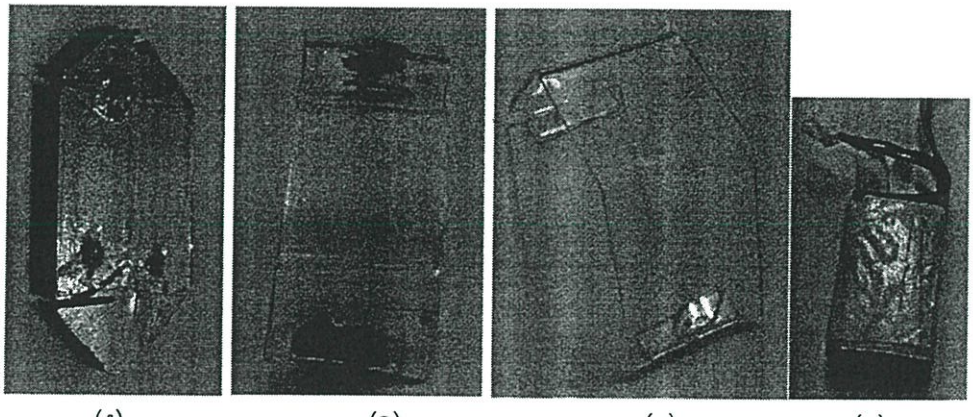


รูปที่ 3.3.60 แสดงรถคันกลาง

รถคันกลางประกอบด้วยอุปกรณ์ดังนี้



(ก) (ข) (ค)

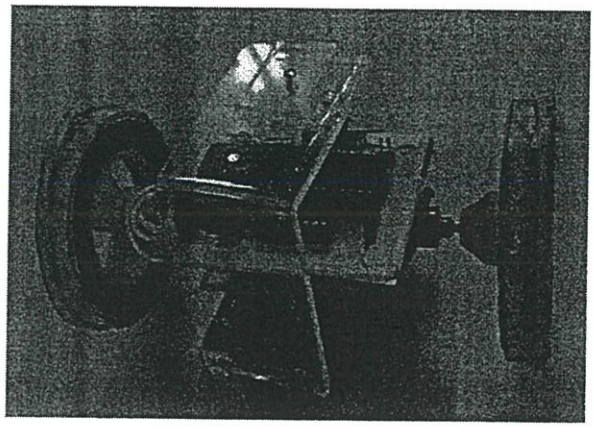


(ง) (จ) (ฉ) (ช)

รูปที่ 3.3.61 แสดงอุปกรณ์ของรถคันกลางโดยประกอบด้วย

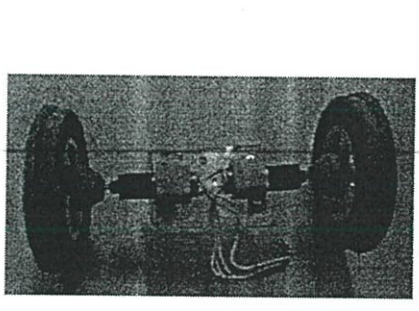
(ก) ชุดมอเตอร์และล้อ (ข) วงจร (ค) ตุ้มถ่วงน้ำหนัก (ง) ฐานแผ่นเหล็ก

(จ) แผ่นอะคิลิกดัดเป็นโครงรถด้านล่าง (ฉ) แผ่นอะคิลิกดัดเป็นโครงรถด้านบน (ช) แบตเตอรี่

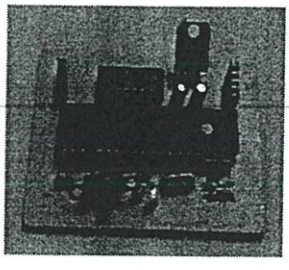


รูปที่ 3.3.62 แสดงรถคันใหญ่

รถคันใหญ่ประกอบด้วยอุปกรณ์ดังนี้

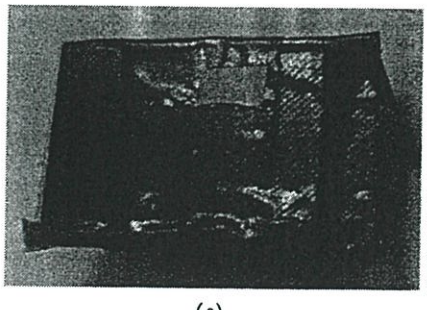


(ก)

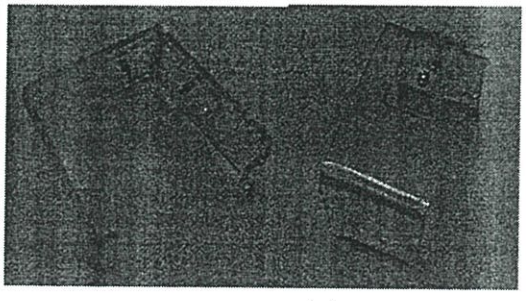


(ข)

(ค)



(ง)



(จ)

รูปที่ 3.3.63 แสดงอุปกรณ์ของรถคันใหญ่โดยประกอบด้วย

(ก) ชุดมอเตอร์และล้อ (ข) วงจร (ค) แบตเตอรี่

(ง) ฐานแผ่นเหล็กและตุ้มถ่วงน้ำหนัก (จ) แผ่นอะคิลิกดัดเป็นโครงรถด้านบน

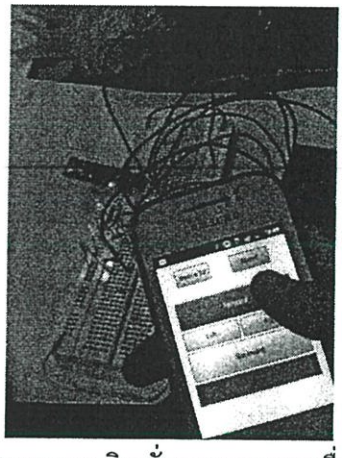
บทที่ 4

ผลการทดลอง

4.1 การทดลองแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control Sphere Robot Application)

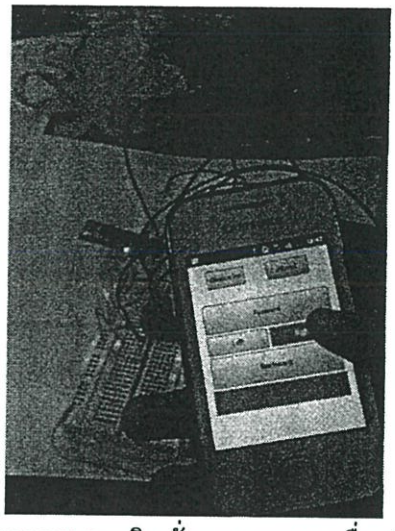
ในส่วนของการทดลองนี้ จะประกอบไปด้วย แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง ซึ่งรันอยู่บนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ และส่วนโปรแกรมสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง ซึ่งรันอยู่บนไมโครคอนโทรลเลอร์ โดยทั้งสองส่วนนี้ จะสื่อสารกันผ่านเทคโนโลยีบลูทูธบนโทรศัพท์มือถือและโมดูลบลูทูธซึ่งถูกประกอบเข้ากับวงจรไฟฟ้าบนรถบังคับ

การทดลองนี้จะใช้หลอด LED แสดงผลการเกิดสัญญาณควบคุม แทนการใช้ DC มอเตอร์ โดยแสดงให้เห็นดังรูปที่ 4.1



รูปที่ 4.1 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า)

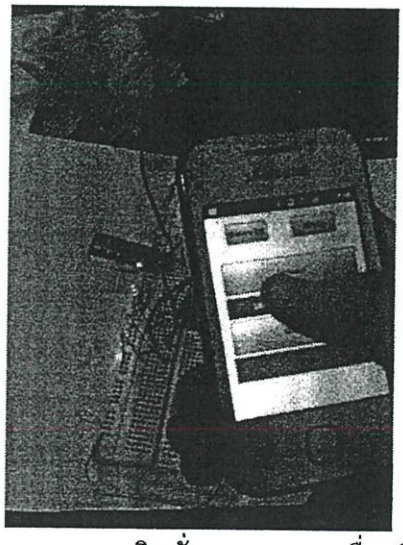
จากรูปภาพที่ 4.1 จะเห็นว่า เมื่อสั่งเคลื่อนที่ไปข้างหน้าผ่านแอพพลิเคชั่นแล้ว หลอด LED หลอดแรก และหลอดที่ 3 จะติด และหลอดที่ 2 และ 4 จะดับ ซึ่งหมายความว่า มอเตอร์ตัวแรก (มอเตอร์ขวา) (หลอด LED 1,2) และมอเตอร์ตัวที่ 2 (มอเตอร์ซ้าย)(หลอด LED 3,4) หมุนไปในทิศทางเดียวกัน



รูปที่ 4.2 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่เลี้ยวไปทางขวา)

ยกตัวอย่าง การเลี้ยวขวาของรถ จากภาพที่ 4.2 จะเห็นว่า เมื่อมีการสั่งการจากแอพพลิเคชั่นให้รถเลี้ยวขวา หลอด LED หลอดที่ 3 นั้นจะติดเพียงหลอดเดียว ซึ่งหมายความว่ามอเตอร์ซ้าย (หลอด LED 3,4) นั้นมีการทำงานและมอเตอร์ขวานั้นหยุดการทำงาน เท่ากับว่า รถนั้นกำลังอยู่ในสภาพเลี้ยวไปทางขวานั่นเอง

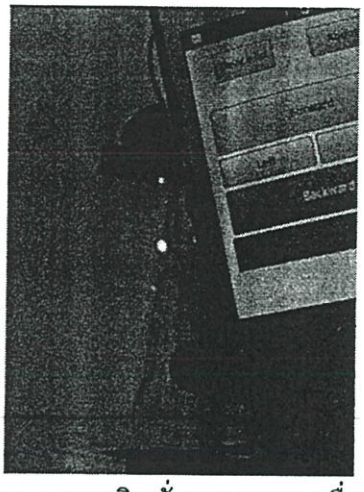
จากนั้น ยกตัวอย่าง การเลี้ยวซ้ายของรถ ซึ่งจะเห็นได้จากภาพที่ 4.3 ว่า เมื่อมีการสั่งการมาจากแอพพลิเคชั่นให้รถเลี้ยวซ้ายนั้น หลอด LED หลอดที่ 1 จะติดเพียงหลอดเดียว ซึ่งหมายความว่ามอเตอร์ขวา (หลอด LED 1,2) นั้นมีการทำงานและมอเตอร์ซ้ายนั้นหยุดการทำงาน เท่ากับว่า รถนั้นกำลังอยู่ในสภาพเลี้ยวไปทางซ้ายนั้นเอง



รูปที่ 4.3 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนที่เลี้ยวไปทางซ้าย)

จากนั้น ยกตัวอย่าง การถอยหลังของรถ ซึ่งจะเห็นได้จากภาพที่ 4.4 ว่า เมื่อมีการสั่งการมาจากแอพพลิเคชั่นให้รถถอยหลังนั้น หลอด LED หลอดที่ 2 และ 4 จะติด ซึ่งหมายความว่ามอเตอร์ขวา (หลอด LED 1,2) และมอเตอร์ซ้าย (หลอด LED 3,4) นั้นมีการทำงานแต่หมุนในทิศทางตรงกันข้ามกับปกติ เท่ากับว่า รถนั้นกำลังอยู่ในสภาพเคลื่อนที่ถอยหลังนั่นเอง

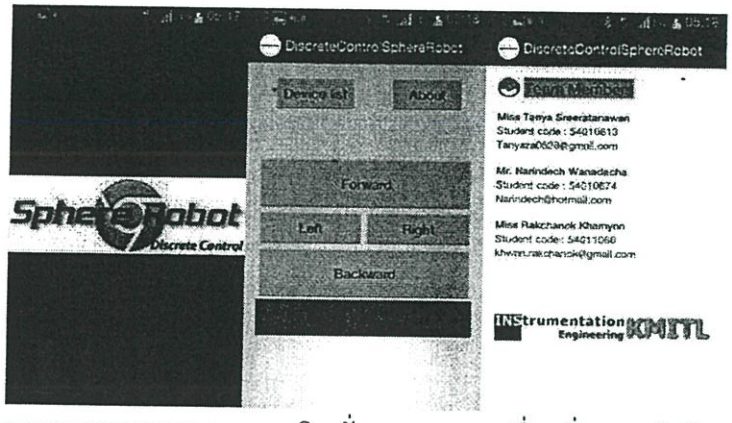
และสุดท้าย สำหรับการหยุดการเคลื่อนที่ของรถ ซึ่งจะเห็นได้จากภาพที่ 4.5 ว่า เมื่อมีการสั่งการมาจากแอพพลิเคชั่นให้รถหยุดนั้น หลอด LED ทุกหลอดจะดับหมด ซึ่งหมายความว่ามอเตอร์ขวา (หลอด LED 1,2) และมอเตอร์ซ้าย (หลอด LED 3,4) หยุดการทำงาน ทำให้รถนั้นหยุดการเคลื่อนที่ไปด้วย



รูปที่ 4.4 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งเคลื่อนทีถอยหลัง)

รูปที่ 4.5 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (สั่งหยุดการเคลื่อนที่)

ผลการทดลองอีกส่วนหนึ่ง จะเป็นผลการทดลองในส่วนแอพพลิเคชั่นบนโทรศัพท์มือถือ ซึ่งมีหน้าตาของหน้าต่างผู้ใช้งานแอพพลิเคชั่นดังต่อไปนี้

