Structure Design and Platform Development of Universal Template for Humanoid Algorithm Interface (UTHAI)

การออกแบบโครงสร้างและพัฒนาระบบพื้นฐานสำหรับหุ่นยนต์ฮิวมานอยด์เพื่อการศึกษาและวิจัย

นายจิรัฏฐ์ ศรีรัตนอาภรณ์

นายเจษฎากร ทาไชยวงค์

นายวุฒิภัทร โชคอนันตทรัพย์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมหุ่นยนต์และระบบอัตโนมัติ

สถาบันวิทยาการหุ่นยนต์ภาคสนาม

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

ปีการศึกษา 2560

Structure Design and Platform Development of Universal Template for Humanoid Algorithm Interface (UTHAI)

การออกแบบโครงสร้างและพัฒนาระบบพื้นฐานสำหรับหุ่นยนต์ฮิวมานอยด์เพื่อการศึกษาและวิจัย

นายจิรัฏฐ์ ศรีรัตนอาภรณ์
นายเจษฎากร ทาไชยวงค์
นายวุฒิภัทร โชคอนันตทรัพย์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
สถาบันวิทยาการหุ่นยนต์ภาคสนาม
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
ปีการศึกษา 2560

การออกแบบโครงสร้างและพัฒนาระบบพื้นฐานสำหรับหุ่นยนต์ฮิวมานอยด์
เพื่อการศึกษาและวิจัย

นายจิรัฏฐ์ ศรีรัตนอาภรณ์
นายเจษฎากร ทาไชยวงค์
นายวุฒิภัทร โชคอนันตทรัพย์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
สถาบันวิทยาการหุ่นยนต์ภาคสนาม
ปีการศึกษา 2560

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

.....
(นายธนัชชา ชูพจน์เจริญ)

อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม

.....
(รศ.ดร.ชิต เหล่าวัฒนา)

คณะกรรมการสอบ

.....ประธานกรรมการ
(ดร.อาบทิพย์ ธีรวงศ์กิจ)

.....กรรมการ
(ดร.ปิติวุฒญ์ ธีรกิตติกุล)

.....กรรมการ
(ดร.สุภชัย วงศ์บุณย์ยง)

ชื่อวิทยานิพนธ์	การออกแบบโครงสร้างและพัฒนาระบบพื้นฐานสำหรับหุ่นยนต์ฮิวมานอยด์เพื่อการศึกษาและวิจัย
หน่วยกิต	6
ผู้เขียน	นายจิรัฏฐ์ ศรีรัตนอาภรณ์ นายเจษฎากร ทาไชยวงค์ นายวุฒิภัทร โชคอนันตทรัพย์
อาจารย์ที่ปรึกษา	ที่ปรึกษาวิทยานิพนธ์หลัก นายธนัชชา ชูพจน์เจริญ ที่ปรึกษาวิทยานิพนธ์ร่วม รศ.ดร.ชิต เหล่าวัฒนา
หลักสูตร	วิศวกรรมศาสตรบัณฑิต
สาขาวิชา	วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
คณะ	สถาบันวิทยาการหุ่นยนต์ภาคสนาม
ปีการศึกษา	2560

บทคัดย่อ

สำหรับเขียนบทคัดย่อ

กิตติกรรมประกาศ

ขอขอบพระคุณอาจารย์ ธนัชชา ชูพจน์เจริญ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก ที่ได้สละเวลามาให้คำปรึกษา ชี้แนะแนวทาง ให้ความรู้ในด้านต่างๆ ที่จำเป็นต่องานวิจัย รวมถึงการให้การสนับสนุนในเรื่องอุปกรณ์ในการทำวิจัย ตลอดจนช่วยตรวจและแก้ไขวิทยานิพนธ์ให้เป็นไปอย่างสมบูรณ์

ขอขอบพระคุณรองศาสตราจารย์ ดร.ชิต เหล่าวัฒนา อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่ได้ชี้แนะแนวทาง ให้คำแนะนำ และให้เกียรติเข้าร่วมการสอบวิทยานิพนธ์

ขอขอบพระคุณอาจารย์ ดร.ถวิดา มณีวรรณ และนายวิษณุ จูธารี ที่ได้ให้คำแนะนำในการแก้ไขปัญหาด้านต่างๆ ที่เกิดขึ้นระหว่างการทำการวิจัย และได้ให้การสนับสนุนอุปกรณ์สำคัญที่ใช้ในการทำวิจัย

ขอขอบพระคุณอาจารย์ อาบทิพย์ ธีรวงศ์กิจ ที่กรุณาให้เกียรติเป็นประธานกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการจัดทำวิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณอาจารย์ ดร.ปิติวุฒญ์ ธีรกิตติกุล ที่กรุณาให้เกียรติเป็นกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการวิจัย และการแก้ไขปรับปรุงงานวิจัย ตลอดจนตรวจแก้วิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณอาจารย์ ดร.สุภชัย วงศ์บุณย์ยง ที่กรุณาให้เกียรติเป็นกรรมการสอบวิทยานิพนธ์ ให้คำแนะนำที่เป็นประโยชน์ต่อการวิจัย และการแก้ไขปรับปรุงงานวิจัย ตลอดจนตรวจแก้วิทยานิพนธ์ให้ดำเนินไปอย่างสมบูรณ์

ขอขอบพระคุณคณาจารย์ และบุคลากรในสถาบันวิทยาการหุ่นยนต์ภาคสนามทุกท่าน ที่ได้ให้คำปรึกษาและช่วยเหลือด้านสถานที่พร้อมทั้งสิ่งอำนวยความสะดวกต่างๆ ในระหว่างการทำวิทยานิพนธ์

ขอขอบคุณนักศึกษาปริญญาตรี สถาบันวิทยาการหุ่นยนต์ภาคสนามทุกท่าน ที่ได้ให้คำแนะนำ ถามไถ่ และเป็นกำลังใจมาโดยตลอด

และสุดท้ายนี้ ขอน้อมรำลึกถึงพระคุณบิดา มารดา และครอบครัว ที่ส่งเสริมให้กำลังใจ และให้การสนับสนุนในเรื่องต่างๆ จนกระทั่งข้าพเจ้าประสบความสำเร็จในการศึกษา

นายจิรัฏฐ์ ศรีรัตนอาภรณ์
นายเจษฎากร ทาไชยวงค์
นายวุฒิภัทร โชคอนันตทรัพย์

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

รายการตาราง

ตาราง หน้า

รายการรูปภาพ

รูป หน้า

รายการสัญลักษณ์

θ	เซต้า
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
Liews	วุฒิภัทร โชคอนันตทรัพย์
θ	เซต้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

หุ่นยนต์ฮิวมานอยด์เป็นหุ่นยนต์ที่สร้างขึ้นเพื่อเลียนแบบสรีระร่างกายของมนุษย์ ซึ่งมีข้อจำนวนมากเพื่อให้มีการเคลื่อนไหวคล้ายมนุษย์ ลักษณะเด่นของหุ่นยนต์ฮิวมานอยด์คือ การเคลื่อนที่ด้วยขาสองข้าง ด้วยการเคลื่อนที่โดยการใช้ขานั้น ทำให้หุ่นยนต์ฮิวมานอยด์สามารถเคลื่อนที่ได้ อย่างคล่องแคล่วในทุกสภาพพื้นผิว ทั้งทางเรียบ ทางขรุขระหรือพื้นต่างระดับ [1] ซึ่งนั่นทำให้หุ่นยนต์ที่เดินสองขาแตกต่างจากหุ่นยนต์ที่เคลื่อนที่ด้วยล้อ ด้วยโครงสร้างของหุ่นยนต์ที่คล้ายมนุษย์นั้นเอง จึงทำให้หุ่นยนต์ฮิวมานอยด์สามารถทำงานได้หลากหลายและยืดหยุ่น สามารถที่จะใช้อุปกรณ์ทั่วไปที่ถูกออกแบบขึ้นมาเพื่อใช้กับมนุษย์ได้ ซึ่งหมายความว่าในอนาคตนั้นหุ่นยนต์ฮิวมานอยด์สามารถที่จะทำงานทดแทนแรงงานของมนุษย์ได้ [2] งานที่หุ่นยนต์ฮิวมานอยด์จะเข้ามาทดแทนแรงงานของมนุษย์นั้น จะเป็นงานที่ต้องทำซ้ำๆ จนเกินความเมื่อยล้า งานที่อยู่ในพื้นที่อันตรายหรือที่เสี่ยงต่อการเกิดอุบัติเหตุ

