



Goggle : People Video Analytics and Deep Learning Platform

– ชื่อภาษาไทย –

นายปฐมพงศ์ สินธุจาม

นายศุภกร เบญจวิกรัย

นายอุกฤษฎ์ เลิศวรรณาการ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมหุ่นยนต์และระบบอัตโนมัติ

สถาบันวิทยาการหุ่นยนต์ภาคนาม

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ปีการศึกษา 2562





Goggle : People Video Analytics and Deep Learning Platform

– ชื่อภาษาไทย –

นายปฐมพงศ์ สินธุจงม

นายศุภกร เบญจวิกรัย

นายอุกฤษฎ์ เลิศวรรณาการ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมหุ่นยนต์และระบบอัตโนมัติ

สถาบันวิทยาการหุ่นยนต์ภาคนาม

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ปีการศึกษา 2562

- ชื่อภาษาไทย -

นายปฐมพงศ์ สินธุจาม

นายศุภกร เบญจวิกรัชย์

นายอุกฤษฎ์ เลิศวรรณาการ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาชีวกรรมหุ่นยนต์และระบบอัตโนมัติ

สถาบันวิทยาการหุ่นยนต์ภาคสนาม

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ปีการศึกษา 2562

---

คณะกรรมการสอบวิทยานิพนธ์

ประธานกรรมการสอบวิทยานิพนธ์

(ดร.อาบทิพย์ ชีรวงศ์กิจ)

อาจารย์ที่ปรึกษาวิทยานิพนธ์

(นายธนัชชา ชูพจน์เจริญ)

อาจารย์ที่ปรึกษาวิทยานิพนธ์

(รศ.ดร.ชิต เหล่าวัฒนา)

กรรมการสอบวิทยานิพนธ์

(ดร.ปิติวุฒิ ชีรากิตติกุล)

กรรมการสอบวิทยานิพนธ์

(ดร.สุรชัย วงศ์บุณย์ยง)

ลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ชื่อวิทยานิพนธ์	- ชื่อภาษาไทย -
หน่วยกิต	6
ผู้เขียน	นายปฐมพงศ์ สินธุจิต นายศุภกร เบญจวิกรัย นายอุตุษฐ์ เลิศวรรณการ
อาจารย์ที่ปรึกษา	ที่ปรึกษาวิทยานิพนธ์หลัก นายธนัชชา ชูพจน์เจริญ ที่ปรึกษาวิทยานิพนธ์ร่วม รศ.ดร.ชิต เหล่าวัฒนา
หลักสูตร	วิศวกรรมศาสตรบัณฑิต
สาขาวิชา	วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
คณะ	สถาบันวิทยาการหุ่นยนต์ภาคสนาม
ปีการศึกษา	2562

---

## บทคัดย่อ

งานวิทยานิพนธ์นี้เป็นงานที่เกี่ยวกับการออกแบบและจัดทำแพลตฟอร์มหุ่นยนต์อิเล็กทรอนิกส์ด้วยเครื่องพิมพ์สามมิติ โดยใช้ชื่อว่า หุ่นยนต์อิเล็กทรอนิกส์ UTHAI และจุดประสงค์เพื่อให้นักวิจัยท่านอื่นสามารถนำไปพัฒนาต่อได้ง่าย ภาพรวมของวิทยานิพนธ์นี้จะแบ่งออกเป็นทั้งหมดสามส่วน คือ ส่วนแรกเป็นส่วนของการออกแบบและจัดสร้าง ส่วนสองของหุ่นยนต์อิเล็กทรอนิกส์ ส่วนที่สองเป็นส่วนของการพัฒนาโปรแกรมที่ใช้ในระบบด้วย ROS และ ส่วนสุดท้ายเป็นส่วนที่ออกแบบระบบพื้นฐานสำหรับการพัฒนาหุ่นยนต์อิเล็กทรอนิกส์ รวมไปถึงมีเอกสาร คู่มือที่อยู่ในรูปแบบออนไลน์

คำสำคัญ : แพลตฟอร์มหุ่นยนต์อิเล็กทรอนิกส์ / ระบบพื้นฐานหุ่นยนต์ / ROS / เครื่องพิมพ์สามมิติ

## กิตติกรรมประกาศ

ขอขอบพระคุณอาจารย์ รนชชา ชูพจน์เจริญ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก ที่ได้สละเวลามาให้คำปรึกษา ชี้แนะแนวทาง ให้ความรู้ในด้านต่างๆ ที่จำเป็นต่องานวิจัย รวมถึงการให้การสนับสนุนในเรื่องอุปกรณ์ในการทำวิจัย ตลอดจนช่วยตรวจสอบและแก้ไขวิทยานิพนธ์ให้เป็นไปอย่างสมบูรณ์

ขอขอบพระคุณรองศาสตราจารย์ ดร.ชิต เหล่าวัฒนา อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่ได้ชี้แนะแนวทางให้คำแนะนำ และให้เกียรติเข้าร่วมการสอบวิทยานิพนธ์

ขอขอบพระคุณอาจารย์ ดร.ภิวดา มณีวรรณ และนายวิษณุ จุราวี ที่ได้ให้คำแนะนำในการแก้ไขปัญหาด้านต่างๆ ที่เกิดขึ้นระหว่างการทำวิจัย และได้ให้การสนับสนุนอุปกรณ์สำคัญที่ใช้ในการทำวิจัย

ขอขอบพระคุณอาจารย์ อับพิพิญ ธิรวงศ์กิจ ที่กรุณาให้เกียรติเป็นประธานกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการจัดทำวิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณอาจารย์ ดร.ปิติวุฒย์ ธีรกิตติกุล ที่กรุณาให้เกียรติเป็นกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการวิจัย และการแก้ไขปรับปรุงงานวิจัย ตลอดจนตรวจแก้วิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณอาจารย์ ดร.สุภาชัย วงศ์บุณย์ยง ที่กรุณาให้เกียรติเป็นกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการวิจัย และการแก้ไขปรับปรุงงานวิจัย ตลอดจนตรวจแก้วิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณคณาจารย์ และบุคลากรในสถาบันวิทยาการหุ่นยนต์ภาควิชานามทุกท่าน ที่ได้ให้คำปรึกษาและช่วยเหลือด้านสถานที่พร้อมทั้งสิ่งอำนวยความสะดวกต่างๆ ในระหว่างการทำวิทยานิพนธ์

ขอขอบคุณนักศึกษาปริญญาตรี สถาบันวิทยาการหุ่นยนต์ภาควิชานามทุกท่าน ที่ได้ให้คำแนะนำ ถ้ามี แล้วเป็นกำลังใจมาโดยตลอด

และสุดท้ายนี้ ขอน้อมรำลึกถึงพระคุณบิดา มารดา และครอบครัว ที่ส่งเสริมให้กำลังใจ และให้การสนับสนุนในเรื่องต่างๆ จนกระทั่งข้าพเจ้าประสบความสำเร็จในการศึกษา

นายปฐมพงศ์ สินธุรงาม  
นายศุภกร เบญจวิกรัย  
นายอุกฤษฎ์ เลิศวรณากุล

## สารบัญ

เรื่อง	หน้า
บทคัดย่อ .....	๑
กิตติกรรมประกาศ .....	๗
สารบัญ .....	๗
รายการรูปภาพ .....	๗
รายการตาราง .....	๘
รายการสัญลักษณ์ .....	๙
ประมวลศัพท์และตัวย่อ .....	๙
<b>บทที่ ๑ บทนำ .....</b>	<b>๑</b>
1.1 ทีมและความสำคัญ .....	๑
1.2 วัตถุประสงค์ .....	๑
1.3 ประโยชน์ที่คาดว่าจะได้รับ .....	๑
1.4 ขอบเขตการดำเนินงาน .....	๒
1.5 ภาพรวมของระบบและขั้นตอนการดำเนินงาน .....	๒
<b>บทที่ ๒ ทฤษฎีและหลักการ .....</b>	<b>๓</b>
2.1 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง .....	๓
2.1.1 หุ่นยนต์อิมานอยด์ .....	๓
2.1.2 ทฤษฎีที่เกี่ยวข้องกับมนุษย์ .....	๗
2.1.3 ทฤษฎีที่เกี่ยวข้องกับหุ่นยนต์อิมานอยด์ .....	๙
2.1.4 ตัวอย่างหุ่นยนต์อิมานอยด์ .....	๑๒
2.2 การออกแบบโครงสร้างของหุ่นยนต์ .....	๑๗
2.2.1 ความแตกต่างระหว่างโครงสร้างของมนุษย์กับโครงสร้างของหุ่นยนต์ .....	๑๗
2.2.2 อุปกรณ์ที่ใช้ในหุ่นยนต์อิมานอยด์ .....	๑๘
2.3 การออกแบบโปรแกรมด้วย ROS .....	๒๑
2.3.1 ระบบที่ใช้ช่วยในการพัฒนาหุ่นยนต์ .....	๒๑
2.3.2 ระบบที่ใช้ในการจำลองการทำงานของหุ่นยนต์ .....	๒๔
2.3.3 Robot Operating System .....	๒๗

## สารบัญ (ต่อ)

เรื่อง	หน้า
2.4 การออกแบบระบบพื้นฐาน .....	33
2.4.1 ความแตกต่างของ Operating Systems.....	33
2.4.2 ข้อแตกต่างระหว่าง Open platform กับ Non-open platform.....	33
2.4.3 มาตรฐานหน่วยวัดและการอภิปรัชต.....	33
2.4.4 Robot Operating System.....	34
บทที่ 3 ระเบียบวิธีวิจัย .....	35
3.1 หน้าที่ความรับผิดชอบ.....	35
3.2 แผนการดำเนินงาน .....	35
3.3 การออกแบบแอพพลิเคชัน labeling tool .....	36
3.3.1 แอพพลิเคชัน labeling tool .....	36
3.4 การออกแบบระบบวิเคราะห์การกรวยทำของมนุษย์.....	40
3.4.1 กำหนดพิกัดเฟรมให้กับหุ่นยนต์อิมานอยด์.....	40
3.4.2 การแปลงข้อมูลให้อยู่ในรูปแบบ URDF .....	41
3.4.3 โครงสร้างการติดต่อสื่อสารระหว่าง Node ใน ROS .....	44
เอกสารอ้างอิง.....	49
ภาคผนวก ก ข้อมูลเบื้องต้นของหุ่นยนต์อิมานอยด์ UTHAI.....	50
ก.1 ค่าคุณสมบัติทางพลศาสตร์.....	50
ประวัติผู้เขียน .....	63

## รายการรูปภาพ

รูป	หน้า
รูปที่ 2.1 แสดงความแตกต่างของหุ่นยนต์ชีวามโนยด์แต่ละประเภท.....	3
รูปที่ 2.2 Honda asimo โดย Kazou Hirai .....	5
รูปที่ 2.3 วัสดุการเดินของมนุษย์.....	7
รูปที่ 2.4 ส่วนประกอบของหุ่นยนต์ชีวามโนยด์ .....	9
รูปที่ 2.5 วัสดุการเดินของหุ่นยนต์ชีวามโนยด์.....	9
รูปที่ 2.6 การควบคุมตำแหน่งของจุดรวมมวลให้อยู่ในพื้นที่ฐาน .....	10
รูปที่ 2.7 การควบคุมตำแหน่งของจุดโมเมนต์ศูนย์ให้ตรงกับแรงปฏิกิริยารวม.....	11
รูปที่ 2.8 หุ่นยนต์ชีวามโนยด์ปือปี้ .....	12
รูปที่ 2.9 หุ่นยนต์ชีวามโนยด์ไอคัพ .....	13
รูปที่ 2.10 หุ่นยนต์ชีวามโนยด์ดาร์วิน .....	14
รูปที่ 2.11 หุ่นยนต์ชีวามโนยด์นิโอะ .....	15
รูปที่ 2.12 หุ่นยนต์ชีวามโนยด์ราบอท .....	16
รูปที่ 2.13 ตัวอย่างตำแหน่งและการหมุนของข้อต่อของหุ่นยนต์เพื่อการอ้างอิง .....	17
รูปที่ 2.14 ตัวขับเคลื่อนที่ใช้ในหุ่นยนต์ชีวามโนยด์ .....	18
รูปที่ 2.15 ตัวประมวลผลระดับสูงของ Thormang Humanoid .....	19
รูปที่ 2.16 ตัวประมวลผลระดับต่ำของ Robotis OP3 Humanoid.....	19
รูปที่ 2.17 ลักษณะโครงสร้างของตัวตรวจจับแรงกด FSR .....	20
รูปที่ 2.18 เช่นเชอร์วัดความเฉียบ .....	20
รูปที่ 2.19 player project middleware .....	21
รูปที่ 2.20 yarp middleware.....	21
รูปที่ 2.21 urbi middleware .....	22
รูปที่ 2.22 miro middleware .....	22
รูปที่ 2.23 openrdk middleware.....	22
รูปที่ 2.24 ROS middleware Rviz .....	23
รูปที่ 2.25 ROS algitecture .....	23
รูปที่ 2.26 ROS Moveit .....	23
รูปที่ 2.27 ผลลัพธ์จากการใช้โปรแกรม USARSim .....	24

## รายการรูปภาพ (ต่อ)

รูป	หน้า
รูปที่ 2.28 ผลลัพธ์จากการใช้โปรแกรม MuRoSimF .....	24
รูปที่ 2.29 ผลลัพธ์จากการใช้โปรแกรม V-REP .....	25
รูปที่ 2.30 V-REP จำลองสายการผลิต.....	25
รูปที่ 2.31 Mobile robot with Gazebo.....	26
รูปที่ 2.32 Quadrotor with Gazebo.....	26
รูปที่ 2.33 ตัวอย่างสถานะปัจจุบันของ ROS.....	27
รูปที่ 2.34 ตัวอย่างไฟล์ package.xml.....	29
รูปที่ 2.35 ตัวอย่างการแสดงผลใน rqt .....	31
รูปที่ 2.36 ตัวอย่างการแสดงผลใน RViz .....	32
รูปที่ 2.37 ตัวอย่างหุ่นยนต์ชีวมโนยด์ Poppy .....	32
รูปที่ 2.38 การตั้งแgnตามกฎหมายขวากฎมือขวา .....	33
รูปที่ 3.1 ภาพรวมระบบของแอพพลิเคชัน labeling tool.....	36
รูปที่ 3.2 หน้าต่างแบบ Select ของแอพพลิเคชัน labeling tool.....	37
รูปที่ 3.3 หลังจากตัดส่วนวิดีโอแล้ว คีย์เฟรมจะถูกเก็บไว้ในช่องรายการตามประเภท .....	38
รูปที่ 3.4 ตัวอย่าง link ใน urdf .....	42
รูปที่ 3.5 การอธิบาย link ใน URDF ไฟล์ .....	42
รูปที่ 3.6 ตัวอย่าง joint ใน urdf .....	43
รูปที่ 3.7 การอธิบาย Joint ใน URDF ไฟล์ .....	43
รูปที่ 3.8 การติดต่อสื่อสารระหว่าง Node .....	44
รูปที่ ก.1 ภาพแสดงช่วงล่างทั้งตัว .....	50
รูปที่ ก.2 ภาพแสดงก้านต่อ Right Hip Yaw .....	51
รูปที่ ก.3 ภาพแสดงก้านต่อ Left Hip Yaw .....	52
รูปที่ ก.4 ภาพแสดงก้านต่อ Right Hip Roll .....	53
รูปที่ ก.5 ภาพแสดงก้านต่อ Left Hip Roll .....	54
รูปที่ ก.6 ภาพแสดงก้านต่อ Right Hip Pitch .....	55
รูปที่ ก.7 ภาพแสดงก้านต่อ Left Hip Pitch .....	56
รูปที่ ก.8 ภาพแสดงก้านต่อ Right Knee Pitch .....	57
รูปที่ ก.9 ภาพแสดงก้านต่อ Left Knee Pitch .....	58

## รายการรูปภาพ (ต่อ)

รูป	หน้า
รูปที่ ก.10 ภาพแสดงก้านต่อ Right Ankle Pitch .....	59
รูปที่ ก.11 ภาพแสดงก้านต่อ Left Ankle Pitch.....	60
รูปที่ ก.12 ภาพแสดงก้านต่อ Right Ankle Roll.....	61
รูปที่ ก.13 ภาพแสดงก้านต่อ Left Ankle Roll .....	62

## รายการตาราง

ตาราง	หน้า
ตารางที่ 2.1 ความสามารถในการหมุนของแต่ละข้อต่อของมนุษย์.....	8
ตารางที่ 2.2 ตัวอย่างชื่อและข้อมูลของ Message .....	28
ตารางที่ 2.3 ตารางแสดงหน่วยวัดมาตราฐาน.....	34
ตารางที่ 3.1 Message Geometry Point .....	44
ตารางที่ 3.2 Message Geometry Quaternion .....	45
ตารางที่ 3.3 Message Geometry Pose.....	45
ตารางที่ 3.4 Message Geometry Vector3.....	45
ตารางที่ 3.5 Message Geometry Twist .....	45
ตารางที่ 3.6 Message Navigation Odometry .....	45
ตารางที่ 3.7 Message Geometry Pose2D.....	46
ตารางที่ 3.8 Message Navigation Path.....	46
ตารางที่ 3.9 Message Geometry PoseStamped.....	46
ตารางที่ 3.10 Message Trajectory JointTrajectory.....	47
ตารางที่ 3.11 Message Trajectory JointTrajectoryPoint.....	47
ตารางที่ 3.12 Message Sensor JointState .....	47
ตารางที่ 3.13 Message Geometry Wrench.....	47
ตารางที่ 3.14 Message Sensor Imu .....	48
ตารางที่ 3.15 Message Sensor MegneticField .....	48
ตารางที่ ก.1 ตารางแสดงค่าพารามิเตอร์ทั้งตัว.....	50
ตารางที่ ก.2 ตารางแสดงค่าพารามิเตอร์ Right Hip Yaw.....	51
ตารางที่ ก.3 ตารางแสดงค่าพารามิเตอร์ Left Hip Yaw .....	52
ตารางที่ ก.4 ตารางแสดงค่าพารามิเตอร์ Right Hip Roll.....	53
ตารางที่ ก.5 ตารางแสดงค่าพารามิเตอร์ Left Hip Roll .....	54
ตารางที่ ก.6 ตารางแสดงค่าพารามิเตอร์ Right Hip Pitch .....	55
ตารางที่ ก.7 ตารางแสดงค่าพารามิเตอร์ Left Hip Pitch.....	56
ตารางที่ ก.8 ตารางแสดงค่าพารามิเตอร์ Right Knee Pitch.....	57
ตารางที่ ก.9 ตารางแสดงค่าพารามิเตอร์ Left Knee Pitch .....	58

## รายการตาราง (ต่อ)

ตาราง	หน้า
ตารางที่ ก.10 ตารางแสดงค่าพารามิเตอร์ Right Ankle Pitch .....	59
ตารางที่ ก.11 ตารางแสดงค่าพารามิเตอร์ Left Ankle Pitch.....	60
ตารางที่ ก.12 ตารางแสดงค่าพารามิเตอร์ Right Ankle Roll.....	61
ตารางที่ ก.13 ตารางแสดงค่าพารามิเตอร์ Left Ankle Roll .....	62

## รายการสัญลักษณ์

$\theta$	เซ็ต้า
$d$	distance
kg	Kilogram
$m^2$	Square Metre

## ประมวลศัพท์และตัวย่อ

UTHAI	Universal Template for Humanoid Algorithm Interface
ROS	Robot Operating System
IMU	Inertial Measurement Unit
Dof	Degree of Freedom
CoM	Center of Mass
ZMP	Zero Moment Point
PLA	Polylactic acid
ABS	Acrylonitrile butadiene styrene
KMUTT	King Mongkut's University of Technology Thonburi
Liws	ลูกิวส์ โซลูชันส์ ทรัพย์
$\theta$	เชิงตัว

## บทที่ 1

### บทนำ

#### 1.1 ที่มาและความสำคัญ

บริษัท เพอเซ็ปท์รา ดำเนินธุรกิจเกี่ยวกับด้าน artificial intelligence service โดยลูกค้านั้นมีความต้องการที่จะให้ทางบริษัทสร้างปัญญาประดิษฐ์(artificial intelligence) เพื่อนำไปใช้งานหรือแก้ปัญหาที่ต่างกันออกไป ทำการสร้างปัญญาประดิษฐ์ (artificial intelligence) เพื่อตอบสนองความต้องการของลูกค้าเหล่านั้นต้องมีข้อมูลที่เหมาะสมกับปัญหานั้นๆ เช่น ร้านขายของแห่งหนึ่งต้องการรู้ว่าในแต่ละวันมีลูกค้าเดินเข้าร้านกี่คน เป็นผู้ชายกี่คน เป็นผู้หญิงกี่คน เป็นต้น ซึ่งการจะได้ข้อมูลที่เหมาะสมกับงานนั้น ต้องใช้มนุษย์ในการสร้างขึ้นมาโดยการเก็บข้อมูลวิดีโอ และสร้าง label สำหรับใช้ในการสร้างโมเดล machine learning ด้วยตัวเอง ถ้าหากวีดีโอดูเป็นจำนวนมาก การที่จะใช้มนุษย์ในการสร้าง label นั้นอาจจะต้องใช้มนุษย์เป็นจำนวนมาก หรือ ก่อให้เกิดภาระแก่มนุษย์ อีกทั้งการสร้าง label

นั้นเป็นงานที่ลำบาก และน่าเบื่อ ทางคณะผู้วิจัยจึงมีความต้องการที่จะออกแบบ และพัฒนา video analytics platform ที่มีเครื่องมือในการสร้าง label สำหรับวิดีโอ เพื่อช่วยแบ่งเบาภาระของผู้พัฒนาในการสร้าง label เพื่อนำไปสร้างโมเดล machine learning สำหรับใช้แก้ปัญหาที่ลูกค้าต้องการ โดยโครงการสหกิจนี้เน้นศึกษาการวิเคราะห์และจัดจำการกระทำของมนุษย์จากภาพเคลื่อนไหวเป็นหลัก

#### 1.2 วัตถุประสงค์

- เพื่อออกแบบ และ สร้างระบบที่สามารถตรวจจับมนุษย์ และจัดจำการกระทำพื้นฐานของมนุษย์ภายในสำนักงาน ประกอบด้วย ยืน นั่ง ใช้คอมพิวเตอร์ เล่นโทรศัพท์ เดิน กินข้าว โดยใช้ปัญญาประดิษฐ์มาประมวลผลกับวิดีโอ
- เพื่อพัฒนาเครื่องมือในการทำ video labeling ใน การสร้างข้อมูลที่ใช้สร้างโมเดลจากวิดีโอ ให้สามารถทำได้ง่าย และ มีประสิทธิภาพที่สูงกว่าเครื่องมือตัวอื่นในปัจจุบัน

#### 1.3 ประโยชน์ที่คาดว่าจะได้รับ

- พัฒนาเครื่องมือในการทำ labeling โดยมี artificial intelligence เข้ามาช่วย ที่สามารถสร้าง label ที่สามารถนำไปใช้สร้างโมเดล machine learning ได้
- พัฒนาต้นแบบของ video analytics platform ที่สามารถรับวิดีโอเข้ามาในระบบแล้วสร้างรายงานเกี่ยวกับกิจกรรมของมนุษย์ในวิดีโอด้วย
- สร้างและทดสอบโมเดลสำหรับทำ action recognition อย่างน้อย 2 โมเดล

## 1.4 ขอบเขตการดำเนินงาน

1. Labeling tool สามารถตัดวิดีโอเฉพาะในช่วงเวลาที่มีมนุษย์อยู่ได้อัตโนมัติ
2. Labeling tool สามารถระบุตำแหน่งได้ว่ามนุษย์แต่ละคนในวิดีโอด้วยตัวเองได้ ประกอบด้วยการทำได้แก่ ยืน นั่ง ใช้คอมพิวเตอร์ เล่นโทรศัพท์ เดิน กินข้าว
3. Label ผลลัพธ์ที่ได้จาก labeling tool ต้องสามารถนำไปใช้ในการสร้างโมเดลต่อได้
4. พัฒนา Labeling tool ด้วยภาษา Python
5. พัฒนา Labeling tool ที่สามารถให้มนุษย์ทำงานแก้ไขได้ เมื่อระบบอัตโนมัติทำงานผิดพลาด
6. สร้างโมเดลสำหรับการทำ action recognition อย่างน้อย 2 โมเดลที่สามารถระบุการกระทำของมนุษย์ ตามที่กำหนดไว้ได้ เพื่อนำไปใช้ใน video analytics platform
7. Video analytics platform ต้องสามารถนำวิดีโอมามีเคราะห์ข้อมูลการกระทำและตำแหน่งของมนุษย์ แต่ละคนได้ และนำข้อมูลเหล่านั้นไปสร้างรายงานออกมาได้
8. ความละเอียดอย่างต่ำของวิดีโอต้องมากกว่า  $640 \times 480$  (ยาว x สูง)
9. วิดีโอจะต้องมีเฟรมเรท (fps) อย่างต่ำ 24 fps

## 1.5 ภาพรวมของระบบและขั้นตอนการดำเนินงาน

งานวิจัยนี้การดำเนินงานวิจัยถูกแบ่งออกเป็นสองส่วน คือ ส่วนที่หนึ่งส่วนเครื่องมือสำหรับการเตรียมชุดข้อมูล (dataset) เป็นส่วนที่ทำเครื่องมือสำหรับช่วยผู้พัฒนาในการสร้างชุดข้อมูล และส่วนที่สองนำชุดข้อมูลไปสร้างโมเดล

ศึกษาค้นคว้าเอกสารและงานวิจัยที่เกี่ยวข้อง

- ศึกษาเกี่ยวกับการวิเคราะห์ผลวิดีโอ (video analytics)
- ศึกษาเกี่ยวกับชุดข้อมูลสำหรับการวิเคราะห์ผลวิดีโอ
- ศึกษาเกี่ยวกับโมเดลใช้ในการวิเคราะห์ผลวิดีโอ
- ศึกษาเครื่องมือที่ใช้สำหรับช่วยสร้างชุดข้อมูล

### 1) ส่วนเครื่องมือสำหรับการเตรียมชุดข้อมูล (dataset)

- ออกแบบหน้าต่างของแอพพลิเคชัน
- สร้างระบบของแอพพลิเคชัน
- ทดสอบการทำงานของแอพพลิเคชัน

### 2) ส่วนนำชุดข้อมูลไปสร้างโมเดล

- สร้างชุดข้อมูลสำหรับสร้างโมเดล
- สร้างโมเดลสำหรับการทำนายการกระทำของมนุษย์
- ทดสอบการทำงานของโมเดล