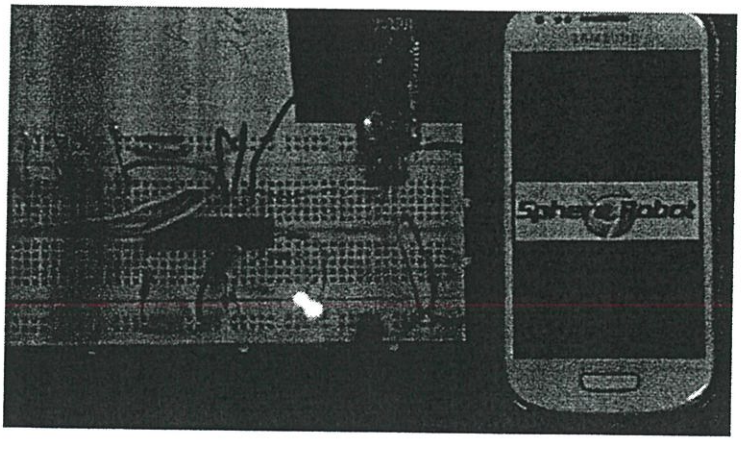


รูปที่ 4.6 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (หน้าต่างผู้ใช้งานแอพพลิเคชั่น)

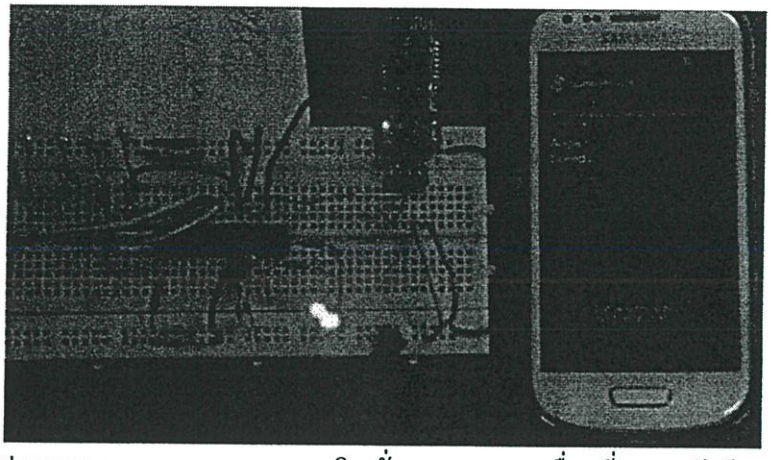
4.2 การทดลองแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (Continuous Control Sphere Robot Application)

ในส่วนของการทดลองนี้ จะประกอบไปด้วย แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง ซึ่งรันอยู่บนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ และส่วนโปรแกรมสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง ซึ่งรันอยู่บนไมโครคอนโทรลเลอร์ โดยทั้งสองส่วนนี้ จะสื่อสารกันผ่านเทคโนโลยีบลูทูธบนโทรศัพท์มือถือและโมดูลบลูทูธซึ่งถูกประกอบเข้ากับวงจรไฟฟ้าบนรถบังคับ

การทดลองนี้จะใช้หลอด LED แสดงผลการเกิดสัญญาณควบคุม แทนการใช้ DC มอเตอร์ โดยแสดงให้เห็นดังรูปที่ 4.7

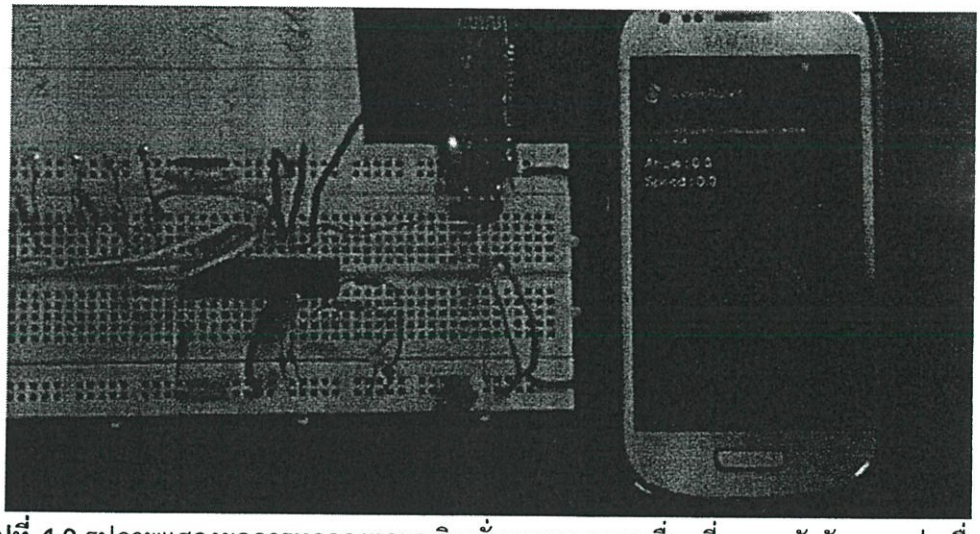


รูปที่ 4.7 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง



รูปที่ 4.8 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (หลังจากแอพพลิเคชั่นกับโมดูลบลูทูธได้เชื่อมต่อกันแล้ว)

จากรูปภาพที่ 4.8 แสดงให้เห็นว่า หลังจากที่แอพพลิเคชั่นบนโทรศัพท์มือถือได้เชื่อมต่อเข้ากับโมดูลบลูทูธ ซึ่งถูกประกอบอยู่กับ PIC ไมโครคอนโทรลเลอร์แล้ว หน้าต่างผู้ใช้งานแอพพลิเคชั่นจะแสดงข้อความแบบ Toast (คล้ายๆข้อความแบบป๊อปอั๊พ) ขึ้นบอก บอกผู้ใช้งานว่า แอพพลิเคชั่นได้เชื่อมต่อเข้ากับโมดูลบลูทูธ ชื่อ NONG ซึ่งมีหมายเลขแอดเดรสเป็น 20:13:06:18:05:81

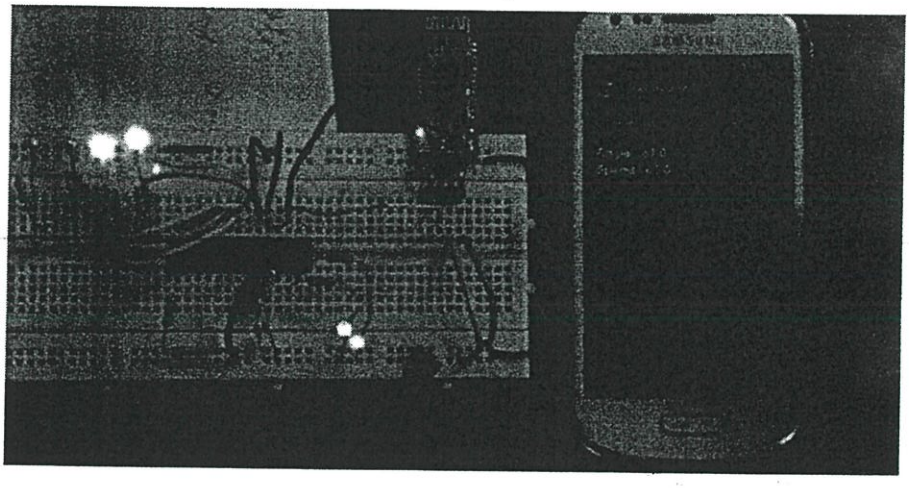


รูปที่ 4.9 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (สั่งหยุดการเคลื่อนที่)

เมื่อทำการเชื่อมต่อบลูทูธกันเสร็จเรียบร้อยแล้ว ก็สามารถทำการสั่งการเคลื่อนที่ผ่านโทรศัพท์มือถือได้เลย โดยการทดลองแรก จะเป็นการสั่งหยุดการเคลื่อนที่ของรถบังคับ โดยจากรูปที่ 4.9 จะเห็นได้ว่า ตัว PIC ไมโครคอนโทรลเลอร์นั้น จะต่ออยู่กับหลอด LED 4 หลอดด้านบน และ 2 หลอดด้านล่าง

โดย หลอด LED 4 หลอดบน นั้นแทนสัญญาณการควบคุมของมอเตอร์ทั้งสองข้าง (ซ้ายและขวา) และหลอด LED 2 หลอดล่างนั้นจะแทนสัญญาณการควบคุมการป้อนสัญญาณ PWM (Pulse Width Modulation) เพื่อเป็นตัวควบคุมการจ่ายไฟเลี้ยงให้กับมอเตอร์ ซึ่งเท่ากับว่าเป็นการควบคุมความเร็วการหมุนของมอเตอร์ไปในตัว

เมื่อผู้ใช้งานสัมผัสหน้าจอของโทรศัพท์มือถือ ส่วนที่เป็นวงกลมสีเหลือง (จอยสติ๊ก) ตัวแอพพลิเคชั่นจะทำการประมวลผลว่า ตำแหน่งของนิ้วที่ผู้ใช้งานสัมผัสนั้น อยู่ในบริเวณใดของวงกลม ถ้าหากสัมผัสที่จุดศูนย์กลางของวงกลม แอพพลิเคชั่นจะทำการคำนวณ และส่งค่ามุม 0.0 และความเร็ว 0.0 ผ่านบลูทูธมายัง PIC ไมโครคอนโทรลเลอร์ ทำให้หลอด LED ทุกหลอดดับ เท่ากับว่ามอเตอร์ทั้งสองตัวหยุดการทำงาน และไม่มีการป้อนสัญญาณ PWM ออกมาควบคุมความเร็วมอเตอร์ จากนั้นเมื่อผู้ใช้งานลากนิ้วไปบนจอยสติ๊กไปทางด้านหน้า แอพพลิเคชั่นก็จะทำการประมวลผลว่าตำแหน่งของนิ้วที่ผู้ใช้งานสัมผัสอยู่บริเวณใดเมื่อเทียบกับจุดศูนย์กลางรูปวงกลม โดยจะได้ค่ามุมองศาและค่าความเร็วออกมา ยิ่งห่างออกจากจุดศูนย์กลางมากเท่าไหร่ ค่าความเร็วยิ่งมากขึ้นเท่านั้น ซึ่งแสดงได้ดังรูปที่ 4.10



รูปที่ 4.10 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า)

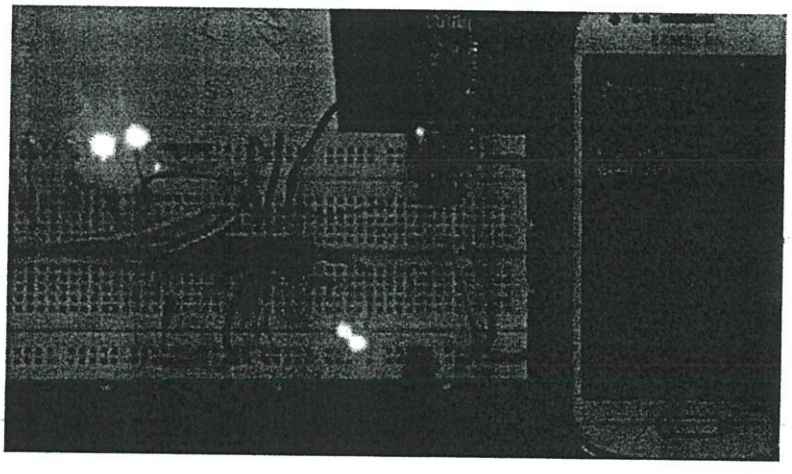
จากรูปที่ 4.10 จะเห็นได้ว่า นิ้วของผู้ใช้งานลากไปด้านหน้าทำให้ได้ค่ามุมองศาออกมาเป็น 84.0 องศาและค่าความเร็วเพิ่มขึ้นจาก 0.0 เป็น 64.0 เนื่องจากตำแหน่งของนิ้วที่สัมผัสออกห่างจากจุดศูนย์กลางรูปภาพ จากนั้นค่ามุมและค่าความเร็วที่ได้ จะถูกส่งผ่านบลูทูธไปยังไมโครคอนโทรลเลอร์ ตัวไมโครคอนโทรลเลอร์เองก็จะทำการประมวลผล และสร้างสัญญาณควบคุมมอเตอร์ (หลอด LED 4 หลอดบน) ให้เคลื่อนที่ไปด้านหน้า และสร้างสัญญาณ PWM (หลอด LED 2 หลอดล่าง)ให้เพิ่มความเร็ว

โดยในการทดลองนี้ จะกำหนดไว้ว่า หลอด LED 4 ตัวบน หลอดที่ 1,3 นับจากขวามือ จะแทนขาของมอเตอร์ซ้าย และหลอด LED 2,4 จะแทนขาของมอเตอร์ขวา ส่วนหลอด LED 2 ตัวล่าง หลอดซ้าย(สีขาว) จะแทนสัญญาณควบคุมความเร็วของมอเตอร์ซ้าย และหลอด LEDหลอดขวา(สี

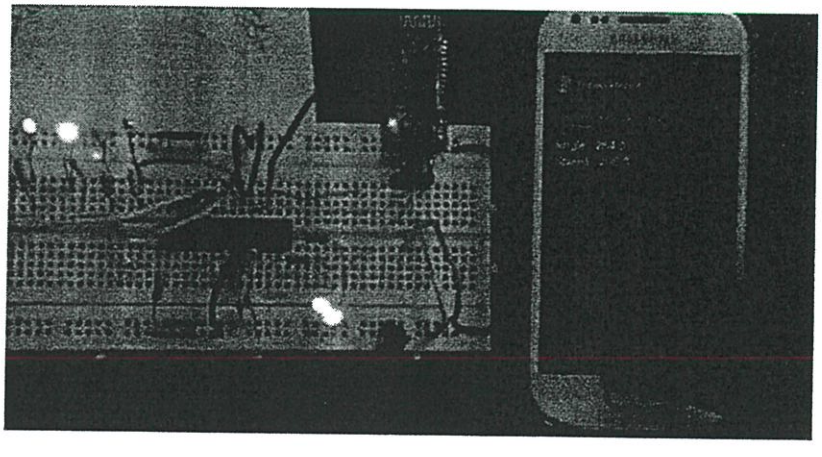
เขียว) จะแทนสัญญาณควบคุมความเร็วของมอเตอร์ขวา โดยผลลัพธ์ที่ได้นั้น ก็คือ มอเตอร์ทั้งสองตัวทำงานโดยมีทิศทางการหมุนไปด้านหน้า (หลอด LED ที่ 1,3 ติด) และไมโครคอนโทรลเลอร์ป้อนสัญญาณ PWM ออกมาควบคุมทั้งมอเตอร์ซ้ายและขวา (หลอด LED 2 หลอดล่างสว่างทั้งคู่)