สถาบันวิจัยหลายแห่งทั่วโลกกำลังให้ความสนับสนุนด้านการศึกษาวิจัยและพัฒนาหุ่นยนต์ฮิวมานอยด์เพื่อให้ทำภารกิจต่างๆ ยกตัวอย่างเช่น DARPA Robotics Challenge (DRC) [3] เป็นรายการแข่งขันหุ่นยนต์กึ่งอัตโนมัติเพื่อทำภารกิจกู้ภัยในสถานการณ์ภัยพิบัติที่อันตราย ซึ่งสถาบันวิจัยหุ่นยนต์ทั่วโลกได้ส่งหุ่นยนต์ฮิวมานอยด์ของตนเข้าร่วมการแข่งขัน ในปัจจุบันได้มีการพัฒนาหุ่นยนต์ฮิวมานอยด์ขึ้นมาหลากหลายตัวเช่น ASIMO, HRP-3, LOLA และ WATHLETE-1 การพัฒนาหุ่นยนต์ฮิวมานอยด์นั้นได้ก่อให้เกิดงานศึกษาวิจัย และทฎษฎีต่อยอด ต่างๆมากมาย เช่น การวางแผนการเดิน การเดินแบบสถิต การเดินแบบพลวัต การติดต่อสื่อสารของระบบ การมองเห็นและการประมวลผลภาพ การพูดคุยโต้ตอบกับมนุษย์ ปัญญาประดิษฐ์ ฯลฯ ซึ่งงานวิจัยเหล่านี้สามารถที่จะนำไปประยุกต์ใช้กับระบบหุ่นยนต์ระบบอื่นๆได้ แม้ว่าจะมีการพัฒนาหุ่นยนต์ฮิวมานอยด์มามากมายแล้ว แต่การเริ่มต้นทำงานวิจัยที่มีความเกี่ยวข้องกับหุ่นยนต์ฮิวมานอยด์นั้น ต้องใช้ความรู้ความสามารถ เครื่องมือ ระยะเวลา งบประมาณ และ ความพยายามเป็นอย่างมาก การสร้างหุ่นยนต์ฮิวมานอยด์ขึ้นมาใหม่นั้นต้องใช้งบประมาณสูง ดั้งนั้นการสร้างระบบจำลองของหุ่นยนต์ฮิวมานอยด์ขึ้นมาเป็นระบบพื้นฐาน ให้มีความพร้อมสำหรับการพัฒนาต่อยอดแก่นักศึกษาหรือนักวิจัย จะช่วยประหยัดเวลาและงบประมาณที่ต้องใช้ได้อย่างมาก ซึ่งนั่นหมายความว่านักวิจัยจะสามารถทำงานวิจัยได้อย่างมีประสิทธิภาพมากขึ้น

ในงานวิจัยนี้เป็นการออกแบบหุ่นยนต์ฮิวมานอยด์และพัฒนาระบบพื้นฐานของหุ่นยนต์ฮิวมานอยด์ สำหรับให้นักศึกษาหรือนักวิจัยสามารถพัฒนาต่อยอดได้ โดยหุ่นยนต์ฮิวมานอยด์ที่ออกแบบมานั้น สามารถที่จะปรับปรุง แก้ไข ดัดแปลงได้ง่าย ตัวโครงสร้างจะใช้เป็น พลาสติก PLA ที่สามารถขึ้นรูปได้ โดยการใช้เครื่องพิมพ์สามมิติ มีเซนเซอร์ตรวจการสัมผัสพื้นที่ฝ่าเท้าของหุ่นยนต์ มีเซนเซอร์สำหรับการวัดมุมเอียง ที่ลำตัวของหุ่นยนต์ และเพื่อที่จะทำให้ง่ายต่อการศึกษาทำความเข้าใจ บำรุงรักษา จึงได้มีการจัดทำคู่มือและเอกสารวิธีการใช้งานอย่างชัดเจน โดยจะเก็บในรูปแบบของเอกสารออนไลน์

---

[1] การเคลื่อนที่ของหุ่นยนต์รูปแบบต่างๆ ราชภัฏสวนดุสิต

[2] ณัฐพงษ์ วารีประเสริฐ และณรงศ์ ล่ำดี (2552: 374)

[3] DARPA 2015 [https://www.darpa.mil/about-us/about-darpa]

1.2 วัตถุประสงค์

ออกแบบโครงสร้างของหุ่นยนต์ฮิวมานอยด์ที่สามารถแก้ไขปรับเปลี่ยนได้ง่าย พัฒนาระบบพื้นฐาน ระบบจำลองสำหรับหุ่นยนต์ฮิวมานอยด์เพื่อใช้ในการศึกษาวิจัย รวบรวมเครื่องมือที่เป็นประโยชน์ต่อการพัฒนาหุ่นยนต์และจัดทำเอกสารออนไลน์ ให้บุคคลที่สนใจสามารถเข้ามาศึกษาได้

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 มีต้นแบบหุ่นยนต์ฮิวมานอยด์สำหรับใช้ในงานวิจัยแขนงต่างๆ

1.3.2 มีระบบพื้นฐานสำหรับพัฒนาหุ่นยนต์ฮิวมานอยด์รุ่นใหม่ในสถาบัน

1.3.3 มีระบบจำลองสำหรับจำลองการทำงานของหุ่นยนต์ฮิวมานอยด์

1.3.4 มีแหล่งรวบรวมเครื่องมือสำหรับการพัฒนาหุ่นยนต์

1.3.5 มีคู่มือ เอกสาร วิธีการใช้งาน และรายละเอียดของหุ่นยนต์สำหรับพัฒนาต่อยอด

1.4 ขอบเขตการดำเนินงาน

1.4.1 ใช้ ROS เป็นกรอบการทำงานสำหรับพัฒนาระบบพื้นฐาน

1.4.2 ออกแบบโครงสร้างให้มีความแข็งแรง สามารถรองรับน้ำหนักอุปกรณ์ที่ติดตั้งอยู่บนตัวหุ่นยนต์ได้

1.4.3 น้ำหนักของหุ่นยนต์รวมกันทั้งตัว ไม่เกิน 5 กิโลกรัม

1.4.4 ใช้ Solidworks 3D เป็นโปรแกรมสำหรับออกแบบโครงสร้าง และคำนวณ

1.4.5 หุ่นยนต์มีความสูงไม่ต่ำกว่า 100 เซนติเมตร และสูงไม่เกิน 120 เซนติเมตร

1.4.6 หุ่นยนต์มี 2 แขน 2 ขา มีองศาอิสระของขาข้างละ 6 และแขนข้างละ 2 องศาอิสระ

1.4.7 หุ่นยนต์สามารถทำงานได้ภายในสภาพแวดล้อมแบบปิด

1.4.8 หุ่นยนต์ใช้พลังงานจากแหล่งจ่ายพลังงานที่มีขนาดแรงดันไฟฟ้า 12 โวลต์

1.4.9 หุ่นยนต์ใช้ตัวขับเคลื่อนแบบดิจิตอลสำหรับแต่ละข้อต่อเป็น Dynamixel Digital Servo

1.4.10 ใช้ Gazebo สำหรับจำลองระบบของหุ่นยนต์

1.4.11 ติดตั้งเซนเซอร์วัดการกด (Ground contact) ที่ฝ่าเท้าของหุ่นยนต์

1.4.12 ติดตั้งเซนเซอร์วัดมุมเอียง (IMU) ที่บริเวณลำตัวของหุ่นยนต์

1.4.13 จัดทำคู่มือ เอกสารการใช้งาน และรายละเอียดส่วนประกอบของหุ่นยนต์

1.5 ภาพรวมของระบบและขั้นตอนการดำเนินงาน

งานวิจัยนี้การดำเนินงานวิจัยถูกแบ่งออกเป็นสามส่วน คือ ส่วนที่หนึ่งส่วนโครงสร้างของหุ่นยนต์ฮิวมานอยด์ เป็นส่วนที่ทำในส่วนของการขึ้นรูปชิ้นงาน ออกแบบโมเดลสามมิติ รวมไปถึงระบบอิเล็กทรอนิกส์ ติดตั้งบอร์ดและเซนเซอร์ไว้ตามจุดต่างๆ เพื่อสร้างโครงสร้างของหุ่นยนต์ให้สามารถรองรับการเดินได้ ส่วนที่สองส่วนโปรแกรมของหุ่นยนต์ฮิวมานอยด์ เป็นส่วนที่ทำในส่วนของการสั่งการตัวขับเคลื่อนต่างๆ อ่านค่าสถานะเซนเซอร์จากคอนโทลเลอร์ รวมไปถึงระบบจำลองการทำงานของหุ่นยนต์ และส่วนที่สามส่วนการออกแบบระบบพื้นฐานเพื่อการพัฒนาต่อยอด ส่วนนี้จะเป็นส่วนที่ทำให้ผู้ที่จะมาวิจัยต่อยอดสามารถทำงานได้ง่ายขึ้น จัดการเอกสารคู่มือ