## บทที่ 2

### ทฤษฎีและหลักการ

#### 2.1 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

##### 2.1.1 หุ่นยนต์ชีวามโนยด์

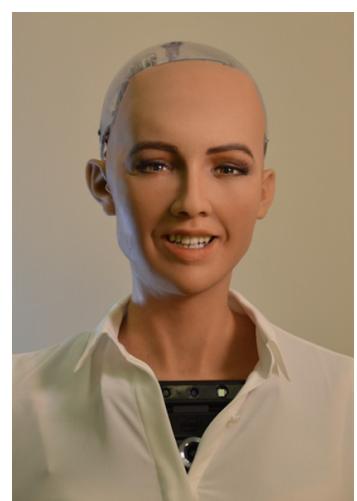
หุ่นยนต์ชีวามโนยด์ คือ หุ่นยนต์ที่ลูกสร้างขึ้นมาให้มีรูปร่างคล้ายคลึงกับสรีระโครงสร้างของมนุษย์ มักถูกออกแบบขึ้นมาเพื่อจุดประสงค์เฉพาะอย่าง เช่น การใช้เครื่องมือต่างๆของมนุษย์ การอยู่ในสภาพแวดล้อมของมนุษย์ การศึกษาการเคลื่อนไหวของร่ายกายมนุษย์ หรือเพื่อวัตถุประสงค์อื่นๆ โดยทั่วไปแล้ว หุ่นยนต์ชีวามโนยด์ประกอบไปด้วย 4 ส่วนคือ หัว ลำตัว เแขน และขา แต่การสร้างหุ่นยนต์ชีวามโนยด์นั้นก็ไม่จำเป็นที่จะต้องมีส่วนประกอบทุกส่วนดังที่กล่าวไว้ ในบางครั้งอาจมีเพียงแค่ส่วนบนเท่านั้น ดังรูปที่ 2.1ก<sup>1</sup> หุ่นยนต์นี้มาจากสถาบันวิทยาการหุ่นยนต์ภาคสนาม เป็นหุ่นยนต์ที่มีส่วนบนเหมือนมนุษย์ แต่มีส่วนล่างเป็นล้อ หรือหุ่นยนต์ชีวามโนยด์ที่มีเพียงแค่ส่วนล่าง ดังรูปที่ 2.1ข<sup>2</sup> หุ่นยนต์สัมภาระ เป็นหุ่นยนต์ชีวามโนยด์ที่มีเพียงแค่ส่วนขาเท่านั้น หรือหุ่นยนต์ชีวามโนยด์ที่มีเพียงใบหน้าเหมือนมนุษย์ ดังรูปที่ 2.1ค<sup>3</sup> หุ่นยนต์โซเฟีย เป็นแอนดรอยด์ที่มีหน้าตาคล้ายมนุษย์มาก มีตา มีปาก สามารถพูดปฏิสัมพันธ์กับมนุษย์ได้



(ก) หุ่นยนต์ประชาสัมพันธ์โรงแรม



(ข) หุ่นยนต์เดินสองขาสัมภาระ



(ค) หุ่นยนต์แอนดรอยด์โซเฟีย

รูปที่ 2.1: แสดงความแตกต่างของหุ่นยนต์ชีวามโนยด์แต่ละประเภท

<sup>1</sup> คนไทยสุดเจ๊!!สร้างหุ่นยนต์ 'น้องโน้ม' ทำหน้าที่ต้อนรับแทนคน ทั้งไทย-พูดหลายภาษา,<https://www.thairath.co.th/content/523340>

<sup>2</sup> หุ่นยนต์ชีวามโนยด์เดินสองขาสัมภาระ,<http://www.fibo.kmutt.ac.th/fiboweb2015/>หุ่นยนต์ชีวามโนยด์เดิน

<sup>3</sup> ชา อุติ อาราเบีย มอบ สัญชาติ ให้ กับ หุ่น ยนต์ ครั้ง แรก ของ โลก “Sophia” สาว อัจฉริยะ,<https://www.ensurecommunication.com/2017/11/02/ชาอุติอาราเบียมอบสัญชาติ>

งานวิจัยทางด้านหุ่นยนต์อิวามานอยด์จากอดีตจนถึงปัจจุบันมีการพัฒนาความสามารถของการเดินของหุ่นยนต์ เช่น เริ่มต้นจากแรกสุดจะเป็นการพัฒนาให้หุ่นยนต์สามารถเดินหน้าได้ ต่อมาเกิดเพิ่มความสามารถให้หุ่นยนต์สามารถเดินบนพื้นอิเยิง พื้นชูชูระ เดินเลี้ยวซ้ายขวา เดินขึ้นลงบันได ฯลฯ เป็นต้น นอกจากนี้ยังมีการพัฒนาปรับปรุงสมดุลของการเดินแบบสองขาอีกด้วย สมดุลของการเดินสามารถแบ่งได้สองแบบหลัก คือ การเดินแบบสมดุลสถิต และการเดินแบบสมดุลพลวัต งานในยุคแรกนั้นจะพัฒนาให้เดินได้แบบสมดุลสถิต ต่อมาเป็นสมดุลกึ่งพลวัต และเป็นสมดุลพลวัต การพัฒนาด้วยความคุ้มการเดินของหุ่นยนต์ จำเป็นที่จะต้องใช้ความรู้ทางด้านกลศาสตร์ ค่อนข้างมาก มีการใช้สมการที่มีความซับซ้อน

Zheng และคณะ (1988) พัฒนาหุ่นยนต์สองขาที่สามารถเดินบนพื้นราบได้ ให้สามารถเดินต่อเนื่องไปบนพื้นอิเยิงได้ด้วย พื้นอิเยิงที่ใช้มีลักษณะเป็นพื้นอิเยิงขึ้น หุ่นยนต์ที่ใช้งานนี้มีข้อต่อสะโพก (hip), ข้อเท้า (ankle) และลำตัว (torso) มีเซนเซอร์วัดแรงกด (force sensor) ติดตั้งอยู่ที่ปลายเท้าและสันเท้าแต่ละข้างเพื่อใช้วัดตำแหน่งของน้ำหนักโดยรวม (center of gravity) ของหุ่นยนต์ การเดินของงานวิจัยจะพิจารณาเฉพาะการเดินในแนวหน้าหลัง โดยมีหลักการคือ การเดินบนพื้นอิเยิงโดยที่หุ่นยนต์ยังเดินในท่าทางเหมือนกับตอนที่เดินบนพื้นราบจะทำให้น้ำหนักโดยรวมของหุ่นยนต์เคลื่อนไปข้างหลัง ดังนั้นการที่หุ่นยนต์ขยับลำตัวไปด้านหน้าจะทำให้น้ำหนักโดยรวมของหุ่นยนต์กลับมาอยู่ตรงกลางของพื้นที่รับน้ำหนักเหมือนเดิม ซึ่งจะทำให้หุ่นยนต์มีความสมดุลได้ด้วยการเดินบนพื้นราบเป็นแบบสมดุลสถิตและการเดินบนพื้นอิเยิงก็ยังคงเป็นแบบสมดุลสถิตเช่นกัน

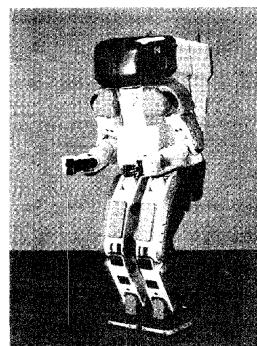
Inaba<sup>4</sup> และคณะ (1995) สร้างหุ่นยนต์เลียนแบบลิง (ape-like biped) ประกอบด้วยสองมือและสองขา มีการเดินแบบสมดุลสถิต งานวิจัยนี้มีความคิดว่าหากการทำให้หุ่นยนต์สองขาเดินได้โดยไม่ล้มแล้ว ควรจะทำหุ่นยนต์ที่สามารถลุกขึ้นเองได้หลังจากที่ล้มแล้วด้วย ดังนั้นในงานนี้ หุ่นยนต์ถูกพัฒนาให้สามารถเดิน เมื่อล้มแล้ว ก็สามารถพลิกตัวและลุกขึ้นมาเดินให้ได้

Kun และ Miller<sup>5</sup> (1996) ได้นำโครงข่ายประสาทเทียม มาประยุกต์ใช้ในการปรับเปลี่ยนท่าทางการเดิน โดยอัตโนมัติของหุ่นยนต์สองขา การที่หุ่นยนต์สามารถปรับเปลี่ยนท่าทางได้โดยอัตโนมัตินี้มีประโยชน์ทำให้หุ่นยนต์เดินได้บนพื้นผิวหลากหลายลักษณะมากขึ้น ในงานนี้พิจารณาทั้งสมดุลในแนวหน้าหลัง (sagittal plane) และแนวซ้ายขวา (frontal plane) และการเดินของหุ่นยนต์เป็นแบบสมดุลพลวัต หลักการทำงานประกอบด้วยตัวสร้างท่าทางการเดินหนึ่งตัว และตัวปรับท่าทางการเดินทั้งแนวหน้าหลังและซ้ายขวาอีกหนึ่งตัว โดยค่าการปรับเปลี่ยนนั้นจะได้มาจากการที่ใช้ในงานนี้ไปใช้กับการเดินของหุ่นยนต์อีกตัว (Kun and Miller, 1997)

<sup>4</sup>Yuki Asano\*, Kei Okada and Masayuki Inaba

<sup>5</sup>Modelling of Walking Humanoid Robot With Capability of Floor Detection and Dynamic Balancing Using Colored Petri Net, Saeid Pashazadeh and Saeed Saeedvand

Hirai<sup>6</sup> และคณะ (1998) พัฒนาหุ่นยนต์ชีวมานอยด์ ซึ่งตัวหุ่นยนต์มีความคล้ายมนุษย์มาก สามารถเดินได้อย่างราบรื่นคล้ายมนุษย์มากที่สุด เช่น สามารถเดินได้ในพื้นผิวนิ่มต่างๆ เดินได้บนพื้นอิฐขึ้นลง เดินขึ้นลงบันไดได้ เดินเข็นรถได้ เป็นต้น การเดินในทุกสถานการณ์เป็นการเดินแบบสมดุลพลวัต หุ่นยนต์สามารถเดินได้ด้วยความเร็วสูงสุด 4.7 กิโลเมตรต่อชั่วโมง หุ่นยนต์ประกอบไปด้วย แขนขาละ 9 องศาอิสระ ขาละ 6 องศาอิสระ ที่ปรับรีเควนหัวมีกล้องติดตั้งอยู่ 4 ตัว นอกจากนี้ยังมีอุปกรณ์ที่ใช้ในการรักษาสมดุลอื่นๆ อีกได้แก่ IMU ที่ติดตั้งบริเวณลำตัว และ Force sensor ที่ติดที่เท้าทั้งสองข้าง



รูปที่ 2.2: Honda asimo โดย Kazou Hirai

องค์ประกอบของหุ่นยนต์ทั่วไปจะประกอบไปด้วยระบบการตอบสนองต่างๆที่เป็นระบบ ซึ่งเราสามารถจำแนกออกเป็นส่วนหลักๆได้สามส่วนคือ ส่วนการรับรู้ ส่วนการประมวลผล และส่วนการขับเคลื่อน ทั้งหมด เมื่อนำมารวมเข้าด้วยกันแล้ว เราสามารถที่จะควบคุมการทำงานของหุ่นยนต์ชีวมานอยด์ได้

#### การรับรู้ของหุ่นยนต์ชีวมานอยด์

การรับรู้ของหุ่นยนต์ชีวมานอยด์นั้นมีความยากมากกว่าหุ่นยนต์ชนิดอื่นๆเนื่องจาก หุ่นยนต์ชีวมานอยด์เกิดจากการนำก้านต่อหularyาชิ้นเข้ามาเขื่อมต่อกันด้วยข้อต่อ ทำให้หุ่นยนต์ชีวมานอยด์สามารถเคลื่อนไหวเป็นท่าทางต่างๆได้ และไม่มีส่วนใดถูกตึงยืดติดกับพื้นโลก ซึ่งทำให้เราไม่สามารถที่จะอ้างอิงท่าทางของหุ่นยนต์ได้ จึงจำเป็นที่จะต้องเพิ่มส่วนของการรับรู้เข้าไปเพื่อช่วยแก้ปัญหาในส่วนนี้ เช่นเซอร์ที่เพิ่มเข้าไปมีหลากหลายชนิด และแต่ละชนิดก็ทำหน้าที่แตกต่างกัน ยกตัวอย่างเช่น เซนเซอร์เอนโคดเดอร์ที่ใช้สำหรับอ่านสถานะตำแหน่งและความเร็วของข้อต่อได้ เช่นเซอร์หน่วยวัดความเฉื่อยที่ใช้สำหรับหามุมเอียงของตัวหุ่นยนต์ และเซนเซอร์วัดแรงที่ฝ่าเท้าที่จะช่วยในการบอกว่าเท้าของหุ่นยนต์สัมผัสพื้นหรือไม่ เป็นต้น

<sup>6</sup>Kazuo Hirai, (1999) "The Honda humanoid robot: development and future perspective", Industrial Robot: An International Journal, Vol. 26 Issue: 4, pp.260-266, <https://doi.org/10.1108/01439919910277431>

### การประมวลผลของหุ่นยนต์ชีวามโนยด์

ในปัจจุบันนี้หน่วยประมวลผลของหุ่นยนต์ชีวามโนยด์มีประสิทธิภาพเพียงพอต่อการควบคุมหุ่นยนต์ชีวามโนยด์แบบเรียลไทม์ได้ การประมวลผลนั้นสามารถที่จะแบ่งออกเป็นหลายๆส่วนได้ ยกตัวอย่างเช่น

หุ่นยนต์ชีวามโนยด์ Thormang ได้แบ่งตัวประมวลผลเป็นตัวประมวลควบคุมสำหรับสั่งการตัวขับเคลื่อน ตัวประมวลผลควบคุมสำหรับอ่านสถานะตัวรับรู้ และตัวประมวลผลควบคุมภายนอกสำหรับคำนวนท่าทางการเดินและการวางแผน

หุ่นยนต์ชีวามโนยด์ Robotis OP3 ได้แบ่งตัวประมวลผลเป็นตัวประมวลผลระดับสูงสำหรับคำนวนท่าทางการเคลื่อนไหว และตัวประมวลผลระดับล่างสำหรับสั่งการตัวขับเคลื่อนและอ่านสถานะตัวรับรู้

หุ่นยนต์ชีวามโนยด์ Poppy ไม่ได้แบ่งตัวประมวลผล แต่ใช้เพียงตัวเดียวในการสั่งการตัวขับเคลื่อนอ่านสถานะตัวรับรู้ และประมวลผลการคำนวนทั้งหมด

นอกจากการประมวลผลและควบคุมแล้ว ยังรวมไปถึงหน่วยแสดงผลที่สามารถนำค่าสถานะต่างๆจากหุ่นยนต์ชีวามโนยด์ออกไปสร้างเป็นกราฟ หรือแบบจำลองสามมิติได้อีกด้วย

### การขับเคลื่อนของหุ่นยนต์ชีวามโนยด์

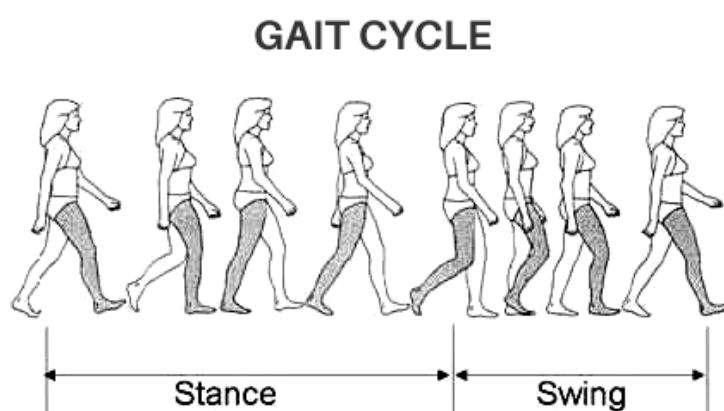
การที่หุ่นยนต์ชีวามโนยด์สามารถที่จะทำท่าทางต่างๆได้นั้น เกิดจากการเปลี่ยนแปลงตำแหน่งของข้อต่อตั้งนั้นการที่หุ่นยนต์จะขับเคลื่อนข้อต่อได้นั้น ก็จึงจำเป็นที่จะต้องมีตัวขับเคลื่อนที่ข้อต่อ โดยที่ไปแล้วเราจะติดมอเตอร์เซอร์โวไว้ที่ตำแหน่งเดียวกับข้อต่อเลย เหตุผลที่นิยมใช้มอเตอร์เซอร์โวนี้องจาก เซอร์โวสามารถที่จะควบคุมตำแหน่งและความเร็วได้

## 2.1.2 ทฤษฎีเกี่ยวกับมนุษย์

### 2.1.2.1 การวิเคราะห์การเดินของมนุษย์

การเคลื่อนที่ของหุ่นยนต์ชีวามโนยดั้นจะเลียนแบบจากการเดินของมนุษย์ ดังนั้นการวิเคราะห์ลักษณะการเดินของมนุษย์ จะเป็นการศึกษาเพื่อทำความเข้าใจถึงธรรมชาติการเดิน ก่อนนำไปทำการออกแบบกลไกทางกลและระบบควบคุมของหุ่นยนต์ชีวามโนยด้วย การก้าวเดินของมนุษย์โดยปกติแล้ว จะมีลักษณะเป็นวัฏจักร วนซ้ำไปเรื่อยๆ ในทิศทางที่ต้องการจนกว่าจะทำการหยุดเดิน การทรงตัวในระหว่างการยืนหรือการเดินนั้น เป็นไปตามสัญชาตญาณซึ่งเกิดจากการรักษาความสมดุลของร่างกายบนพื้นที่<sup>7</sup> ส่งสัญญาณผ่านเส้นประสาทไปยังกล้ามเนื้อส่วนต่างๆ ที่ทำหน้าที่ให้เกิดการเคลื่อนที่

การเคลื่อนที่ของมนุษย์ในการเดินไปข้างหน้าสามารถแบ่งออกเป็นช่วงๆ ดังนี้



รูปที่ 2.3: วัฏจักรการเดินของมนุษย์

1. ช่วงเริ่มการวางเท้าเพื่อเข้าสู่ช่วงเริ่มต้นเหวี่ยงเท้า เป็นช่วงที่เท้าเกิดการกระแทกลงบนพื้นหลังจากทำการเหวี่ยงมาจากด้านหลัง โดยธรรมชาติมนุษย์จะทำการวางสันเท้าลงเพื่อลดแรงกระแทกที่เกิดขึ้นในช่วงนี้ ดังนั้นทางกายภาพในส่วนของสันเท้ามนุษย์จึงมีลักษณะอ่อนนุ่ม
2. ช่วงเริ่มต้นเหวี่ยงเท้าเพื่อเข้าสู่ช่วงเหวี่ยงเท้า หลังจากทำการวางสันเท้าลงกับพื้นแล้ว ข้อเข้าจะปรับมุมเพื่อให้ฝ่าเท้าแนวพื้นสนิท ขณะเดียวกันขาอีกข้างจะยกสูงขึ้นเพื่อถ่ายเทน้ำหนักไปยังเท้าที่เพิ่งวางลง
3. ช่วงเหวี่ยงเท้า เป็นช่วงที่ขาหนีงยกอย้อยู่ในอากาศและขาที่วางแนบกับพื้นจะรองรับน้ำหนักทั้งหมดของร่างกาย
4. ช่วงเตรียมการวางเท้า เป็นช่วงที่ขาที่วางอยู่เหวี่ยงไปข้างหน้าเพื่อเตรียมเข้าสู่ช่วงรองรับ ในขณะเดียวกันขาที่รับน้ำหนักอยู่จะทำการผลักตัวเพื่อเริ่มทำการถ่ายเทน้ำหนักไปข้างหน้า

<sup>7</sup>Rose, J. and Gamble, J., 1993, Human Walking, Williams & Wilkins, Philadelphia, pp. 10-44.

### 2.1.2.2 การวิเคราะห์องศาสอิสระของมนุษย์

การที่มนุษย์สามารถเคลื่อนที่ได้เป็นผลเนื่องมาจากการเคลื่อนที่ของข้อต่อต่างๆ ซึ่งประกอบไปด้วย ข้อต่อส่วนสะโพก ส่วนหัวเข่า และส่วนข้อเท้า แรงบิดที่เกิดขึ้นของแต่ละข้อต่อมีความสัมพันธ์ต่อกัน ส่งผลให้เกิดเสถียรภาพในการเดินของมนุษย์ เมื่อวิเคราะห์ลักษณะโครงสร้างในแต่ละส่วน พบว่าข้อต่อส่วนสะโพกมีลักษณะเป็นทรงกลม ทำให้ข้อต่อส่วนสะโพกสามารถหมุนได้ 3 องศาอิสระ ส่วนหัวเข่าของมนุษย์มีจุดต่อของข้อที่มีลักษณะเป็นทรงกลม สองลูกประกอบเข้าด้วยกันทำให้การเคลื่อนที่ถูกบังคับให้สามารถเคลื่อนที่ได้เพียง 1 องศาอิสระ ในส่วนของข้อเท้ามีลักษณะการเคลื่อนที่เหมือนสะโพกคือสามารถเคลื่อนที่ได้ 3 องศาอิสระ

จากทั้งหมดที่ได้ทำการวิเคราะห์มาข้างต้นพบว่าในขาหนึ่งข้างของมนุษย์ประกอบด้วย 7 องศาสอิสระ ซึ่งส่งผลให้การเคลื่อนที่ของมนุษย์มีความคล่องแคล่วสูง แต่ในทางออกแบบกลไกการเดินและการควบคุม ของหุ่นยนต์สองขาถือว่ามีจำนวนองศาสอิสระเกินความจำเป็นในการเคลื่อนที่บนปริภูมิ (space) และยากต่อการควบคุม (underactuated) ดังนั้นการกำหนดจำนวนองศาสอิสระเพื่อให้หุ่นยนต์เดินได้สมดุลมนุษย์จึงมีผลในการออกแบบกลไกทางกลและการควบคุมของหุ่นยนต์สองขา

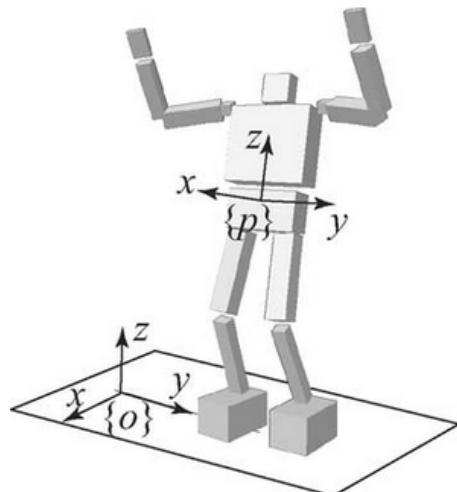
ข้อต่อ	องศาสอิสระ	องศาสาการหมุน	
		สูงสุด	ต่ำสุด
หัว	$\theta_x$	+60	-30
	$\theta_y$	+70	-70
	$\theta_z$	+80	-80
หลัง	$\theta_x$	+30	-30
	$\theta_y$	+55	-55
	$\theta_z$	+45	-45
หัวไหล่	$\theta_x$	+180	-80
	$\theta_y$	+45	-135
	$\theta_z$	+30	0
ศอก	$\theta_x$	0	-155
สะโพก	$\theta_x$	+120	-40
	$\theta_y$	+40	-50
	$\theta_z$	+60	-50
หัวเข่า	$\theta_x$	0	-130
ข้อเท้า	$\theta_x$	+30	-60
	$\theta_y$	+45	-20
	$\theta_z$	+20	-60

ตารางที่ 2.1: ความสามารถในการหมุนของแต่ละข้อต่อของมนุษย์

ผู้เขียนได้ข้อสรุปในการออกแบบขาหนึ่งข้างของหุ่นยนต์ให้มีองศาสอิสระเท่ากับ 6 องศาสอิสระ และได้ใช้ติดต่อโลเซอร์ไวของบริษัท Robotis เป็นตัวขับเคลื่อนข้อต่อ เนื่องจากภายในเซอร์ไวมีตัวรับรู้สถานะของตัวเอง และเซอร์ไวน์ถูกออกแบบมาให้สามารถติดตั้ง และสั่งการได้ง่าย

### 2.1.3 ทฤษฎีเกี่ยวกับหุ่นยนต์ชีวมโนยด์

#### 2.1.3.1 ส่วนประกอบของหุ่นยนต์ชีวมโนยด์

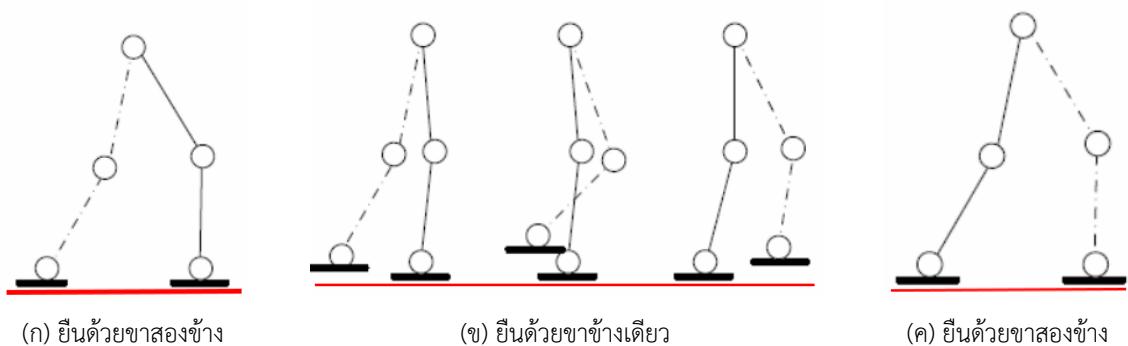


รูปที่ 2.4: ส่วนประกอบของหุ่นยนต์ชีวมโนยด์

หุ่นยนต์ชีวมโนยด์ประกอบด้วยก้านต่อห้ายๆ ก้านที่นำมาต่อกัน ลักษณะโครงสร้างนั้นจะเป็นแบบโซ่อเปิด (Open kinematic chain) และแต่ละก้านต่อจะเชื่อมต่อกันด้วยข้อต่อแบบหมุน เราสามารถแบ่งโครงสร้างของหุ่นยนต์ชีวมโนยด์ออกเป็นส่วนหลักๆ เป็น 2 ส่วน ส่วนแรกคือ ส่วนก้านต่อของลำตัวหุ่นยนต์ (Torso) ซึ่งเราสามารถที่จะรวมไปถึงส่วนแขนกับหัวด้วย และในส่วนที่สองคือ ส่วนก้านต่อของขาหุ่นยนต์ (Legs) ซึ่งเป็นส่วนของหุ่นยนต์ทั้งสองข้างที่สามารถนำไปที่สัมผัสกับพื้นได้ ทั้งสองก้านต่ออนีกูกะเชื่อมต่อกันด้วยส่วนของสะโพก (Hip) ที่อยู่ระหว่างส่วนลำตัวกับส่วนของขาหุ่นยนต์ ดังรูปที่ 2.4

#### 2.1.3.2 วิธีการเดินของหุ่นยนต์ชีวมโนยด์

วิธีการเดินของหุ่นยนต์ คือ การที่หุ่นยนต์จะต้องมีการถ่ายน้ำหนักไปมาระหว่างเท้าซ้ายและเท้าขวา มีบางช่วงที่น้ำหนักตกลงบนเท้าข้างใดข้างหนึ่งหรือทั้งสองข้างพร้อมกัน สามารถแบ่งออกเป็นช่วงได้สองช่วง คือ ช่วงการยืนด้วยขาข้างเดียว และช่วงการยืนด้วยขาทั้งสองข้าง



รูปที่ 2.5: วิธีการเดินของหุ่นยนต์ชีวมโนยด์

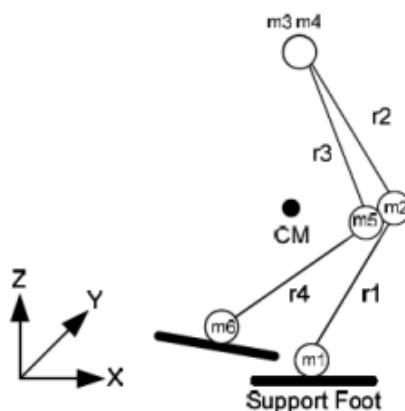
1) การยืนด้วยขาข้างเดียว : เป็นช่วงที่มีเท้าของทุ่นยนต์สัมผัสพื้นเพียงข้างเดียว ส่วนเท้าอีกข้างของทุ่นยนต์จะถูกยกกลอยจากพื้น โดยที่ไม่มีส่วนใดๆของขาข้างนั้นสัมผัสกับพื้นเลย ช่วงนี้จะเกิดขึ้นเมื่อมีการแกว่งเท้าจากข้างหลังไปข้างหน้า ดังรูปที่ 2.5x

2) การยืนด้วยขาสองข้าง : เป็นช่วงที่เท้าทั้งสองข้างของทุ่นยนต์สัมผัสกับพื้น ช่วงนี้จะเกิดตั้งแต่ทุ่นยนต์วางเท้าขณะที่สันเท้าแตะกับพื้น ไปจนถึง ปลายเท้าของขาอีกข้างหลุดออกจากพื้น

การเดินได้โดยไม่ล้มนั้น ตัวทุ่นยนต์จะต้องรักษาสมดุลของการเดินให้ได้ตลอดช่วงเวลาของการเดิน ซึ่งสมดุลของการเดินแบบสองขาสามารถแบ่งตามลักษณะการเดินและการถ่ายน้ำหนักได้เป็น 2 รูปแบบหลัก คือ การเดินแบบสมดุลสถิต (static balance walking) และ การเดินแบบสมดุลพลวัต (dynamic balance walking)

#### 2.1.3.3 การสร้างและควบคุมการเดินแบบสมดุลสถิต

การเดินของทุ่นยนต์ในลักษณะนี้ จุดศูนย์กลางมวล (CoM) ของตัวทุ่นยนต์จะไม่มีการเคลื่อนไหวออกนอกบริเวณฐานรับน้ำหนัก (Supporting Area) ตลอดช่วงเวลาการเดิน ไม่ว่าจะเป็นช่วงเวลาที่รับน้ำหนักด้วยเท้าข้างเดียวหรือทั้งสองข้างก็ตาม หมายความว่า โครงสร้างของทุ่นยนต์จะไม่ล้มแน่นอน เนื่องจากการสร้างรูปแบบการเดินด้วยวิธีนี้จะควบคุมให้ตำแหน่งของจุดศูนย์กลางมวล อยู่ภายใต้พื้นที่ฐานรับน้ำหนักของทุ่นยนต์ตลอดเวลา ??