หลังจากนั้น ยิ่งผู้ใช้งานลากนิ้วขึ้นไปทาด้านบน ห่างออกจากจุดศูนย์กลางเรื่อยๆ ก็ไม่ได้เปลี่ยนแปลงทิศทางการเคลื่อนที่แต่อย่างใด (ยังคงอยู่ในทิศทางการเคลื่อนที่ไปด้านหน้า) แต่ความเร็วของรถจะเพิ่มขึ้นเรื่อยๆ เนื่องจากสัญญาณ PWM ที่ถูกป้อนออกมาจะเพิ่มขึ้น (สังเกตได้จากความสว่างของหลอด LED 2 หลอดล่างที่สว่างขึ้นเรื่อยๆ)

แต่อย่างไรก็ตาม เมื่อผู้ใช้งานลากนิ้วห่างจากจุดศูนย์กลางมากจนถึงระยะหนึ่ง ตัวแอพพลิเคชั่นนั้นจะทำการประมวลผล และจำกัดค่าความเร็วที่จะถูกส่งไปยังไมโครคอนโทรลเลอร์ไว้ที่ 255.0 ซึ่งเป็นค่าสูงสุดสำหรับการสร้างสัญญาณ PWM ในไมโครคอนโทรลเลอร์ ดังรูปที่ 4.11

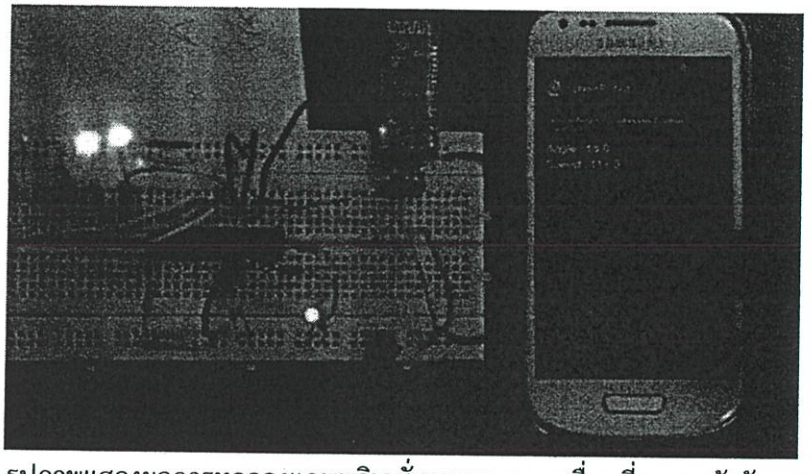


รูปที่ 4.11 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (สั่งเคลื่อนที่ไปข้างหน้า และเพิ่มความเร็ว)



รูปที่ 4.12 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (สั่งเคลื่อนที่ถอยหลัง และเพิ่มความเร็ว)

เช่นเดียวกัน กับการเคลื่อนที่เป็นข้างหน้า การสั่งเคลื่อนที่ถอยหลังก็สามารถทำได้โดยผู้ใช้งานลากนิ้วลงมาด้านล่างของจอยสติ๊ก ตัวแอพพลิเคชั่นก็จะทำการประมวลผลหาค่า มุมองศาและความเร็วจากตำแหน่งของนิ้วมือที่สัมผัสเปรียบเทียบกับจุดศูนย์กลางของรูปวงกลม จากนั้นส่วนไมโครคอนโทรลเลอร์ เมื่อได้รับค่าผ่านโมดูลบลูทูธเข้ามาแล้ว ก็นำค่าไปทำการประมวลผลสั่งการทำงานของมอเตอร์ว่าควรหมุนไปในทิศทางใด ความเร็วเท่าใด โดยจากรูปที่ 4.12 จะเห็นได้ว่า หลอด LDE 4 หลอดบน หลอดที่ 3,4 นับจากขวามือนั้นติด เท่ากับว่ามอเตอร์นั้นหมุนทิศตรงกันข้ามกับปกติทำให้รถเคลื่อนที่ถอยหลัง ในขณะเดียวกัน หลอด LED 2 หลอดล่างก็จะสว่างตามการกำเนิดสัญญาณ PWM ที่สอดคล้องกับค่าความเร็วที่ส่งมาจากแอพพลิเคชั่นนั่นเอง

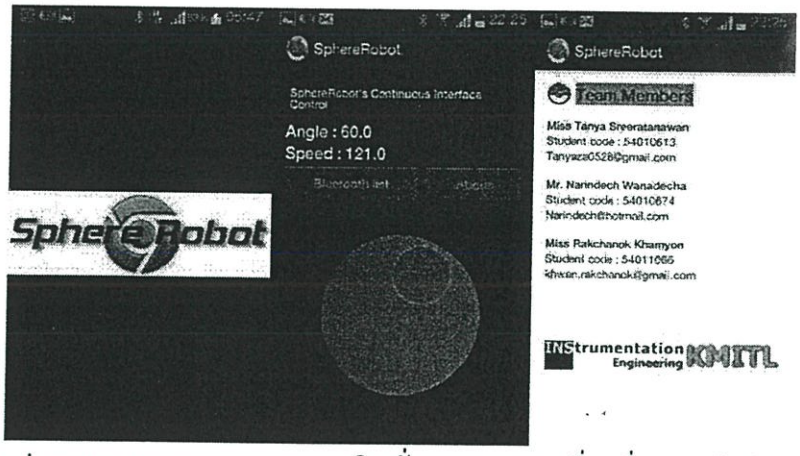


รูปที่ 4.13 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (สั่งเคลื่อนที่เดินหน้า ในทิศทางเฉียงไปทางขวา และเพิ่มความเร็ว)

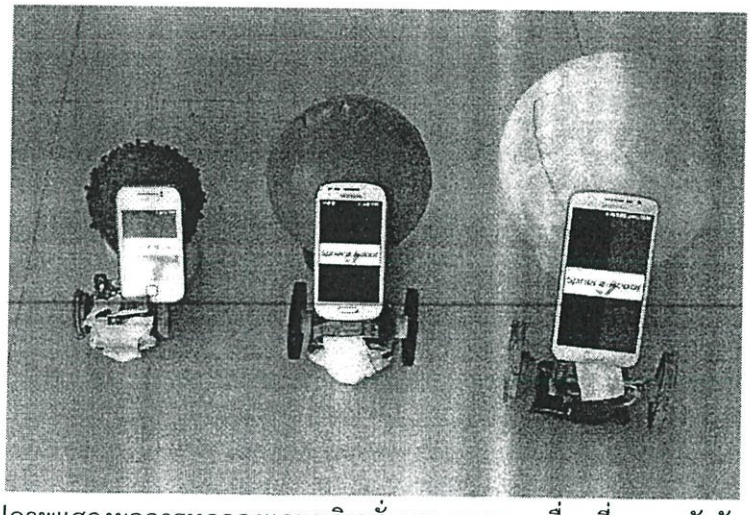
จากนั้น ยกตัวอย่างการเคลื่อนที่ไปข้างหน้าเฉียงไปทางขวา ซึ่งจากรูปที่ 4.13แสดงให้เห็นว่าแอพพลิเคชั่นจะส่งค่ามุมองศาและความเร็วออกมาเป็น 16.0 และ 117.0 ตามลำดับสอดคล้องกับนิ้วของผู้ใช้งานที่สัมผัสบนรูปวงกลม (จอยสติ๊ก) ค่ามุมองศาและความเร็วที่ได้นั้นจะถูกส่งไปยังไมโครคอนโทรลเลอร์ ตัวไมโครคอนโทรลเลอร์นั้นก็จะทำกาคำนวณว่า ควรสั่งการหมุนของมอเตอร์ไปในทิศทางใด ด้วยความเร็วเท่าใด ซึ่งจะเห็นได้ว่าหลอด LED 4 หลอดบน หลอดที่ 1,2 นับจากขวามือนั้นติด เท่ากับว่ามอเตอร์นั้นมีทิศทางการหมุนไปด้านหน้า เพียงแต่ว่าหลอด LED 2 หลอดล่าง หลอดสีขาว (มอเตอร์ซ้าย) นั้นติดในขณะที่ หลอดสีเขียว(มอเตอร์ขวา)นั้นดับ เท่ากับว่ามอเตอร์ซ้ายนั้นทำงานอยู่ข้างเดียว ส่งผลให้การเคลื่อนที่ของรถนั้นจะเบนออกไปทางขวาด้วย

แต่อย่างไรก็ตาม การเคลื่อนที่ไปด้านหน้าเบนขวาของรถบังคับนั้น ก็จะมีความเร็วสอดคล้องกับค่าความเร็วที่ถูกสั่งมาจากโทรศัพท์มือถือนั่นเอง

ส่วนผลการทดลองของฝั่งแอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับแบบต่อเนื่องบนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์นั้น มีหน้าตาของหน้าต่างผู้ใช้งานแอพพลิเคชั่นดังแสดงในรูปที่ 4.14



รูปที่ 4.14 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (หน้าต่างผู้ใช้งานแอพพลิเคชั่น)



รูปที่ 4.15 รูปภาพแสดงผลการทดลองแอพพลิเคชั่นควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (ทดสอบบนโทรศัพท์มือถือ Samsung Galaxy Y(Android 2.3.6 "Froyo"), Samsung Galaxy S3 mini(Android 4.2 "Jellybean") และ Samsung Galaxy GRAND 2(Android 4.3 "Jellybean")

*** หมายเหตุ: หุ่นยนต์ทรงกลมทั้ง 3 แบบนี้ใช้แอพพลิเคชั่นควบคุมการเคลื่อนที่ของหุ่นยนต์แบบต่อเนื่องทั้งหมด

4.3 คุณสมบัติของหุ่นยนต์ Sphere Robot ในรูปแบบต่างๆ (Sphere Robot's Specification)

ผลการทดลองในส่วนนี้เป็นการบอกถึงรายละเอียดโดยรวมของหุ่นยนต์ Sphere Robot แต่ละตัวที่ทำขึ้นมาเปรียบเทียบกัน สามารถแสดงให้เห็นตามตารางที่ 4.1

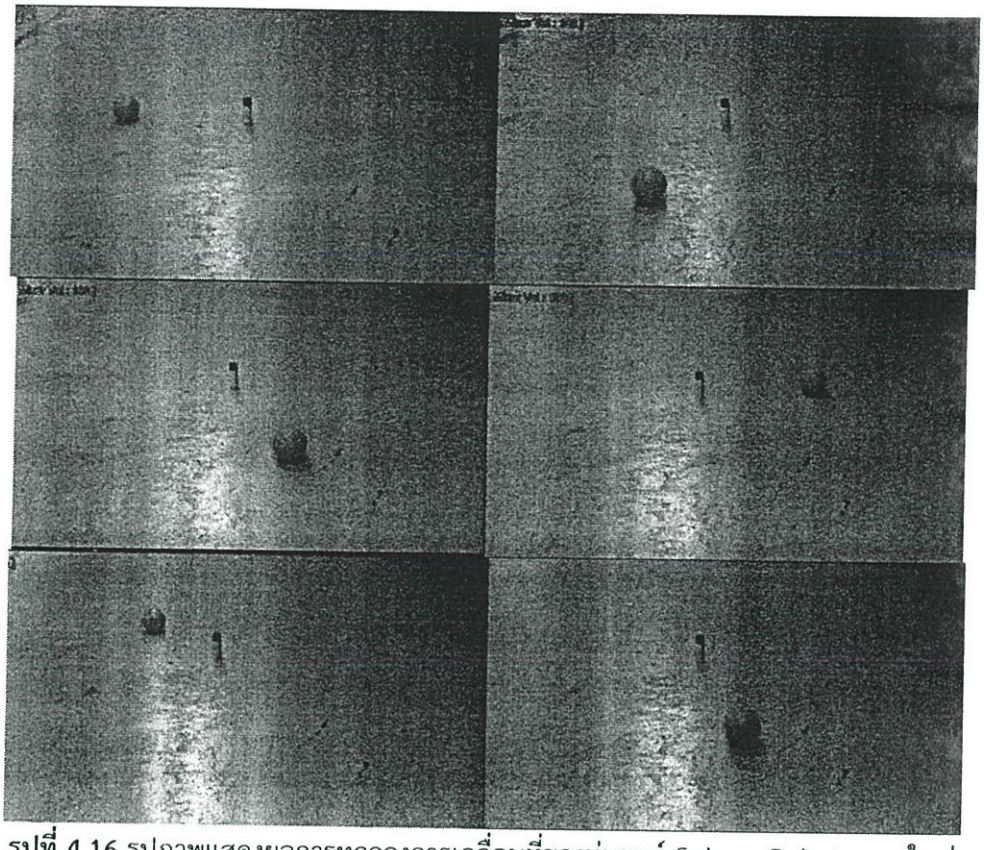
ส่วนประกอบ/ชนิดของหุ่นยนต์	หุ่นยนต์เล็ก	หุ่นยนต์กลาง	หุ่นยนต์ใหญ่
ฐานของรถ	140 g / 7.6x8.2x6 cm(ฐานเหล็ก)	70 g(ฐานอะคริลิค)	270 g(ฐานเหล็ก)
ล้อรวมยาง 2 ข้าง	20.5 g	80 g	80 g
มอเตอร์ 2 ข้าง	20 g	20 g	60 g
วงจรไฟฟ้า	22 g	20 g	30 g
ตุ้มถ่วงน้ำหนัก	100 g	230 g	-
แบตเตอรี่ Li-Po	40 g	40 g	40 g
ขนาดเส้นผ่านศูนย์กลางของโครงทรงกลม	10 cm	15 cm	20 cm
น้ำหนักรวม	352 g	475 g	500 g
ลักษณะพื้นผิวภายนอกของหุ่นยนต์	ยางพารามีตุ่ม	ยางพาราผิวเรียบ	เม็ดพลาสติกทรงกระบอกและตุ่มยางพารา