การใช้งานต่างๆให้เป็นระบบระเบียบ สามารถแยกขั้นตอนการทำงานของแต่ละส่วนออกเป็นข้อดังนี้

1.5.1 ส่วนโครงสร้างของหุ่นยนต์ฮิวมานอยด์

1 ศึกษาค้นคว้าเอกสารและงานวิจัยที่เกี่ยวข้อง

- ศึกษาลักษณะโครงสร้างของมนุษย์
- ศึกษาลักษณะโครงสร้างของหุ่นยนต์ฮิวมานอยด์
- ศึกษาการออกแบบฝ่าเท้าและเซนเซอร์ตรวจจับพื้น
- ศึกษาการเลือกใช้ตัวขับเคลื่อนและวิธีการไดรว์มอเตอร์
- ศึกษาการส่งกำลังไปยังแต่ละข้อต่อ
- ศึกษาการออกแบบระบบจ่ายพลังงาน

2 ออกแบบโครงสร้างทางกล

- ออกแบบขนาดของหุ่นยนต์
- ตรวจสอบความเป็นไปได้ทาง Kinematics
- เลือกขนาดของมอเตอร์
- ออกแบบการยึดมอเตอร์กับโครงสร้าง
- ออกแบบโครงสร้างของหุ่นยนต์
- สร้าง CAD Model ของหุ่นยนต์
- จำลองและทดสอบความแข็งแรง
- ปรับปรุงโครงสร้าง

1.5.2 ส่วนโปรแกรมของหุ่นยนต์ฮิวมานอยด์

1 ศึกษาค้นคว้าเอกสารและงานวิจัยที่เกี่ยวข้อง

- ศึกษาระบบที่ใช้ในการพัฒนาหุ่นยนต์ฮิวมานอยด์
- ศึกษาระบบ

1.5.3 ส่วนการออกแบบระบบพื้นฐานเพื่อการพัฒนาต่อยอด

1 ติดตั้งระบบ

- Setup Version Control [GitHub]
- ติดตั้ง ROS ในคอมพิวเตอร์ที่ใช้ควบคุม
- ติดตั้ง ROS ในคอมพิวเตอร์ที่ควบคุมหุ่นยนต์
- ตั้งค่า ROS Workspace
- ตั้งค่าการติดต่อสื่อสารระหว่างระบบ

2 วางระบบพื้นฐาน

3 จัดทำคู่มือ

- รายการของที่จำเป็นต้องใช้
- คู่มือการประกอบของหุ่นยนต์
- คู่มือรายระเอียดข้อมูลของหุ่นยนต์

- คู่มือลิมิตของแต่ละข้อต่อของหุ่นยนต์
- คู่มือการวางโครงสร้างของระบบ

4 สื่อการสอน

- การติดตั้ง ROS
- การใช้งาน ROS เบื้องต้น
- การใช้งาน Simulation ของหุ่นยนต์
- การใช้งาน ROS กับหุ่นยนต์ฮิวมานอยด์
- การใช้งาน ROS บน Mbed

1.6 นิยามศัพท์

Gait Pattern : คือ รูปแบบการเดินของขาทั้งสองข้างให้เป็นรูปแบบการก้าวเดินเดียวกัน

Embedded System : คือ ระบบสมองกลฝังตัว ใช้ในการควบคุมอุปกรณ์ต่างๆ สามารถนำไปใช้ในการควบคุมเฉพาะทาง การควบคุมแบบแยกส่วน โดยอาศัยตัวประมวลผล ที่ทำหน้าที่วิเคราะห์และประมวลผลข้อมูลตามโปรแกรมที่ได้ออกแบบไว้

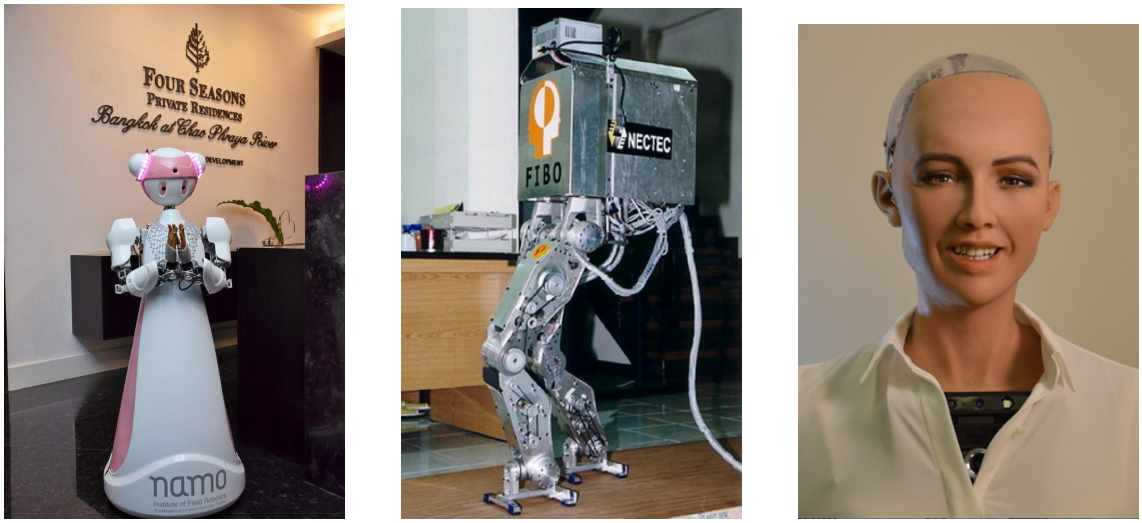
บทที่ 2

ทฤษฎีและหลักการ

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 หุ่นยนต์ฮิวมานอยด์

หุ่นยนต์ฮิวมานอยด์ คือ หุ่นยนต์ที่ถูกสร้างขึ้นมาให้มีรูปร่างคล้ายคลึงกับสรีระโครงสร้างของมนุษย์ มักถูกออกแบบขึ้นมาเพื่อจุดประสงค์เฉพาะอย่าง เช่น เพื่อให้ใช้เครื่องมือต่างๆของมนุษย์ เพื่อให้อยู่ในสภาพแวดล้อมของมนุษย์ เพื่อศึกษาการเคลื่อนไหนของร่ายกายมนุษย์ เพื่อศึกษาการมองเห็นของมนุษย์ เพื่อทำงานในสิ่งที่มนุษย์ทำได้ยาก หรือเพื่อวัตถุประสงค์อื่นๆ โดยทั่วไปแล้ว หุ่นยนต์ฮิวมานอยด์จะประกอบไปด้วย 4 ส่วนคือ ส่วนของหัว ส่วนของลำตัว ส่วนของแขน และส่วนของขา แต่การสร้างหุ่นยนต์ฮิวมานอยด์นั้นก็ไม่จำเป็นที่จะต้องมีส่วนประกอบทุกส่วนดังที่กล่าวไป ในบางครั้งอาจมีเพียงแค่ส่วนบนเท่านั้น ดังรูปที่ 2.1ก หุ่นยนต์นะโมจากสถาบันวิทยาการหุ่นยนต์ภาคสนาม เป็นหุ่นยนต์ที่มีส่วนบนเหมือนมนุษย์ แต่มีส่วนล่างเป็นล้อ หรือหุ่นยนต์ฮิวมานอยด์ที่มีเพียงแค่ส่วนล่าง ดังรูปที่ 2.1ข หุ่นยนต์สัมจุก เป็นหุ่นยนต์ฮิวมานอยด์ที่มีเพียงแค่ส่วนขาเท่านั้น หรือหุ่นยนต์ฮิวมานอยด์ที่มีเพียงใบหน้าเหมือนมนุษย์ ดังรูปที่ 2.1ข หุ่นยนต์โซเฟีย เป็นแอนดรอยด์ที่มีหน้าตาคล้ายมนุษย์มาก มีตา มีปาก สามารถพูดปฏิสัมพันธ์กับมนุษย์ได้