รูปที่ 2.6: การควบคุมตำแหน่งของจุดรวมมวลให้อยู่ในพื้นที่ฐาน

ข้อดีของการสร้างและควบคุมการเดินของทุ่นยนต์ด้วยวิธีนี้คือ สามารถสร้างรูปแบบการเดินได้โดยที่มีความซับซ้อนไม่มากนัก สามารถสั่งให้ทุ่นยนต์หยุดค้างในท่าทางใดๆก็ได้ตลอดเวลาโดยทุ่นยนต์ไม่ล้ม ทุ่นยนต์ที่มีฝ่าเท้าใหญ่จะทำให้ง่ายต่อการก้าวเดินมากขึ้น นอกจากการควบคุมการก้าวขาแล้วอาจเพิ่มการควบคุมส่วนลำตัวเพิ่มเติม เพื่อเป็นการเพิ่มเสถียรภาพในการเดินและการถ่ายเท้น้ำหนัก โดยที่อาจจะมีการเพิ่มเซนเซอร์วัดแรงที่ฝ่าเท้าเพื่อตรวจสอบการกระจายแรงกดที่ฝ่าเท้า เพื่อตรวจสอบว่าตำแหน่งของจุดรวมน้ำหนักอยู่บนพื้นที่ฝ่าเท้าหรือไม่ หรือเพื่อตรวจสอบเสถียรภาพของการเดินเพื่อแก้ไขท่าทางการเดินไม่ให้เกิดการล้ม

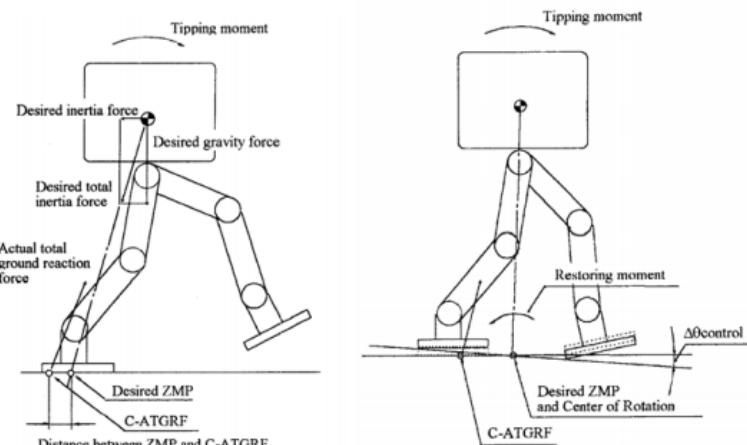
ข้อเสียของการควบคุมการเดินด้วยวิธีนี้คือ ทุ่นยนต์จะใช้เวลาในการก้าวเดินมาก ใช้พลังงานในการเดินมากกว่าการเดินแบบสมดุลพลวัต และท่าทางที่ได้จะมีความแตกต่างจากท่าทางการเดินของมนุษย์

#### 2.1.3.4 การสร้างและควบคุมการเดินแบบสมดุลพลวัต

การสร้างรูปแบบการเดินและควบคุมการเดินในลักษณะนี้ท่าทางการเดินของทุ่นยนต์นั้นจะคล้ายกับการเดินของมนุษย์มากกว่าแบบสถิต เนื่องจากมีหลักการในการสร้างท่าทางที่เหมือนกับการเดินของมนุษย์ซึ่งมีขั้นตอนดังนี้คือ เอียงตัวให้ล้มไปในทิศทางที่ต้องการเดิน เมื่อเริ่มเกิดการล้มขึ้นทุ่นยนต์จะเปลี่ยนตำแหน่งการวางเท้าไปยังตำแหน่งใหม่ เพื่อปรับให้โครงสร้างเข้าสู่ภาวะสมดุลอีกรั้ง

โดยธรรมชาติแล้วมนุษย์มีการถ่ายน้ำหนักในขณะที่เคลื่อนที่หรือยืนอยู่กับที่เพื่อรักษาสมดุลของท่าทางนั้นไว้ แต่หากการถ่ายโอนน้ำหนักนั้นเกิดสภาวะไม่สมดุล ร่างกายจะปรับสภาพโดยการเคลื่อนตำแหน่งของเท้าซึ่งเป็นพื้นที่ฐานออกจากเดิมไปยังตำแหน่งใหม่ เพื่อรักษาสมดุลไว้ หลักการดังกล่าวถูกนำมาใช้กับการควบคุมการเดินของหุ่นยนต์อิวามานอยด์ ในขณะที่หุ่นยนต์กำลังเคลื่อนไหว ผลจากแรงเฉือนของการเคลื่อนที่และผลจากแรงดึงดูดของโลกมีผลต่อการเพิ่มและลดความเร่งให้การเดินของหุ่นยนต์ แรงเหล่านี้เรียกว่าแรงเฉือนของการเคลื่อนที่ และเมื่อเท้าหุ่นยนต์สัมผัสกับพื้นจะได้รับผลกระทบของแรงนี้ เรียกว่า แรงปฏิกิริยาจากพื้น

การตัดกันระหว่างแรงปฏิกิริยาจากพื้นและแนวแรงเฉือนรวม ตำแหน่งนั้นหากทำให้โมเมนต์เท่ากับศูนย์เรียกจุดตัดนี้ว่าจุดโมเมนต์ศูนย์ ( $ZMP_{robot}$ ) และจุดที่แรงปฏิกิริยาลงสู่พื้นว่า จุดปฏิกิริยาพื้นฐาน ท่าทางการเดินของหุ่นยนต์จะถูกกำหนดและถูกส่งให้กับชุดควบคุมข้อต่อจุดต่างๆ ของหุ่นยนต์ โดยให้สอดคล้องกับแรงเฉือนรวมที่เกิดขึ้นจากการคำนวณ เรียกว่าแรงเฉือนรวม เป้าหมาย และจุดโมเมนต์ศูนย์ที่ได้จากการคำนวณเรียกว่าจุดโมเมนต์ศูนย์เป้าหมาย ( $ZMP_{target}$ ) เมื่อหุ่นยนต์เกิดสมดุลในขณะที่ทำการเดินได้อย่างสมบูรณ์ แนวแกนของแรงเฉือนรวมเป้าหมายและแรงปฏิกิริยาที่พื้นจะเป็นตำแหน่งเดียวกัน แต่ในขณะที่หุ่นยนต์เดินผ่านพื้นผิวที่มีความชรุขระหรือไม่เรียบตำแหน่งสองจุดดังกล่าว จะไม่ใช่ตำแหน่งเดียวกันทำให้หุ่นยนต์เกิดการล้มได้ แรงที่ทำให้เกิดการล้มนี้เกิดจากตำแหน่งของจุดโมเมนต์ศูนย์และตำแหน่งแรงปฏิกิริยารวมที่พื้นไม่ตรงกัน ซึ่งเป็นสาเหตุหลักที่ทำให้เกิดความไม่สมดุลขึ้น และเมื่อหุ่นยนต์เสียสมดุลระบบที่จะสามารถป้องกันการล้มและทำให้หุ่นยนต์เดินต่อไปได้อย่างต่อเนื่องคือ ระบบควบคุมแรงปฏิกิริยา ระบบควบคุมจุดโมเมนต์ศูนย์ และระบบควบคุมการวางแผนเท้า??



รูปที่ 2.7: การควบคุมตำแหน่งของจุดโมเมนต์ศูนย์ให้ตรงกับแรงปฏิกิริยารวม

อย่างไรก็ตาม การสร้างท่าทางการเดินในลักษณะนี้ต้องใช้สมการในการคำนวณที่ซับซ้อนมาก เนื่องจากต้องหาความสัมพันธ์ระหว่างองค์ประกอบหลายส่วน เช่น น้ำหนักของโครงสร้างในแต่ละส่วน แรงบิดที่แต่ละข้อต่อ และโมเมนต์โดยรวมของระบบ นอกจากนี้ยังต้องใช้อุปกรณ์การตรวจวัดต่างๆ เช่น เชเซอร์วัดแรง เชเซอร์วัดมุม เชเซอร์วัดแรงบิด ติดตั้งตามจุดต่างๆ ของโครงสร้างเพื่อวัดค่าอุกมา ก่อนที่จะทำการคำนวณตำแหน่ง และสร้างท่าทางการเดินของหุ่นยนต์อิวามานอยด์ ท่าทางการเดินที่ได้จากการควบคุมด้วยวิธีนี้ จะมีความคล้ายคลึงกับท่าทางการเดินของมนุษย์มาก

### 2.1.3.5 จุดศูนย์กลางมวลของหุ่นยนต์

หากต้องการให้หุ่นยนต์สามารถที่จะทรงตัวอยู่ได้โดยไม่ล้มนั้น จึงต้องรู้ตำแหน่งจุดศูนย์กลางมวลของหุ่นยนต์ตลอดเวลา และต้องให้จุดศูนย์กลางมวลพยายามตกลงบริเวณฐานรับน้ำหนักของหุ่นยนต์โดยหากพื้นที่ฝ่าเท้าสัมผัสกับพื้น วิธีการนี้เป็นวิธีการทางสถิตศาสตร์

#### 2.1.4 ตัวอย่างหุ่นยนต์ชีวามโนยด์

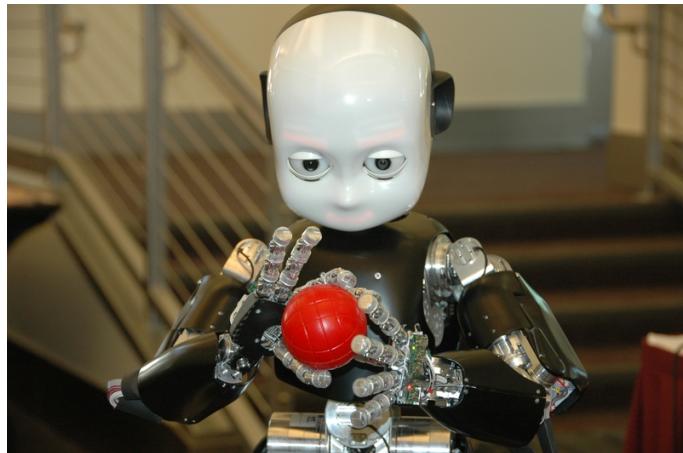
Poppy Humanoid



รูปที่ 2.8: หุ่นยนต์ชีวามโนยด์ปีอปปี้

หุ่นยนต์ชีวามโนยด์ปีอปปี้ ถูกสร้างขึ้นมาเพื่อใช้ในงานศิลปะ การวิจัยและการศึกษาโดยเฉพาะ หุ่นยนต์ปีอปปี้ประกอบด้วยส่วนของฮาร์ดแวร์และซอฟแวร์ที่เปิดเป็นโอเพนซอร์ซให้ผู้ที่สนใจสามารถเข้ามาศึกษาได้ โปรแกรมของหุ่นยนต์ใช้โมดูลที่มีชื่อว่า Pypot ที่เป็นส่วนเสริมของภาษา Python ในการพัฒนาซอฟแวร์ ทุกคนสามารถเข้าถึงข้อมูลเชิงเทคนิคของหุ่นยนต์ชีวามโนยด์ปีอปปี้ได้ เช่น ส่วนรายละเอียดการทำงาน คลิปวีดีโอสอน การประกอบ การใช้ระบบจำลอง และการพัฒนาต่างๆผ่านทางเว็บไซต์ <http://www.poppy-project.org> หุ่นยนต์ปีอปปี้มีส่วนของโครงสร้างที่ผลิตมาจากพลาสติก PLA และ ABS โดยใช้เทคนิคการฉีดรูปด้วยเครื่องพิมพ์สามมิติ ตัวขับเคลื่อนข้อต่อต่างๆใช้เป็น Dynamixel Digital Servo และควบคุมคำสั่งของตัวขับเคลื่อนด้วยคอมพิวเตอร์ขนาดเล็ก Odroid UX4 ใช้ระบบปฏิบัติการ Ubuntu 14.04 ตัวของหุ่นยนต์มีความสูง 83 เซนติเมตร น้ำหนัก 3.5 กิโลกรัม ใช้เซนเซอร์วัดมุมอิเล็กทรอนิกส์ IMU ที่มีองศาอิสระเท่ากับ 9 องศาอิสระ ในการควบคุมเส้นสายภาพในการเดินของตัวเอง มีองศาอิสระหรือจำนวนตัวขับเคลื่อนทั้งหมด 25 องศา ประกอบไปด้วย ขาข้างละ 6 องศาอิสระ แขนข้างละ 4 องศาอิสระ ลำตัว 3 องศาอิสระ และ หัว 2 องศาอิสระ??

## iCub Humanoid



รูปที่ 2.9: หุ่นยนต์อิวามานอยด์ไอคัพ

หุ่นยนต์อิวามานอยด์ไอคัพ ถูกออกแบบโดยมหาวิทยาลัยหลายแห่งในยุโรปรวมกลุ่มกันขึ้นมาในชื่อ RobotCub และถูกสร้างขึ้นโดย Istituto Italiano di Tecnologia (IIT) ตัวหุ่นยนต์ไอคัพนั้นมีความสูงอยู่ที่ 1 เมตร น้ำหนักโดยรวมทั้งหมดประมาณ 22 กิโลกรัม วัสดุที่ใช้ในการสร้างแตกต่างกันไปในแต่ละส่วนของร่างกายโดยจะใช้ aluminum alloy Al6082 สำหรับส่วนที่ต้องรับภาระความเครียดน้อย ใช้ aluminum alloy 7075 (Ergal) สำหรับส่วนที่ต้องรับภาระความเครียดปานกลางถึงสูง และใช้ Stainless Steel 17-4PH ในส่วนของเพลาข้อต่อต่างๆ เพื่อให้มีความแข็งแรงสูง ตัวหุ่นยนต์ถูกออกแบบให้มีลักษณะเหมือนเด็กอายุ 3-4 ขวบ ควบคุมโดยใช้บอร์ดไมโครคอนโทรเลอร์เป็นรุ่น PC104 Controller ภาษาที่ใช้ในการพัฒนาใช้เป็นภาษา C++ ในการเขียนโปรแกรม การติดต่อสื่อสารกับตัวขับเคลื่อนหรือมอเตอร์ตามข้อต่อต่างๆ และเซนเซอร์ ผ่านทางโปรโตคอล CAN Bus เพื่อทำให้ใช้งานง่ายลง ใช้เส้นเอ็นในการส่งถ่ายแรงขับเคลื่อนไปยังส่วนของข้อต่อส่วนมือและขา นิ้วของหุ่นยนต์ถูกร้อยด้วยสายเคเบิลเคลือบ Teflon อยู่ภายใต้ และความต้านทานของสายเคเบิลที่ต้องการ ใช้เซนเซอร์วัดมุมของข้อต่อแต่ละตัวใช้การออกแบบให้มี Hall-effect ติดอยู่ ช่วยในการอ่านค่าของตำแหน่งและความเร็วที่เกิดขึ้นที่ข้อต่อนั้น หุ่นยนต์ไอคัพมีองศาสตร์รวมกันทั้งหมด 53 องศาสตร์ ประกอบไปด้วย แขนข้างละ 7 องศาสตร์ มือข้างละ 9 องศาสตร์ หัว 6 องศาสตร์ ลำตัว 3 องศาสตร์ และขาข้างละ 6 องศาสตร์ ในส่วนของหัวจะประกอบไปด้วย กล้องสองตัวเพื่อทำการสืบสาน ไมโครโฟนสำหรับรับเสียงจากสภาพแวดล้อมภายนอก และไฟแสดงอารมณ์บริเวณปากและคิ้ว หุ่นยนต์นี้ไม่ได้ถูกออกแบบให้มีการทำงานเป็นแบบอัตโนมัติ ซึ่งก็คือตัวหุ่นยนต์นั้นไม่มีแบตเตอรี่ภายในตัว แต่ใช้แหล่งพลังงานจากการส่งเข้าไปผ่านสายเคเบิล และเชื่อมต่อกับอินเทอร์เน็ตผ่านสายแลน (LAN) ??

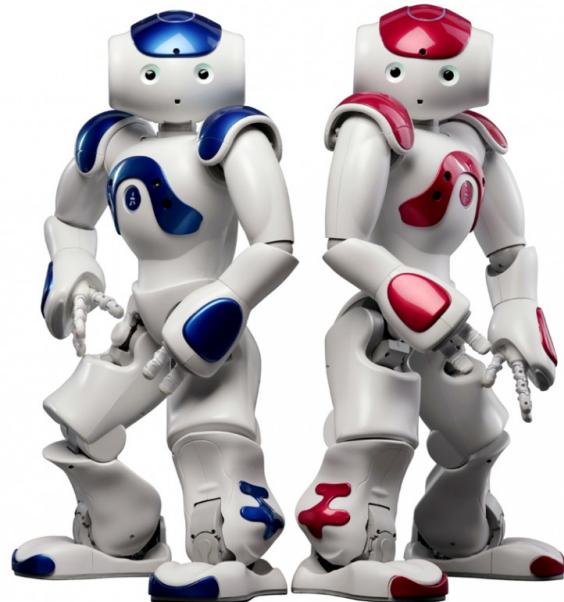
Darwin-OP Humanoid



รูปที่ 2.10: หุ่นยนต์อิวามาโนย์ดาร์วิน

หุ่นยนต์อิวามาโนย์ดาร์วิน (Darwin-OP) เป็นชื่อที่ย่อมาจากคำว่า Dynamic Anthropomorphic Robot with Intelligence–Open Platform เป็น OpenSource Platform ที่ถูกออกแบบและพัฒนาโดย Korean robot manufacturer Robotis โดยมีความร่วมมือกับ Virginia polytechnic institute and state university, Purdue university และ University of Pennsylvania หุ่นยนต์อิวามาโนย์ดาร์วินมีความสามารถในการรับภาระโหลดได้สูง เนื่องจากมีการพัฒนามอเตอร์เป็นของตัวเอง อีกทั้งยังมีความสามารถในการเคลื่อนที่แบบ พลวัต (Dynamic) หุ่นยนต์дар์วิน มีองศาอิสระทั้งหมด 20 องศาอิสระ ซึ่งประกอบไปด้วย ขาข้างละ 6 องศาอิสระ แขนข้างละ 3 องศาอิสระ และหัว 2 องศาอิสระ ขับเคลื่อนข้อต่อต่างๆด้วยเซอร์โวมอเตอร์ Dynamixel MX-28T ที่มีการเชื่อมต่อแบบ RS485 ในการประยัดสายที่ใช้ในการสั่งการ มอเตอร์แต่ละตัวมีเซนเซอร์วัดตำแหน่ง และความเร็วอยู่ภายใน ตัวหุ่นยนต์มีความสูงทั้งหมด 45 เซนติเมตร มีน้ำหนักโดยประมาณ 2.9 กิโลกรัม ระบบภายในใช้คอมพิวเตอร์ขนาดเล็กเป็น 1.6 GHz Intel Atom Z530 (32 bit) ใช้คอนโทรลเลอร์ ARM CortexM3 STM32F103RE 72 MHz และมีเซนเซอร์วัดมุมเอียงเป็น 3-axis gyro, 3-axis accelerometer เพื่อช่วยในการควบคุมเสถียรภาพในการเดิน ??

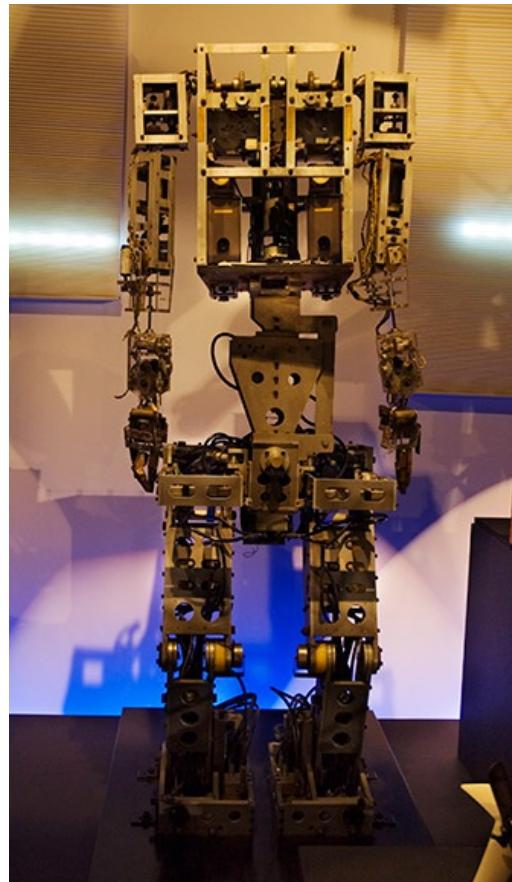
Nao Humanoid



รูปที่ 2.11: หุ่นยนต์อิวามานอยด์นาโอะ

หุ่นยนต์อิวามานอยด์นาโอะ เป็นหุ่นยนต์อิวามานอยด์ขนาดกลาง ถูกผลิตมาจากประเทศฝรั่งเศษ พัฒนาโดยบริษัท Aldebaran Robotics เมื่อปี 2004 และในปี 2007 หุ่นยนต์อิวามานอยด์นาโอะได้นำไปแทนที่หุ่นยนต์สูนัขของ Sony ซึ่ง Aibo ขณะนั้นใช้ในการแข่งขัน RoboCup Standard Platform League (SPL) หุ่นยนต์นาโอะได้ถูกนำไปใช้ใน Robocup 2008 และ 2009 หุ่นยนต์นาโอะถูกพัฒนาออกแบบมาหลายรุ่น มีองศาอิสระตั้งแต่ 14 องศาอิสระ 21 องศาอิสระ และ 25 องศาอิสระ สำหรับเพื่องานวิจัยนั้นมีถึง 25 องศาอิสระ โดยเพิ่มเติมมือสองข้างเอวเข้าไปเพื่อให้สามารถยกจับสิ่งของได้ ภายในหุ่นยนต์ถูกควบคุมด้วยระบบปฏิบัติการ NAO 2.0 (Linux-based) ตัวหุ่นยนต์มีความสูง 58 เซนติเมตร น้ำหนัก 4.3 กิโลกรัม ส่วนเซนเซอร์การรับรู้ต่างๆ จะประกอบไปด้วยเซนเซอร์วัดมุมอุ่ย 3-axis gyro, 3-axis accelerometer, Ultrasound captors, ไมโครโฟน 4 ตัว ลำโพง 2 ตัว กล้อง 2 ตัว เพื่อใช้ประโยชน์ในการทำงานวิจัยต่างๆ ตอนนี้ความสามารถของหุ่นยนต์นาโอะที่ทำได้คือ สามารถเห็นสีได้ เดินขึ้นลงบันไดและทางลาดชันได้ ระหว่างการเดินนั้นสามารถวางแผนการวางเท้าได้อย่างรวดเร็ว อีกทั้งยังสามารถที่จะเดินหลบหลีกสิ่งกีดขวางได้ด้วย ??