ตารางที่ 4.1 ตารางแสดงผลการทดลองคุณลักษณะและรายละเอียดโดยรวมของหุ่นยนต์ Sphere Robot ในแต่ละรูปแบบ

จากผลการทดลองในส่วนนี้ จะเห็นได้ว่า คุณสมบัติและรายละเอียดของหุ่นยนต์ทรงกลมแต่ละรูปแบบนั้น มีความแตกต่างกัน เช่น การถ่วงน้ำหนักของรถเพื่อให้ได้ศูนย์ถ่วงที่เหมาะสมกับขนาดของโครงทรงกลม ขนาดของวงจรเพื่อให้สอดคล้องกับขนาดของโครงทรงกลม และรูปแบบผิวภายนอกของโครงทรงกลม โดยคุณสมบัติแต่ละอย่างนั้น ส่งผลถึงการเคลื่อนที่ของรถด้วย ซึ่งจะกล่าวถึงในการทดลองส่วนต่อไป

4.4 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดใหญ่

การทดลองในส่วนนี้จะเป็นการทดสอบการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดใหญ่ โดยเงื่อนไขของการทดสอบคือ การวิ่งเป็นวงกลมให้มีขนาดรัศมีของวงกลมทั้งเล็กและใหญ่ โดยจุดประสงค์ของการทดสอบนี้คือ เพื่อแสดงให้เห็นว่า หุ่นยนต์มีการเคลื่อนที่ได้ดีมากน้อยเพียงใด มีความนุ่มนวลในการเคลื่อนที่มากน้อยเพียงใด และผลของพื้นผิวภายนอกของโครงทรงกลมนั้นส่งผลกระทบอย่างไรต่อการเคลื่อนที่



รูปที่ 4.16 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดใหญ่

จากรูปที่ 4.16 จะเห็นได้ว่า การทดสอบนั้นทำขึ้นโดยให้ หุ่นยนต์ Sphere Robot ขนาดใหญ่นั้นวิ่งวนรอบๆกระป๋องสีสเปรย์ ซึ่งผลลัพธ์จากการทดสอบนั้นเป็นที่น่าพอใจ มีข้อเสียบ้างบางประการ โดยรายละเอียดของผลการทดลอง มีดังต่อไปนี้

1. หุ่นยนต์สามารถวิ่งวนรอบกระป๋องสีสเปรย์ ได้ระยะของรัศมีที่คงที่ ที่ระดับความเร็วบนแอพพลิเคชั่นควบคุมการเคลื่อนที่ของหุ่นยนต์แบบต่อเนื่อง ประมาณ 140-160

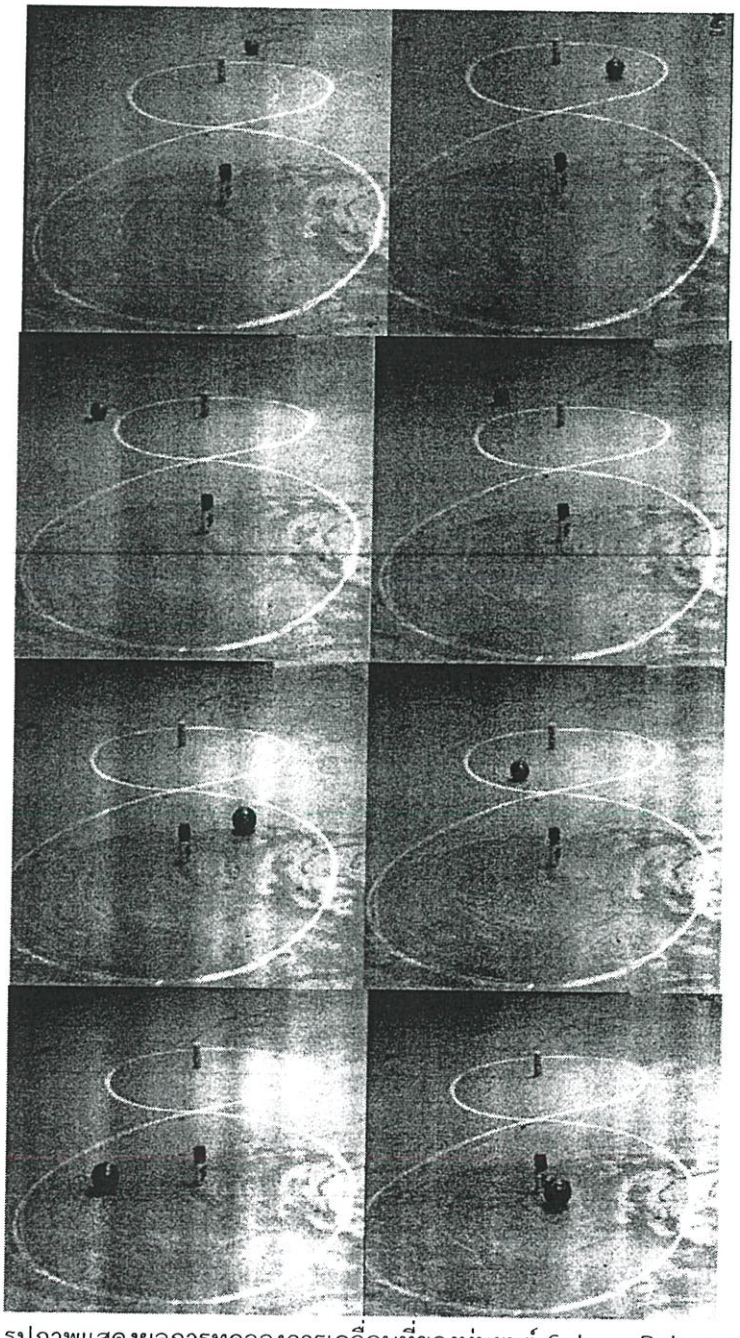
2. เนื่องจากโครงสร้างของรถบังคับภายในทรงกลม ทั้งในด้านของผิวสัมผัสของล้อที่กระทำต่อผิวภายในทรงกลม การถ่วงน้ำหนัก และผิวภายนอกของโครงทรงกลม (ตุ่มเม็ดพลาสติกรวมกับตุ่มยางพารา) ที่มีความถี่ (ระยะห่างระหว่างตุ่ม) มากพอสมควร ทำให้หุ่นยนต์สามารถเคลื่อนที่ได้อย่างราบเรียบ

3. หุ่นยนต์สามารถเปลี่ยนรัศมีการเคลื่อนที่รอบกระป๋องสีสเปรย์ได้ แต่ทั้งนี้ต้องขึ้นอยู่กับการควบคุมด้วย เพราะหากควบคุมได้ไม่ดีเท่าที่ควร หุ่นยนต์ก็จะหมดความเสถียร

4. ความเร็วของมอเตอร์ทั้งสองข้างของรถบังคับภายในทรงกลม ยังไม่ดีเท่าที่ควร เพราะบางครั้งหุ่นยนต์ยังคงมีการสั่น และเคลื่อนที่ผิดแปลกไปจากการควบคุมอยู่ ซึ่งเกิดจากความเร็วของมอเตอร์ทั้งสองข้างนั้นไม่เท่ากัน

4.5 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดกลาง

การทดลองในส่วนนี้จะเป็นการทดสอบการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดกลาง โดยเงื่อนไขของการทดสอบคือ การวิ่งวนรอบกระป๋องสีสเปรย์ 2 กระป๋องแบบเลขแปด โดยจุดประสงค์ของการทดสอบนี้คือ เพื่อแสดงให้เห็นว่า หุ่นยนต์สามารถควบคุมได้ดีมากน้อยเพียงใด การเคลื่อนที่มีความนุ่มนวลในการเคลื่อนที่มากน้อยเพียงใด และผลของพื้นผิวภายนอกของโครงทรงกลมนั้นส่งผลกระทบอย่างไรต่อการเคลื่อนที่



รูปที่ 4.17 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดกลาง

จากรูปที่ 4.17 จะเห็นได้ว่า การทดสอบนั้นทำขึ้นโดยให้ หุ่นยนต์ Sphere Robot ขนาดกลางนั้นวิ่งวนรอบๆกระป๋องสีสเปรย์ 2 กระป๋องในลักษณะเป็นเลข 8 ซึ่งผลลัพธ์จากการทดสอบนั้นเป็นที่น่าพอใจ มีข้อเสียบ้างบางประการ โดยรายละเอียดของผลการทดลอง มีดังต่อไปนี้

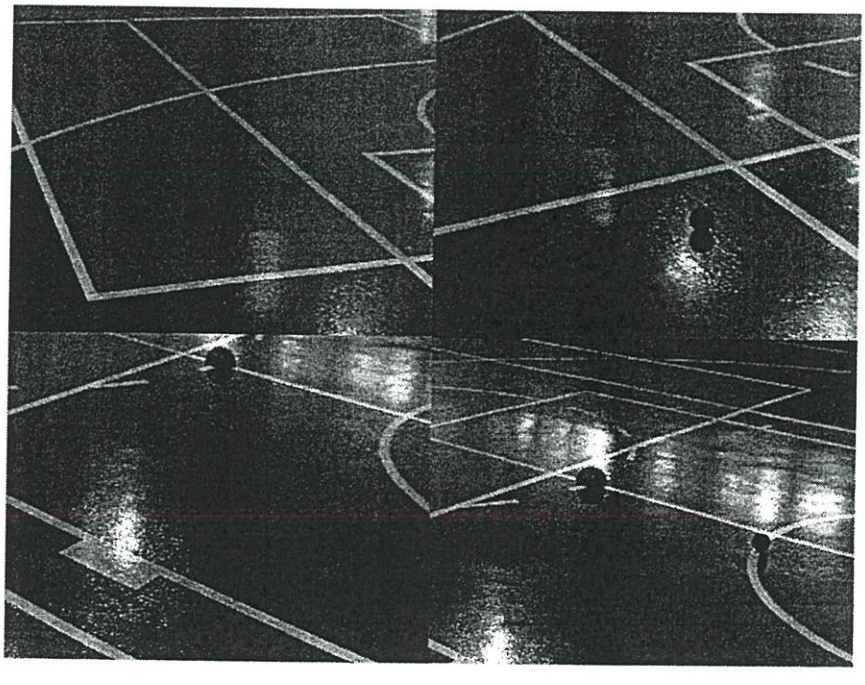
1. หุ่นยนต์ Sphere Robot ขนาดกลางสามารถเคลื่อนที่เป็นเลขแปด ล้อมรอบกระป๋องสีสเปรย์ทั้งสองกระป๋องได้ แต่ยังคงหลุดขอบเขตที่ได้วาดเอาไว้อยู่บ้างในบางรอบ แต่โดยรวมแล้วสามารถเคลื่อนที่เป็นเลขแปดได้

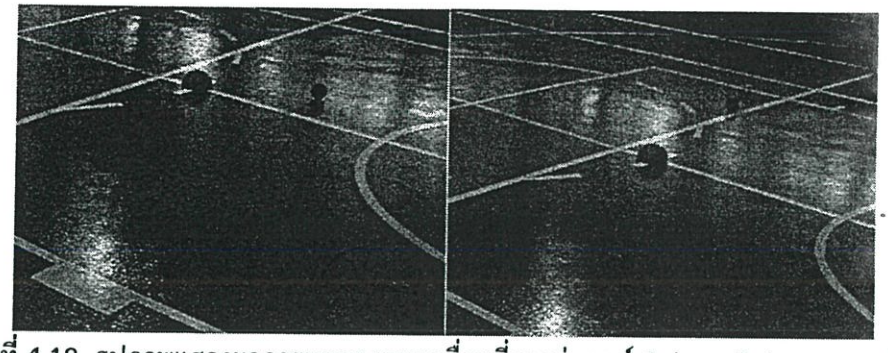
2. การเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดกลางนี้ มีผลการตอบสนองเป็นที่น่าพอใจ เนื่องจากรถบังคับภายในมีโครงสร้างและการถ่วงน้ำหนักที่เหมาะสม บวกกับผิวภายนอกของโครงทรงกลมมีลักษณะเป็นยางเรียบ เลยไม่ค่อยส่งผลกระทบใดๆต่อการเคลื่อนที่

3. การวิ่งเป็นเลขแปดไม่สามารถใช้ความเร็วสูงได้ เนื่องจากว่า การควบคุมการเคลื่อนที่นั้นยังไม่สามารถทำได้แม่นยำมากพอ อีกทั้งการใช้ความเร็วสูงอาจทำให้รถบังคับภายในหมุนรอบตัวเอง แทนที่จะขับเคลื่อนผิวทรงกลมได้ในบางครั้ง สาเหตุมาจาก ผิวสัมผัสของล้อและแรงของมอเตอร์ที่สร้างแรงหมุนรอบตัวเองได้อย่างอิสระมากกว่าที่จะขับผิวภายในทรงกลม

4.6 การทดลองการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดเล็ก

การทดลองในส่วนนี้จะเป็นการทดสอบการเคลื่อนที่ของหุ่นยนต์ทรงกลมขนาดเล็ก โดยเงื่อนไขของการทดสอบคือ การวิ่งแบบอิสระ โดยจุดประสงค์ของการทดสอบนี้คือ เพื่อแสดงให้เห็นว่าหุ่นยนต์สามารถควบคุมได้ดีมากน้อยเพียงใด มีความเร็วในการการเคลื่อนที่เท่าใด มีความนุ่มนวลในการเคลื่อนที่มากน้อยเพียงใด และผลของพื้นผิวภายนอกของโครงทรงกลมนั้นส่งผลกระทบอย่างไรต่อการเคลื่อนที่





รูปที่ 4.18 รูปภาพแสดงผลการทดลองการเคลื่อนที่ของหุ่นยนต์ Sphere Robot ขนาดเล็ก

จากรูปที่ 4.18 จะเห็นได้ว่า การทดสอบนั้นทำขึ้นโดยให้ หุ่นยนต์ Sphere Robot ขนาดเล็กนั้นวิ่งแบบอิสระ ซึ่งผลลัพธ์จากการทดสอบนั้นเป็นที่น่าพอใจ มีข้อเสียบ้างบางประการ โดยรายละเอียดของผลการทดลอง มีดังต่อไปนี้