(ก) หุ่นยนต์ประชาสัมพันธ์นะโม (ข) หุ่นยนต์เดินสองขาสัมจุก (ฆ) หุ่นยนต์แอนดรอยด์โซเฟีย

รูปที่ 2.1: แสดงความแตกต่างของหุ่นยนต์ฮิวมานอยด์แต่ละประเภท

งานวิจัยทางด้านหุ่นยนต์ฮิวมานอยด์จากอดีตจนถึงปัจจุบันส่วนใหญ่จะเป็นการพัฒนาความสามารถของการเดินของหุ่นยนต์ เช่น เริ่มต้นจากแรกสุดจะเป็นการพัฒนาให้หุ่นยนต์สามารถเดินหน้าได้ ต่อมาก็เพิ่มความสามารถให้หุ่นยนต์สามารถเดินบนพื้นเอียง พื้นขรุขระ เดินเลี้ยวซ้ายขวา เดินขึ้นลงบันได ฯลฯ เป็นต้น นอกจากนี้ยังมีการพัฒนาปรับปรุงสมดุลของการเดินแบบสองขาอีกด้วย สมดุลของการเดินสามารถแบ่งได้สองแบบหลัก คือ การเดินแบบสมดุลสถิต และการเดินแบบสมดุลพลวัต งานในยุคแรกนั้นจะพัฒนาให้เดินได้แบบสมดุลสถิต ต่อมาเป็นสมดุลกึ่งพลวัต และเป็นสมดุลพลวัต การพัฒนาตัวควบคุมการเดินของหุ่นยนต์ จำเป็นที่จะต้องใช้ความรู้ทางด้านกลศาสตร์ค่อนข้างมาก มีการใช้สมการที่มีความซับซ้อน

Zheng และคณะ (1988) พัฒนาหุ่นยนต์สองขาที่สามารถเดินบนพื้นราบได้ ให้สามารถเดินต่อเนื่องไปบนพื้นเอียงได้ด้วย พื้นเอียงที่ใช้มีลักษณะเป็นพื้นเอียงขึ้น หุ่นยนต์ที่ใช้ในงานนี้มีข้อต่อสะโพก (hip), ข้อเท้า (ankle) และลำตัว (torso) มีเซนเซอร์วัดแรงกด (force sensor) ติดตั้งอยู่ที่ปลายเท้าและส้นเท้าแต่ละข้างเพื่อใช้วัดตำแหน่งของน้ำหนักโดยรวม (center of gravity) ของหุ่นยนต์ การเดินของงานวิจัยจะพิจารณาเฉพาะการเดินในแนวหน้าหลัง โดยมีหลักการคือ การเดินบนพื้นเอียงโดยที่หุ่นยนต์ยังเดินในท่าทางเหมือนกับตอนที่เดินบนพื้นราบจะทำให้น้ำหนักโดยรวมของหุ่นยนต์เลื่อนไปข้างหลัง ดังนั้นการที่หุ่นยนต์ขยับลำตัวไปด้านหน้าจำทำให้น้ำหนักโดยรวมของหุ่นยนต์กลับมาอยู่ตรงกลางของพื้นที่รับน้ำหนักเหมือนเดิม ซึ่งจะทำให้หุ่นยนต์มีความสมดุลได้ ดั้งนั้นข้อมูลที่ได้จากหน่วยวัดแรงกดที่เท้าจะถูกนำมาคำนวณตลอดการเดินเพื่อใช้ในการปรับเปลี่ยนมุมการขยับของลำตัว การเดินบนพื้นราบเป็นแบบสมดุลสถิตและการเดินบนพื้นเอียงก็ยังคงเป็นแบบสมดุลสถิตเช่นกัน

Inaba และคณะ (1995) สร้างหุ่นยนต์เลียนแบบลิง (ape-like biped) ประกอบด้วยสองมือและสองขา มีการเดินแบบสมดุลสถิต งานวิจัยนี้มีความคิดว่านอกจากการทำให้หุ่นยนต์สองขาเดินได้โดยไม่ล้มแล้ว ควรจะทำหุ่นยนต์ที่สามารถลุกขึ้นเองได้หลังจากที่ล้มแล้วด้วย ดังนั้นในงานนี้ หุ่นยนต์ถูกพัฒนาให้สามารถเดิน เมื่อล้มแล้วก็สามารถพลิกตัวและลุกขึ้นมาเดินให้ได้

Kun และ Miller (1996) ได้นำโครงข่ายประสาทเทียม มาประยุกต์ใช้ในการปรับเปลี่ยนท่าทางการเดินโดยอัตโนมัติของหุ่นยนต์สองขา การที่หุ่นยนต์สามารถปรับเปลี่ยนท่าทางได้โดยอัตโนมัตินี้มีประโยชน์ทำให้หุ่นยนต์เดินได้บนพื้นผิวหลากหลายลักษณะมากขึ้น ในงานนี้พิจารณาทั้งสมดุลในแนวหน้าหลัง (sigittal plane) และแนวซ้ายขวา (frontal plane) และการเดินของหุ่นยนต์เป็นแบบสมดุลพลวัต หลักการทำงานประกอบด้วยตัวสร้างท่าทางการเดินหนึ่งตัว และตัวปรับท่าทางการเดินทั้งแนวหน้าหลังและซ้ายขวาอีกหนึ่งตัว โดยค่าการปรับเปลี่ยนนั้นจะได้มาจาก แรงกดที่เท้า ความยาวการก้าวเท้า ความสูงของการยกเท้า เป็นต้น นอกจากนี้ ในปีถัดมาทั้งสิงได้ใช้หลักการที่ใช้ในงานนี้ไปใช้กับการเดินของหุ่นยนต์อีกตัว (Kun and Miller, 1997)

Hirai และคณะ (1998) พัฒนาหุ่นยนต์ฮิวมานอยด์ ซึ่งตัวหุ่นยนต์มีความคล้ายมนุษย์มาก สามารถเดินได้อย่างราบลื่นคล้านมนุษย์มากที่สุด เช่น สามารถเดินได้ในพื้นผิวชนิดต่างๆ เดินได้บนพื้นเอียงขึ้นเอียงลง เดินขึ้นลงบันได้ได้ เดินเข็นรถได้ เป็นต้น การเดินในทุกสถานการณ์เป็นการเดินแบบสมดุลพลวัต หุ่นยนต์สามารถเดินได้ด้วยความเร็วสูงสุด 4.7 กิโลเมตรต่อชั่วโมง หุ่นยนต์ประกอบไปด้วย แขนข้างละ 9 องศาอิสระ ขาข้างละ 6 องศาอิสระ ที่บริเวณหัวมีกล้องติดตั้งอยู่ 4 ตัว นอกจากนี้ยังมีอุปกรณ์ที่ใช้ในการรักษาสมดุลอื่นๆ อีกได้แก่ IMU ที่ติดตั้งบริเวณลำตัว และ Force sensor ที่ติดที่เท้าทั้งสองข้าง

ส่วนประกอบของหุ่นยนต์ฮิวมานอยด์สามารถจำแนกออกเป็นส่วนหลักๆได้สามส่วนคือ ส่วนการรับรู้ ส่วนการประมวลผล และส่วนการขับเคลื่อน

การรับรู้ของหุ่นยนต์ฮิวมานอยด์

Sensing on humanoid robots is more difficult than on other types of robots since the movement of the robot leads to noise in the sensors. For example, camera images get blurry if the shutter speed is too slow or if it is a rolling shutter sensor. Also, odometry data is less certain than on wheeled robots. Furthermore, the position of sensors in relation to the world is not stable. On a wheeled robot, the camera is usually a fixed hight above the ground. A humanoid robot has to do forward kinematics from its support feet to the camera to get the position and orientation. This also requires knowing which leg is the supporting leg and if the

robot is standing at all. For this an inertia measurement unit (IMU) and position encoders in the joints are crucial. These are described in the following. The IMU is a combination of a gyroscope and an accelerometer. It measures angular velocities and linear accelerations. It is normally installed inside the torso near the center of mass. The sensor values can be used to detect falls and to identify the side on which the robot landed after a fall. Furthermore, the positions of the robot's joints have to be measured. This is usually realized by a combination of a magnet on the end of the outermost gear and a hall sensor [Ramsden, 2011]. When the gear rotates, the magnet field rotates accordingly. The hall sensor measures this and a microcontroller can compute the position of the servo based on this data.