Wabot



รูปที่ 2.12: หุ่นยนต์อิวามานอยด์ว้าบอท

หุ่นยนต์อิวามานอยด์มีการพัฒนาในช่วงแรกเริ่มมาตั้งแต่ปี 1973 หุ่นยนต์อิวามานอยด์ ตัวแรกชื่อ Wabot-1 เริ่มสร้างโดยมหาวิทยาลัย Waseda ที่ประเทศญี่ปุ่น ตัวของหุ่นยนต์มีความสูง 180 เซนติเมตร น้ำหนัก 210 กิโลกรัม โดยหุ่นยนต์สามารถติดต่อสื่อสารกับมนุษย์ได้ด้วยภาษาญี่ปุ่น สามารถวัดระยะและทิศทางได้โดยใช้การรับรู้ผ่านทางตาและหูเทียม หุ่นยนต์ Wabot-1 นั้นสามารถเดินได้ด้วยขาของตนเองที่มีสองข้าง สามารถหยิบและเคลื่อนย้ายวัสดุด้วยมือ ต่อมาในปี 1984 มหาวิทยาลัย Waseda ได้พัฒนาหุ่นยนต์อิวามานอยด์ที่ชื่อ Wabot-2 โดยหุ่นยนต์สามารถสื่อสารกับมนุษย์ได้ สามารถถ่ายโน้ตเพลงและเล่นดนตรีโดยใช้ electronic organ แบบง่ายๆ ได้ และในปี 1985 บริษัท Hitachi ได้สร้างหุ่นยนต์ WHL-11 ที่มีสองขาเหมือนมนุษย์ ซึ่งสามารถเดินแบบสมดุลสถิต (Static Walking) บนพื้นราบได้ด้วยความเร็ว 13 วินาทีต่อหนึ่งก้าว และสามารถเลี้ยวได้ซ้ายและขวาได้ ??

## 2.2 การออกแบบโครงสร้างของหุ่นยนต์

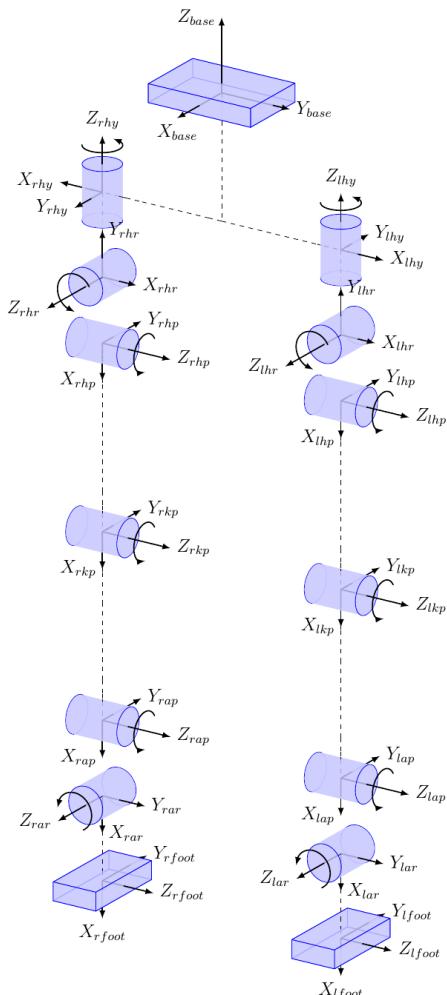
### 2.2.1 ความแตกต่างระหว่างโครงสร้างของมนุษย์กับโครงสร้างของหุ่นยนต์

#### 2.2.1.1 ความแตกต่างขององค์ประกอบ

เนื่องจากลักษณะข้อต่อของมนุษย์มีความซับซ้อนมากกว่าโครงสร้างของหุ่นยนต์ ทำให้ข้อต่อแต่ละจุดของมนุษย์นั้นสามารถหมุนได้หลายทิศทาง รวมถึงขอบเขตของการหมุนของข้อต่อในแต่จุดก็มีความแตกต่างกัน ใน การนำรูปแบบการเดินของมนุษย์ไปใช้กับหุ่นยนต์จึงต้องปรับค่ามุมที่ข้อต่อให้มีความเหมาะสมกับโครงสร้าง และ ข้อจำกัดเกี่ยวกับการหมุนของข้อต่อจุดต่างๆของหุ่นยนต์ที่จะใช้ทดสอบด้วย

#### 2.2.1.2 ความแตกต่างของอัตราส่วน

นอกจากความแตกต่างขององค์ประกอบ (DOF) ระหว่างมนุษย์กับหุ่นยนต์แล้ว ความแตกต่างของอัตราส่วนระหว่างโครงสร้างแต่ละส่วนของมนุษย์กับหุ่นยนต์เป็นอีกสาเหตุหนึ่ง ที่ต้องทำการปรับแต่งใหม่ มีความเหมาะสมกับความยาวของโครงสร้างแต่ละส่วน รวมทั้งระยะห่างระหว่างจุดหมุนแต่ละจุดของมนุษย์กับหุ่นยนต์ที่มีความแตกต่างกัน ดังนั้นจึงต้องกำหนดระบบพิกัดสำหรับหุ่นยนต์ที่มีความยาวต่างกัน เช่น ขาหน้าและขาหลัง ที่มีความยาวต่างกัน แต่ทั้งสองขาต้องมีความยาวที่ใกล้เคียงกัน เพื่อให้ในการอ้างอิงจุดหมุน และความยาวของโครงสร้างในส่วนต่างๆ



รูปที่ 2.13: ตัวอย่างตำแหน่งและการหมุนของข้อต่อของหุ่นยนต์เพื่อการอ้างอิง

### 2.2.1.3 กำลังและประสิทธิภาพของมอเตอร์

ความสามารถในการรับน้ำหนักของข้อต่อแต่ละจุดมีความแตกต่างกัน การเคลื่อนไหวของมนุษย์นั้นจะมีกล้ามเนื้อ และเลี้นเอ็นเป็นตัวออกแรงดึงส่วนต่างๆของร่างกายเพื่อทำให้เกิดการเคลื่อนไหวซึ่งจะมีความยืดหยุ่นและแรงดึงที่มีค่าสูง สำหรับการเคลื่อนไหวของหุ่นยนต์ จะใช้การบิดแกนของเซอร์โวมอเตอร์ (Servo Motor) หรือมอเตอร์ที่ติดอยู่ที่ข้อต่อจุดต่างๆ ทำให้ความสามารถในการรับน้ำหนัก แรงบิดและความยืดหยุ่นที่ข้อต่อขึ้นกับกำลังของมอเตอร์เป็นหลัก การสร้างท่าทางของหุ่นยนต์จึงต้องคำนึงถึงความสามารถในการรับน้ำหนักและกำลังของเซอร์โวมอเตอร์ที่ใช้ด้วยเข่นกัน

### 2.2.2 อุปกรณ์ที่ใช้ในหุ่นยนต์ชีวามโนยด์

#### ตัวขับเคลื่อน

ในการสร้างหุ่นยนต์ชีวามโนยด์นั้นระบบการขับเคลื่อนถือว่าเป็นเรื่องสำคัญ เนื่องจากว่าถ้าหากระบบขับเคลื่อนไม่สามารถทำงานได้อย่างเต็มประสิทธิภาพ หรือหากมีการอุบัติเหตุที่ผิดพลาด จะส่งผลทำให้หุ่นยนต์ชีวามโนยด์นั้นมีประสิทธิภาพในการทำงานลดลงตามไปด้วย ภายในงานวิจัยนี้ทางผู้จัดทำได้ใช้ตัวขับเคลื่อนเป็น Dynamixel digital servo EX-106 ซึ่งเป็นเซอร์โวมอเตอร์ที่เหมาะสมสำหรับทำหุ่นยนต์โดยเฉพาะ ภายในประกอบไปด้วย มอเตอร์กระแสตรง ชุดเฟืองมอเตอร์ ไดเรอර์คอนโทรลเลอร์ สามารถเชื่อมต่อกันผ่าน BUS RS-485 มีการควบคุมแบบ PID และแรงบิดที่สูง<sup>8</sup>



รูปที่ 2.14: ตัวขับเคลื่อนที่ใช้ในหุ่นยนต์ชีวามโนยด์

<sup>8</sup>Robot Actuator [[http://support.robotis.com/en/product/actuator/dynamixel/ex\\_series/ex-106.htm](http://support.robotis.com/en/product/actuator/dynamixel/ex_series/ex-106.htm)]

### หน่วยประมวลผลควบคุม

ในการควบคุมหุ่นยนต์ชีวามโนยด้วยความสามารถทำกิจกรรมต่างๆ คือ หน่วยประมวลผลระบบควบคุม ถ้าหากไม่มีระบบประมวลผลควบคุมแล้ว อุปกรณ์ต่างๆ ที่ติดตั้งอยู่ภายในตัวของหุ่นยนต์ชีวามโนยด้วยจะไม่สามารถติดต่อสื่อสารกันได้ ซอฟท์แวร์ของหุ่นยนต์ที่พัฒนามาทั้งหมดจะไม่สามารถใช้ได้ ทำให้หุ่นยนต์ชีวามโนยด้วยไม่สามารถทำงานในสิ่งที่ต้องการ การวางแผนระบบควบคุมที่นิยมใช้ในระบบหุ่นยนต์ชีวามโนยด้วยส่วนใหญ่ จะแบ่งออกเป็น 2 ส่วนด้วยกัน คือส่วนของหน่วยประมวลผลควบคุมระดับสูง และหน่วยประมวลผลควบคุมระดับต่ำ

#### หน่วยประมวลผลควบคุมระดับสูง (High level controller)

หน่วยประมวลผลควบคุมระดับสูงเป็นส่วนที่ใช้ประมวลผลการทำงานที่มีความซับซ้อนของระบบ เช่น จลนศาสตร์ของหุ่นยนต์ การคำนวณหาเส้นทางการเดิน ในการคำนวณทางคณิตศาสตร์ของระบบเหล่านี้จำเป็นต้องมีการประมวลผลที่เร็ว และมีประสิทธิภาพ ในสมัยที่มีการพัฒนาหุ่นยนต์ชีวามโนยด้วยคุ้นเคยเริ่มนั้น หน่วยประมวลผลควบคุมระดับสูง จะใช้คอมพิวเตอร์เป็นตัวในการประมวลผลการคำนวณ ซึ่งคอมพิวเตอร์สมัยนั้นมีขนาดใหญ่ น้ำหนักมาก และต้องใช้พลังงานสูง ซึ่งต่างจากปัจจุบันนี้ที่มีการพัฒนาของเทคโนโลยีที่ก้าวหน้ามากขึ้น ทำให้คอมพิวเตอร์มีขนาดเล็กลงเทียบเท่ากับบอร์ดคอนโทรลเลอร์ทั่วไป<sup>9</sup>



รูปที่ 2.15: ตัวประมวลผลระดับสูงของ Thormang Humanoid

#### หน่วยประมวลผลควบคุมระดับต่ำ (Low level controller)

หน่วยประมวลผลควบคุมระดับต่ำเป็นส่วนที่รับคำสั่งมาจากหน่วยประมวลผลควบคุมระดับสูง มีประสิทธิภาพในการประมวลผลการคำนวณที่น้อยกว่า เนื่องจากการออกแบบสถาปัตยกรรมภายในระบบไม่เอื้ออำนวยต่อการคำนวณที่มีความซับซ้อน แต่มีความสามารถในการประมวลผลระบบที่เป็นควบคู่อ่อนโยน ในการด้านการทำหุ่นยนต์ชีวามโนยดันนั้นมากจะใช้หน่วยประมวลผลควบคุมระดับต่ำ ในการติดต่อกับอุปกรณ์ต่างๆ บนตัวของหุ่นยนต์ชีวามโนยด้วยตรง เช่น ตัวขับเคลื่อน เชนเชอร์รับค่า หรือไฟแสดงสถานะต่างๆ ของหุ่นยนต์



รูปที่ 2.16: ตัวประมวลผลระดับต่ำของ Robotis OP3 Humanoid

<sup>9</sup>Robot controller Robotis, Thormang, [http://jp.robotis.com/index/product.php?cate\\_code=111410](http://jp.robotis.com/index/product.php?cate_code=111410)

### เซนเซอร์ตรวจหน้าสัมผัสที่พื้น

เซนเซอร์ตรวจหน้าสัมผัสที่พื้นเป็นเซนเซอร์ที่ถูกติดตั้งบริเวณฝ่าเท้า เพื่อตรวจสอบการเดินของหุ่นยนต์อิวามานอยด์ว่าขณะนี้มีการสัมผัสของฝ่าเท้าของหุ่นยนต์กับพื้นหรือไม่ ซึ่งในงานวิจัยนี้ได้ใช้หลักการตัวตรวจจับแรงกดแบบค่าความต้านทานหรือ Force Sensing Resistor (FSR)<sup>10</sup> ที่ใช้เทคโนโลยีฟิล์มโพลีเมอร์แบบหนาโดยที่เซนเซอร์สามารถเปลี่ยนแรงที่มากระทำให้อยู่ในรูปของการเปลี่ยนแปลงค่าความต้านทานไฟฟ้า ตัวเซนเซอร์มีลักษณะเป็นแผ่น มีโครงสร้าง 5 ชั้น โดยสองชั้นนอกสุดเป็นฟิล์มของโพลีเอสเตอร์ ส่วนชั้นถัดเข้ามาเป็นฟิล์มของโลหะที่เป็นตัวนำไฟฟ้า และชั้นในสุดเป็นหมึกที่มีความไวในการตอบสนองต่อแรงภายนอกที่มากระทำ (Pressure sensitive ink) และโครงสร้างทั้ง 5 ชั้น ถูกรวบเข้าด้วยกันด้วยวิธีลามิเนท จึงทำให้เซนเซอร์วัดแรงนี้มีลักษณะแบบมีความยืดหยุ่นสูง ด้วยเหตุนี้จึงทำให้เซนเซอร์สามารถโค้งงอได้ง่าย แรงดันไฟฟ้าที่ต่อกลับจะลดลง เมื่อมีแรงกดมากระทำบนแผ่นตรวจจับ มีโครงสร้างของตัวตรวจจับแสดงในรูปที่ 2.17



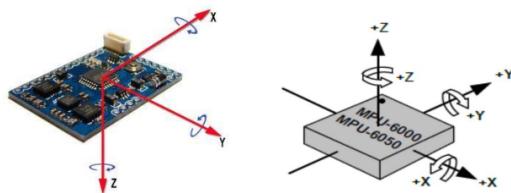
รูปที่ 2.17: ลักษณะโครงสร้างของตัวตรวจจับแรงกด FSR

### เซนเซอร์วัดความเร็ว

Inertial Measurement Unit (IMU) เป็นส่วนประกอบหลักที่ใช้ในการนำร่องเครื่องบิน ยาน-อวกาศ ดาวเทียม เรือ ขีปนาวุธ ซึ่งในตัวของ IMU ประกอบไปด้วยสองส่วนหลักคือ Accelerometers 3 ทิศทาง ใน การรับความเร่งเชิงเส้น และ Gyroscopes 3 ทิศทาง ในการบอกความเร็วเชิงมุม เซนเซอร์ตัวนี้สามารถนำมาใช้ในการหาทิศทางการหมุนของตัวหุ่นยนต์อิวามานอยด์ได้

เซนเซอร์วัดความเร็ว (Gyroscope)<sup>11</sup> เป็นอุปกรณ์สำหรับการวัดความเร็ว หรือการรักษาการปรับทิศทาง ขึ้นอยู่กับหลักการของการอนุรักษ์โมเมนตัมเชิงมุม ถ้าไม่มีการเคลื่อนที่ อัตราการเปลี่ยนแปลงมุมจะมีค่าเท่ากับศูนย์

เซนเซอร์วัดความเร่ง (Accelerometer)<sup>12</sup> เป็นอุปกรณ์ที่ใช้วัดความเร่งเชิงเส้น โดยอาศัยการวัดแรงที่กระทำต่อน้ำหนัก อ้างอิงที่เกิดจากแรงโน้มถ่วงโลก ซึ่งแรงโน้มถ่วงของโลกจะเป็นเวกเตอร์ซึ่งไปที่แกนกลางโลกเสมอ ตามกฎของนิวตัน



รูปที่ 2.18: เซนเซอร์วัดความเร็ว

<sup>10</sup>[UNICON] Force sensor with UNICON [http://doc.inex.co.th/force-sensor-with-unicon/]

<sup>11</sup>Mechanic gyroscope two-degree of freedom [https://www.bosch-sensortec.com/bst/products/motion/gyro-scope/overview\_gyrosopesensors]

<sup>12</sup>Accelerometer and Gyroscopes Sensor [https://www.maximintegrated.com/en/app-notes/index.mvp/id/5830]

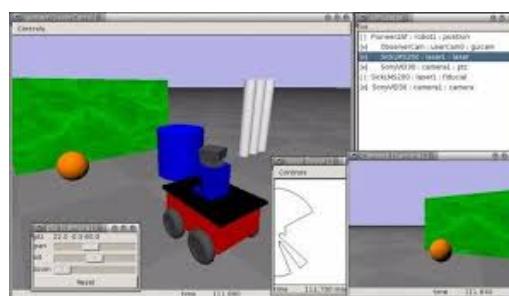
## 2.3 การออกแบบโปรแกรมด้วย ROS

### 2.3.1 ระบบที่ใช้ช่วยในการพัฒนาหุ่นยนต์

Robot Middleware เป็นกรอบการทำงาน (framework) ที่มีความยืดหยุ่นสำหรับการพัฒนาซอฟแวร์ที่ซับซ้อนในการควบคุมของหุ่นยนต์ ตัว Robot Middleware ถูกออกแบบมาให้ใช้ในการจัดการระบบที่มีความยุ่งยาก โดยมีเครื่องมือที่ช่วยติดต่อสื่อสารระหว่างอุปกรณ์ต่างๆของหุ่นยนต์ Robot Middleware ส่วนใหญ่จะใช้การติดต่อสื่อสารผ่านระบบเครือข่ายเน็ตเวิร์ก ทำให้การสื่อสารในระบบพื้นฐานเป็นอิสระต่อกัน และสามารถติดต่อสื่อสารกันกับอุปกรณ์ที่อยู่ภายนอกผ่านเครือข่ายเดียวกันได้

ปัจจุบันมี Robot Middleware ที่ถูกพัฒนาขึ้นมาให้ใช้อยู่หลายตัว เช่น

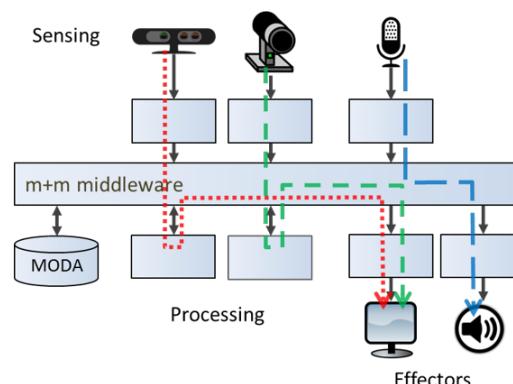
Player Project



รูปที่ 2.19: player project middleware

เป็นโปรเจกท์ที่ใช้ในการสร้างซอฟแวร์เพื่อการศึกษาวิจัยที่มีความเกี่ยวข้องกับหุ่นยนต์และระบบเชนเชอร์ ภายในประกอบไปด้วยระบบตัวกลาง และระบบจำลองการทำงานของหุ่นยนต์

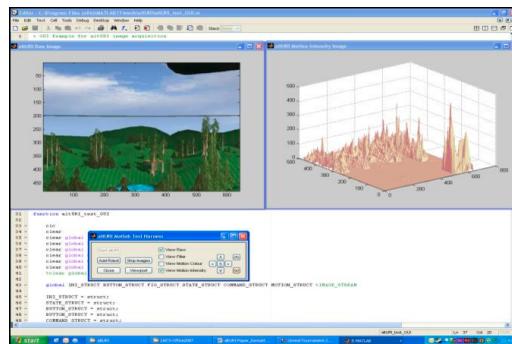
YARP



รูปที่ 2.20: yarp middleware

เป็น open source ที่เขียนด้วยภาษา C++ ในการเข้ามาร่วมต่อกับเชนเชอร์ หน่วยประมวลผล และตัวขับเคลื่อนของหุ่นยนต์

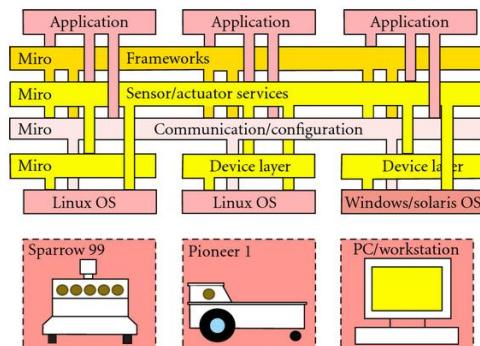
## URBI



รูปที่ 2.21: urbi middleware

เป็น open source สำหรับพัฒนาแอพพลิเคชันที่เกี่ยวข้องกับหุ่นยนต์หรือระบบที่มีความซับซ้อน ใช้ภาษาพื้นฐานเป็นภาษา C++ ติดต่อสื่อสารได้ภายในเครือข่ายเดียวกันเท่านั้น (Local Network)

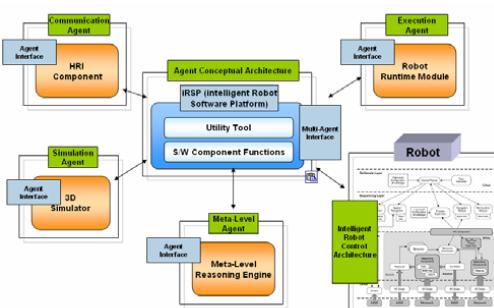
## MIRO



รูปที่ 2.22: miro middleware

เป็นกรอบการทำงานของหุ่นยนต์ที่เคลื่อนที่โดยใช้ในลักษณะเป็น OOP

## OpenRDK

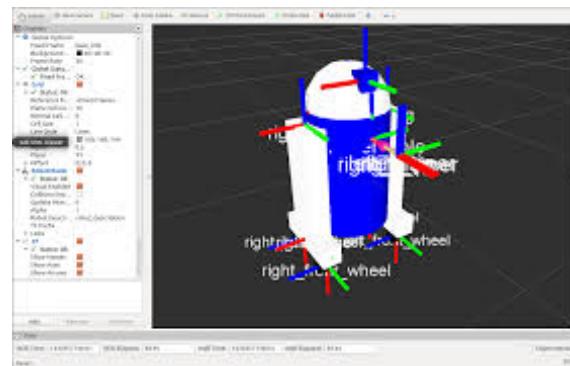


รูปที่ 2.23: openrdk middleware

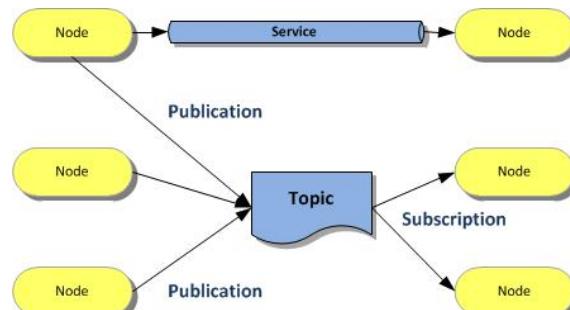
เป็น open source สำหรับพัฒนาระบบที่มีความเป็นอิสระต่อกัน (Modules) สามารถใช้ช่องทางการติดต่อสื่อสารและหน่วยความจำร่วมกันได้

## Robot Operating System

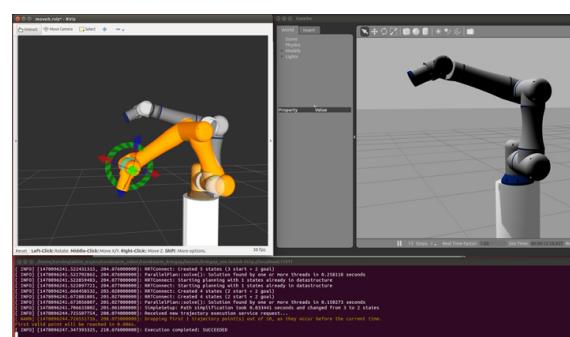
Robot Operating System หรือ ROS ถูกพัฒนาโดยบริษัท Willow Garage, แต่เดิมแล้ว ROS ถูกพัฒนาเพื่อใช้งานกับหุ่นยนต์ PR2 ในปี 2007 ซึ่งพัฒนาเป็น Open Source framework สำหรับนักพัฒนาซอฟแวร์ที่เกี่ยวข้อง กับหุ่นยนต์ มีความสามารถในการทำงานแบบ parallel บนคอมพิวเตอร์หลายเครื่องได้ สามารถทำงานได้ทั้งหลาย OS นอกจากรถยนต์ยังมีคลังที่ครอบคลุมของเฟอร์ต่างๆไว้เป็น libraries อีกด้วย การใช้ ROS จะช่วยทำให้เราสามารถพัฒนาหุ่นยนต์ได้อย่างรวดเร็วมากขึ้น ประหยัดเวลา ประหยัดทรัพยากร



รูปที่ 2.24: ROS middleware Rviz



รูปที่ 2.25: ROS algitecture



รูปที่ 2.26: ROS Moveit

### 2.3.2 ระบบที่ใช้ในการจำลองการทำงานของหุ่นยนต์

โปรแกรมจำลองการทำงานของหุ่นยนต์นั้นเป็นเครื่องมือที่สำคัญสำหรับนักวิจัยที่ทำงานวิจัยเกี่ยวกับหุ่นยนต์ การใช้โปรแกรมจำลองนั้นจะช่วยเพิ่มประสิทธิภาพในการทำงานหลายอย่าง เช่น ให้รู้ว่าหุ่นยนต์ที่ออกแบบนั้นสามารถทำงานได้อย่างที่ต้องการหรือไม่ กระบวนการคิดถูกต้องหรือไม่ โปรแกรมจำลองระบบส่วนใหญ่จะคำนวณพลวัตของหุ่นยนต์โดยใช้เครื่องมือคำนวณ Open Dynamics Engine (ODE)

USARSim



รูปที่ 2.27: ผลลัพธ์จากการใช้โปรแกรม USARSim

USARSim เป็นโอเพนซอร์ซและเหมาะสมสำหรับทำหุ่นยนต์ประเภทกู้ภัยในชากเมือง โดยมีฐานการพัฒนามาจาก Unreal Tournament game engine ภายใต้โปรแกรมมีเครื่องมือสำหรับการทำงานวิจัย มีเซนเซอร์ของหุ่นยนต์ที่หลากหลาย เช่น เซนเซอร์รับภาพ หรือเซนเซอร์ตรวจความเคลื่อนไหว

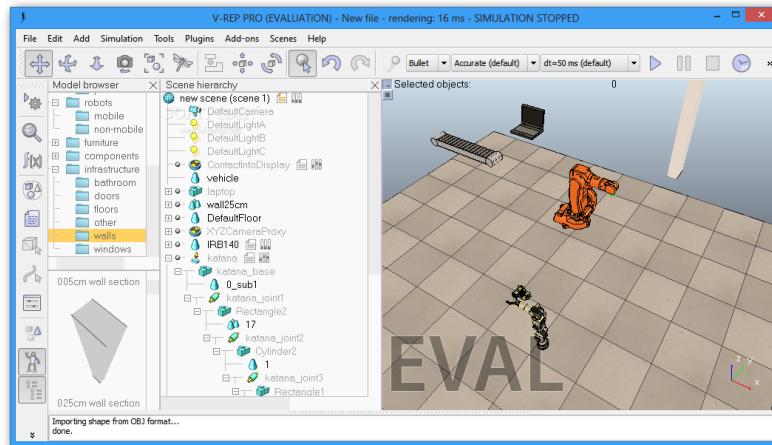
MuRoSimF



รูปที่ 2.28: ผลลัพธ์จากการใช้โปรแกรม MuRoSimF

MuRoSimF ย่อมาจากคำว่า Multi-Robot Simulation Framework เป็นเครื่องมือที่ช่วยทำระบบจำลองจาก Darmstadt University โปรแกรมระบบจำลองนี้มีการใช้งานที่ง่าย เหมาะสำหรับหุ่นยนต์หลายประเภท เช่น หุ่นยนต์เคลื่อนที่ด้วยล้อ หุ่นยนต์สองขา หรือหุ่นยนต์หลายขา สามารถคำนวณพลวัตร และการกระแทกกันของก้านต่อต่างๆได้