1. หุ่นยนต์สามารถเคลื่อนที่ได้โดยใช้ความเร็วสูงสุด ซึ่งมีค่าความเร็วเป็น 255.0 (อ้างอิงจากค่าความเร็วบนหน้าต่างผู้ใช้งานแอพพลิเคชัน) โดยเมื่อเปรียบเทียบกันระหว่างหุ่นยนต์ขนาดใหญ่แขนาดกลางแล้ว หุ่นขนาดเล็กสามารถทำความเร็วได้มากที่สุด

2. แต่มีข้อเสียคือ การควบคุมการเคลื่อนที่นั้นทำได้ยาก เนื่องจากโครงสร้างภายในของหุ่นยนต์ ผิวสัมผัสของล้อที่กระทำต่อพื้นผิวภายในทรงกลมและความสมดุลของหุ่นยนต์เมื่อเปรียบเทียบกับหุ่นยนต์ขนาดใหญ่และขนาดกลางแล้วยังไม่ดีเท่าที่ควร

บทที่ 5

สรุปผลการทดลอง และข้อเสนอแนะ

5.1 สรุปผลการทดลอง

จากการทดลองในโครงงานนี้ ได้แสดงให้เห็นว่า หุ่นยนต์ทรงกลมขนาดใหญ่ จะมีการเคลื่อนที่ที่ราบเรียบ และนุ่มนวลมากที่สุด อันเนื่องมาจาก ขนาดและน้ำหนักโดยรวมของตัวหุ่นยนต์ รวมไปถึงสมดุลของตัวหุ่นยนต์ที่อยู่ในระดับที่ดี ทำให้การเคลื่อนที่นั้นราบเรียบ แต่ควบคุมการเลี้ยวไปในทิศทางอื่นๆได้ลำบาก เพราะความเร็วมอเตอร์ที่ไม่เท่ากันของล้อแต่ละข้าง ในขณะที่ หุ่นยนต์ทรงกลมขนาดกลาง สามารถทำได้ดีในการบังคับเลี้ยว อันเนื่องมาจาก ความเร็วของมอเตอร์ที่ใกล้เคียงกัน พื้นผิวภายนอกทรงกลมที่ทำจากยางพาราแบบเรียบ และน้ำหนักกับสมดุลของหุ่นยนต์ที่พอประมาณ แต่ยังคงมีข้อเสียบ้างในด้านความราบเรียบและนุ่มนวลในการเคลื่อนที่ ถ้าเทียบกับหุ่นยนต์ทรงกลมขนาดใหญ่ เนื่องจากการถ่วงน้ำหนักที่เบากว่า และมีข้อเสียในด้านความเร็ว เพราะไม่สามารถใช้ความเร็วสูงได้ หากต้องการการเคลื่อนที่ที่แม่นยำ ในขณะเดียวกัน หุ่นยนต์ทรงกลมขนาดเล็ก สามารถทำความเร็วได้สูง เพราะมีที่น้ำหนักเบา แต่ในด้านการบังคับเลี้ยวแทบจะไม่สามารถทำได้เลย เนื่องจาก น้ำหนักที่เบา และความสมดุลที่ไม่ดีเท่าที่ควร รวมถึงผิวภายนอกของทรงกลมที่ทำจากยางพารามีปุ่ม ทำให้บังคับเลี้ยวยากและไม่มีความราบเรียบในการเคลื่อนที่

5.2 ข้อเสนอแนะ

1. หุ่นยนต์ทรงกลม Sphere Robot ทั้งสามรูปแบบนั้นสามารถเพิ่มส่วนของการควบคุมความเร็วมอเตอร์ทั้ง 2 ข้างให้เท่ากันได้ เพื่อเพิ่มความสามารถในการเคลื่อนที่ให้แม่นยำมากขึ้น

2. ในส่วนของวงจรไฟฟ้า ควรเพิ่มความแข็งแรงทนทาน หรือส่วนป้องกันให้กับวงจร และส่วนประกอบอื่นๆบนวงจร ยกตัวอย่างเช่น โมดูลบลูทูธ เพราะภายในหุ่นยนต์ทรงกลมนั้น มีการเคลื่อนที่ของรถบังคับตลอดเวลา

3. มอเตอร์ที่ใช้ในการสร้างแรงขับเคลื่อนให้แก่หุ่นยนต์นั้น อาจจะเพิ่มขนาดแรงบิด หรือรอบการหมุนให้สูงมากกว่านี้ก็ได้ เพื่อการสร้างแรงขับเคลื่อนที่มากขึ้นสำหรับพื้นผิวที่มีแรงเสียดทานมากกว่าพื้นเรียบ เช่น พื้นหญ้า พื้นลาดเอียง พื้นผิวขรุขระ เป็นต้น

4. การถ่วงน้ำหนักเพื่อให้ได้จุดศูนย์ถ่วงของหุ่นยนต์นั้น ควรคำนึงถึงการรับน้ำหนักของเพลามอเตอร์ด้วย เช่น หุ่นยนต์ทรงกลมขนาดใหญ่ อาจจะใช้ขนาดเพลาของมอเตอร์ที่ใหญ่ขึ้นด้วย เพื่อรองรับน้ำหนักโดยรวมของตัวรถบังคับได้

5. ในการควบคุมการเคลื่อนที่ที่ดีขึ้น อาจจะนำทฤษฎีการสร้างแรงขับเคลื่อนอื่นๆเข้ามาร่วมด้วย เช่น ทฤษฎีลูกตุ้มเพนดูลัม เป็นต้น

บรรณานุกรม

[1] PIC Microcontroller Lerning-By-Doing ด้วยภาษา C. พิมพ์ครั้งที่ 1. กรุงเทพฯ : สมาร์ทเลิร์นนิ่ง, 2550

[2] ดอนสัน ปงผาบ. ไมโครคอนโทรลเลอร์ PIC และการประยุกต์ใช้งาน - - กรุงเทพฯ : สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น), 2550

[3] ประจิน พลังสันติกุล, PIC Works Examples and C Source Code. กรุงเทพฯ : บริษัท แอพซอฟต์เทคจำกัด, 2537

[4] ประจิน พลังสันติกุล, All About CCS C PIC C Programming with CCS C Conpiler. กรุงเทพฯ : บริษัท แอพซอฟต์เทคจำกัด, 2551

[5] พร้อมเลิศ หล่อวิจิตร, คู่มือ เขียนแอพ Android ฉบับสมบูรณ์. พิมพ์ครั้งที่ 1. กรุงเทพฯ : ซีเอ็ดยูเคชั่น, 2556

[6] Sleeping For Less. 2556. "สารบัญบทความแอนดรอยด์" [ระบบออนไลน์]. แหล่งที่มา http://www.akexorcist.com/2013/02/android-article-index.html

[7] บริษัท เอ็มเบ็ดเด็ด เทคโนโลยี จำกัด. "Altium Designer 14" [ระบบออนไลน์]. แหล่งที่มา http://www.altiumthai.com/node/209

ภาคผนวก

โปรแกรม

โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control) บนบอร์ด Arduino DUE

```
void setup()                          //ฟังก์ชั่นสำหรับตั้งค่าตัวแปรต่างๆ
 {

  Serial.begin(9600);                 //เริ่มการทำงานSerialPort baudrate9600
  Serial2.begin(9600);                //เริ่มการทำงานSerialPort2 baudrate9600
  pinMode(4, OUTPUT);                 //ตั้งค่าขา 4 เป็นเอาท์พุทของมอเตอร์ขวา
  pinMode(5, OUTPUT);                 //ตั้งค่าขา 5 เป็นเอาท์พุทของมอเตอร์ขวา
  pinMode(8, OUTPUT);                 //ตั้งค่าขา 8 เป็นเอาท์พุทของมอเตอร์ซ้าย
  pinMode(9, OUTPUT);                 //ตั้งค่าขา 9 เป็นเอาท์พุทของมอเตอร์ซ้าย

}
void loop()                           //ฟังก์ชั่นวนลูปการทำงาน
{

 if (Serial2.available())             //ถ้าพอร์ทSerial2 พร้อมใช้งาน
 {
   char toSend = (char)Serial2.read();    //อ่านค่าจากพอร์ท Serial2 มาเก็บไว้
                                          //ในตัวแปร toSend ซึ่งเป็นตัวแปรชนิด character

   Serial.print(toSend);              //ให้พอร์ทSerialแสดงค่าที่เก็บไว้ในตัวแปร toSend

   if(toSend == '1')                  //ถ้าตัวแปร toSend มีค่า '1'
   {
          drive_forward();            //เรียกฟังก์ชั่น drive_forward()
   }
   if(toSend == '2')                  //ถ้าตัวแปร toSend มีค่า '2'
   {
          drive_turnleft();           //เรียกฟังก์ชั่น drive_turnleft()
   }
   if(toSend == '3')                  //ถ้าตัวแปร toSend มีค่า '3'
   {
          drive_turnright();          //เรียกฟังก์ชั่น drive_turnright()
   }
```

```
      if(toSend == '4')        //ถ้าตัวแปร toSend มีค่า '4'
      {
            drive_backward();      //เรียกฟังก์ชั่น drive_backward()
      }
      if(toSend == '5')        //ถ้าตัวแปร toSend มีค่า '5'
      {
            motor_stop();          //เรียกฟังก์ชั่น motor_stop()
      }
    }
  }
  void drive_forward()       // ขับเคลื่อนไปด้านหน้า
  {
    digitalWrite(4,HIGH);    //ตั้งค่าขา 4 เป็นเอาท์พุท HIGH
    digitalWrite(5,LOW);     //ตั้งค่าขา 5 เป็นเอาท์พุท LOW
    digitalWrite(8,HIGH);    //ตั้งค่าขา 8 เป็นเอาท์พุท HIGH
    digitalWrite(9,LOW);     //ตั้งค่าขา 9 เป็นเอาท์พุท LOW
  }
  void drive_backward()      // ขับเคลื่อนไปด้านหลัง
  {
    digitalWrite(4,LOW);     //ตั้งค่าขา 4 เป็นเอาท์พุท LOW
    digitalWrite(5,HIGH);    //ตั้งค่าขา 5 เป็นเอาท์พุท HIGH
    digitalWrite(8,LOW);     //ตั้งค่าขา 8 เป็นเอาท์พุท LOW
    digitalWrite(9,HIGH);    //ตั้งค่าขา 9 เป็นเอาท์พุท HIGH
  }
  void drive_turnleft()      // เลี้ยวซ้าย
  {
    digitalWrite(4,HIGH);    //ตั้งค่าขา 4 เป็นเอาท์พุท HIGH
    digitalWrite(5,LOW);     //ตั้งค่าขา 5 เป็นเอาท์พุท LOW
    digitalWrite(8,LOW);     //ตั้งค่าขา 8 เป็นเอาท์พุท LOW
    digitalWrite(9,LOW);     //ตั้งค่าขา 9 เป็นเอาท์พุท LOW
  }
  void drive_turnright()     // เลี้ยวขวา
  {
    digitalWrite(4,LOW);     //ตั้งค่าขา 4 เป็นเอาท์พุท LOW
    digitalWrite(5,LOW);     //ตั้งค่าขา 5 เป็นเอาท์พุท LOW
    digitalWrite(8,HIGH);    //ตั้งค่าขา 8 เป็นเอาท์พุท HIGH
```

```
  digitalWrite(9,LOW);      //ตั้งค่าขา 9 เป็นเอาท์พุท LOW
}
void motor_stop()
{
  digitalWrite(4,LOW);      //ตั้งค่าขา 4 เป็นเอาท์พุท LOW
  digitalWrite(5,LOW);      //ตั้งค่าขา 5 เป็นเอาท์พุท LOW
  digitalWrite(8,LOW);      //ตั้งค่าขา 8 เป็นเอาท์พุท LOW
  digitalWrite(9,LOW);      //ตั้งค่าขา 9 เป็นเอาท์พุท LOW
}
```

โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง (Discrete Control) บน PIC ไมโครคอนโทรลเลอร์

```
#include <16F877A.h> //เรียกใช้ไฟล์ Header ของ PIC หมายเลข 16F876A
#fuses NOWDT //เรียกใช้ fuses clock แบบไม่เปิดการใช้งาน watchdog timer
#use delay (clock=4000000) //เรียกใช้ delay กำหนดสัญญาณนาฬิกาเป็น 4MHz
#fuses HS          //เรียกใช้ fuses clock แบบ High-speed
#define TX1 PIN_C6        //กำหนด TX1 เป็น ขา C6
#define RX1 PIN_C7        //กำหนด RX1 เป็น ขา C7
#use fast_io(b)           //เปิดการใช้งาน fast_io ที่พอร์ท B
#use rs232(baud = 9600, xmit = TX1, rcv = RX1, bits = 8, stop = 1)
          // ใช้ฟังก์ชั่น rs232 โดยกำหนด baudrate=9600, ขา TX1, RX1, จำนวน
          //  ข้อมูลการรับส่ง 8 bit(1 byte), กำหนดให้มี stop bit

void main()        //ฟังก์ชั่นหลัก
{

  char data;       //กำหนดตัวแปรชื่อ data เป็นชนิด character

  set_tris_c(0B10000000);          //กำหนดขา I/O ที่พอร์ท C
  set_tris_b(0B00000000);          //กำหนดขา I/O ที่พอร์ท B

  output_b(0b00000000);            //เคลียร์เอาท์พุทของพอร์ท B

  while(true)              //วนลูปตลอดเวลา
  {

    data = getc();         //รับข้อมูลตัวอักษรมาจาก SerialPort เก็บไว้ที่ data

    if(data == '1')        //ถ้า data = 1
    {
      output_b(0b00001001);    //สั่งการทำงานมอเตอร์ให้หมุนไปข้างหน้า
                               //หุ่นยนต์เคลื่อนที่ไปข้างหน้า
    }
    if(data == '2')        //ถ้า data = 2
    {
```

```
        output_b(0b00001000);    //สั่งการทำงานมอเตอร์ขวาอย่างเดียวให้หมุนไป
                                 //ข้างหน้า (เลี้ยวขวา)
      }
      if(data == '3')          //ถ้า data = 3
      {
        output_b(0b00000001);    //สั่งการทำงานมอเตอร์ซ้ายอย่างเดียวให้หมุนไป
                                 //ข้างหน้า (เลี้ยวซ้าย)
      }
      if(data == '4')          //ถ้า data = 4
      {
        output_b(0b00000110);    //สั่งการทำงานมอเตอร์ให้หมุนไปข้างหลัง
                                 //เคลื่อนที่ถอยหลัง
      }
      if(data == '5')          //ถ้า data = 5
      {
        output_b(0x00);    //สั่งหยุดการทำงานของมอเตอร์

      }
    }
}
```

โปรแกรมควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง (Continuous Control) บน PIC ไมโครคอนโทรลเลอร์

```
#include <16F876A.h>    //เรียกใช้ไฟล์ Header ของ PIC หมายเลข 16F876A
#fuses HS,NOWDT,NOPROTECT,NOLVP //เรียกใช้ fuses clock รูปแบบต่างๆ เช่น
        //fuses clock แบบ high-speed ,ไม่เปิดการใช้งาน
        //watchdog timer, ไม่เปิดการ
        //ป้องกันการคัดลอกโค้ด, ไม่เปิดการใช้งานการโปรแกรมมิ่งแบบ
        //Low-Voltage programming
#use delay(clock=20000000)      // เรียกใช้งานฟังก์ชั่น delay
                                //โดยกำหนดสัญญาณนาฬิกาเป็น 20MHz
#define TX1 PIN_C6      //กำหนดขา TX1 ที่ขา C6
#define RX1 PIN_C7      //กำหนดขา RX1 ที่ขา C7
#use fast_io(B)         //เปิดการใช้งาน fast_io ที่พอร์ท B
#use fast_io(C)         //เปิดการใช้งาน fast_io ที่พอร์ท C

#use rs232(baud = 9600, xmit = TX1, rcv = RX1, bits = 8, stop = 1)
        // ใช้ฟังก์ชั่น rs232 โดยกำหนด baudrate=9600, ขา TX1, RX1, จำนวน
        //  ข้อมูลการรับส่ง 8 bit(1 byte), กำหนดให้มี stop bit
#include <stdio.h>      // เรียกไฟล์ Header ชื่อ stdio.h
#include <string.h>     // เรียกไฟล์ Header ชื่อ string.h
#include <stdlib.h>     // เรียกไฟล์ Header ชื่อ stdlib.h

char BUFFER[15];        //กำหนดตัวแปร array ชื่อ BUFFER ขนาด 15 ช่อง
char DATA_1[10];        //กำหนดตัวแปร array ชื่อ DATA_1 ขนาด 10 ช่อง
char DATA_2[10];        //กำหนดตัวแปร array ชื่อ DATA_2 ขนาด 10 ช่อง
int STRLEN_DATA1;       //กำหนดตัวแปร STRLEN_DATA1 เป็น integer
int STRLEN_DATA2;       //กำหนดตัวแปร STRLEN_DATA2 เป็น integer
float DATA_1_NUM;       //กำหนดตัวแปร DATA_1_NUM เป็น float
int DATA_2_NUM;         //กำหนดตัวแปร DATA_2_NUM เป็น int
int8 MotorLeft;         //กำหนดตัวแปร MotorLeft เป็น signed 8bit integer
int8 MotorRight;        //กำหนดตัวแปร MotorRight เป็น signed 8bit integer
int BUFFER_LEN,DATA1_LEN,DATA2_LEN;

void MotorForward()     //PIN_B0,B1 = Left/ PIN_B1,B2 = Right ฟังก์ชั่นขับ
                        //มอเตอร์ไปข้างหน้า
```

```
{
   output_high(PIN_B0);   //ขา B0 เป็น HIGH
   output_low(PIN_B2);    //ขา B2 เป็น LOW
   output_high(PIN_B1);   //ขา B1 เป็น HIGH
   output_low(PIN_B3);    //ขา B3 เป็น LOW
}
void MotorBackward()      //PIN_B0,B1 = Left/ PIN_B1,B2 = Right ฟังก์ชั่นขับ
                          //มอเตอร์ไปด้านหลัง
{
   output_low(PIN_B0);
   output_high(PIN_B2);
   output_low(PIN_B1);
   output_high(PIN_B3);
}
void MotorStop()          //ฟังก์ชั่นหยุดการทำงานของมอเตอร์
{
   output_low(PIN_B0);
   output_low(PIN_B2);
   output_low(PIN_B1);
   output_low(PIN_B3);
}
void MotorCondition()     //ฟังก์ชั่นตรวจสอบเงื่อนไขการทำงานของมอเตอร์
{
  if(DATA_1_NUM==0.0)  //ถ้า DATA_1_NUM = 0.0
  {
     MotorStop();          //เรียกฟังก์ชั่นหยุดการทำงานของมอเตอร์
  }
  if(DATA_1_NUM>=1.0  &&  DATA_1_NUM<=180.0)  //ถ้ามีค่าระหว่าง  1.0-
180.0
  {
     MotorForward();       //เรียกฟังก์ชั่นสั่งมอเตอร์หมุนไปด้านหน้า
  }
  if(DATA_1_NUM>=181.0 && DATA_1_NUM<=360.0) //ถ้ามีค่าระหว่าง 181.0-
                                             //360.0
  {
     MotorBackward();      //เรียกฟังก์ชั่นสั่งมอเตอร์หมุนไปด้านหลัง
```

```
    }
  }
  void SteeringCondition()  //ฟังก์ชั่นตรวจสอบมุมองศาเพื่อสั่งควบคุมสัดส่วน
                            //ความเร็วของมอเตอร์ทั้ง 2 ข้าง
  {
    if(DATA_1_NUM>=0.0 && DATA_1_NUM<=30.0 || DATA_1_NUM>=331.0
&& DATA_1_NUM<=360.0)    //เลี้ยวขวาเต็มตัว ถ้าค่า DATA_1_NUM อยู่ระหว่าง 0.0-30.0
                         //หรือ 331.0-360.0
    {
      MotorRight = DATA_2_NUM;        //ตัวแปรมอเตอร์ขวาทำงานเต็มความเร็ว
      MotorLeft = 0;                  //ตัวแปรมอเตอร์ซ้ายหยุดการทำงาน
    }
    if(DATA_1_NUM>=31.0 && DATA_1_NUM<=60.0 || DATA_1_NUM>=301.0
&& DATA_1_NUM<=330.0)    //เลี้ยวขวา 45 องศา ถ้าค่าอยู่ระหว่าง 31.0-60.0 หรือ 301.0-
                         //330.0
    {
      MotorRight = DATA_2_NUM;        //ตัวแปรมอเตอร์ขวาทำงานเต็มความเร็ว
      MotorLeft = DATA_2_NUM/2;       //ตัวแปรมอเตอร์ซ้ายทำงานครึ่งหนึ่งของ
                                      //ความเร็ว
    }
    if(DATA_1_NUM>=61.0 && DATA_1_NUM<=90.0 || DATA_1_NUM>=271.0
&& DATA_1_NUM<=300.0)    //เคลื่อนที่ไปด้านหน้า ถ้าค่าอยู่ระหว่าง 61.0-90.0 หรือ 271.0-
                         //300.0
    {
      MotorRight = DATA_2_NUM;        //ตัวแปรมอเตอร์ขวาทำงานเต็มความเร็ว
      MotorLeft = DATA_2_NUM;         //ตัวแปรมอเตอร์ซ้ายทำงานเต็มความเร็ว
    }
    if(DATA_1_NUM>=91.0 && DATA_1_NUM<=120.0 || DATA_1_NUM>=241.0
&& DATA_1_NUM<=270.0)    //เคลื่อนที่ไปด้านหน้า ถ้าค่าอยู่ระหว่าง 91.0-120.0 หรือ 241.0-
                         //270.0
    {
      MotorRight = DATA_2_NUM;        //ตัวแปรมอเตอร์ขวาทำงานเต็มความเร็ว
      MotorLeft = DATA_2_NUM;         //ตัวแปรมอเตอร์ซ้ายทำงานเต็มความเร็ว
    }
    if(DATA_1_NUM>=121.0 && DATA_1_NUM<=150.0 ||
DATA_1_NUM>=211.0 && DATA_1_NUM<=240.0)   //เลี้ยวซ้าย 45 องศา ถ้าค่าอยู่ระหว่าง
```

```
                                        //121.0-150.0 หรือ 211.0-240.0
        {
          MotorRight = DATA_2_NUM/2;      //ตัวแปรมอเตอร์ขวาทำงานครึ่งหนึ่งของ
                                                  //ความเร็ว
          MotorLeft = DATA_2_NUM;         //ตัวแปรมอเตอร์ซ้ายทำงานเต็มความเร็ว
        }
        if(DATA_1_NUM>=151.0          &&        DATA_1_NUM<=180.0          ||
DATA_1_NUM>=181.0 && DATA_1_NUM<=210.0)   //เลี้ยวซ้ายเต็มตัว ถ้าค่าอยู่ระหว่าง
                                        //151.0-180.0 หรือ 181.0-210.0
        {
          MotorRight = 0;                 //ตัวแปรมอเตอร์ขวาหยุดการทำงาน
          MotorLeft = DATA_2_NUM;         //ตัวแปรมอเตอร์ซ้ายทำงานเต็มความเร็ว
        }
      }

      void SeperateDATA()           //ฟังก์ชั่นแยก ข้อมูลที่ได้ออกมาเป็น 2 ตัวแปร
      {
        do{                         // วนลูป Do-while ไปเรื่อยๆจนกว่าจะจบข้อความ

          if(BUFFER[BUFFER_LEN]=='@')       //ถ้าหากตัวแปร BUFFER เจอ '@'
          {
            BUFFER_LEN++;             //เลื่อนตำแหน่งใน array ไป 1 ตำแหน่ง
            do                        // เก็บค่ามุมในตัวแปร DATA_1 จนกว่าจะเจอ '|'
            {
              DATA_1[DATA1_LEN] = BUFFER[BUFFER_LEN];
              DATA1_LEN++;
              BUFFER_LEN++;

            }while(BUFFER[BUFFER_LEN]!='|');
            STRLEN_DATA1 = strlen(DATA_1); //วัดความยาวของข้อมูลใน DATA_1
            BUFFER_LEN = BUFFER_LEN+1; //เลื่อนตำแหน่งใน BUFFER ไปตำแหน่ง
                                              //ต่อไป
            do                        // เก็บค่าความเร็วใน DATA_2 จนกว่าจะเจอ '.'
            {
              DATA_2[DATA2_LEN] = BUFFER[BUFFER_LEN];
              DATA2_LEN++;
```

```
            BUFFER_LEN++;

          }while(BUFFER[BUFFER_LEN]!='.');
          STRLEN_DATA2 = strlen(DATA_2); //วัดความยาวของข้อมูลใน DATA_2
        }
      BUFFER_LEN++;          //เลื่อนตำแหน่งของ array ไปเรื่อยๆ
      }while(BUFFER[BUFFER_LEN]!='%');      // ถ้าเจอคำว่า '%' เท่ากับจบข้อความ
    }

    void ClrDATAForNextCycle()          // เคลียร์ข้อมูลสำหรับการับข้อมูลรอบต่อไป
    {
      int DATA_ptr;
      for(DATA_ptr=0;DATA_ptr<STRLEN_DATA1;DATA_ptr++) //วนลูปตามจำนวน
                                                       //ข้อมูลในตัวแปร
      {
        DATA_1[DATA_ptr] = 0;      //เคลียร์ค่า DATA_1
      }

      for(DATA_ptr=0;DATA_ptr<9;DATA_ptr++) //วนลูปตามจำนวนข้อมูลในตัวแปร
      {
        DATA_2[DATA_ptr] = 0;      //เคลียร์ค่า DATA_2
      }

      for(DATA_ptr=0;DATA_ptr<14;DATA_ptr++)//วนลูปตามจำนวนข้อมูลในตัวแปร
      {
        BUFFER[DATA_ptr] = 0;      // เคลียร์ตัวแปร BUFFER
      }

    }

    void ConvertToInt()                 //ฟังก์ชั่นแปลงข้อมูล
    {
      DATA_1_NUM = atof(DATA_1);          //แปลงตัวแปร DATA_1_NUM เป็น
                                          //float
      DATA_2_NUM = atoi(DATA_2);          //แปลงตัวแปร DATA_2 เป็น integer
```

```
}

void main()        //ฟังก์ชั่นหลัก
{

  set_tris_b(0B00000000);       // Bit 0(L),1(L),2(R),3(R) = MotorSignal
  set_tris_c(0B10000000);       //RC1, RC2 = CCP2,CCP1

  setup_ccp1(CCP_PWM);          // ตั้งค่า PWM ของมอเตอร์ซ้าย
  setup_ccp2(CCP_PWM);          // ตั้งค่า PWM ของมอเตอร์ขวา
  setup_timer_2(T2_DIV_BY_16,255,1); //ตั้งค่า timer2 สำหรับเปิดใช้งาน PWM
  set_timer2(0);                // ให้ timer2 เริ่มนับจาก 0

  while(1)                      //วนลูปตลอดเวลา
  {
      gets(BUFFER);             // รับข้อมูลจาก SerialPort เก็บไว้ใน BUFFER
      BUFFER_LEN = 0;           //เคลียร์ค่า BUFFER_LEN ให้เป็น 0
      DATA1_LEN = 0;            //เคลียร์ค่า DATA1_LEN ให้เป็น 0
      DATA2_LEN = 0;            //เคลียร์ค่า DATA2_LEN ให้เป็น 0

      SeperateDATA();           //เรียกฟังก์ชั่นแยกข้อมูล

      ConvertToInt();           //เรียกฟังก์ชั่นแยกข้อมูล
      MotorCondition();         //เรียกฟังก์ชั่นตรวจสอบเงื่อนไขการทำงานของ
                                //มอเตอร์
      SteeringCondition(); //ฟังก์ชั่นตรวจสอบมุมองศาเพื่อสั่งควบคุมสัดส่วน
                           //ความเร็วของมอเตอร์ทั้ง 2 ข้าง


      ClrDATAForNextCycle();// เคลียร์ข้อมูลสำหรับการรับข้อมูลรอบต่อไป

      set_pwm2_duty(MotorRight);//สั่งการกำเนิดสัญญาณPWMของมอเตอร์ขวา
      set_pwm1_duty(MotorLeft); //สั่งการกำเนิดสัญญาณPWMของมอเตอร์ซ้าย
  }
}
```

แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบไม่ต่อเนื่อง(Discrete Control Sphere Robot Application) บนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

โปรแกรมส่วน SplashScreen.java

```
package com.example.discretecontrolsphererobot;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

import android.app.Activity;
import android.content.Intent;
import android.os.Handler;
import android.view.Window;

public class SplashScreen extends Activity {
Handler handler;
Runnable runnable;
Long delay_time;
Long time = 3000L;

public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
      requestWindowFeature(Window.FEATURE_NO_TITLE);
        setContentView(R.layout.activity_splash_screen);
        handler = new Handler();

        runnable = new Runnable() {
       public void run() {
        Intent intent = new Intent(SplashScreen.this, firstpage.class);
                                startActivity(intent);
                                finish();
      }
  };
```

```
 }

 public void onResume() {
         super.onResume();
         delay_time = time;
    handler.postDelayed(runnable, delay_time);
    time = System.currentTimeMillis();
 }

 public void onStop() {
         super.onStop();
         handler.removeCallbacks(runnable);
          time = delay_time - (System.currentTimeMillis() - time);
         }
}
```

โปรแกรมส่วน firstpage.java

```
package com.example.discretecontrolsphererobot;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;
import app.akexorcist.bluetoothspp.BluetoothSPP;
import app.akexorcist.bluetoothspp.BluetoothSPP.BluetoothConnectionListener;
import app.akexorcist.bluetoothspp.BluetoothState;
import app.akexorcist.bluetoothspp.DeviceList;

public class firstpage extends Activity{

    BluetoothSPP bt;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button deviceList = (Button)findViewById(R.id.deviceList);
        Button button1 = (Button)findViewById(R.id.aboutPageBtn);

        button1.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {
                Intent i = new Intent(getApplicationContext(),
about.class);
                startActivity(i);
```

```
                    }
                });

                deviceList.setOnClickListener(new OnClickListener() {

                    public void onClick(View v) {
                        if(bt.getServiceState() ==
BluetoothState.STATE_CONNECTED) {
                bt.disconnect();
            } else {
                Intent intent = new Intent(getApplicationContext(), DeviceList.class);
                startActivityForResult(intent,
BluetoothState.REQUEST_CONNECT_DEVICE);
            }
                    }
                });

                bt = new BluetoothSPP(this);
                if(!bt.isBluetoothAvailable()) {
                    Toast.makeText(getApplicationContext(), "Bluetooth is not available",
Toast.LENGTH_SHORT).show();
                    finish();
                }

                bt.setBluetoothConnectionListener(new BluetoothConnectionListener()
{
        public void onDeviceConnected(String name, String address) {
            Toast.makeText(getApplicationContext(), "Connected to " + name + "\n" +
address
                    , Toast.LENGTH_SHORT).show();
        }

        public void onDeviceDisconnected() {
            Toast.makeText(getApplicationContext(), "Connection lost"
                    , Toast.LENGTH_SHORT).show();
        }
```

```
    public void onDeviceConnectionFailed() {
        Toast.makeText(getApplicationContext(), "Unable to connect"
                , Toast.LENGTH_SHORT).show();
    }
});

  }

  public void onStart() {
     super.onStart();
     if(!bt.isBluetoothEnabled()) {
        Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(intent, BluetoothState.REQUEST_ENABLE_BT);
     } else {
          if(!bt.isServiceAvailable()) {
        bt.setupService();
        bt.startService(BluetoothState.DEVICE_OTHER);
        setup();
          }
     }
  }

  private void setup() {
          Button forward = (Button)findViewById(R.id.forwardBtn);
          Button left = (Button)findViewById(R.id.leftBtn);
          Button right = (Button)findViewById(R.id.rightBtn);
          Button back = (Button)findViewById(R.id.backwardBtn);
          Button stop = (Button)findViewById(R.id.stopBtn);
          forward.setOnClickListener(new OnClickListener(){
                  public void onClick(View v) {
                          bt.send("1",true);

                  }
          });
          left.setOnClickListener(new OnClickListener(){
```

```
                public void onClick(View v) {
                        bt.send("2", true);

                }
        });
        right.setOnClickListener(new OnClickListener(){
                public void onClick(View v) {
                        bt.send("3", true);

                }
        });
        back.setOnClickListener(new OnClickListener(){
                public void onClick(View v) {
                        bt.send("4", true);

                }
        });
        stop.setOnClickListener(new OnClickListener(){
                public void onClick(View v) {
                        bt.send("5", true);

                }
        });


}

public void onDestroy() {
    super.onDestroy();
    bt.stopService();
}

public void onActivityResult(int requestCode, int resultCode, Intent data) {
    if(requestCode == BluetoothState.REQUEST_CONNECT_DEVICE) {
        if(resultCode == Activity.RESULT_OK)
            bt.connect(data);
```

```
        } else if(requestCode == BluetoothState.REQUEST_ENABLE_BT) {
            if(resultCode == Activity.RESULT_OK) {
                bt.setupService();
                bt.startService(BluetoothState.DEVICE_ANDROID);
                setup();
            } else {
                Toast.makeText(getApplicationContext(), "Bluetooth was not enabled."
                        , Toast.LENGTH_SHORT).show();
                finish();
            }
        }
    }

}
```

โปรแกรมส่วน firstpage.java

```
package com.example.discretecontrolsphererobot;

import android.app.Activity;
import android.os.Bundle;

public class about extends Activity{

        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.aboutpage);
        }

}
```

โปรแกรมส่วน activity_splash_screen.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="#420300" >

   <ImageView
      android:id="@+id/imageView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:src="@drawable/discretesphererobotlogo" />

          </RelativeLayout>
```

โปรแกรมส่วน activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.bungkeesgift.MainActivity" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/deviceList"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginLeft="20dp"
            android:text="Device list"
            android:textAlignment="center" />

        <Button
            android:id="@+id/aboutPageBtn"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="45dp"

            android:text="About"
            android:textAlignment="center" />

    </LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

    <Button
        android:id="@+id/forwardBtn"
        android:layout_width="wrap_content"
        android:layout_height="71dp"
        android:layout_weight="0.19"
        android:text="Forward"
        android:textAlignment="center"
        android:layout_marginTop="30dp"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

    <Button
        android:id="@+id/leftBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Left"
        android:textAlignment="center"
        android:layout_weight="0.5" />

    <Button
        android:id="@+id/rightBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Right"
        android:textAlignment="center"
        android:layout_weight="0.5" />

</LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

    <Button
        android:id="@+id/backwardBtn"
        android:layout_width="wrap_content"
        android:layout_height="71dp"
        android:layout_weight="0.19"
        android:text="Backward"
        android:textAlignment="center" />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <Button
        android:id="@+id/stopBtn"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:background="#FF0000"
        android:text="Stop"
        android:textAlignment="center" />

</LinearLayout>

        </LinearLayout>
```

โปรแกรมส่วน aboutpage.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#ffffff" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="15dp" >

        <LinearLayout
            android:layout_width="30dp"
            android:layout_height="match_parent"
            android:layout_marginRight="5dp"
            android:background="@drawable/applogo"
            android:orientation="vertical" >
        </LinearLayout>

        <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="#2eec71"
            android:text="Team Members"
            android:textColor="#000000"
            android:textSize="20dp"
            android:textStyle="bold" />

    </LinearLayout>

    <LinearLayout
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Miss Tanya Sreeratanawan" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Student code : 54010613" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tanyaza0528@gmail.com" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_margin="15dp" >

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Mr. Narindech Wanadecha" />

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Student code : 54010674" />

    <TextView
        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Narindech@hotmail.com" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"  >

    <TextView
        android:id="@+id/textView8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Miss Rakchanok Khamyon" />

    <TextView
        android:id="@+id/textView9"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:text="Student code : 54011066" />

        <TextView
            android:id="@+id/textView10"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="khwan.rakchanok@gmail.com" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="40dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:layout_marginTop="70dp"
        android:orientation="vertical"
        android:background="@drawable/inslogo" >


    </LinearLayout>

</LinearLayout>
```

โปรแกรมส่วน AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.discretecontrolsphererobot"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".SplashScreen"
            android:label="@string/app_name" android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="firstpage"
android:screenOrientation="portrait"></activity>
        <activity android:name="about" android:screenOrientation="portrait"></activity>
    </application>

</manifest>
```

โปรแกรมส่วน project.properties

```
# This file is automatically generated by Android Tools.
# Do not modify this file -- YOUR CHANGES WILL BE ERASED!
#
# This file must be checked in Version Control Systems.
#
# To customize properties used by the Ant build system edit
# "ant.properties", and override values to adapt the script to your
# project structure.
#
# To enable ProGuard to shrink and obfuscate your code, uncomment this (available
properties: sdk.dir, user.home):
#proguard.config=${sdk.dir}/tools/proguard/proguard-android.txt:proguard-project.txt

# Project target.
target=android-20
manifestmerger.enabled=true
android.library.reference.1=..\\appcompat_v7
android.library.reference.2=../BluetoothSPP
```

แอพพลิเคชั่นสำหรับควบคุมการเคลื่อนที่ของรถบังคับ แบบต่อเนื่อง(Continuous Control Sphere Robot Application) บนโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์

โปรแกรมส่วน SplashScreen.java

```
package com.example.newsphererobotforandroidv_43;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

import android.app.Activity;
import android.content.Intent;
import android.os.Handler;
import android.view.Window;

public class SplashScreen extends Activity {
        Handler handler;
        Runnable runnable;
        Long delay_time;
        Long time = 3000L;

        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
      requestWindowFeature(Window.FEATURE_NO_TITLE);
                setContentView(R.layout.activity_splash_screen);
                handler = new Handler();

                runnable = new Runnable() {
               public void run() {
                 Intent intent = new Intent(SplashScreen.this, ControlPage.class);
                                        startActivity(intent);
                                        finish();
               }
           };
```

```
        }

        public void onResume() {
                super.onResume();
                delay_time = time;
            handler.postDelayed(runnable, delay_time);
            time = System.currentTimeMillis();
        }

        public void onStop() {
                super.onStop();
                handler.removeCallbacks(runnable);
            time = delay_time - (System.currentTimeMillis() - time);
        }
}
```