การประมวลผลของหุ่นยนต์ฮิวมานอยด์

In the early days of humanoid robots, having enough computational power was difficult because the space and mass for the computational unit are very limited. Today, single-board computers, e.g. Raspberry Pi [Upton and Halfacree, 2014], and Intel NUCs [Perico et al., 2014] are mostly used, depending on the size of the robot. All these boards provide GPUs and multiple CPU cores. Therefore the interest in parallelizing software and outsourcing computation to the GPU, especially for neural networks and computer vision, rose in the last years. Sometimes multiple cheap single-board computers are installed and connected using an Ethernet network, to get a high performance for low cost. This comes with the disadvantage that the software has to be able to run in parallel on a distributed system.

การขับเคลื่อนของหุ่นยนต์ฮิวมานอยด์

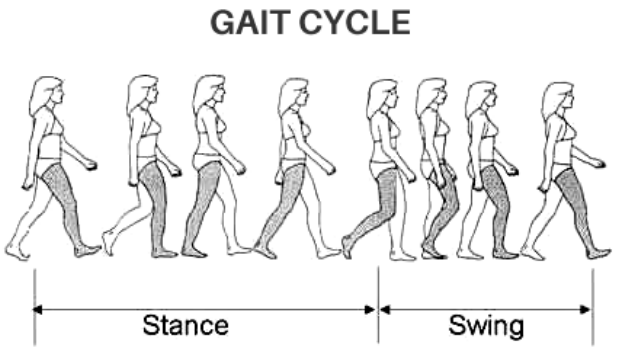
A humanoid robot is typically composed of multiple joints. Each joint has to be actuated which can be done in two different ways. Either the joint can be moved by tendons, similar to the human movement, or a motor can be placed directly into the joint axis. The later approach is used more often in smaller robots, since it needs less space. Using tendons makes exact positioning of the joint more difficult, but also enables to use bigger and linear motors, since they don't have to be placed inside the joint. Actuators need a high strength in relation to their weight, since they have to carry themselves. Multiple actuators are typically needed for one limb, resulting in a high number of cables, which have to be laid over multiple joints. Therefore a bus system is often used to limit the number of cables. The bus requires the motors to have a micro controller, to receive goal positions and to send the current position. Micro controller, motor and a gearbox are usually grouped together in a case and called servo . For reaching the goal position and for holding it, a controller is required in the firmware of the servo. This is usually a PID controller [Wescott, 2000].

2.1.2 ทฤษฎีที่เกี่ยวข้องกับมนุษย์

2.1.2.1 การวิเคราะห์การเดินของมนุษย์

การเคลื่อนที่ของหุ่นยนต์ฮิวมานอยด์นั้นจะเลียนแบบมาจากการเดินของมนุษย์ ดั้งนั้นการวิเคราะห์ลักษณะการเดินของมนุษย์ จะเป็นการศึกษาเพื่อทำความเข้าใจถึงธรรมชาติการเดิน ก่อนนำไปทำการออกแบบกลไกทางกลและระบบควบคุมของหุ่นยนต์ฮิวมานอยด์ การก้าวเดินของมนุษย์โดยปกติแล้ว จะมีลักษณะเป็นวัฏจักรวนซ้ำไปเรื่อยๆ ในทิศทางที่ต้องการจนกว่าจะทำการหยุดเดิน การทรงตัวในระหว่างการยืนหรือการเดินนั้น เป็นไปตามสัญชาติญาณซึ่งเกิดจากการรักษาความสมดุลของระดับน้ำในหู[1] ส่งสัญญาณผ่านเส้นประสาทไปยังกล้ามเนื้อส่วนต่างๆ ที่ทำหน้าที่ให้เกิดการเคลื่อนที่

การเคลื่อนที่ของมนุษย์ในการเดินไปข้างหน้าสามารถแบ่งออกเป็นช่วงต่างๆดังนี้



รูปที่ 2.2: วัฏจักรการเดินของมนุษย์

1. ช่วงเริ่มการวางเท้าเพื่อเข้าสู่ช่วงเริ่มต้นเหวี่ยงเท้า เป็นช่วงที่เท้าเกิดการกระแทกลงบนพื้นหลังจากทำการเหวี่ยงมาจากด้านหลัง โดยธรรมชาติมนุษย์จะทำการวางส้นเท้าลงเพื่อลดแรงกระแทกที่เกิดขึ้นในช่วงนี้ ดังนั้นทางกายภาพในส่วนของส้นเท้ามนุษย์จึงมีลักษณะอ่อนนุ่ม
2. ช่วงเริ่มต้นเหวี่ยงเท้าเพื่อเข้าสู่ช่วงเหวี่ยงเท้า หลังจากทำการวางส้นเท้าลงกับพื้นแล้ว ข้อเข้าจะปรับมุมเพื่อให้ฝ่าเท้าแนบพื้นสนิท ขณะเดียวกันขาอีกข้างจะยกสูงขึ้นเพื่อถ่ายเทน้ำหนักไปยังเท้าที่เพิ่งวางลง
3. ช่วงเหวี่ยงเท้า เป็นช่วงที่ขาหนึ่งยกลอยอยู่ในอากาศและขาที่วางแนบกับพื้นจะรองรับน้ำหนักทั้งหมดของร่างกาย
4. ช่วงเตรียมการวางเท้า เป็นช่วงที่ขาข้างที่ลอยอยู่เหวี่ยงไปข้างหน้าเพื่อเตรียมเข้าสู่ช่วงรองรับ ในขณะเดียวกันขาที่รับน้ำหนักอยู่จะทำการผลักตัวเพื่อเริ่มทำการถ่ายเทน้ำหนักไปข้างหน้า

2.1.2.2 การวิเคราะห์องศาอิสระของมนุษย์

การที่มนุษย์เราสามารถเคลื่อนที่ได้นั้น เป็นผลเนื่องมาจากการเคลื่อนที่ของข้อต่อต่าง ๆ ที่อยู่ บนขา ซึ่งประกอบไปด้วย ข้อต่อส่วนสะโพก ข้อต่อส่วนหัวเข่า และข้อต่อส่วนข้อเท้า แรงบิดที่เกิด ขึ้นของแต่ละข้อต่อมีความสัมพันธ์ต่อกัน ส่งผลให้เกิดเสถียรภาพในการเดินของมนุษย์ เมื่อวิเคราะห์ ลักษณะโครงสร้างในแต่ละส่วนพบว่าข้อต่อส่วนสะโพกมีลักษณะเป็นทรงกลม ทำให้ข้อต่อส่วน สะโพกสามารถหมุนได้ 3 องศาอิสระ ส่วนหัวเข่าของมนุษย์ มีจุดต่อของข้อที่มีลักษณะเป็นทรงกลม สองลูกประกอบเข้าด้วยกันทำให้การเคลื่อนที่ถูกบังคับ

[1] text

ให้สามารถเคลื่อนที่ได้เพียง 1 องศาอิสระ ใน ส่วนของข้อเท้ามีลักษณะการเคลื่อนที่เหมือนสะโพกคือสามารถเคลื่อนที่ได้ 3 องศาอิสระ

จากทั้งหมดที่ได้ทำการวิเคราะห์มาข้างต้นพบว่าในขาหนึ่งข้างของมนุษย์ประกอบด้วย 7 องศาอิสระ ซึ่งส่งผลให้การเคลื่อนที่ของมนุษย์มีความคล่องแคล่วสูง แต่ในทางออกแบบกลไกการเดินและการควบคุม ของหุ่นยนต์สองขาถือว่ามีจำนวนองศาอิสระเกินความจำเป็นในการเคลื่อนที่บนปริภูมิ(space) และยากต่อ การควบ-คุม(under actuated) ดังนั้นการกำหนดจำนวนองศาอิสระเพื่อให้หุ่นยนต์เดินได้เสมือนมนุษย์จึง มีผลในการออกแบบกลไกทางกลและการควบคุมของหุ่นยนต์สองขา