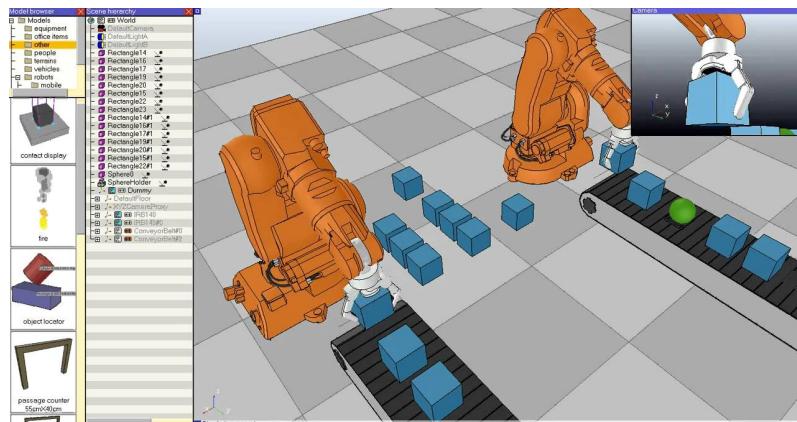
## V-Rep



รูปที่ 2.29: ผลลัพธ์จากการใช้โปรแกรม V-REP

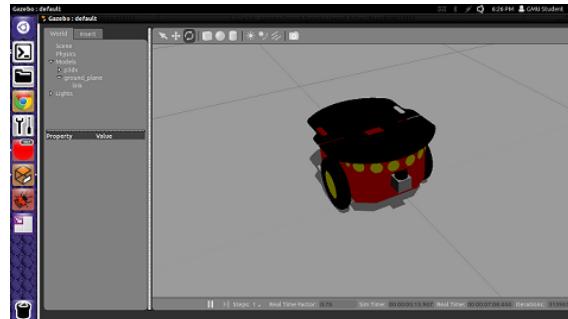
VREP เป็นเครื่องมือสำหรับใช้ในการจำลองระบบหุ่นยนต์ ที่กำลังได้รับความนิยม โดยมีการเพิ่มระบบควบคุมผ่านโปรแกรมจากภายนอกเข้าไป สามารถที่จะเชื่อมต่อกับ ROS ได้และยังสามารถที่จะเขียน Script เพื่อควบคุมหุ่นยนต์ผ่าน API ได้

VREP มีความเร็วในการประมวลผลที่สูง จำลองสายการผลิตในโรงงานอุตสาหกรรม และเหมาะสมสำหรับหุ่นยนต์ที่มีการเคลื่อนที่ด้วยขา มีเวอร์ชัน Education ที่เป็น Open source แต่หากจะใช้เพื่อการค้าจำเป็นจะต้องมี license เพื่อที่จะใช้งาน ราคาค่อนข้างสูง

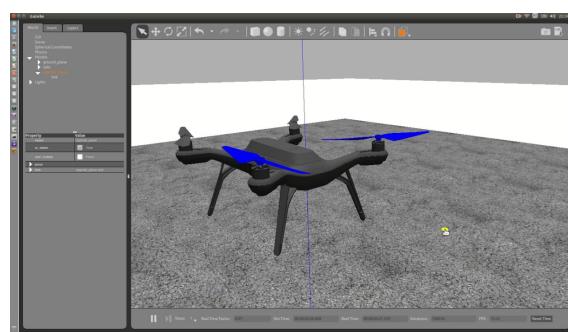


รูปที่ 2.30: V-REP จำลองสายการผลิต

## Gazebo



รูปที่ 2.31: Mobile robot with Gazebo



รูปที่ 2.32: Quadrotor with Gazebo

Gazebo เป็นโปรแกรมจำลองการทำงานของหุ่นยนต์ ที่มีความสามารถในการคำนวณการเดินและการเคลื่อนที่ของหุ่นยนต์ที่ слับซับซ้อนได้ สามารถเห็นภาพกราฟฟิคของหุ่นยนต์ขณะทำงาน โดยผู้ใช้สามารถกำหนดค่าตัวแปรทางฟิสิกส์ต่าง ๆได้ เช่น น้ำหนัก ค่าความเรื้อย แรงเสียดทานของข้อต่อ ทำให้การออกแบบหุ่นยนต์หรือทดลองโปรแกรมได้เหมือนกับโลกจริง มีแสง มีเงา และ พื้นผิวของวัตถุ และที่พิเศษคือสามารถสังเคราะห์ค่าของเซนเซอร์ เช่นเซอร์พร้อมสัญญาณรบกวน ค่าระยะทาง แรงบิด และอื่นๆ คำนวณพลศาสตร์ของหุ่นยนต์โดยใช้ตัวคำนวณทางฟิสิกส์เป็น Bullet หรือ Simbody ในการจำลองหุ่นยนต์ในโปรแกรมนี้จำเป็นต้องได้รับไฟล์ข้อมูลของหุ่นยนต์มาก่อนซึ่งอยู่ในรูปแบบของ URDF ซึ่ง URDF คือ ประเภทของไฟล์ที่บ่งบอกถึงความสัมพันธ์ของข้อต่อและก้านต่อแต่ละชิ้นในตัวหุ่นยนต์ มีความสามารถในการอธิบายถึงกลศาสตร์และการเคลื่อนที่ของหุ่นยนต์ รวมถึงตรวจสอบการกระแทกกันของก้านต่อในหุ่นยนต์ได้ ภายในไฟล์นี้จะประกอบไปด้วย

Link : คือก้านต่อของหุ่นยนต์ซึ่งภายในจะสามารถบอกขนาด รูปร่าง สี และสามารถ import 3d mesh เข้ามาได้ด้วย อีกทั้งยังสามารถใส่รายละเอียดของการเคลื่อนที่ของก้านต่อได้เช่น inertial matrix และ collision properties

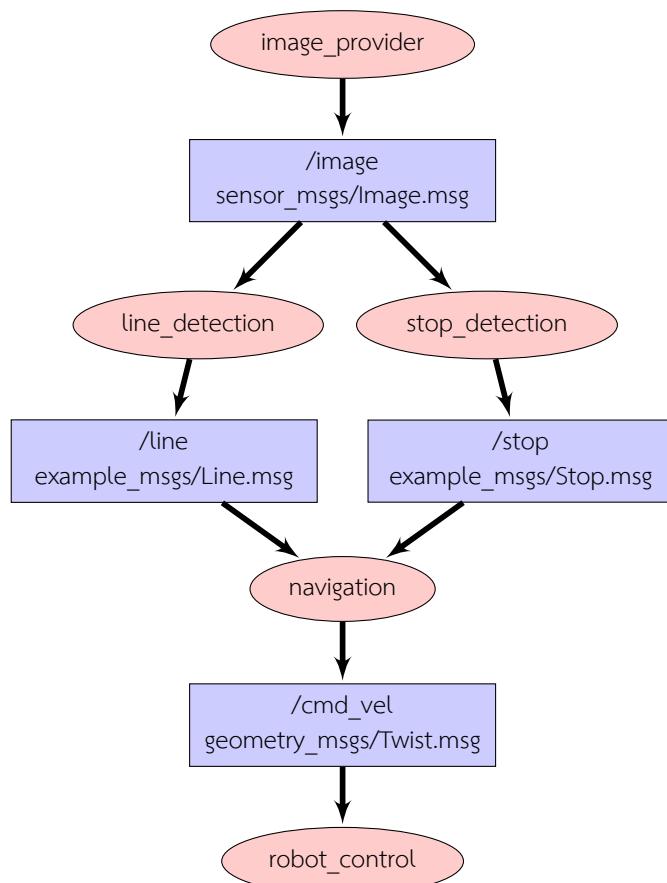
Joint : คือข้อต่อของหุ่นยนต์สามารถกำหนดกลศาสตร์และการเคลื่อนที่ได้ เช่น Joint limits ของข้อต่อที่กำลังหมุนและความเร็วการหมุน ซึ่งข้อต่อมีหลายแบบที่สามารถกำหนดได้ เช่น ข้อต่อแบบหมุน, ข้อต่อแบบเลื่อน, ข้อต่อแบบบิดติด, ข้อต่อแบบต่อเนื่อง

### 2.3.3 Robot Operating System

Robot Operating System หรือ ROS ถูกพัฒนาโดยบริษัท Willow Garage, แต่เดิมแล้วเป็นเครื่องมือเพื่อใช้งานกับหุ่นยนต์ PR2 ในปี 2007 ซึ่งพัฒนาเป็น open source framework สำหรับนักพัฒนาซอฟแวร์ที่เกี่ยวข้องกับหุ่นยนต์ มีความสามารถในการทำงานแบบ parallel บนคอมพิวเตอร์หลายเครื่องได้ สามารถทำงานได้หลาย OS แต่ที่ซัพพอร์ทจริงๆ ก็คือ Ubuntu และ Debian นอกจากนี้ยังมีคลังที่ค่อยเก็บซอฟแวร์ต่างๆไว้เป็น libraries อีกด้วย การใช้ ROS จะช่วยทำให้เราสามารถพัฒนาหุ่นยนต์ได้อย่างรวดเร็วมากขึ้น ประหยัดเวลา ประหยัดทรัพยากรในส่วนนี้จะกล่าวถึง ROS คร่าวๆ

#### Node

Node เป็นเมืองหน่วยประมวลผลในระบบ ROS, Node สามารถที่จะส่งข้อมูลหา Node อื่นๆได้ ผ่าน Topics หรือ Services ในทางปฏิบัติแล้ว Node เป็นตัวประมวลผลอยู่อย่างที่ค่อยทำหน้าที่เฉพาะ ยกตัวอย่างเช่น Node ตัวแรกเชื่อมต่อกับกล้อง เพื่อที่จะนำภาพจากกล้องออกมานode ตัวที่สองใช้ในการหาลูกบล็อกที่อยู่ในภาพที่ได้มาจากการตัวแรก และ Node ตัวที่สามใช้ในการคำนวณหาตำแหน่งของลูกบล็อกที่อยู่บนโลกจริงๆ จากตำแหน่งของลูกบล็อกที่ได้มาจากการตัวที่สอง ดังนั้นจะเห็นว่าแต่ละ Node จะทำงานเฉพาะของตัวเอง ซึ่งสามารถนำมาร่วมกันได้ การเขียนเป็นแบบ Node จะช่วยทำให้เราสามารถที่จะนำไปrogramกลับมา แก้ไขปรับปรุงให้ใช้ใหม่ได้จ่าย ในกรณีที่จะนำไปทำงานอย่างอื่น ยกตัวอย่างเช่น Node ที่เอากาฬจากกล้องออกมาน่าจะมี Node อีกตัว ทำหน้าที่ในการหาโกล์ดเป้าหมาย และหาทิศทางการเคลื่อนที่ของหุ่นยนต์ได้ ดังนั้นการพัฒนา Node เป็นส่วนย่อยๆเล็กๆ ก็เพื่อที่จะทำให้การแก้ไขหรือปรับปรุงได้จ่าย



รูปที่ 2.33: ตัวอย่างสถาปัตยกรรมของ ROS

จากตัวอย่างสถาปัตยกรรมของ ROS ดังรูปที่ 2.33 นั้นสามารถอธิบายได้ว่า หุ่นยนต์เคลื่อนที่ด้วยล้อมีภารกิจคือ เคลื่อนที่ตามเส้นไปเรื่อยๆจนกว่าจะเจอเครื่องหมายหยุด Node คือตัวที่แสดงด้วยรูปวงรี ข้างในเป็นชื่อ Node ส่วน Topic จะแสดงด้วยรูปสี่เหลี่ยม ซึ่งข้างในเป็นชื่อของ Topic และชนิดของ Message ที่ใช้ในการส่งข้อมูล มาดูกันก่อนอื่น ภาพถูกส่งมาจากกล้อง และก็มี Node สองตัวในการดูเส้น และเครื่องหมายหยุด จากภาพที่ได้มา เมื่อ Node ได้ข้อมูลแล้วก็นำมาประมวลผลการเดินของหุ่นยนต์โดยส่งไปยัง node navigation และ Node นี้ก็จะทำหน้าที่คำนวณความเร็วและทิศทางของหุ่นยนต์ ส่งไปยัง node robot\_control ซึ่งเป็นตัวสั่งการมอเตอร์ของหุ่นยนต์อีกด้วย

Twist.msg			Stop.msg
geometry_msgs/Vector3		linear	uint8 RED = 0
geometry_msgs/Vector3		angular	uint8 GREEN = 1
(ก) Message Twist			uint8 color
			float32 distance
(ข) Message Stop			

ตารางที่ 2.2: ตัวอย่างชื่อและข้อมูลของ Message

ตัวอย่างของ Message สองอันนี้ Twist message ดังรูปที่ 2.2ก คือ message ที่เอาไว้บอกความเร็วเชิงเส้น และความเร็วเชิงมุม ซึ่ง ROS มี message ชนิดนี้ให้อยู่แล้ว ส่วน Stop message ดังรูปที่ 2.2ข คือ message ที่เอาไว้บอกระยะทางและสีของป้าย Stop ซึ่ง message นี้ถูกสร้างขึ้นมาใหม่เพื่อใช้กับงานนี้โดยเฉพาะ

### Topics and Messages

Messages เป็นตัวหลักสำคัญในการติดต่อสื่อสารกันระหว่าง Node ใน ROS โดยที่ message จะถูกส่งผ่านไปยัง topic เสมอ แต่ละ Node สามารถที่จะ subscribe หรือ publish ไปกับ topic ก็ได้ การเชื่อมต่อกันระหว่าง Node นั้นสามารถส่งอยู่ภายในเครื่องคอมพิวเตอร์เครื่องเดียวกัน หรือเครื่องอื่นได้ที่อยู่ใน network เดียวกัน โดยจะติดต่อสื่อสารโดยใช้ TCP/IP การใช้คอมพิวเตอร์หลายเครื่องก็จะช่วยให้การประมวลผลมีประสิทธิภาพมากยิ่งขึ้น นอกจากนั้นยังสามารถที่จะแบ่งหน้าที่การทำงานออกจากกันได้ เราสามารถที่จะสร้าง Topic หรือ Message ขึ้นมาเองได้ หากต้องการใช้งานที่เฉพาะทาง

### roscore

roscore เป็นส่วนกลางในการรันระบบทั้งหมด เราจะเรียกว่า rosmaster ซึ่งมีหน้าที่ในการจัดการ topics ทั้งหมด ที่ต้องการจะเชื่อมต่อกันไม่ว่าจะเป็นการ publish หรือ subscribe แต่ rosmaster จะเป็นแค่ตัวจัดการเท่านั้นไม่ได้เป็นตัวที่เก็บ message ต่างๆที่ส่งไปมา ดังนั้น rosmaster จะไม่ทำให้เกิดคอกขวด เวลา.ran ระบบ ในกระบวนการที่คือ subscribe node จะถาม rosmaster ว่ามี topic ที่ต้องการรับข้อมูลใหม่ ส่วนตัว master ที่เก็บค่า topic message เอาไว้ ก็จะส่งไปยัง subscribe node ถ้าหากมีข้อความที่ร้องขอมา และ rosmaster ก็จะจำไว้ว่ามี node ไหนซึ่งมี topic ที่ต้องการรับข้อมูลนั้น

rosparameter server เป็นตัวในการเก็บค่าต่างๆที่เป็น global key-value ซึ่งช่วยให้ node ทุกตัวสามารถใช้ข้อมูลตัวเดียวกันได้ สามารถปรับเปลี่ยนระหว่างการทำงานอยู่ได้ โดยใช้ rqt plugin ซึ่งจะกล่าวในส่วนถัดไป

roslog เป็นตัวที่ใช้สำหรับ logging ข้อมูลต่างๆ ซึ่งจะถูก publish ออกมายัง topic /rosout ซึ่งเราสามารถที่จะเขียนโปรแกรม subscribe จากตัว topic นี้ไปเก็บเป็นไฟล์ได้

## Services

Services หรืออีกชื่อหนึ่งคือ remote procedure calls (RPC) เป็นการส่ง messages แบบที่ไม่ได้เจาะจงว่าจะส่งไปที่ไหน เมื่อ service ถูกเรียกแล้วระบบจะรอนกว่าจะมีการตอบกลับ เราจะเรียกกระบวนการนี้ว่า request และ response message Node ที่อยู่ทำงานเมื่อมีการเรียกใช้ service จะเรียกว่า service server และ node ที่เรียก service จะเรียกว่า service client การใช้งาน service เหมาะสำหรับงานที่ต้องการความรวดเร็ว (fast task) แต่ไม่ควรใช้กับระบบที่ต้องใช้เวลานาน เพราะระบบจะหยุดไม่ยอมทำต่อ ต้องรอให้ service ทำงานเสร็จก่อน สำหรับงานที่ต้องใช้เวลาในการคำนวณนานจะไปใช้ action แทน จะกล่าวในส่วนถัดไป

## Actions

Actions จะใช้กับการทำงาน การประมวลผลที่ต้องใช้เวลาในการทำงาน หรือที่เรียกว่า asynchronously task ในแต่ละ action จะมี message อよุ่ 3 ชนิด คือ goal, feedback และ result Node ที่เป็นตัวรับและรอให้ node อื่นมาเรียก จะเรียกว่า action server ส่วน node ที่เรียกการทำงาน action จะเรียกว่า action client การใช้งาน action จะเริ่มจาก action client จะส่ง message goal ไปยัง action server แล้ว action server จะพยายามทำงานตาม goal ที่ได้รับมา ในระหว่างที่ action client ก็จะทำงานของตัวเองต่อไป แต่จะได้รับ feedback จาก action server อยู่ตลอดเวลา และเมื่อถึง goal ที่กำหนดแล้ว server จะแจ้งมาทาง result message

## Code Organization

ส่วนที่เล็กที่สุดของการจัดการซอฟแวร์ใน ROS ก็คือ package ภายใน package จะมีไฟล์ที่ชื่อว่า package.xml ซึ่งไฟล์นี้จะทำหน้าที่ในการ อธิบายและบอกข้อมูลต่างๆที่เกี่ยวกับ package นี้ ยกตัวอย่างเช่น ชื่อของ package, ชื่อของผู้เขียน, ลิขสิทธิ์ และ dependencies ที่ต้องใช้กับ package นี้ นอกจากนี้ยังสามารถใส่ข้อมูลอื่นๆเกี่ยวกับ node ลงไปเพิ่มเติมได้

```
<package>
    <name>example_package</name>
    <version>1.0.0</version>
    <description>Short example for a package.xml.</description>
    <maintainer emanil="ex@example.org">Jane Doe</maintainer>
    <license>BSD</license>
    <buildtool_depend>catkin</buildtool_depend>
    <build_depend>example_2</build_depend>
    <run_depend>std_msgs</run_depend>
</package>
```

รูปที่ 2.34: ตัวอย่างไฟล์ package.xml

แต่ละ tags ใช้ในการบอกข้อมูลของ package นี้ ใครเป็นเจ้าของ ใครเป็นคนเขียน รวมไปถึง dependencies ที่จำเป็นต้องใช้ของ package นี้ด้วย ดังรูปที่ 2.34

## Code Distribution

การที่จะนำ Nodes กลับมาใช้ใหม่หรือเอาอกมาแบ่งปันให้ผู้อื่นได้นั้น จะต้องมีการทำเอกสารของ Packages นั้นๆด้วย โดยปกติแล้วจะถูกนำมาเปิดกว้างที่ GitHub และ package dependencies จะบอกไว้ในไฟล์ package.xml เรียบร้อยแล้ว เพื่อให้ง่ายต่อการนำไปติดตั้ง หากผู้ที่นำไปใช้พัฒนาต่อหรือแก้ไขข้อผิดพลาดก็สามารถที่จะช่วยกันได้ โดยการ Pull request หรือ Report issues ได้

### ROS Packages ที่ใช้ในงานวิจัย

Package คือพื้นฐานของ ROS, แอพพลิเคชันทั้งหมดใน ROS จะพัฒนาโดยมี package เป็นรากฐาน ใน package นั้นจะเก็บพวกไฟล์ configuration ไปจนถึงไฟล์ launch ที่สามารถไปรัน package หรือ node อื่นๆ ได้ ตอนนี้ ROS มี packages มากกว่า 5000 packages แล้ว

Metapackage เป็นการรวมกันของ packages ที่ทำหน้าที่คล้ายๆกันหลายๆตัวมารวมไว้ที่เดียวเพื่อจะได้ใช้งานง่าย ตัวอย่าง Navigation metapackage ประกอบไปด้วย 10 packages เช่น [AMCL (partical filter), DWA, EKF (extended kalman filter) และ map\_server] ซึ่งหากติดตั้ง metapackage ตัวนี้ก็จะได้มาหมดเลย

ในส่วนนี้จะอธิบายคร่าวๆถึง ROS standard packages ที่จะนำมาใช้ในงานวิจัยครั้งนี้

`rosbag` rosbag เป็นแพกเกจที่สามารถบันทึก message ที่ส่งหากันในระหว่างที่ ROS กำลังทำงานได้ไฟล์ที่บันทึกเรียกว่า `rosbag` ประโยชน์ของมันคือเราสามารถเอาเข้ามาใช้ในการตรวจสอบ หรือนำมาเล่นซ้ำได้อีกทั้งยังง่ายต่อการค้นหาข้อผิดพลาดอีกด้วย

`tf2` tf2 เป็นแพกเกจที่สามารถติดตามการเปลี่ยนแปลงของ Coordinate frame เราสามารถใช้ในการหาความสัมพันธ์ระหว่าง frame ได้ ยกตัวอย่างเช่นหากเราต้องการหาตำแหน่งของ foot เทียบกับ pelvis ก็สามารถใช้ tf2 หาได้

`robot_state_publisher` robot\_state\_publisher แพกเกจที่ subscribe JointState message เพื่อที่จะนำตำแหน่งของข้อต่อ และแปลงให้อยู่ในรูปข้อมูลของ tf2, tf2 สามารถเรียกจาก Node ใดๆก็ได้เพื่อที่จะหา Coordinate frame ที่ต้องการได้

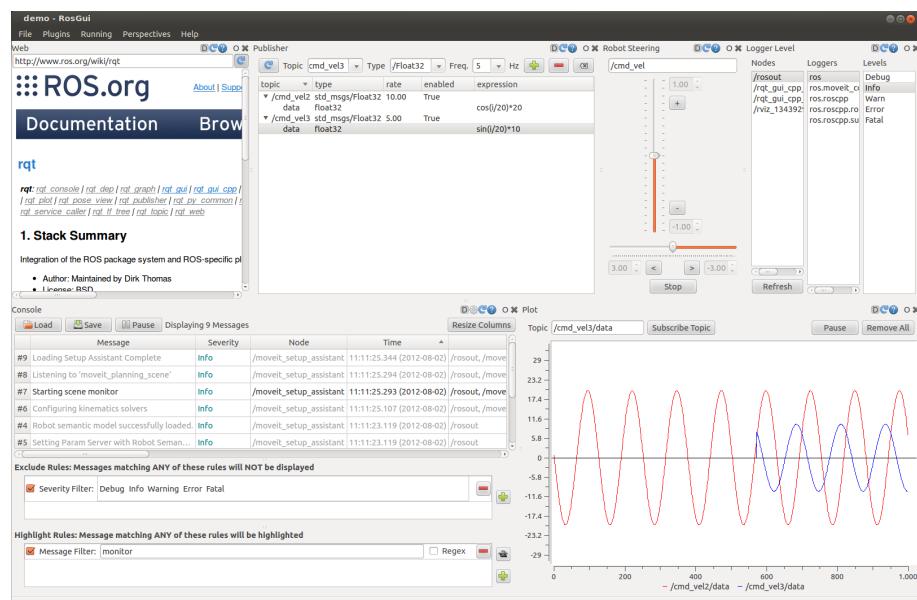
`URDF` Unified Robot Description Format (URDF) เป็นไฟล์ XML ที่เอาไว้อธิบายลักษณะของหุ่นยนต์ ใน ROS มีแพกเกจที่ใช้สำหรับการอ่านไฟล์ คือ urdf\_parser แต่ไฟล์นี้มีการใช้งานโดย tf2 เช่นกัน

`xacro` xacro เป็นไฟล์ XML เช่นเดียวกับ URDF โดยไฟล์ xacro นี้มีประโยชน์มากในการใช้งานใน ROS เพราะว่าทำให้การเขียนไฟล์ URDF ง่ายขึ้น เพราะสามารถทำเป็นมาโครได้ สามารถปรับแต่งค่าตัวแปรต่างๆได้ง่ายขึ้น