โปรแกรมส่วน JoyStickClass.java

```
package com.example.newsphererobotforandroidv_43;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.view.ViewGroup.LayoutParams;

public class JoyStickClass {

        public static final int STICK_NONE = 0;
    public static final int STICK_UP = 1;
    public static final int STICK_UPRIGHT = 2;
    public static final int STICK_RIGHT = 3;
    public static final int STICK_DOWNRIGHT = 4;
    public static final int STICK_DOWN = 5;
    public static final int STICK_DOWNLEFT = 6;
    public static final int STICK_LEFT = 7;
    public static final int STICK_UPLEFT = 8;

    private int STICK_ALPHA = 200;
    private int LAYOUT_ALPHA = 200;
    private int OFFSET = 0;

    private Context mContext;
    private ViewGroup mLayout;
    private LayoutParams params;
    private int stick_width, stick_height;

    private int position_x = 0, position_y = 0, min_distance = 0;
```

```
private int distance – 0, angle – 0,

private DrawCanvas draw;
private Paint paint;
private Bitmap stick;

private boolean touch_state = false;

public JoyStickClass (Context context, ViewGroup layout
        , int stick_res_id)
{
    mContext = context;

    stick = BitmapFactory.decodeResource(mContext.getResources(),
            stick_res_id);

    stick_width = stick.getWidth();
    stick_height = stick.getHeight();

    draw = new DrawCanvas(mContext);
    paint = new Paint();
    mLayout = layout;
    params = mLayout.getLayoutParams();
}

public void drawStick(MotionEvent arg1)
{
    position_x = (int) (arg1.getX() - (params.width / 2));
    position_y = (int) (arg1.getY() - (params.width / 2));
    distance = (int) Math.sqrt(Math.pow(position_x, 2)
            + Math.pow(position_y, 2));
    angle = (int) cal_angle(position_x, position_y);


    if(arg1.getAction() == MotionEvent.ACTION_DOWN)
    {
```

```
        if(distance <= (params.width / 2) - OFFSET)
        {
            draw.position(arg1.getX(), arg1.getY());
            draw();
            touch_state = true;
        }
    } else if(arg1.getAction()
            == MotionEvent.ACTION_MOVE && touch_state)
    {
        if(distance <= (params.width / 2) - OFFSET)
        {
            draw.position(arg1.getX(), arg1.getY());
            draw();
        }
        else if(distance > (params.width / 2) - OFFSET)
        {
            float x = (float) (Math.cos(Math.toRadians(cal_angle
                    (position_x, position_y)))
                    * ((params.width / 2) - OFFSET));
            float y = (float) (Math.sin(Math.toRadians(cal_angle
                    (position_x, position_y)))
                    * ((params.height / 2) - OFFSET));
            x += (params.width / 2);
            y += (params.height / 2);
            draw.position(x, y);
            draw();
        }
        else
        {
            mLayout.removeView(draw);
        }
    }
    else if(arg1.getAction() == MotionEvent.ACTION_UP)
    {
        mLayout.removeView(draw);
        touch_state = false;
```

```
        }
    }

    public int[] getPosition() {
        if(distance > min_distance && touch_state) {
            return new int[] { position_x, position_y };
        }
        return new int[] { 0, 0 };
    }

    public int getX() {
        if(distance > min_distance && touch_state) {
            return position_x;
        }
        return 0;
    }

    public int getY() {
        if(distance > min_distance && touch_state) {
            return position_y;
        }
        return 0;
    }

    public float getAngle() {
        if(distance > min_distance && touch_state) {
            return Math.abs(angle);
        }
        return 0;
    }

    public float getDistance() {
        if(distance > min_distance && touch_state) {
            if(distance>255)
            {
                distance=255;
```

```
        }
        return distance;
    }
    return 0;
}

public void setMinimumDistance(int minDistance) {
    min_distance = minDistance;
}

public int getMinimumDistance() {
    return min_distance;
}

public int get8Direction() {
    if(distance > min_distance && touch_state)
    {
        if(angle >= 247.5 && angle < 292.5 ) {
            return STICK_UP;
        } else if(angle >= 292.5 && angle < 337.5 ) {
            return STICK_UPRIGHT;
        } else if(angle >= 337.5 || angle < 22.5 ) {
            return STICK_RIGHT;
        } else if(angle >= 22.5 && angle < 67.5 ) {
            return STICK_DOWNRIGHT;
        } else if(angle >= 67.5 && angle < 112.5 ) {
            return STICK_DOWN;
        } else if(angle >= 112.5 && angle < 157.5 ) {
            return STICK_DOWNLEFT;
        } else if(angle >= 157.5 && angle < 202.5 ) {
            return STICK_LEFT;
        } else if(angle >= 202.5 && angle < 247.5 ) {
            return STICK_UPLEFT;
        }
    } else if(distance <= min_distance && touch_state) {
        return STICK_NONE;
```

```
        }
      return 0;
   }

   public int get4Direction() {
      if(distance > min_distance && touch_state) {
         if(angle >= 225 && angle < 315 ) {
            return STICK_UP;
         } else if(angle >= 315 || angle < 45 ) {
            return STICK_RIGHT;
         } else if(angle >= 45 && angle < 135 ) {
            return STICK_DOWN;
         } else if(angle >= 135 && angle < 225 ) {
            return STICK_LEFT;
         }
      } else if(distance <= min_distance && touch_state) {
         return STICK_NONE;
      }
      return 0;
   }

   public void setOffset(int offset) {
      OFFSET = offset;
   }

   public int getOffset() {
      return OFFSET;
   }

   public void setStickAlpha(int alpha) {
      STICK_ALPHA = alpha;
      paint.setAlpha(alpha);
   }

   public int getStickAlpha() {
      return STICK_ALPHA;
```

```
    }

    public void setLayoutAlpha(int alpha) {
        LAYOUT_ALPHA = alpha;
        mLayout.getBackground().setAlpha(alpha);
    }

    public int getLayoutAlpha() {
        return LAYOUT_ALPHA;
    }

    public void setStickSize(int width, int height) {
        stick = Bitmap.createScaledBitmap(stick, width, height, false);
        stick_width = stick.getWidth();
        stick_height = stick.getHeight();
    }

    public void setStickWidth(int width) {
        stick = Bitmap.createScaledBitmap(stick, width, stick_height
                , false);
        stick_width = stick.getWidth();
    }

    public void setStickHeight(int height) {
        stick = Bitmap.createScaledBitmap(stick, stick_width, height
                , false);
        stick_height = stick.getHeight();
    }

    public int getStickWidth() {
        return stick_width;
    }

    public int getStickHeight() {
        return stick_height;
    }
```

```
    public void setLayoutSize(int width, int height) {
        params.width = width;
        params.height = height;
    }

    public int getLayoutWidth() {
        return params.width;
    }

    public int getLayoutHeight() {
        return params.height;
    }

    private double cal_angle(float x, float y)
    {
        if(x >= 0 && y >= 0)
            return Math.toDegrees(Math.atan(y / x)) - 360;
        else if(x < 0 && y >= 0)
            return Math.toDegrees(Math.atan(y / x)) - 180;
        else if(x < 0 && y < 0)
            return Math.toDegrees(Math.atan(y / x)) - 180;
        else if(x >= 0 && y < 0)
            return Math.toDegrees(Math.atan(y / x));
        return 0;
    }


        private void draw() {
        try {
            mLayout.removeView(draw);
        } catch (Exception e) { }
        mLayout.addView(draw);
    }

    private class DrawCanvas extends View{
```

```
        float x, y;

        private DrawCanvas(Context mContext) {
            super(mContext);
        }

        public void onDraw(Canvas canvas) {
            canvas.drawBitmap(stick, x, y, paint);
        }

        private void position(float pos_x, float pos_y) {
            x = pos_x - (stick_width / 2);
            y = pos_y - (stick_height / 2);
        }
    }
}
```

โปรแกรมส่วน ControlPage.java

```
package com.example.newsphererobotforandroidv_43;


import android.app.Activity;
import android.os.Bundle;


import android.view.MotionEvent;
import android.view.View;
import android.view.Window;
import android.view.View.OnTouchListener;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;

import android.widget.Button;
import android.view.View.OnClickListener;

import android.widget.Toast;
import app.akexorcist.bluetoothspp.BluetoothSPP;
import app.akexorcist.bluetoothspp.BluetoothSPP.BluetoothConnectionListener;
import app.akexorcist.bluetoothspp.BluetoothState;
import app.akexorcist.bluetoothspp.DeviceList;
import android.bluetooth.BluetoothAdapter;
import android.content.Intent;



public class ControlPage extends Activity {

        RelativeLayout layout_joystick;
    ImageView image_joystick, image_border;
    TextView textView1, textView2, textView3, textView4, textView5;
```

```
JoyStickClass js;
BluetoothSPP bt;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_control_page);

    Button about = (Button)findViewById(R.id.About);

    about.setOnClickListener(new OnClickListener(){
      public void onClick(View v){
            Intent i = new Intent(getApplicationContext(), About.class);
            startActivity(i);
      }
    });


    Button deviceList = (Button)findViewById(R.id.Device_list);

    deviceList.setOnClickListener(new OnClickListener() {
                    public void onClick(View v) {
                        if(bt.getServiceState() ==
BluetoothState.STATE_CONNECTED) {
            bt.disconnect();
        } else {
            Intent intent = new Intent(getApplicationContext(), DeviceList.class);
            startActivityForResult(intent,
BluetoothState.REQUEST_CONNECT_DEVICE);
        }
                    }
            });

    bt = new BluetoothSPP(this);
```

```
                if(!bt.isBluetoothAvailable()) {
                    Toast.makeText(getApplicationContext(), "Bluetooth is not available",
Toast.LENGTH_SHORT).show();
                    finish();
                }

                bt.setBluetoothConnectionListener(new BluetoothConnectionListener()
{
        public void onDeviceConnected(String name, String address) {
            Toast.makeText(getApplicationContext(), "Connected to " + name + "\n" +
address
                    , Toast.LENGTH_SHORT).show();
        }

        public void onDeviceDisconnected() {
            Toast.makeText(getApplicationContext(), "Connection lost"
                    , Toast.LENGTH_SHORT).show();
        }

        public void onDeviceConnectionFailed() {
            Toast.makeText(getApplicationContext(), "Unable to connect"
                    , Toast.LENGTH_SHORT).show();
        }
    });


    textView1 = (TextView)findViewById(R.id.Angle);
    textView2 = (TextView)findViewById(R.id.Speed);


    layout_joystick = (RelativeLayout)findViewById(R.id.layout_joystick);

    js = new JoyStickClass(getApplicationContext(), layout_joystick,
R.drawable.image_button);
    js.setStickSize(120, 120);
    js.setLayoutSize(500, 500);
```

```
        js.setLayoutAlpha(150);
        js.setStickAlpha(100);
        js.setOffset(90);
        js.setMinimumDistance(40);

        layout_joystick.setOnTouchListener(new OnTouchListener() {
            public boolean onTouch(View arg0, MotionEvent arg1) {
                js.drawStick(arg1);
                if(arg1.getAction() == MotionEvent.ACTION_DOWN
                        || arg1.getAction() == MotionEvent.ACTION_MOVE) {

                    textView1.setText("Angle : " + String.valueOf(js.getAngle()));
                    textView2.setText("Speed : " + String.valueOf(js.getDistance()));

                    bt.send(String.valueOf("@"+ js.getAngle()) + "|" +
String.valueOf(js.getDistance()) +"/"+"%" ,true);


                return true;
            }
        });
    }

    public void onStart() {
            super.onStart();
            if(!bt.isBluetoothEnabled()) {
                Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(intent, BluetoothState.REQUEST_ENABLE_BT);
            } else {
                  if(!bt.isServiceAvailable()) {
                bt.setupService();
                bt.startService(BluetoothState.DEVICE_OTHER);
                setup();
                  }
            }
        }
```

```
    private void setup() {

        }

    public void onDestroy() {
            super.onDestroy();
            bt.stopService();
        }

    public void onActivityResult(int requestCode, int resultCode, Intent data) {
            if(requestCode == BluetoothState.REQUEST_CONNECT_DEVICE) {
                if(resultCode == Activity.RESULT_OK)
                    bt.connect(data);
            } else if(requestCode == BluetoothState.REQUEST_ENABLE_BT) {
                if(resultCode == Activity.RESULT_OK) {
                    bt.setupService();
                    bt.startService(BluetoothState.DEVICE_ANDROID);
                    setup();
                } else {
                    Toast.makeText(getApplicationContext(), "Bluetooth was not enabled."
                            , Toast.LENGTH_SHORT).show();
                    finish();
                }
            }
        }

}
```

โปรแกรมส่วน AboutPage.java

```
package com.example.newsphererobotforandroidv_43;

import android.app.Activity;
import android.os.Bundle;

public class About extends Activity{


        protected void onCreate(Bundle savedInstanceState) {

                super.onCreate(savedInstanceState);
                setContentView(R.layout.about);
        }

}
```

โปรแกรมส่วน activity_splash_screen.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="#420300" >

   <ImageView
      android:id="@+id/imageView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:src="@drawable/sphererobotlogo" />

</RelativeLayout>
```

โปรแกรมส่วน activity_control_page.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="#080060" >

   <LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_alignParentLeft="true"
      android:layout_alignParentTop="true"
      android:layout_margin="10dp"
      android:background="#000000"
      android:orientation="vertical" >

      <TextView
         android:id="@+id/textView1"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:textColor="#FdFF00"
         android:layout_margin="10dp"
         android:textStyle="bold"
         android:text="SphereRobot's Continuous Control Interface" />

      <TextView
         android:id="@+id/Angle"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Angle :"
         android:textColor="#FFFFFF"
         android:layout_marginLeft="5dp"
         android:textSize="20dp"
         android:textStyle="bold" />
```

```
<TextView
    android:id="@+id/Speed"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Speed :"
    android:textColor="#FFFFFF"
    android:layout_marginLeft="5dp"
    android:textSize="20dp"
    android:textStyle="bold" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#000000" >

    <Button
        android:id="@+id/Device_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Bluetooth list"
        android:textSize="15dp"
        android:textColor="#00dfff"
        android:textStyle="bold" />

    <Button
        android:id="@+id/About"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="About"
        android:textSize="15dp"
        android:textColor="#FFFF4444"
        android:textStyle="bold" />
```

```
        </LinearLayout>

    </LinearLayout>


    <RelativeLayout
        android:id="@+id/layout_joystick"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_below="@+id/linearLayout1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="50dp"
        android:background="@drawable/image_button_bg" >
    </RelativeLayout>


</RelativeLayout>
```

โปรแกรมส่วน about.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#ffffff" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="15dp" >

        <LinearLayout
            android:layout_width="30dp"
```

```
        android:layout_height="match_parent"
        android:layout_marginRight="5dp"
        android:background="@drawable/applogo"
        android:orientation="vertical" >
    </LinearLayout>

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#2eec71"
        android:text="Team Members"
        android:textColor="#000000"
        android:textSize="20dp"
        android:textStyle="bold" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Miss Tanya Sreeratanawan" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:text="Student code : 54010613" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tanyaza0528@gmail.com" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_margin="15dp" >

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Mr. Narindech Wanadecha" />

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Student code : 54010674" />

    <TextView
        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Narindech@hotmail.com" />

</LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"  >

    <TextView
        android:id="@+id/textView8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textStyle="bold"
        android:text="Miss Rakchanok Khamyon" />

    <TextView
        android:id="@+id/textView9"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Student code : 54011066" />

    <TextView
        android:id="@+id/textView10"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="khwan.rakchanok@gmail.com" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="40dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:layout_marginTop="70dp"
    android:orientation="vertical"
```

```
        android:background="@drawable/inslogo" >


    </LinearLayout>

</LinearLayout>
```

โปรแกรมส่วน AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.newsphererobotforandroidv_43"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="21" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
        <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".SplashScreen"
            android:label="@string/app_name" android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```
        </activity>
        <activity android:name="ControlPage"
 android:screenOrientation="portrait"></activity>
        <activity android:name="About" android:screenOrientation="portrait"></activity>
    </application>

</manifest>
```

โปรแกรมส่วน project.properties

```
# This file is automatically generated by Android Tools.
# Do not modify this file -- YOUR CHANGES WILL BE ERASED!
#
# This file must be checked in Version Control Systems.
#
# To customize properties used by the Ant build system edit
# "ant.properties", and override values to adapt the script to your
# project structure.
#
# To enable ProGuard to shrink and obfuscate your code, uncomment this (available
properties: sdk.dir, user.home):
#proguard.config=${sdk.dir}/tools/proguard/proguard-android.txt:proguard-project.txt

# Project target.
target=android-20
manifestmerger.enabled=true
android.library.reference.1=..\\appcompat_v7
android.library.reference.2=..\\appcompat_v7
android.library.reference.3=../BluetoothSPP
```