2.1.2.3 กายวิภาคศาสตร์

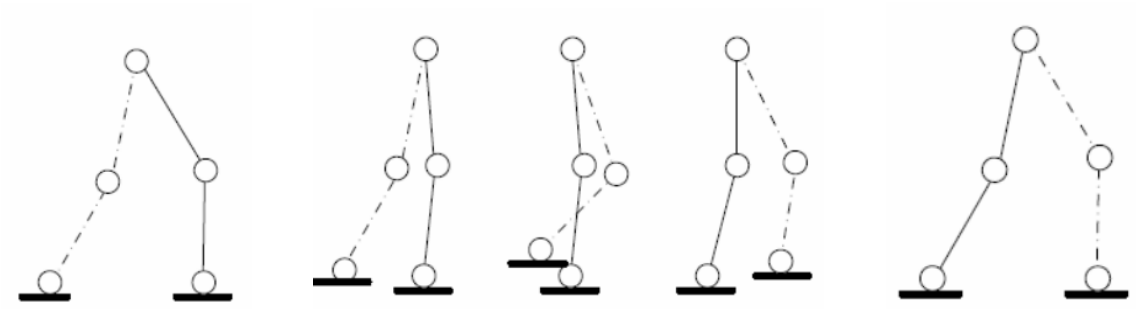
2.1.3 ทฤษฎีที่เกี่ยวข้องกับหุ่นยนต์ฮิวมานอยด์

2.1.3.1 วัฏจักรการเดินของหุ่นยนต์ฮิวมานอยด์

วัฏจักรการเดินของหุ่นยนต์ คือ การที่หุ่นยนต์จะต้องมีการถ่ายน้ำหนักไปมาระหว่างเท้าซ้ายและเท้าขวา มีบางช่วงที่น้ำหนักตกลงบนเท้าข้างใดข้างหนึ่งหรือทั้งสองข้างพร้อมกัน สามารถแบ่งออกเป็นช่วงได้สองช่วง คือช่วงการยืนด้วยขาข้างเดียว และช่วงการยืนด้วยขาทั้งสองข้าง

1) การยืนด้วยขาข้างเดียว : เป็นช่วงที่มีเท้าของหุ่นยนต์สัมผัสพื้นเพียงข้างเดียว ส่วนเท้าอีกข้างของหุ่นยนต์จะถูกยกลอยจากพื้น โดยที่ไม่มีส่วนใดๆของขาข้างนั้นสัมผัสกับพื้นเลย ช่วงนี้จะเกิดขึ้นเมื่อมีการแกว่งเท้าจากข้างหลังไปข้างหน้า ดังจากภาพที่ 2.3ข

2) การยืนด้วยขาสองข้าง : เป็นช่วงที่เท้าทั้งสองข้างของหุ่นยนต์สัมผัสกับพื้น ช่วงนี้จะเกิดตั้งแต่หุ่นยนต์วางเท้าขณะที่ส้นเท้าแตะกับพื้น ไปจนถึง ปลายเท้าของขาอีกข้างหลุดออกจากพื้น



(ก) ยืนด้วยขาสองข้าง (ข) ยืนด้วยขาข้างเดียว (ฆ) ยืนด้วยขาสองข้าง

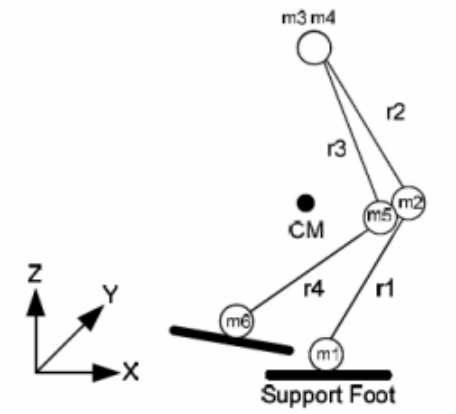
รูปที่ 2.3: วัฏจักรการเดินของหุ่นยนต์ฮิวมานอยด์

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

2.1.3.2 การสร้างและการควบคุมการเดินแบบสมดุลสถิต

การเดินของหุ่นยนต์ในลักษณะนี้ น้ำหนักตัวหุ่นยนต์จะไม่มีการเคลื่อนไหวออกนอกบริเวณฐานรับน้ำหนัก (Supporting Area) ตลอดช่วงเวลาการเดิน ไม่ว่าจะเป็นช่วงเวลาที่รับน้ำหนักด้วยเท้าข้างเดียวหรือทั้ง

สองข้างก็ตาม หมายความว่า โครงสร้างของหุ่นยนต์จะไม่ล้มแน่นอน เนื่องจากการสร้างรูปแบบการเดินด้วยวิธีนี้จะควบคุมให้ตำแหน่งของจุดรวมมวล (CoM) อยู่ภายในพื้นที่ฐานรับน้ำหนักของหุ่นยนต์ตลอดเวลา



รูปที่ 2.4: การควบคุมตำแหน่งของจุดรวมมวลให้อยู่ในพื้นที่ฐาน

ข้อดีของการสร้างและควบคุมการเดินของหุ่นยนต์ด้วยวิธีนี้คือ สามารถสร้างรูปแบบการเดินได้โดยที่มีความซับซ้อนไม่มากนัก สามารถสั่งให้หุ่นยนต์หยุดค้างในท่าทางใดๆก็ได้ตลอดเวลาโดยหุ่นยนต์ไม่ล้ม หุ่นยนต์ที่มีฝ่าเท้าใหญ่จะทำให้ง่ายต่อการก้าวเดินมากขึ้น นอกจากการควบคุมการก้าวขาแล้วอาจเพิ่มการควบคุมส่วนลำตัวเพิ่มเติม เพื่อเป็นการเพิ่มเสถียรภาพในการเดินและการถ่ายเทน้ำหนัก โดยที่อาจจะมีการเพิ่มเซนเซอร์วัดแรงที่ฝ่าเท้าเพื่อตรวจสอบการกระจายแรงกดที่ฝ่าเท้า เพื่อตรวจสอบว่าตำแหน่งของจุดรวมน้ำหนักอยู่บนพื้นที่ฝ่าเท้าหรือไม่ หรือเพื่อตรวจสอบเสถียรภาพของการเดินเพื่อแก้ไขท่าทางการเดินไม่ให้เกิดการล้ม

ข้อเสียของการควบคุมการเดินด้วยวิธีนี้คือ หุ่นยนต์จะใช้เวลาในการก้าวเดินมาก ใช้พลังงานในการเดินมากกว่าการเดินแบบสมดุลพลวัต และท่าทางที่ได้จะมีความแตกต่างจากท่าทางการเดินของมนุษย์

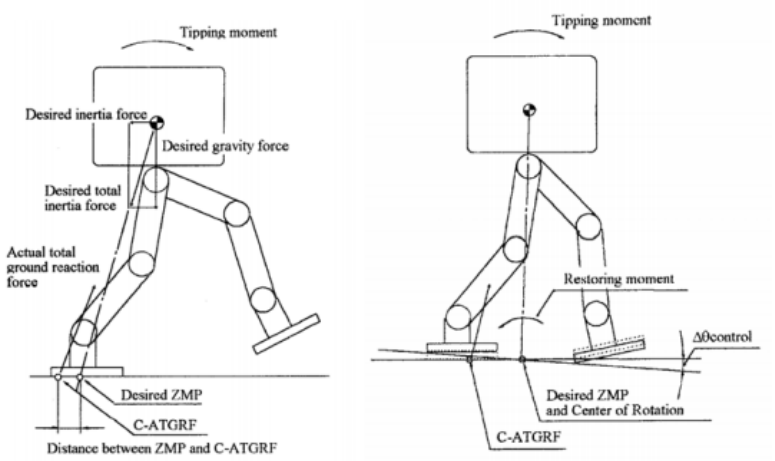
2.1.3.3 การสร้างและการควบคุมการเดินแบบสมดุลพลวัต