## การแสดงผลด้วยภาพ

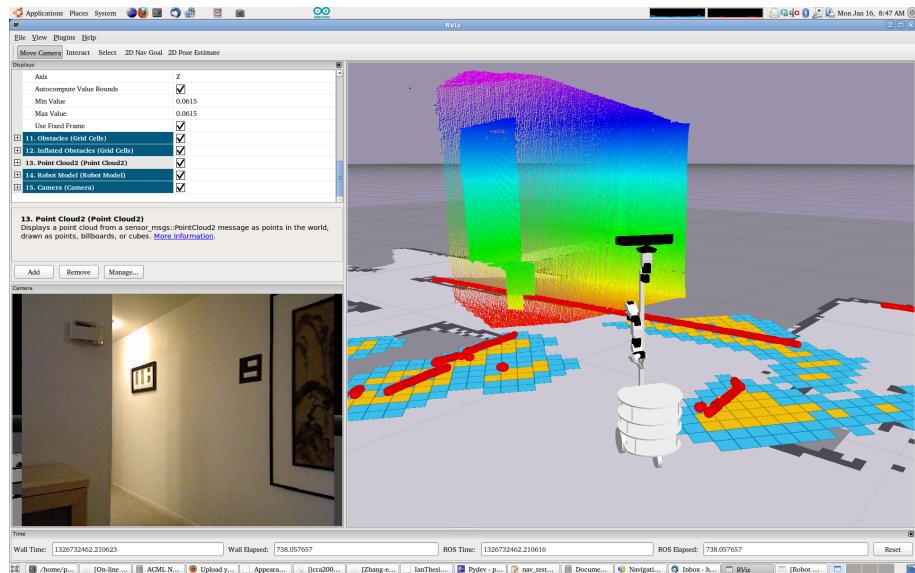
จุดแข็งสำคัญของ ROS อยู่ที่ว่ามีเครื่องมือที่ช่วยในการแสดงผลด้วยภาพได้ที่นอกเหนือจากระบบ publisher-subscriber การใช้เครื่องมือการแสดงผลด้วยภาพนี้จะช่วยให้การทำงานง่ายขึ้นและประหยัดเวลามากขึ้น ในกรณีที่ต้องรับข้อมูลจากหุ่นยนต์อุปกรณ์แสดงผล เพราะว่าเครื่องมือแสดงผลภาพนี้สามารถที่จะ subscribe จาก topic ที่มีการใช้งานอยู่แล้วมาแสดงผลได้ทันที ใน ROS มีเครื่องมือสำคัญอยู่ 2 ตัวที่ใช้สำหรับการแสดงผลด้วยภาพ ซึ่งสามารถที่จะปรับแต่งให้ถูกต้องเป็นเวอร์ชันของเรารองได้

rqt เป็น UI ที่มีฐานมาจาก QT ซึ่งมาพร้อมกับการเชื่อมต่อ ROS เป็นส่วนเสริมในรูปแบบของ QWidget เราสามารถที่จะแสดงผลหลายๆ widgets ได้ภายในเวลาเดียวกัน สามารถที่จะย่อขยาย เปลี่ยนตำแหน่ง ลากวางได้ การเชื่อมต่อกับ ROS นั้นสามารถนำการแสดงผลภาพแบบ 2D ไปแสดงได้ดังรูปที่ 2.35 เป็นการแสดงภาพของกราฟที่ได้รับข้อมูลมาจาก topic หลายๆตัว และสามารถที่จะปรับแต่งค่าและ publish ออกไปได้ด้วยการเขียนโปรแกรมเข้าไป ซึ่งจะเป็นประโยชน์อย่างมากเวลาที่ใช้ในการปรับจูนพารามิเตอร์ต่างๆ เพราะว่าเราสามารถที่จะเปลี่ยนค่าได้ทันที ไม่ต้องรันโปรแกรมใหม่ ในรูปที่ 2.35 เป็นการนำ rqt มาเขียนเป็น GUI ให้ผู้ใช้สามารถใช้งานได้ง่ายและสามารถที่จะปรับแต่งพารามิเตอร์ต่างๆได้เรียลไทม์



รูปที่ 2.35: ตัวอย่างการแสดงผลใน rqt

RViz RViz เป็นการแสดงผลด้วยภาพแบบสามมิติของสถานที่ของหุ่นยนต์และสภาพแวดล้อม โดยใช้ไฟล์ URDF เป็นมาตรฐานการแสดงถึงหุ่นยนต์ ซึ่งสามารถที่จะแสดงตำแหน่งปัจจุบันของข้อต่อต่างๆในหุ่นยนต์ได้ สามารถที่จะแสดงค่าเซนเซอร์เป็น marker ได้ การใช้งานจะเป็นเหมือนการบอกพิกัดเพร์ม ลักษณะการแสดงผลใน RViz มีหลากหลายรูปแบบไม่ว่าจะเป็น camera images, depth clouds, laser scans หรือ point clouds อย่างไรก็ตามการแสดงผลใน Rviz นั้นจะไม่ได้คำนึงถึงแรงที่เข้ามายกระทำกับตัวของหุ่นยนต์ แต่ถ้าเป็นการเคลื่อนที่ที่มีพิกัดเพร์มแล้วสามารถนำมาแสดงได้ ดังรูปที่ 2.36 เป็นตัวอย่างของหุ่นยนต์เคลื่อนที่ด้วยล้อและทำแผนที่ด้วยข้อมูลความลึกที่ได้มาจากการ Kinect

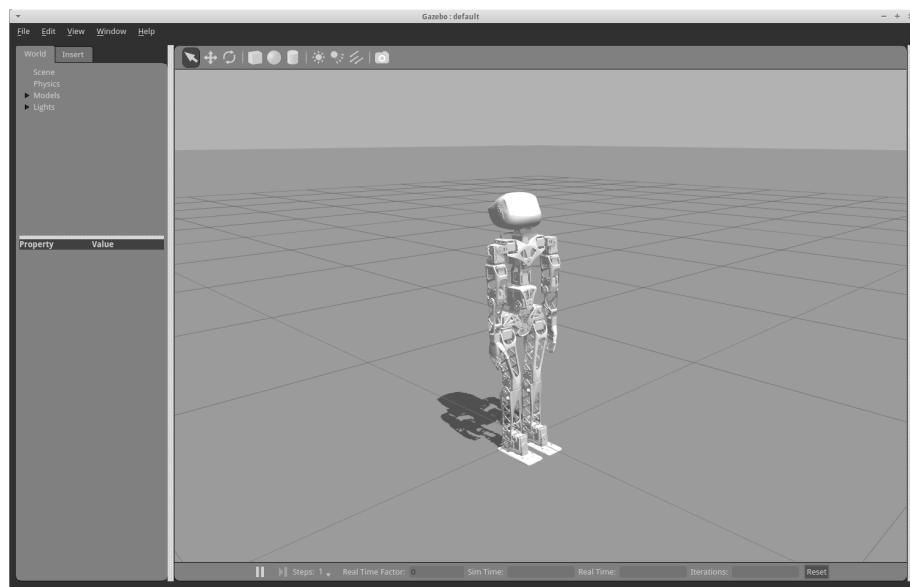


รูปที่ 2.36: ตัวอย่างการแสดงผลใน RViz

### ระบบจำลอง

ระบบจำลองเป็นส่วนที่สำคัญมากสำหรับการพัฒนาโปรแกรมของหุ่นยนต์ เพราะว่าเราสามารถที่จะสร้างโปรแกรมและทดสอบได้โดยไม่จำเป็นต้องมีฮาร์ดแวร์ ซึ่งในส่วนนี้จะช่วยลดความเสียหายจากบักษือโปรแกรมที่ผิดพลาด ที่อาจจะเกิดขึ้นกับหุ่นยนต์ได้ การจำลองจะช่วยลดเวลาในการพัฒนาลงได้ ระบบจำลองปัจจุบันมีมากมายหลายตัวแต่ ตัวที่ได้รับคำแนะนำมากที่สุดคือ Gazebo เพราะว่า Gazebo สามารถที่จะเชื่อมต่อกับ ROS ได้โดยตรง และนักพัฒนาส่วนใหญ่ใช้ Gazebo

การจะใช้ Gazebo ได้นั้นเราจะต้องใช้ไฟล์ URDF ซึ่งเป็นไฟล์ที่เอาไว้แสดงหุ่นยนต์ในระบบจำลอง และสามารถที่จะคำนวนหา collision ให้เราได้อีกด้วย



รูปที่ 2.37: ตัวอย่างหุ่นยนต์หิวามันอยด์ Poppy

## 2.4 การออกแบบระบบพื้นฐาน

### 2.4.1 ความแตกต่างของ Operating Systems

เป็นที่รู้กันโดยทั่วไปว่า hardware และ software ของคอมพิวเตอร์นั้นถูกจัดการโดยโปรแกรมในคอมพิวเตอร์ ซึ่ง operating system (OS) งานพื้นฐานที่ OS ทำก็เช่น การควบคุมและจองหน่วยความจำ การจัดลำดับความสำคัญของระบบ อยู่ดูแลควบคุมอุปกรณ์ต่างๆ ที่เชื่อมต่อเข้ากับคอมพิวเตอร์ การเชื่อมต่อระบบเน็ตเวิร์ค การจัดการไฟล์ข้อมูล อีกทั้งยังรวมไปถึงการให้บริการต่างๆ เช่น การจัดการกระบวนการประมวลผล จัดการไฟล์ ของระบบต่างๆ ระบบป้องกัน อื่นๆ

ปัจจุบันมี OS อยู่หลายตัวเช่น Windows, Mac OS X, UNIX, Solaris BS3000, MS-Dos และอื่นๆ ซึ่งทั้งหมดนี้เป็นส่วนหนึ่งของระบบคอมพิวเตอร์ที่จะคอยช่วยจัดการและควบคุมดูแลการทำงานต่างๆ ของคอมพิวเตอร์ ระบบคอมพิวเตอร์นั้นอาจจะอยู่ในรูปแบบอื่นๆ เช่น เครื่องที่ใช้ทำงาน, เครื่องเซิฟเวอร์, เครื่องคอมพิวเตอร์ส่วนบุคคล, โทรศัพท์เคลื่อนที่, อุปกรณ์นำทาง หรือแม้กระทั่งระบบที่มีความฉลาดในตัวมันเอง เช่น หุ่นยนต์ และ OS นั้นจะสามารถทำงานบน hardware อุปกรณ์ใดๆ ก็ได้

### 2.4.2 ข้อแตกต่างระหว่าง Open platform กับ Non-open platform

หุ่นยนต์ Open platform คือ การออกแบบระบบพื้นฐานของหุ่นยนต์ที่เปิดให้ผู้ที่ต้องการศึกษาหรือผู้ใช้ทั่วไปสามารถเข้าถึงข้อมูลต่างๆ ที่เกี่ยวข้องกับหุ่นยนต์นั้นๆ ได้ ผู้ใช้สามารถที่จะนำข้อมูลเหล่านั้นมาแก้ไข ปรับปรุง แต่งตั้ง หรือเรียนรู้และพัฒนาตามได้ด้วยตนเอง ซึ่งข้อมูลที่กล่าวมานั้นสามารถได้จากเว็บไซต์ของผู้พัฒนาหุ่นยนต์ ปัจจุบันมีหุ่นยนต์ที่มีความสามารถอยู่ที่เป็นเปิดให้เข้าถึงหลายรูปแบบแตกต่างกันไป

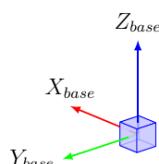
ส่วนหุ่นยนต์ Non-open source platform คือหุ่นยนต์ที่สร้างมาเฉพาะเจาะจงสำหรับการวิจัย การสำรวจ หรือการแข่งขันโดยเฉพาะ ไม่เปิดให้บุคคลภายนอกเข้าศึกษาหรือแก้ไขปรับปรุง ซึ่งทำให้หุ่นยนต์ประเภทนี้ไม่เหมาะสมสำหรับผู้วิจัยที่จะเรียนรู้และศึกษาด้วยตนเอง เพราะมีขั้นตอนใหญ่ ใช้ทรัพยากร่มาก และการออกแบบมีความซับซ้อน เรียนรู้ยากกว่าหุ่นยนต์แบบ Open platform

### 2.4.3 มาตรฐานหน่วยวัดและการบวกพิกัด

การใช้หน่วยวัดที่ไม่ตรงกันอาจจะทำให้เกิดปัญหาขึ้นได้ เนื่องจากเป็นแพลตฟอร์มนั้นจะมีบุคคลอื่นช่วยกันพัฒนาหลายคน จึงควรที่จะมีมาตรฐานในการวัดและการกำหนดพิกัดต่างๆ ที่ตรงกันเพื่อให้เกิดความชัดเจนในการทำความเข้าใจ

หน่วยวัด การวัดนั้นใช้มาตราฐานการวัดเป็น SI Units ซึ่งมาตราฐานนี้ใช้กันอย่างแพร่หลายและเป็นสากล โดยหน่วยการวัดนี้ได้รับการยืนยันจาก Bureau International des Poids et Mesures ตามตารางที่ 2.3

พิกัดเพรม การบวกพิกัดทางการหมุนนั้นใช้หลักตามกฎมือขวา โดยการตั้งแกนนั้นหากเทียบกับมือแล้ว X ไปข้างหน้า Y ไปทางซ้าย Z พุ่มขึ้น ดังรูปที่ 2.38



รูปที่ 2.38: การตั้งแกนตามกฎมือขวา

ปริมาณ (Quantity)	หน่วยวัด (Unit)	สัญลักษณ์ (Symbol)
ความยาว (Length)	เมตร (metre)	$m$
มวล (Mass)	กิโลกรัม (kilogram)	$kg$
เวลา (Time)	วินาที (second)	$s$
กระแสไฟฟ้า (Electric Current)	แอมเปอร์ (ampere)	$A$
มุม (Angle)	เรเดียน (radian), องศา (degree)	$rad, deg$
ความถี่ (Frequency)	เฮิร์ต (Hertz)	$Hz$
แรง (Force)	นิวตัน (Newton)	$N$
กำลัง (Power)	วัตต์ (Watt)	$W$
แรงดันไฟฟ้า (Voltage)	โวลต์ (Volt)	$V$
อุณหภูมิ (Temperature)	เซลเซียส (Celsius)	$^{\circ}C$

ตารางที่ 2.3: ตารางแสดงหน่วยวัดมาตรฐาน

#### 2.4.4 Robot Operating System

ROS เป็นกรอบการทำงานที่ได้รับความนิยมและมีประสิทธิภาพในการทำงานมากที่สุดในปัจจุบัน เนื่องจาก ROS ได้รวมซอฟต์แวร์เครื่องมือที่หลากหลายเอาไว้เป็นหมวดหมู่ เช่น การเชื่อมต่อกับฮาร์ดแวร์ การสร้างระบบควบคุมให้กับอุปกรณ์ต่างๆ อีกทั้งสามารถที่จะเขียนโปรแกรมแล้วนำกลับมาใช้ใหม่ได้ ภายในระบบมีกระบวนการรับส่งข้อมูลต่างๆ เป็นของตัวเอง ทำให้ช่วยลดความซับซ้อนและเพิ่มประสิทธิภาพในการทำงานกับแพลตฟอร์มของหุ่นยนต์ กระบวนการเขียนโปรแกรมของ ROS นั้นจะใช้รูปแบบ Graph architecture ซึ่งจะทำให้สามารถแบ่งโปรแกรมต่างๆ ออกเป็นส่วนๆ เช่น เชนเซอร์หลายๆ ตัว ระบบควบคุม ระบบวางแผน ระบบขับเคลื่อน ระบบสื่อสารภายนอก ด้วยตัวระบบของ ROS นั้น ไม่ใช่ Real Time OS แต่อย่างไรก็ตามเราสามารถใช้งานผสมกับ Real Time ได้

ROS ประกอบไปด้วยแพ็กเกจเจกต่างๆ มาประกอบกันเป็น Node โดยมีตัวกลางทำหน้าที่ในการติดต่อสื่อสารระหว่างอุปกรณ์ที่เป็น Node ต่างๆ ให้สามารถส่งข้อมูลหากันได้ รูปแบบการสื่อสารใน ROS จะใช้หลักการแบบ Publish/Subscribe ทำให้ไม่จำเป็นที่จะต้องระบุโปรแกรมที่จะรับ ภาษาในการพัฒนามีให้เลือกที่หลากหลาย เช่น C++, Python, Lisp, MATLAB หรือ JavaScript

#### ประโยชน์จากการใช้ ROS

ROS เป็นกรอบการทำงาน ที่อยู่ระหว่าง OS และ Robot ทำให้เราไม่ต้องกังวลเรื่องการจัดการระบบภายใน เพราะ ROS จะช่วยจัดการให้เราทั้งหมด ก่อนจะมี ROS นั้น นักวิจัยจะต้องใช้เวลาไปกับพัฒนาพื้นฐานให้หุ่นยนต์ ซึ่งจะต้องมีทักษะทางด้านเครื่องกล ไฟฟ้า และโปรแกรม ซึ่งบอยครั้งที่นักวิจัยหรือนักพัฒนานั้นไม่มีความรู้ หรือประสบการณ์ในการสร้างหุ่นยนต์ ทำให้การทำงานเป็นไปด้วยความลำบาก

## บทที่ 3

### ระเบียบวิธีวิจัย

ในการทำโครงการวิจัยแอพพลิเคชั่นสำหรับวิเคราะห์วิดีโอ(video analytics) จะมีการทำงานหลากหลายส่วนมาทำงานร่วมกัน ซึ่งทำให้จำเป็นจะต้องมีระเบียบวิจัยสำหรับอธิบายภาพรวม โดยในระเบียบวิจัยนี้จะมีหัวข้อ และระเบียบวิจัยดังนี้

- แผนการดำเนินงาน
- เครื่องมือที่ใช้ในการดำเนินงานวิจัย
- ภาพรวมของแอพพลิเคชั่น
- รายละเอียดของโมเดล

#### 3.1 หน้าที่ความรับผิดชอบ

ปฐมพงศ์ สินธุ์งาม สร้างและทดสอบโมเดล adjudication ทำมนุษย์ 3D และออกแบบพร้อมทั้งสร้างระบบ Tracker

ศุภกร เบญจวิกรัย รวบรวมฟังก์ชันต่างๆ ของแอพพลิเคชั่น และออกแบบพร้อมทั้งสร้างระบบแอพพลิเคชั่นในส่วน Selection และ Detection

อุกฤษฎ์ เลิศวรรณการ สร้างและทดสอบโมเดล adjudication ทำมนุษย์ Resnet-50 และออกแบบพร้อมทั้งสร้างระบบ Person ReID

#### 3.2 แผนการดำเนินงาน

โดยจากที่กล่าวไปตอนต้นในบทนำ การดำเนินงานและการออกแบบการสร้าง labeling tool และระบบวิเคราะห์การทำมนุษย์ในวิดีโอ มีแผนการทำงานซึ่งถูกแบ่งออกเป็นสามส่วนดังนี้ ส่วนแรกคือ ส่วนของการศึกษาหาความเป็นไปได้ และเทคโนโลยีในปัจจุบันที่เกี่ยวกับการสร้างแอพพลิเคชั่น และการจัดทำมนุษย์ด้วยปัญญาประดิษฐ์ เพื่อนำมาประยุกต์ใช้กับงานวิจัยนี้ ส่วนที่สองคือ ส่วนของการออกแบบและสร้างแอพพลิเคชั่นที่ใช้ในการสร้างชุดข้อมูลสำหรับการเทรนโมเดลจากวิดีโอ ส่วนที่สามคือ ส่วนของการออกแบบและสร้างระบบแพลตฟอร์มวิเคราะห์การกระทำของมนุษย์โดยมีข้อกำหนดตามที่กล่าวไว้ในบทนำ

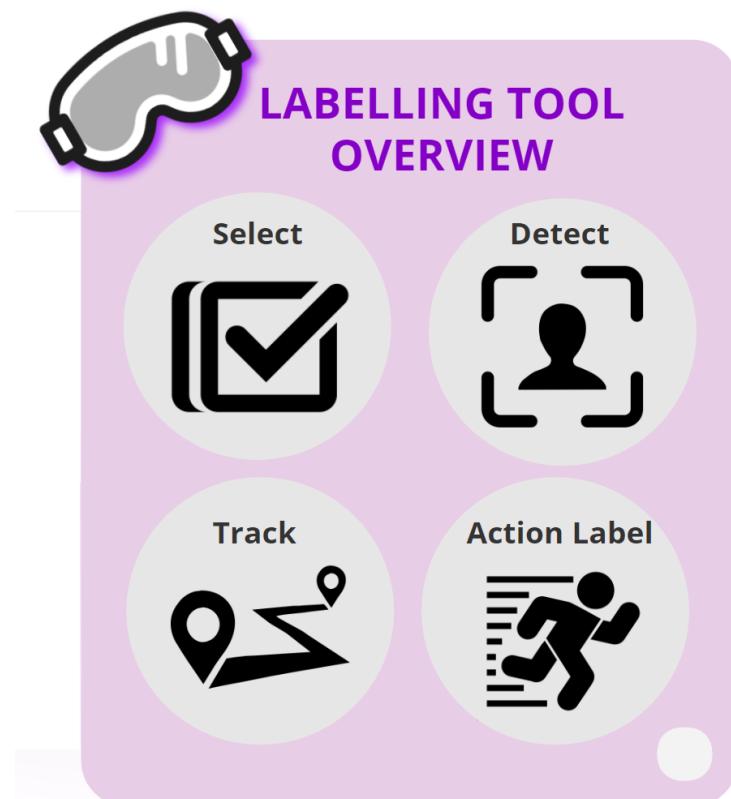
ในการเริ่มทำงานวิจัยนี้นั้นสิ่งจำเป็นที่ต้องทำในอันดับแรกคือการศึกษาสิ่งที่เคยมีอยู่ หรืองานวิจัยอื่นที่ทำเอาไว้แล้ว เพื่อศึกษาและทำความเข้าใจ ข้อดี-ข้อเสีย ของเทคนิคหรือกระบวนการต่างๆ เพื่อนำมาประยุกต์ใช้กับงานวิจัยนี้ ในการศึกษาเกี่ยวกับการออกแบบและการสร้างแอพพลิเคชั่นที่ใช้ในการสร้างชุดข้อมูลสำหรับการเทรนโมเดลจากวิดีโอ ที่มีอยู่แล้วสิ่งที่ต้องให้ความสนใจคือฟังก์ชันการทำงาน การออกแบบและการจัดวางองค์ประกอบต่างๆ ในหน้าต่าง UI และความสะดวกในการใช้งาน จากนั้นจึงเริ่มศึกษาเกี่ยวกับแพตฟอร์มที่ใช้ในการสร้างแอพพลิเคชั่น ส่วนการศึกษาเกี่ยวกับการสร้างระบบวิเคราะห์การกระทำมนุษย์นั้น จะมุ่งความสนใจไปที่ชุดข้อมูลสำหรับการวิเคราะห์วิดีโอ โมเดลสำหรับการวิเคราะห์วิดีโอ เทคนิคในการสร้างโมเดล เทคโนโลยีในการทำระบบวิเคราะห์วิดีโอ เพื่อใช้ในการออกแบบและสร้างระบบวิเคราะห์การกระทำของมนุษย์ในวิดีโอด้วยมีประสิทธิภาพ ในบทนี้ก็จะกล่าวถึงกระบวนการออกแบบและการดำเนินการตามแผนที่วางแผนไว้

### 3.3 การออกแบบแอพพลิเคชัน labeling tool

การออกแบบ labeling tool นั้น ผู้วิจัยได้เลือกใช้แพลตฟอร์ม PyQt และภาษา Python ในการพัฒนาเนื่องจากแพลตฟอร์ม PyQt นั้นเป็นแพลตฟอร์มที่มีผู้พัฒนาใช้กันอย่างแพร่หลาย จึงทำให้สะดวกในการศึกษาและหาข้อมูลผ่านอินเตอร์เน็ต อีกทั้งยังเป็นแพลตฟอร์มที่สามารถพัฒนาด้วยภาษา Python ได้ และเป็นแพลตฟอร์มที่ใช้งานง่าย สามารถปรับปรุงแก้ไขได้ง่าย เนื่องจากการสร้างแอพพลิเคชันนั้นจำเป็นต้องมีการปรับแก้หน้าต่างอยู่เสมอ

#### 3.3.1 แอพพลิเคชัน labeling tool

ภาพรวมของแอพพลิเคชันที่สร้างขึ้นประกอบด้วยส่วน Select, Detect, Track และ Action label เพื่อช่วยแบ่งเบาภาระของผู้พัฒนาในการสร้าง label สำหรับสร้างโมเดลจากข้อมูลประเกวิดีโอ โดยส่วน Select จะต้องสามารถตัดวิดีโอด้วยที่ไม่มีมนุษย์อยู่ออกจากวิดีโอด้วย Detect ต้องสามารถหาตำแหน่งของมนุษย์ภายในวิดีโอด้วย Track ต้องสามารถทำนายตำแหน่งต่อไปของมนุษย์ข้อมูลตำแหน่งของมนุษย์จาก Detect ได้ Action label ต้องสามารถทำนายการกระทำการของมนุษย์ได้ในระดับหนึ่ง โดยทุกส่วนการทำงานมนุษย์ต้องสามารถทำงานร่วมกับระบบได้ ดังรูปที่ 3.1

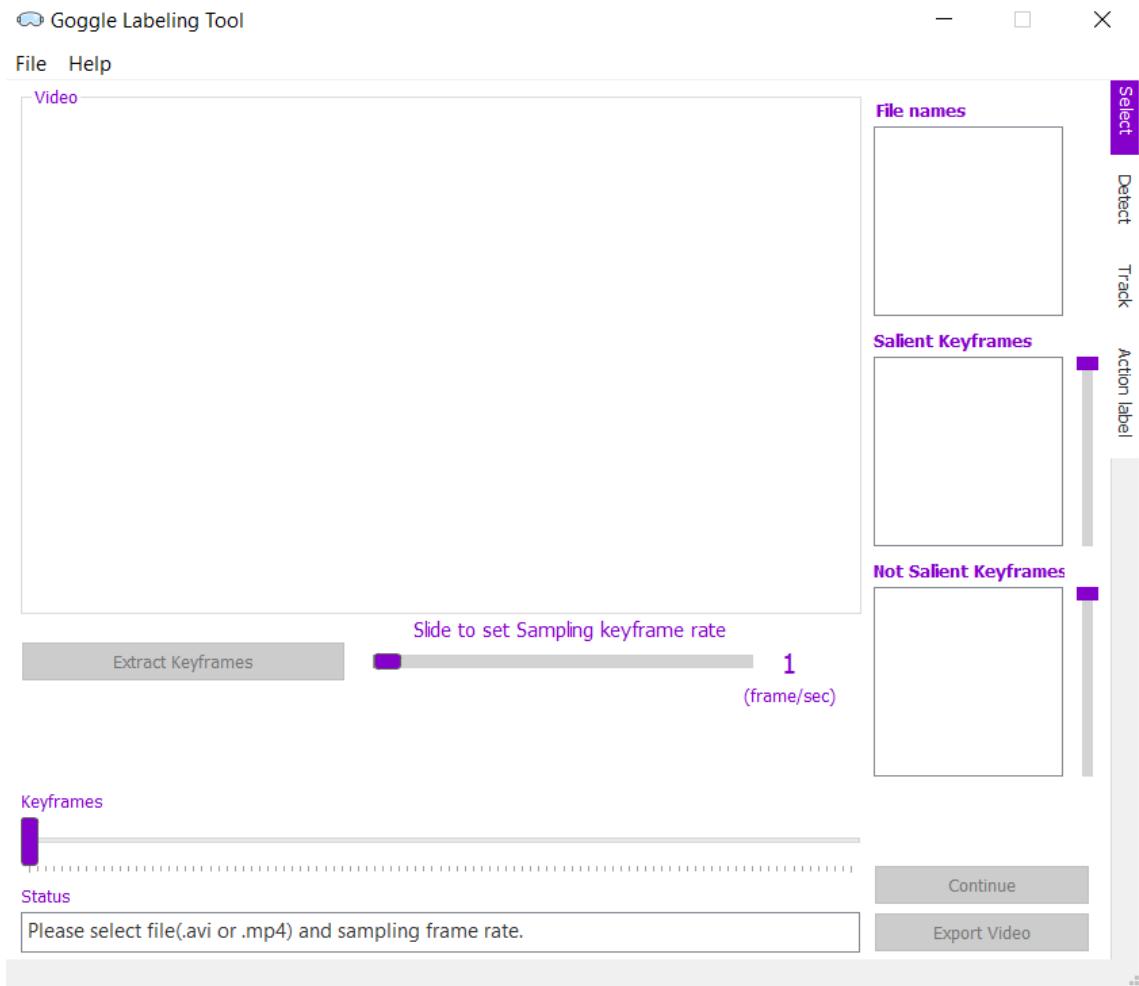


รูปที่ 3.1: ภาพรวมระบบของแอพพลิเคชัน labeling tool

โดยแต่ละส่วนจะมีรายละเอียดดังนี้

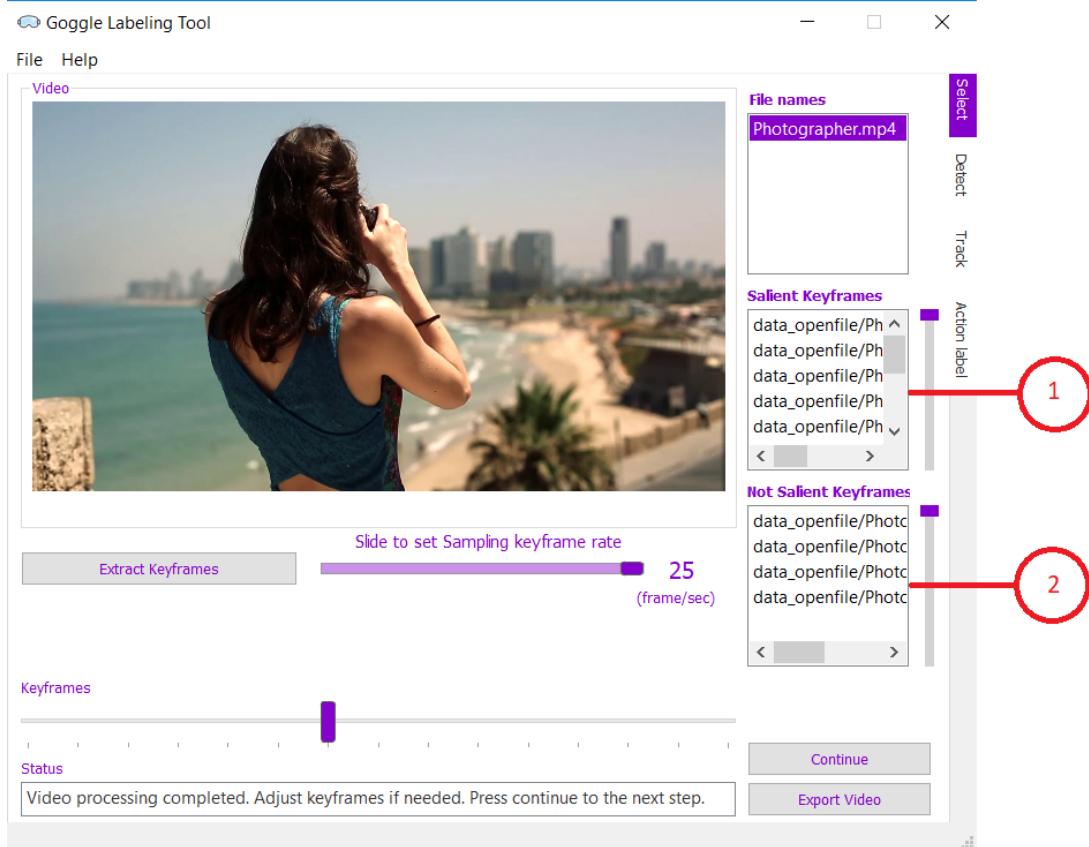
### 3.3.1.1 Select

ในส่วนของแบบ Select จะมีหน้าต่างเป็นดังรูปที่ 3.2 โดยในส่วนนี้จะมีหน้าที่ในการโหลดวิดีโอที่ต้องการ กำหนดอัตราการหยับตัวอย่างเฟรมของวิดีโอแล้วเก็บเฟรมเหล่านั้นเป็นคีย์เฟรม(Keyframe) และตัดวิดีโอส่วนที่ไม่มีมนุษย์อยู่ออกໄປ



รูปที่ 3.2: หน้าต่างแบบ Select ของแอพพลิเคชัน labeling tool

ซึ่งในขั้นตอนการตัดส่วนวิดีโอที่ไม่มีมนุษย์อยู่นั้น ได้ใช้โมเดล YoLo-v3 320 สำหรับตรวจหามนุษย์ ในแต่ละคีย์เฟรม จากนั้นจะแยกคีย์เฟรมที่มีมนุษย์อยู่ และที่ไม่มีมนุษย์อยู่ออกมา แล้วเก็บไว้ในช่องรายการหมายเลข 1 และ 2 ตามลำดับ ดังรูปที่ 3.3



รูปที่ 3.3: หลังจากตัดส่วนวิดีโอลاء คีย์เฟรมจะถูกเก็บไว้ในช่องรายการตามประเภท

### 3.3.1.2 Detect

ในส่วนของแบบ Delect จะมีหน้าต่างเป็นดังรูปที่ ??

### 3.4 การออกแบบระบบวิเคราะห์การกระทำของมนุษย์

การออกแบบแบบโปรแกรมด้วย ROS ของหุ่นยนต์ชีวามโนยด์ UTHAI นั้น ผู้วิจัยได้วางการทำงานโดย เริ่มจาก การสร้างแบบจำลองเพื่อใช้ในการจำลองการทำงานของหุ่นยนต์ชีวามโนยด์ UTHAI โดยการจัดสร้างไฟล์ URDF ขึ้น ทดสอบการเคลื่อนไหวของหุ่นยนต์ในการแสดงผลด้วยภาพผ่านโปรแกรม RViz ทดสอบการสั่งการดิจิตอล เชอร์โว ทดสอบการอ่านตำแหน่งจากดิจิตอลเชอร์โว ทดสอบการส่งค่าตำแหน่งจากดิจิตอลเชอร์โวไปประมวลผล หากดศูนย์กลางมวลในโปรแกรม MATLAB และเขียนโปรแกรมอ่านค่าเซนเซอร์ตรวจสอบการสัมผัสพื้น

#### 3.4.1 กำหนดพิกัดเฟรมให้กับหุ่นยนต์ชีวามโนยด์

การกำหนดเฟรมให้กับหุ่นยนต์ชีวามโนยด์ UTHAI นั้น ในวิทยานิพนธ์เล่มนี้ ผู้วิจัยจะใช้หลักตามของ ROS Enhancement Proposals (REPs)<sup>1</sup> ซึ่งการใช้หลักการนี้จะทำให้ การเขียนเป็นระบบระเบียบสามารถหยิบเครื่อง มือต่างๆ ที่สร้างขึ้นมาใช้งานร่วมกันได้ และช่วยทำให้เกิดความเข้าใจเวลาสื่อสาร

`base_link` เป็นเฟรมที่ติดอยู่กับฐานของหุ่นยนต์ชีวามโนยด์ โดยจะติดตำแหน่งหรือมุมเอียงได้ โดย ส่วนใหญ่แล้วจะติดเฟรม `base_link` ไว้ที่สะโพกของหุ่นยนต์

`base_footprint` เป็นเฟรมที่แสดงว่าหุ่นยนต์อยู่ตรงไหนเมื่อเทียบกับโลก โดยจะมีระดับอยู่ที่จุดต่ำสุด ของฝ่าเท้า  $z = \min(l\_sole\_z, r\_sole\_z)$  โดย  $l\_sole\_z$  และ  $r\_sole\_z$  คือความสูงของฝ่าเท้า

`l_wrist, r_wrist` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของแขนซ้ายและขวาของหุ่นยนต์ชีวามโนยด์ โดยไม่ต้องคำนึงถึงการติดตั้งอุปกรณ์ใดๆเข้าไปที่ปลายแขนของหุ่นยนต์ชีวามโนยด์

`l_gripper, r_gripper` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของที่ปลายแขน (End effector) ถ้ามีอุปกรณ์อยู่ เฟรมนี้จะใช้ในการอ้างอิงตำแหน่งของอุปกรณ์นั้นๆ แต่ในวิทยานิพนธ์นี้ ไม่ได้ใช้แขนของหุ่นยนต์ในการหยิบจับเครื่องมือหรือวัตถุ จึงไม่ได้ใช้เฟรมนี้

`l_ankle, r_ankle` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของขาซ้ายและขวาโดยไม่ได้คำนึงว่าจุดรับน้ำหนักของตัวอยู่ที่ไหน

`l_sole, r_sole` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของขาซ้ายและขวาที่รองรับน้ำหนักตัวอยู่ โดยจะ บอกการฉายลงในระนาบของ  $X, Y$  ที่สัมผัสพื้นและ  $Z$  จะอยู่ระหว่างตัวเดียวกับพื้นสัมผัส

`l_toe, r_toe` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของปลายเท้าซ้ายและขวา

`gaze` เป็นเฟรมที่บอกตำแหน่งและมุมเอียงของหัว โดยการอ้างนั้นจะบอกทิศทางของหัว โดยไม่ได้สนใจเซนเซอร์ว่าจะติดตั้งอย่างไร แต่ในวิทยานิพนธ์นี้ หุ่นยนต์ชีวามโนยด์ UTHAI ไม่มีหัว จึงไม่ได้ใช้เฟรมนี้

`torso` เป็นเฟรมที่ติดอยู่กับลำตัวช่วงล่างของหุ่นยนต์ชีวามโนยด์ โดยจะเป็นเฟรมที่ใช้เชื่อม ขา แขน หัว เข้าหากันได้

<sup>1</sup><http://www.ros.org/reps/rep-0000.html>

### 3.4.2 การแปลงข้อมูลให้อยู่ในรูปแบบ URDF

เมื่อออคแบบโครงสร้างของหุ่นยนต์ชีวามโนยด์ UTHAI ด้วยโปรแกรม Solidworks เสร็จแล้ว ต่อไปเป็นการนำเอาไฟล์ STL ออคมาเพื่อใช้ในการทำระบบจำลองการทำงานของหุ่นยนต์ โดยการใช้งานระบบจำลองเนื่องจากทำให้ผู้วิจัยสามารถที่จะเห็นการทำงานของหุ่นยนต์ชีวามโนยด์ได้ การสร้างแบบจำลองโดยการใช้เครื่องมือที่มาพร้อมกับ ROS ด้วยโมดูล URDF

#### 3.4.2.1 แพกเกจ ROS สำหรับสร้างแบบจำลอง

ROS ได้ให้เครื่องมือที่ช่วยให้ สามารถที่จะสร้างแบบจำลองของหุ่นยนต์ชีวามโนยด์สามมิติได้ เครื่องมือใน ROS ที่ชื่อว่า robot\_model ภายในมีแพกเกจต่างๆที่ใช้สำหรับสร้างแบบจำลองของหุ่นยนต์สามมิติอยู่อย่างครบถ้วน ทำให้เราสามารถทำงานได้สะดวก และรวดเร็วมากขึ้น

urdf เป็นหนึ่งในหลายแพกเกจที่อยู่ใน robot\_model, URDF เป็นไฟล์ XML ที่เอาไว้ใช้ประกอบลักษณะทางกายภาพของหุ่นยนต์ ซึ่งย่อมาจาก Unified Robot Description Format (URDF) การบอกรูปแบบของหุ่นยนต์ด้วย URDF จะใช้การบอกรูปแบบของหุ่นยนต์ในรูปแบบ XML ที่สามารถอ่านและแก้ไขได้ ทำให้เราสามารถนำเข้ามาใช้ใน ROS ได้

joint\_state\_publisher เครื่องมือนี้มีประโยชน์มากในการสร้างแบบจำลองหุ่นยนต์ด้วย URDF เนื่องจากสามารถนำตำแหน่งของข้อต่อ มาแสดงเป็น GUI ได้ ทำให้เราสามารถเลื่อนๆหมุนๆไปมาได้ อีกทั้งยังสามารถใช้งานร่วมกับโปรแกรมแสดงผลภาพ RViz ได้

robot\_state\_publisher เป็นเครื่องมือที่ใช้ในการ publish ตำแหน่งของก้านต่อต่างๆในแบบจำลองของหุ่นยนต์ชีวามโนยด์ออกมาใน TF อีกทั้งยังให้ความสัมพันธ์ระหว่างเฟรมของหุ่นยนต์ได้ด้วย

xacro ย่อมาจาก XML Macros หรือเราสามารถเรียกว่าเครื่องมือเสริมสำหรับ URDF ซึ่งลักษณะการเขียนเหมือนกับไฟล์ URDF แต่การเขียนนั้นจะสั้นกว่า อ่านง่ายกว่า และสามารถใช้เพื่อทำให้สร้างหุ่นยนต์ที่มีความซับซ้อนง่ายขึ้น สามารถแปลงไฟล์ xacro เป็น urdf ได้ถ้าต้องการ

#### 3.4.2.2 URDF

ในส่วนนี้จะเป็นการอธิบายระบบทางกลของหุ่นยนต์ชีวามโนยด์เป็นไฟล์ที่ใช้ร่วมกับ ROS เพื่อที่จะสามารถนำไปใช้กับระบบจำลองการทำงานของหุ่นยนต์ในอนาคตได้ ในการอธิบายระบบทางกลนั้นผู้วิจัยได้ใช้ไฟล์ URDF ซึ่งใช้ภาษาการเขียนเป็น XML ในการบอกรูปแบบของหุ่นยนต์

##### Link

ไฟล์ URDF แต่ละชิ้นส่วนของหุ่นยนต์เราจะเรียกว่า link และใน link จะประกอบไปด้วยส่วนย่อยๆ 3 ส่วนคือ `<inertia>` ที่เอาไว้บอกรถึงค่าตัวแปรทางฟิสิกส์, `<visual>` ที่เอาไว้แสดงผลให้เราเห็น, `<collision>` ที่เอาไว้ตรวจสอบว่าหุ่นยนต์มีการชนกันกับสิ่งแวดล้อมใหม่ ดังรูปที่ 3.4

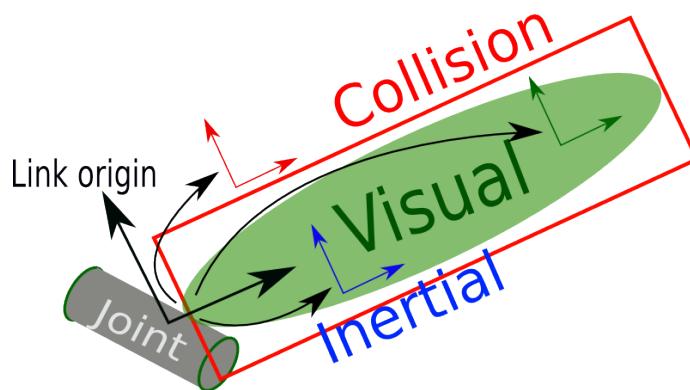
```

<link name="my_link">
  <inertia>
    <origin xyz="0 0 0.5" rpy="0 0 0"/>
    <mass value="1"/>
    <inertia ixx="100" ixy="0" ixz="0" iyy="100" iyz="0" izz="100"/>
  </inertia>
  <visual>
    <origin xyz="0 0 0" rpy="0 0 0"/>
    <geometry>
      <box size="1 1 1" />
    </geometry>
    <material name="Cyan">
      <color rgba="0 1.0 1.0 1.0"/>
    </material>
  </visual>
  <collision>
    <origin xyz="0 0 0" rpy="0 0 0"/>
    <geometry>
      <cylinder radius="1" length="0.5"/>
    </geometry>
  </collision>
</link>

```

รูปที่ 3.4: ตัวอย่าง link ใน urdf

ยังมีอีกหลายตัวที่ใช้ในการอธิบายแต่ละชิ้นส่วนของหุ่นยนต์ แต่ตัวอย่างเป็นเพียงแค่ส่วนหนึ่งเท่านั้น ในความเป็นจริงแล้วเราจะเขียน tags ต่างๆ ก็ตามที่เราต้องการ โดยใน URDF ไฟล์นั้นจะเอาไว้เก็บข้อมูลลักษณะเฉพาะของหุ่นยนต์เอาไว้ และยังสามารถใช้กับซอฟแวร์ตัวอื่นๆ อีกด้วย<sup>2</sup>



รูปที่ 3.5: การอธิบาย link ใน URDF ไฟล์

<sup>2</sup><http://wiki.ros.org/urdf>

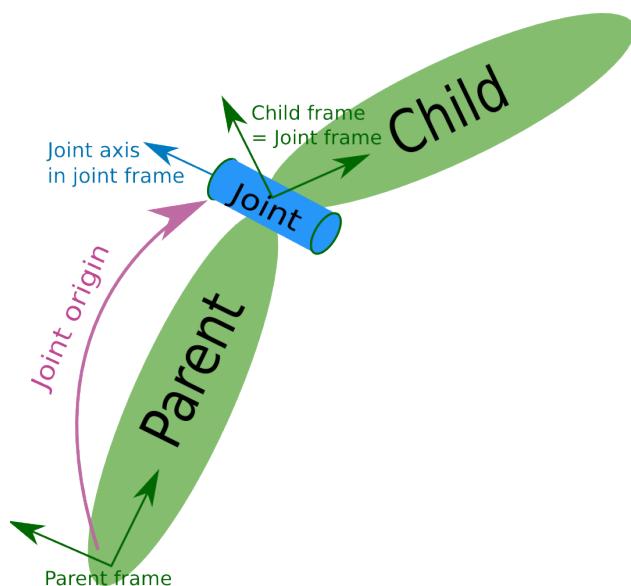
## Joint

อีกส่วนที่สำคัญสำหรับการสร้างไฟล์หุ่นยนต์ด้วย URDF ก็คือ Joint tag โดย tag นี้จะอธิบายถึงความสัมพันธ์ระหว่างก้านต่อสองอัน ส่วนนี้ไม่ได้มีเพียงแค่ทำข้อต่อให้เป็นแบบหมุนได้อย่างเดียว ยังมี Fix, Revolution, Linear และ Planar นอกเหนือจากนี้ เราจึงสามารถที่จะเพิ่มองศาสูงสุดต่ำสุดของข้อต่อ รวมไปถึง dynamic properties ต่างๆ ตามที่เห็นดังรูปที่ 3.6

```
<joint name="my_joint" type="floating">
    <origin xyz="0 0 1" rpy="0 0 3.1416"/>
    <parent link="link1"/>
    <child link="link2"/>
    <calibration rising="0.0"/>
    <dynamics damping="0.0" friction="0.0"/>
    <limit effort="30" velocity="1.0" lower="-2.2" upper="0.7"/>
    <safety_controller k_velocity="10" k_position="15"
        soft_lower_limit="-2.0" soft_upper_limit="0.5"/>
</joint>
```

รูปที่ 3.6: ตัวอย่าง joint ใน urdf

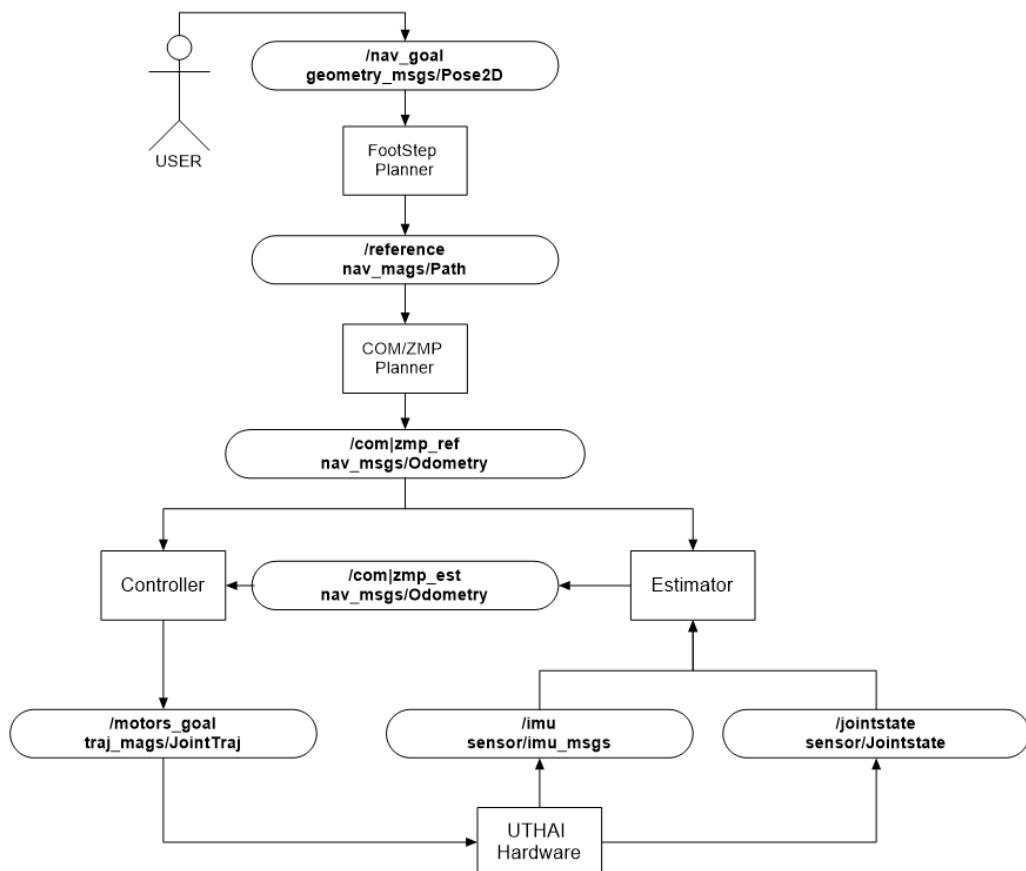
เมื่อเรานำ Joint และ Link มารวมกันเราจะต้องพิจารณาว่ามีว่างรูปแบบเป็นไปตามรูปที่ 3.7 โดยจะมีระยะระหว่างแกนของแต่ละข้อต่อ กับ ก้านต่อ ขึ้นส่วนแรกของการสร้างไฟล์ URDF จะมีชื่อว่า base\_link และเฟรม origin จะเป็นเฟรมอ้างอิง เมื่อเราต่อ Joint เข้ากับ Link จะเรียกว่า parent โดยเฟรม origin ของข้อต่อจะอยู่จุดเดียวกับเฟรม origin ของก้านต่อ ในสถานะเดียวกันก้านต่อที่นำมาต่อจากข้อต่อ เราจะเรียกว่า child และเฟรม origin ของก้านต่อ child จะอยู่ที่จุดเดียวกับเฟรม origin ของข้อต่อ



รูปที่ 3.7: การอธิบาย Joint ใน URDF ไฟล์

### 3.4.3 โครงสร้างการติดต่อสื่อสารระหว่าง Node ใน ROS

การติดต่อสื่อสารกันภายใน ROS นั้นจะใช้การส่ง message หากัน ซึ่ง message แต่ละตัวก็จะใช้ในงานที่ต่างกัน ตามระบบที่ต้องการส่ง จากรูปที่ 3.8 เป็นโครงสร้างการส่งข้อมูลหกันของหุ่นยนต์อิริวามานอยด์ ที่ผู้วิจัยได้ออกแบบไว้ โดยเริ่มจากผู้ใช้งานส่งตำแหน่งที่หุ่นยนต์จะต้องเดินไปเป็น Node ที่ทำการคำนวณและสร้างตำแหน่งการวางเท้าของหุ่นยนต์ และหลังจากนั้นจะส่งข้อมูลออกไปเป็น Path เส้นทางไปยัง Node ที่ทำการค้นหาตำแหน่งของ com, zmp ของหุ่นยนต์ เพื่อทำการควบคุมและส่งการหุ่นยนต์ต่อไป



รูปที่ 3.8: การติดต่อสื่อสารระหว่าง Node

#### การบอกตำแหน่งและมุมเอียง

การบอกตำแหน่งใน 3 มิติ Point คือการบอก  $x, y, z$  และการบومุมเอียงจะใช้ Quaternion ในการบอกโดยใช้ตัวแปรสีตัว คือ  $x,y,z,w$  หากนำทั้งสองมารวมกันเราจะเรียกว่า Pose

geometry_msgs/Point	
float64	x
float64	y
float64	z

ตารางที่ 3.1: Message Geometry Point

geometry_msgs/Quaternion	
float64	x
float64	y
float64	z
float64	w

ตารางที่ 3.2: Message Geometry Quaternion

geometry_msgs/Pose	
geometry_msgs/Point	position
geometry_msgs/Quaternion	orientation

ตารางที่ 3.3: Message Geometry Pose

### การบอกรความเร็วเชิงเส้นและเชิงมุม

การบอกรความเร็วเชิงเส้นใน 3 มิติ คือการบอกรความเร็วตามแนวแกน  $x, y, z$  และการบอกรความเร็วเชิงมุม คือการบอกรความเร็วการหมุนรอบแกน  $x, y, z$  หากนำทั้งสองมาร่วมกันเราจะเรียกว่า Twist

geometry_msgs/Vector3	
float64	x
float64	y
float64	z

ตารางที่ 3.4: Message Geometry Vector3

geometry_msgs/Twist	
geometry_msgs/Vector3	linear
geometry_msgs/Vector3	angular

ตารางที่ 3.5: Message Geometry Twist

### การบอกรตำแหน่งและความเร็ว

หากนำทั้งสองมาร่วมกันจะรู้ว่า ตำแหน่ง (Pose) และความเร็ว (Twist) เราจะเรียกว่า Odometry แต่ที่เพิ่มเข้ามาคือ Covariance ซึ่งอาจทำให้เกิดความสับสนได้

nav_msgs/Odometry	
std_msgs/Header	header
geometry_msgs/PoseWithCovariance	pose
geometry_msgs/TwistWithCovariance	twist

ตารางที่ 3.6: Message Navigation Odometry

### ตำแหน่งของหุ่นยนต์

การบอกตำแหน่งของหุ่นยนต์บนระนาบ 2 มิติ คือการบอก  $x$ ,  $y$  และ  $\theta$  การบอกนั้นจะบอกว่าตำแหน่งที่หุ่นยนต์อยู่นั้นอยู่ตรงไหนหากเทียบกับแผนที่ รวมไปถึงตำแหน่งของหุ่นยนต์ที่ต้องการจะเดินไปด้วย ซึ่งอ้างอิงมาจากการที่หุ่นยนต์มีความสามารถในการรับและส่งข้อมูลทางบูติค เช่น บอทหุ่นยนต์ที่ต้องการเดินไปตามเส้นทางที่กำหนดให้

geometry_msgs/Pose2D	
float64	$x$
float64	$y$
float64	$\theta$

ตารางที่ 3.7: Message Geometry Pose2D

### ตำแหน่งการวางแผนเดินของหุ่นยนต์

การจะให้หุ่นยนต์นำเท้าไปวางในตำแหน่งที่เราต้องการจากที่ได้จากการคำนวณนั้น จะต้องบอกตำแหน่งและบอกรูปแบบของจุดที่จะไป จากการสร้างจะได้เป็นรายการของเท้าซ้ายและขวา โดยอิงจาก ตารางที่ 3.3

nav_msgs/Path	
std_msgs/Header	header
geometry_msgs/PoseStamped[]	poses

ตารางที่ 3.8: Message Navigation Path

geometry_msgs/PoseStamped	
std_msgs/Header	header
geometry_msgs/Pose	pose

ตารางที่ 3.9: Message Geometry PoseStamped

### ตำแหน่งจุดศูนย์กลางมวลของหุ่นยนต์

ใน Message นี้ใช้อยู่ 2 ที่คือ Message ที่ได้จากการวางแผนของ Node CoM Planner และ Node CoM Estimator โดยทั้งสองจุดใช้ Message เมื่อกันส่งไปยัง Controller เพื่อควบคุมท่าทางต่างๆของหุ่นยนต์ต่อไป Message ที่ใช้คือ Message จากตารางที่ 3.6

nav_msgs/Odometry	
std_msgs/Header	header
geometry_msgs/PoseWithCovariance	pose
geometry_msgs/TwistWithCovariance	twist