การสร้างรูปแบบการเดินและควบคุมการเดินในลักษณะนี้ท่าทางการเดินของหุ่นยนต์นั้นจะคล้ายกับการเดินของมนุษย์มากกว่าแบบสถิต เนื่องจากมีหลักการในการสร้างท่าทางที่เหมือนกับการเดินของมนุษย์ซึ่งมีขั้นตอนดังนี้คือ เอียงตัวให้ล้มไปในทิศทางที่ต้องการเดิน เมื่อเริ่มเกิดการล้มขึ้นหุ่นยนต์จะเปลี่ยนตำแหน่งการวางเท้าไปยังตำแหน่งใหม่ เพื่อปรับให้โครงสร้างเข้าสู่สภาวะสมดุลอีกครั้ง

โดยธรรมชาติแล้วมนุษย์มีการถ่ายน้ำหนักในขณะที่เคลื่อนที่หรือยืนอยู่กับที่เพื่อรักษาสมดุลของท่าทางนั้นไว้ แต่หากการถ่ายโอนน้ำหนักนั้นเกิดสภาวะไม่สมดุล ร่างกายจะปรับสภาพโดยการเคลื่อนตำแหน่งของเท้าซึ่งเป็นพื้นที่ฐานออกจากเดิมไปยังตำแหน่งใหม่ เพื่อรักษาสมดุลไว้ หลักการดังกล่าวถูกนำมาใช้กับการควบคุมการเดินของหุ่นยนต์ฮิวมานอยด์ ในขณะที่หุ่นยนต์กำลังเคลื่อนไหว ผลจากแรงเฉื่อยของการเคลื่อนที่และผลจากแรงดึงดูดของโลกมีผลต่อการเพิ่มและลดความเร่งให้การเดินของหุ่นยนต์ แรงเหล่านี้เรียกว่าแรงเฉื่อยรวมของการเคลื่อนที่ และเมื่อเท้าหุ่นยนต์สัมผัสกับพื้นจะได้รับผลกระทบของแรงนี้ เรียกว่า แรงปฏิกิริยาจากพื้น

การตัดกันระหว่างแรงปฏิกิริยาจากพื้นและแนวแรงเฉื่อยรวม ตำแหน่งนั้นหากทำให้โมเมนต์เท่ากับศูนย์เรียกจุดตัดนี้ว่าจุดโมเมนต์ศูนย์ (ZMP) และจุดที่แรงปฏิกิริยาลงสู่พื้นว่า จุดปฏิกิริยาพื้นฐาน ท่าทางการเดินของหุ่นยนต์จะถูกกำหนดและถูกส่งให้กับชุดควบคุมข้อต่อจุดต่างๆของหุ่นยนต์ โดยให้สอดคล้องกับแรงเฉื่อยรวมที่เกิดขึ้นจากการคำนวณ เรียกว่าแรงเฉื่อยรวมเป้าหมาย และจุดโมเมนต์ศูนย์ที่ได้จากการคำนวณเรียกว่าจุดโมเมนต์ศูนย์เป้าหมาย (ZMP_{target}) เมื่อหุ่นยนต์เกิดสมดุลในขณะที่ทำการเดินได้อย่างสมบูรณ์ แนวแกนของ

แรงเฉื่อยรวมเป้าหมายและแรงปฏิกิริยาที่พื้นจะเป็นตำแหน่งเดียวกัน แต่ในขณะที่หุ่นยนต์เดินผ่านพื้นผิวที่มีความขรุขระหรือไม่เรียบตำแหน่งสองจุดดังกล่าว จะไม่ใช่ตำแหน่งเดียวกันทำให้หุ่นยนต์เกิดการล้มได้ แรงที่ทำให้เกิดการล้มนี้เกิดจากตำแหน่งของจุดโมเมนต์ศูนย์และตำแหน่งแรงปฏิกิริยารวมที่พื้นไม่ตรงกัน ซึ่งเป็นสาเหตุหลักที่ทำให้เกิดความไม่สมดุลขึ้น และเมื่อหุ่นยนต์เสียสมดุลระบบที่จะสามารถป้องกันการล้มและทำให้หุ่นยนต์เดินต่อไปได้อย่างต่อเนื่องคือ ระบบควบคุมแรงปฏิกิริยา ระบบควบคุมจุดโมเมนต์ศูนย์ และระบบควบคุมการวางเท้า



รูปที่ 2.5: การควบคุมตำแหน่งของจุดโมเมนต์ศูนย์ให้ตรงกับแรงปฏิกิริยารวม

อย่างไรก็ตาม การสร้างท่าทางการเดินในลักษณะนี้ต้องใช้สมการในการคำนวณที่ซับซ้อนมาก เนื่องจากต้องหาความสัมพันธ์ระหว่างองค์ประกอบหลายส่วน เช่น น้ำหนักของโครงสร้างในแต่ละส่วน แรงบิดที่แต่ละข้อต่อ และโมเมนต์โดยรวมของระบบ นอกจากนี้ยังต้องใช้อุปกรณ์การตรวจวัดต่างๆ เช่น เซนเซอร์วัดแรง เซนเซอร์วัดมุม เซนเซอร์วัดแรงบิด ติดตั้งตามจุดต่างๆของโครงสร้างเพื่อวัดค่าออกมา ก่อนที่จะทำการคำนวณตำแหน่ง และสร้างท่าทางการเดินของหุ่นยนต์ฮิวมานอยด์ ท่าทางการเดินที่ได้จากการควบคุมด้วยวิธีนี้ จะมีความคล้ายคลึงกับท่าทางการเดินของมนุษย์มาก

2.1.3.4 จุดศูนย์กลางมวลของหุ่นยนต์

หากต้องการให้หุ่นยนต์สามารถที่จะทรงตัวอยู่ได้โดยไม่ล้มนั้น จึงต้องรู้ตำแหน่งจุดศูนย์กลางมวลของหุ่นยนต์ตลอดเวลา และต้องให้จุดศูนย์กลางมวลฉายตกในบริเวณฐานรับน้ำหนักของหุ่นยนต์โดยหาจากพื้นที่ที่ฝ่าเท้าสัมผัสกับพื้น