Message Navigation Odometry

### การควบคุมข้อต่อของหุ่นยนต์

ในการควบคุมข้อต่อแต่ละข้อของหุ่นยนต์ชีวามาโนyd'n'จะใช้ Message trajectory\_msgs/JointTrajectory ซึ่งสามารถส่ง ตำแหน่ง ความเร็ว ความเร่ง และ แรงบิด ไปได้ ทำให้หากต้องการเปลี่ยนระบบใหม่สามารถทำได้โดยง่าย

trajectory_msgs/JointTrajectory	
std_msgs/Header	header
string[]	joint_names
trajectory_msgs/JointTrajectoryPoint[]	points

ตารางที่ 3.10: Message Trajectory JointTrajectory

trajectory_msgs/JointTrajectoryPoint	
float64[]	positions
float64[]	velocities
float64[]	accelerations
float64[]	effort
duration	time_from_start

ตารางที่ 3.11: Message Trajectory JointTrajectoryPoint

### ค่าเซนเซอร์ข้อต่อของหุ่นยนต์

ที่ข้อต่อของหุ่นยนต์ชีวามาโนyd'm'i'เซนเซอร์ที่เอาไว้ใช้ในการอ่านค่าตำแหน่ง ความเร็ว และแรง อยู่ด้วย เราสามารถที่จะใช้ Message sensor\_msgs/JointState สำหรับอ่านค่าตำแหน่ง ความเร็ว แรง ของตัวขับเคลื่อนแล้วส่งให้ Estimator Node ได้

sensor_msgs/JointState	
std_msgs/Header	header
float64[]	position
float64[]	velocity
float64[]	effort

ตารางที่ 3.12: Message Sensor JointState

### ค่าเซนเซอร์ผ้าเท้าของหุ่นยนต์

ที่ผ้าเท้าของหุ่นยนต์ชีวามาโนyd'm'i'เซนเซอร์ที่เอาไว้ใช้ในการอ่าน แรงกดที่ผ้าเท้า ใช้ในการเอามากกว่า เท้าสัมผัสพื้นหรือไม่

geometry_msgs/Wrench	
geometry_msgs/Vector3	force
geometry_msgs/Vector3	torque

ตารางที่ 3.13: Message Geometry Wrench

### ค่าเซนเซอร์ IMU ของทุนยนต์

เซนเซอร์ IMU เป็นเซนเซอร์ที่เอาไว้ใช้ในการวัด ความเร็วเชิงมุม และ ความเร่งเชิงเส้น หากนำทั้งคู่มารวมกันจะสามารถที่จะแปลงให้วัดมุมอิริยาบถของเซนเซอร์ได้ โดยจะใช้ Message std\_msgs/Imu ในการส่งให้ Node Estimator จากตัวทุนยนต์

sensor_msgs/Imu	
std_msgs/Header	header
geometry_msgs/Quaternion float64[9]	orientation
geometry_msgs/Vector3 float64[9]	orientation_covariance
geometry_msgs/Vector3 float64[9]	angular_velocity
	angular_velocity_covariance
	linear_acceleration
	linear_acceleration_covariance

ตารางที่ 3.14: Message Sensor Imu

sensor_msgs/MagneticField	
std_msgs/Header	header
geometry_msgs/Vector3 float64[9]	magnetic_field
	magnetic_field_covariance

ตารางที่ 3.15: Message Sensor MagneticField

## เอกสารอ้างอิง

- [1] Discover nao.
- [2] นายชาญชัย ชัยสุขโภศล. การสังเคราะห์โปรแกรมควบคุมหุ่นยนต์เดินสองขา แบบสมดุลสถิตโดยอัตโนมัติ ด้วยการคำนวณเชิงวิวัฒน์. <http://ezproxy.car.chula.ac.th:2074/bitstream/123456789/11758/1/chanchai.pdf>, 2544.
- [3] นายปรีดา เลิศพงศ์วิภูษณะ. การออกแบบและสร้างหุ่นยนต์สองขาและการสังเคราะห์โปรแกรมการเดิน. <http://ezproxy.car.chula.ac.th:2074/bitstream/123456789/6246/1/Preeda.pdf>, 2546.
- [4] ยศวีร์ แก้วมณี. การสร้างรูปแบบการเดินส่วนขาของหุ่นยนต์คล้ายมนุษย์โดยเลียนแบบจากภาพท่าทางของมนุษย์. <http://kb.psu.ac.th/psukb/bitstream/2010/5569/1/307650.pdf>, 2552.
- [5] นส.สุนัณญา จริยาวัฒนากุล. การควบคุมหุ่นยนต์เชมิชิวามอยด์ระยะไกล. doi.nrct.go.th, 2556.
- [6] Humanoid history - wabot-, 9 1970.
- [7] Darwin-op, 9 2010.
- [8] Antoine Petit Chairman. Poppy-project, 9 2016.
- [9] Antoine Petit Chairman and CEO of Inria. Poppy-project, 9 2016.
- [10] Albert Einstein. Zur elektrodynamik bewegter körper. (german) [on the electrodynamics of moving bodies]. 322(10):891–921, 1905.
- [11] Michel Goossens, Frank Mittelbach, and Alexander Samarin. The  $\text{\LaTeX}$  Companion. Addison-Wesley, 1993.
- [12] Seward D.W. A. Bradshaw F. Margrave. The anatomy of a humanoid robot. pages 437–443, 1995.
- [13] None. Darpa web site, June 2015.
- [14] Ng Buck Sin. Robo-erectus tr-2010 teensize team description paper. 2010.
- [15] Consortium the RobotCub. icub.org - an opensource cognitive humanoid robotic platform, 9 2017.
- [16] Hayashibara Yasuo. Cit brains (kid size league). 2015.

ภาคผนวก

## ภาคผนวก ก

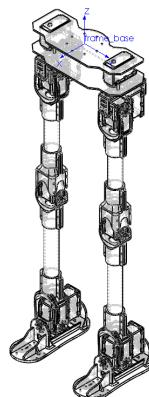
### ข้อมูลเบื้องต้นของหุ่นยนต์ชีวมานอยด์ UTHAI

#### ก.1 ค่าคุณสมบัติทางพลศาสตร์

ข้อมูลพลศาสตร์ของหุ่นยนต์ชีวมานอยด์ UTHAI ซึ่งจะนำไปใช้ในการทำระบบจำลองด้วยโปรแกรม Gazebo ใน ROS และใช้ในการคำนวณทางคณิตศาสตร์เพื่อทำให้การเดินมีเสถียรภาพ โดยข้อมูลดูดันนี้ได้มาจากการคำนวณ Mass Properties ในโปรแกรม SolidWorks และปรับมีค่าใกล้เคียงกับของจริงโดยการเทียบกับเครื่องซึ่งน้ำหนัก

ข้อมูลดูดันนี้ประกอบไปด้วย มวล จุดศูนย์กลางมวล และโมเมนต์ความเฉื่อย อีกทั้งข้อมูลยังบอกในมาตรฐาน URDF กับ DH-Parameter ซึ่งทำให้ใช้งานในระบบการคำนวณที่ต่างกันได้

Overall Humanoid

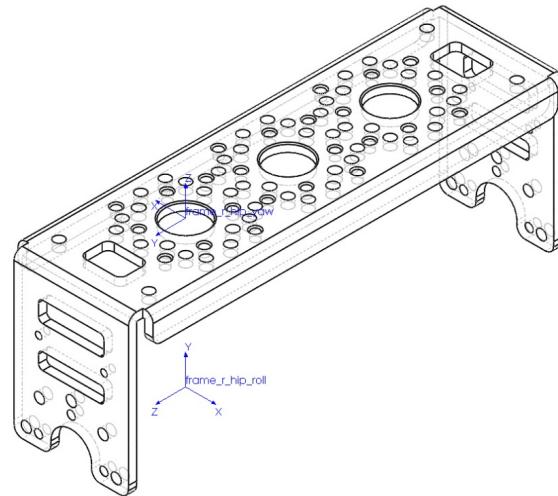


รูปที่ ก.1: ภาพแสดงช่วงล่างทั้งตัว

Link	All Link
Mass (kg)	3.31477475
CoM X (m)	-0.00855772
CoM Y (m)	0.00000000
CoM Z (m)	-0.33375492
Inertia Ixx	0.28641029
Inertia Ixy	-0.00000302
Inertia Ixz	-0.00048106
Inertia Iyy	0.26207601
Inertia Iyz	-0.00061103
Inertia Izz	0.02925799

ตารางที่ ก.1: ตารางแสดงค่าพารามิเตอร์ทั้งตัว

## Right Hip Yaw



รูปที่ ก.2: ภาพแสดงก้านต่อ Right Hip Yaw

Link	r_hip_yaw
Mass (kg)	0.09100000
CoM X (m)	0.00000000
CoM Y (m)	0.02864983
CoM Z (m)	-0.02500000
Inertia Ixx	0.00014158
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00014316
Inertia Iyz	0.00000000
Inertia Izz	0.00002022

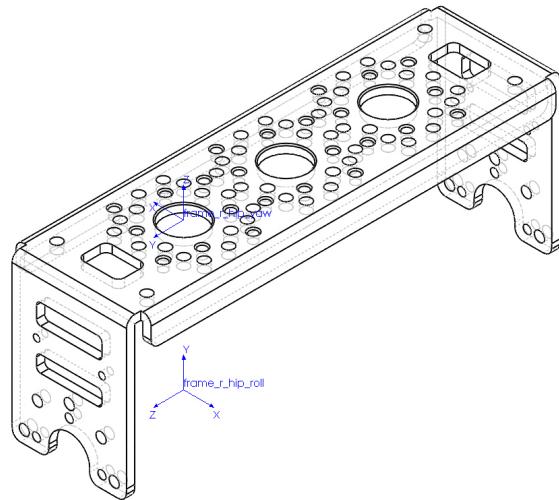
(ก) DH Parameter

Link	r_hip_yaw
Mass (kg)	0.09100000
CoM X (m)	0.00000000
CoM Y (m)	-0.02500000
CoM Z (m)	-0.00735017
Inertia Ixx	0.00014158
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00002022
Inertia Iyz	0.00000000
Inertia Izz	0.00014316

(ข) URDF

ตารางที่ ก.2: ตารางแสดงค่าพารามิเตอร์ Right Hip Yaw

## Left Hip Yaw



รูปที่ ก.3: ภาพแสดงก้านต่อ Left Hip Yaw

Link	l_hip_yaw
Mass (kg)	0.09100000
CoM X (m)	0.00000000
CoM Y (m)	0.02864983
CoM Z (m)	-0.02500000
Inertia Ixx	0.00014158
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00014316
Inertia Iyz	0.00000000
Inertia Izz	0.00002022

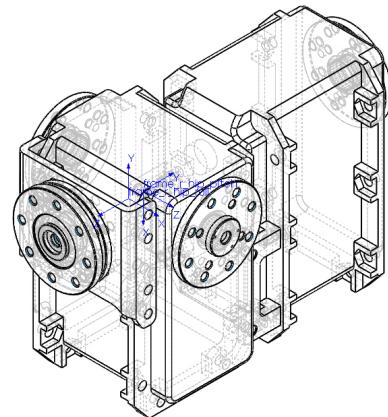
(ก) DH Parameter

Link	l_hip_yaw
Mass (kg)	0.09100000
CoM X (m)	0.00000000
CoM Y (m)	0.02500000
CoM Z (m)	-0.00735017
Inertia Ixx	0.00014158
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00002022
Inertia Iyz	0.00000000
Inertia Izz	0.00014316

(ข) URDF

ตารางที่ ก.3: ตารางแสดงค่าพารามิเตอร์ Left Hip Yaw

## Right Hip Roll



รูปที่ ก.4: ภาพแสดงก้านต่อ Right Hip Roll

Link	r_hip_roll
Mass (kg)	0.34300000
CoM X (m)	0.01526237
CoM Y (m)	0.02152630
CoM Z (m)	0.00000000
Inertia Ixx	0.00026846
Inertia Ixy	0.00000219
Inertia Ixz	-0.00000081
Inertia Iyy	0.00014760
Inertia Iyz	0.00000000
Inertia Izz	0.00032448

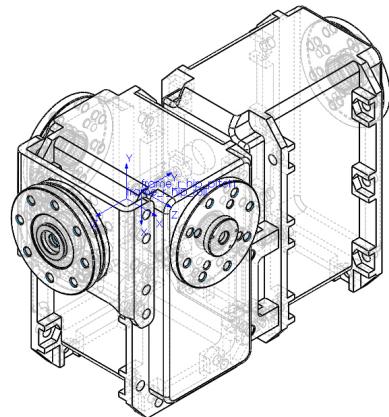
(ก) DH Parameter

Link	r_hip_roll
Mass (kg)	0.34300000
CoM X (m)	0.00000000
CoM Y (m)	-0.01526237
CoM Z (m)	-0.02652630
Inertia Ixx	0.00032448
Inertia Ixy	0.00000081
Inertia Ixz	0.00000000
Inertia Iyy	0.00026846
Inertia Iyz	0.00000219
Inertia Izz	0.00014760

(ข) URDF

ตารางที่ ก.4: ตารางแสดงค่าพารามิเตอร์ Right Hip Roll

## Left Hip Roll



รูปที่ ก.5: ภาพแสดงก้านต่อ Left Hip Roll

Link	l_hip_roll
Mass (kg)	0.34300000
CoM X (m)	0.01526237
CoM Y (m)	0.02152630
CoM Z (m)	0.00000000
Inertia Ixx	0.00026846
Inertia Ixy	0.00000219
Inertia Ixz	-0.00000081
Inertia Iyy	0.00014760
Inertia Iyz	0.00000000
Inertia Izz	0.00032448

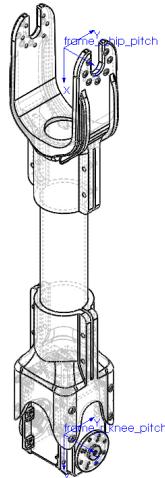
(ก) DH Parameter

Link	l_hip_roll
Mass (kg)	0.34300000
CoM X (m)	0.00000000
CoM Y (m)	-0.01526237
CoM Z (m)	-0.02652630
Inertia Ixx	0.00032448
Inertia Ixy	0.00000081
Inertia Ixz	0.00000000
Inertia Iyy	0.00026846
Inertia Iyz	0.00000219
Inertia Izz	0.00014760

(ข) URDF

ตารางที่ ก.5: ตารางแสดงค่าพารามิเตอร์ Left Hip Roll

## Right Hip Pitch



รูปที่ ก.6: ภาพแสดงก้านต่อ Right Hip Pitch

Link	r_hip_pitch
Mass (kg)	0.31800000
CoM X (m)	-0.07862011
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000078
Inertia Iyy	0.00254669
Inertia Iyz	0.00000000
Inertia Izz	0.00250848

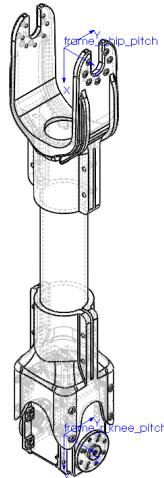
(ก) DH Parameter

Link	r_hip_pitch
Mass (kg)	0.31800000
CoM X (m)	0.22137989
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000078
Inertia Iyy	0.00254669
Inertia Iyz	0.00000000
Inertia Izz	0.00250848

(ข) URDF

ตารางที่ ก.6: ตารางแสดงค่าพารามิเตอร์ Right Hip Pitch

## Left Hip Pitch



รูปที่ ก.7: ภาพแสดงก้านต่อ Left Hip Pitch

Link	l_hip_pitch
Mass (kg)	0.31800000
CoM X (m)	-0.07862011
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000078
Inertia Iyy	0.00254669
Inertia Iyz	0.00000000
Inertia Izz	0.00250848

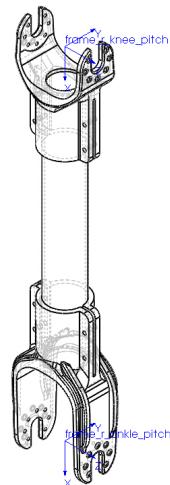
(ก) DH Parameter

Link	l_hip_pitch
Mass (kg)	0.31800000
CoM X (m)	0.22137989
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000078
Inertia Iyy	0.00254669
Inertia Iyz	0.00000000
Inertia Izz	0.00250848

(ข) URDF

ตารางที่ ก.7: ตารางแสดงค่าพารามิเตอร์ Left Hip Pitch

### Right Knee Pitch



รูปที่ ก.8: ภาพแสดงก้านต่อ Right Knee Pitch

Link	r_knee_pitch
Mass (kg)	0.13800000
CoM X (m)	-0.15211782
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00127592
Inertia Iyz	0.00000000
Inertia Izz	0.00124960

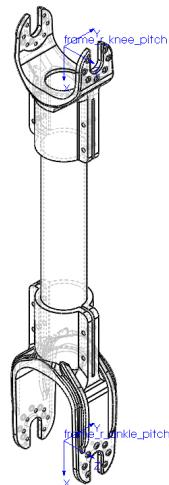
(ก) DH Parameter

Link	r_knee_pitch
Mass (kg)	0.13800000
CoM X (m)	0.16288218
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00005794
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00127592
Inertia Iyz	0.00000000
Inertia Izz	0.00124960

(ข) URDF

ตารางที่ ก.8: ตารางแสดงค่าพารามิเตอร์ Right Knee Pitch

## Left Knee Pitch



รูปที่ ก.9: ภาพแสดงก้านต่อ Left Knee Pitch

Link	l_knee_pitch
Mass (kg)	0.13800000
CoM X (m)	-0.15211782
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00011525
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00127592
Inertia Iyz	0.00000000
Inertia Izz	0.00124960

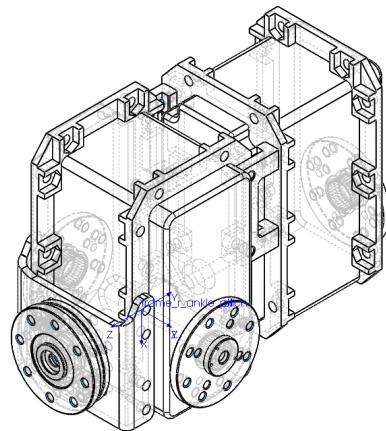
(ก) DH Parameter

Link	l_knee_pitch
Mass (kg)	0.13800000
CoM X (m)	0.16288218
CoM Y (m)	0.00000000
CoM Z (m)	0.00000000
Inertia Ixx	0.00005794
Inertia Ixy	0.00000000
Inertia Ixz	0.00000000
Inertia Iyy	0.00127592
Inertia Iyz	0.00000000
Inertia Izz	0.00124960

(ข) URDF

ตารางที่ ก.9: ตารางแสดงค่าพารามิเตอร์ Left Knee Pitch

## Right Ankle Pitch



รูปที่ ก.10: ภาพแสดงก้านต่อ Right Ankle Pitch

Link	r_ankle_pitch
Mass (kg)	0.33138738
CoM X (m)	-0.01526237
CoM Y (m)	0.00000000
CoM Z (m)	-0.02152630
Inertia Ixx	0.00025937
Inertia Ixy	0.00000000
Inertia Ixz	0.00000079
Inertia Iyy	0.00031349
Inertia Iyz	0.00000000
Inertia Izz	0.00014261

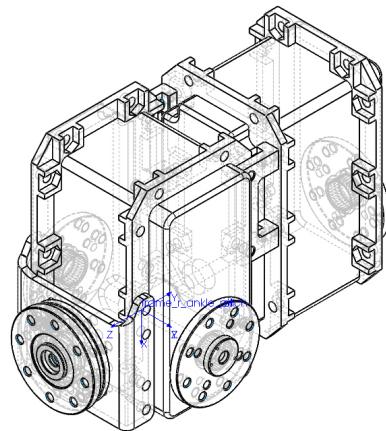
(ก) DH Parameter

Link	r_ankle_pitch
Mass (kg)	0.33138738
CoM X (m)	-0.01526237
CoM Y (m)	0.02152630
CoM Z (m)	0.00000000
Inertia Ixx	0.00025937
Inertia Ixy	0-0.00000212
Inertia Ixz	0.00000079
Inertia Iyy	0.00014261
Inertia Iyz	0.00000000
Inertia Izz	0.00031349

(ข) URDF

ตารางที่ ก.10: ตารางแสดงค่าพารามิเตอร์ Right Ankle Pitch

## Left Ankle Pitch



รูปที่ ก.11: ภาพแสดงก้านต่อ Left Ankle Pitch

Link	l_ankle_pitch
Mass (kg)	0.33138738
CoM X (m)	-0.01526237
CoM Y (m)	0.00000000
CoM Z (m)	-0.02152630
Inertia Ixx	0.00025937
Inertia Ixy	0.00000000
Inertia Ixz	0.00000079
Inertia Iyy	0.00031349
Inertia Iyz	0.00000000
Inertia Izz	0.00014261

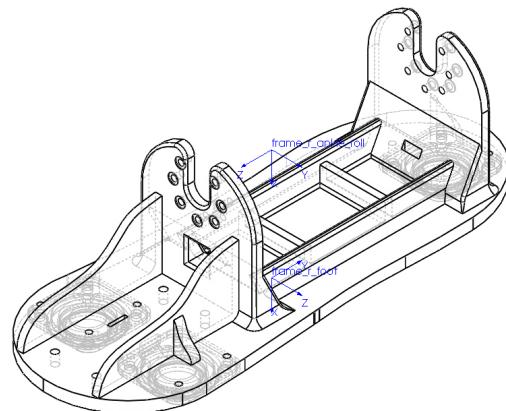
(ก) DH Parameter

Link	l_ankle_pitch
Mass (kg)	0.33138738
CoM X (m)	-0.01526237
CoM Y (m)	0.02152630
CoM Z (m)	0.00000000
Inertia Ixx	0.00025937
Inertia Ixy	0-0.00000212
Inertia Ixz	0.00000079
Inertia Iyy	0.00014261
Inertia Iyz	0.00000000
Inertia Izz	0.00031349

(ข) URDF

ตารางที่ ก.11: ตารางแสดงค่าพารามิเตอร์ Left Ankle Pitch

## Right Ankle Roll



รูปที่ ก.12: ภาพแสดงก้านต่อ Right Ankle Roll

Link	r_ankle_roll
Mass (kg)	0.10500000
CoM X (m)	-0.01454118
CoM Y (m)	-0.00034576
CoM Z (m)	-0.00019548
Inertia Ixx	0.00034591
Inertia Ixy	-0.00000857
Inertia Ixz	-0.00000013
Inertia Iyy	0.00004813
Inertia Iyz	-0.00000120
Inertia Izz	0.00032705

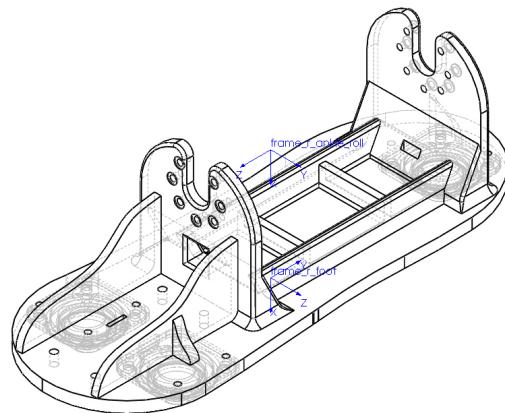
(ก) DH Parameter

Link	r_ankle_roll
Mass (kg)	0.10500000
CoM X (m)	0.03625882
CoM Y (m)	-0.00019548
CoM Z (m)	0.00034576
Inertia Ixx	0.00034591
Inertia Ixy	-0.00000013
Inertia Ixz	0.00000857
Inertia Iyy	0.00032705
Inertia Iyz	0.00000120
Inertia Izz	0.00004813

(ข) URDF

ตารางที่ ก.12: ตารางแสดงค่าพารามิเตอร์ Right Ankle Roll

## Left Ankle Roll



รูปที่ ก.13: ภาพแสดงก้านต่อ Left Ankle Roll

Link	l_ankle_roll
Mass (kg)	0.10500000
CoM X (m)	-0.01454118
CoM Y (m)	-0.00034576
CoM Z (m)	-0.00019548
Inertia Ixx	0.00034591
Inertia Ixy	-0.00000857
Inertia Ixz	-0.00000013
Inertia Iyy	0.00004813
Inertia Iyz	-0.00000120
Inertia Izz	0.00032705

(ก) DH Parameter

Link	l_ankle_roll
Mass (kg)	0.10500000
CoM X (m)	0.03625882
CoM Y (m)	-0.00019548
CoM Z (m)	0.00034576
Inertia Ixx	0.00034591
Inertia Ixy	-0.00000013
Inertia Ixz	0.00000857
Inertia Iyy	0.00032705
Inertia Iyz	0.00000120
Inertia Izz	0.00004813

(ข) URDF

ตารางที่ ก.13: ตารางแสดงค่าพารามิเตอร์ Left Ankle Roll

## ประวัติผู้เขียน

นายจิรภูร์ ศรีรัตนอาภรณ์



ชื่อ สกุล

รหัสนักศึกษา

วุฒิการศึกษา

ชื่อสถาบัน

ปีที่สำเร็จการศึกษา

นายจิรภูร์ ศรีรัตนอาภรณ์

57340500067

วิศวกรรมศาสตรบัณฑิต

วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

2560

## ประวัติผู้เขียน

นายเจษฎากร ท่าไชยวงศ์



ชื่อ สกุล	นายเจษฎากร ท่าไชยวงศ์
รหัสนักศึกษา	57340500067
วุฒิการศึกษา	วิศวกรรมศาสตรบัณฑิต
ชื่อสถาบัน	วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
ปีที่สำเร็จการศึกษา	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
	2560

## ประวัติผู้เขียน

นายวุฒิภัทร โชคอนันตทรัพย์



ชื่อ สกุล

รหัสนักศึกษา

วุฒิการศึกษา

ชื่อสถานบัน

ปีที่สำเร็จการศึกษา

นายวุฒิภัทร โชคอนันตทรัพย์

57340500067

วิศวกรรมศาสตรบัณฑิต

วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

2560