2.2 งานวิจัยที่เกี่ยวข้อง

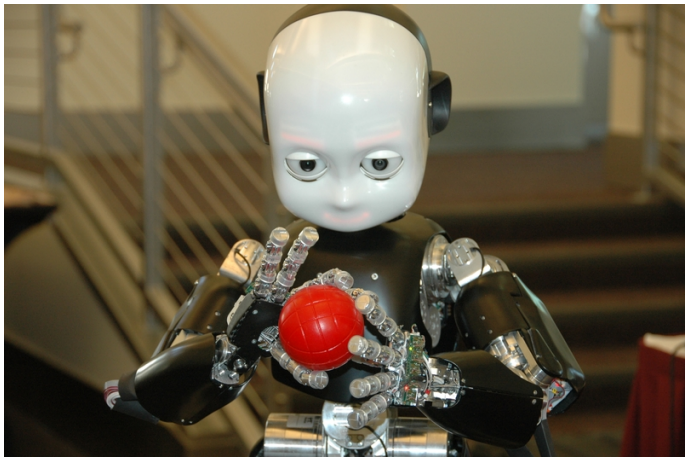
2.2.1 ตัวอย่างหุ่นยนต์ฮิวมานอยด์

Poppy Humanoid

รูปที่ 2.6: หุ่นยนต์ฮิวมานอยด์ป๊อปปี้

หุ่นยนต์ฮิวมานอยด์ป๊อปปี้ ถูกสร้างขึ้นมาเพื่อใช้ในงานศิลปะ การวิจัยและการศึกษาโดยเฉพาะ หุ่นยนต์ป๊อปปี้ประกอบด้วยส่วนของฮาร์ทแวร์และซอฟแวร์ที่เปิดเป็นโอเพนซอร์ซให้ผู้ที่สนใจสามารถเข้ามาศึกษาได้ โปรแกรมของหุ่นยนต์ใช้โมดูลที่มีชื่อว่า Pypot ที่เป็นส่วนเสริมของภาษา Python ในการพัฒนาซอฟแวร์ ทุกคนสามารถเข้าถึงข้อมูลเชิงเทคนิคของหุ่นยนต์ฮิวมานอยด์ป๊อปปี้ได้ เช่น ส่วนรายละเอียดการทำงาน คลิปวีดีโอสอนการประกอบ การใช้ระบบจำลอง และการพัฒนาต่างๆผ่านทางเว็บไซต์ http://www.poppy-project.org หุ่นยนต์ป๊อปปี้มีส่วนของโครงสร้างที่ผลิตมาจากพลาสติด PLA และ ABS โดยใช้เทคนิคการขึ้นรูปด้วยเครื่องพิมพ์สามมิติ ตัวขับเคลื่อนข้อต่อต่างๆใช้เป็น Dynamixel Digital Servo และควบคุมคำสั่งของตัวขับเคลื่อนด้วย คอม-พิวเตอร์ขนาดเล็ก Odroid UX4 ใช้ระบบปฏิบัติการ Ubuntu 14.04 ตัวของหุ่นยนต์มีความสูง 83 เซนติเมตร น้ำหนัก 3.5 กิโลกรัม ใช้เซนเซอร์วัดมุมเอียงเป็น IMU ที่มีองศาอิสระเท่ากับ 9 องศาอิสระ ในการควบคุมเสถียรภาพในการเดินของตัวเอง มีองศาอิสระหรือจำนวนตัวขับเคลื่อนทั้งหมด 25 องศา ประกอบไปด้วย ขาข้างละ 6 องศาอิสระ แขนข้างละ 4 องศาอิสระ ลำตัว 3 องศาอิสระ และ หัว 2 องสาอิสระ

iCub Humanoid



รูปที่ 2.7: หุ่นยนต์ฮิวมานอยด์ไอคัพ

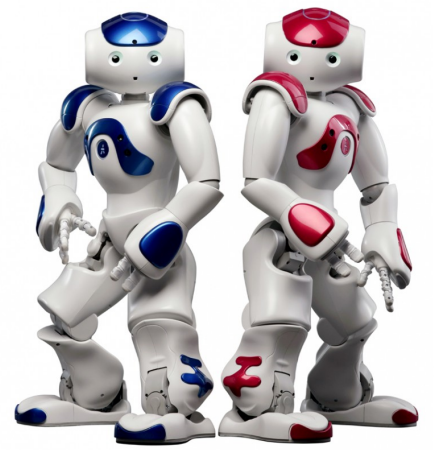
หุ่นยนต์ฮิวมานอยด์ไอคัพ ถูกออกแบบโดยมหาวิทยาลัยหลายแห่งในยุโรปรวมกลุ่มกันขึ้นมาในชื่อ RobotCub และถูกสร้างขึ้นโดย Istituto Italiano di Tecnologia (IIT) ตัวหุ่นยนต์ไอคัพนั้นมีความสูงอยู่ที่ 1 เมตร น้ำหนักโดยรวมทั้งหมดประมาณ 22 กิโลกรัม วัสดุที่ใช้ในการสร้างแตกต่างกันไปในแต่ละส่วนของร่างกายโดยจะใช้ aluminum alloy Al6082 สำหรับส่วนที่ต้องรับภาระความเครียดน้อย ใช้ aluminum alloy 7075(Ergal) สำหรับส่วนที่ต้องรับภาระความเครียดปานกลางถึงสูง และใช้ Stainless Steel 17-4PH ในส่วนของเพลาข้อต่อต่างๆเพื่อให้มีความแข็งแรงสูง ตัวหุ่นยนต์ถูกออกแบบให้มีลักษณะเหมือนเด็กอายุ 3-4 ขวบ ควบคุมโดยใช้บอร์ดไมโครคอนโทรเลอร์เป็นรุ่น PC104 Controller ภาษาที่ใช้ในการพัฒนาใช้เป็นภาษา C++ ในการเขียนโปรแรม การติดต่อสื่อสารกับตัวขับเคลื่อนหรือมอเตอร์ตามข้อต่อต่างๆ และเซนเซอร์ ผ่านทางโปรโตคอล CAN Bus เพื่อทำให้ใช้สายน้อยลง ใช้เส้นเอ็นในการส่งถ่ายแรงขับเคลื่อนไปยังส่วนของข้อต่อส่วนมือและไหล่ นิ้วของหุ่นยนต์ถูกร้อยด้วย สายเคเบิลเคลือบ Teflon อยู่ภายใน และคลายตัวกลับสู่สภาวะสมดุลได้ด้วยแรงของสปริง เซนเซอร์วัดมุมของข้อต่อแต่ละตัวใช้การออกแบบให้มี Hall-effect ติดอยู่ ช่วยในการอ่านค่าของตำแหน่งและความเร็วที่เกิดขึ้นที่ข้อต่อนั้น หุ่นยนต์ไอคัพมีองศาอิสระรวมกันทั้งหมด 53 องศาอิสระ ประกอบไปด้วย แขนข้างละ 7 องศาอิสระ มือข้างละ 9 องศาอิสระ หัว 6 องศาอิสระ ลำตัว 3 องศาอิสระ และขาข้างละ 6 องศาอิสระ ในส่วนของหัวจะประกอบไปด้วย กล้องสองตัวเพื่อทำสเตอริโอวิชั่น ไมโครโฟนสำหรับรับเสียงจากสภาพแวดล้อมภายนอก และไฟแสดงอารมณ์บริเวณปากและคิ้ว หุ่นยนต์นี้ไม่ได้ถูกออกแบบให้มีการทำงานเป็นแบบอัตโนมัติ ซึ่งก็คือตัวหุ่นยนต์นั้นไม่มีแบตเตอรี่ภายในตัว แต่ใช้แหล่งพลังงานจากภายนอกโดยการส่งเข้าไปผ่านสายเคเบิล และเชื่อมต่อกับอินเทอร์เน็ตผ่านสายแลน (LAN)

Darwin-OP Humanoid

รูปที่ 2.8: หุ่นยนต์ฮิวมานอยด์ดาร์วิน

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

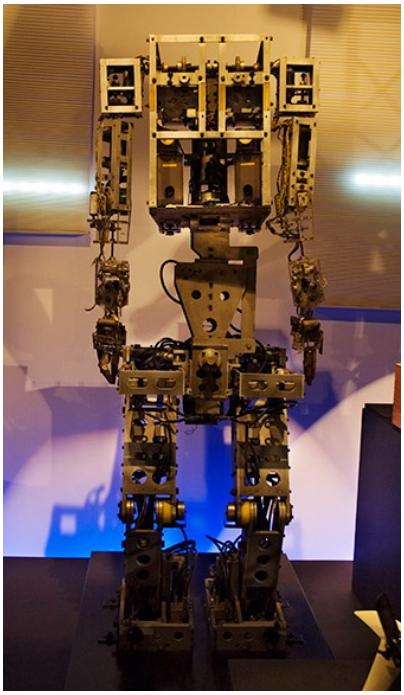
Nao Humanoid



รูปที่ 2.9: หุ่นยนต์ฮิวมานอยด์นาโอะ

หุ่นยนต์ฮิวมานอยด์นาโอะ เป็นหุ่นยนต์ฮิวมานอยด์ขนาดกลาง ถูกผลิตมาจากประเทศฝรั่งเศษ พัฒนาโดยบริษัท Aldebaran Robotics เมื่อปี 2004 และในปี 2007 หุ่นยนต์ฮิวมานอยด์นาโอะได้นำไปแทนที่หุ่นยนต์สุนัขของ Sony ชื่อ Aibo ขณะถูกใช้ในรายการแข่งขัน RoboCup Standard Platform League (SPL) หุ่นยนต์นาโอะได้ถูกนำไปใช้ใน Robocup 2008 และ 2009 หุ่นยนต์นาโอะถูกพัฒนาออกมาหลายรุ่นมีองศาอิสระตั้งแต่ 14 องศาอิสระ 21 องศาอิสระ และ 25 องศาอิสระ สำหรับเพื่องานวิจัยนั้นมีถึง 25 องศาอิสระ โดยเพิ่มเติมมือสองข้างเอาเข้าไปเพื่อให้สามารถหยิบจับสิ่งของได้ ภายในหุ่นยนต์ถูกควบคุมด้วยระบบปฏิบัติการ NAO 2.0 (Linux-based) ตัวหุ่นยนต์มีความสูง 58 เซนติเมตร น้ำหนัก 4.3 กิโลกรัม ส่วนเซนเซอร์การรับรู้ต่างๆจะประกอบไปด้วย เซนเซอร์วัดมุมเอียง 3-axis gyro, 3-axis accelerometer, Ultrasound captors, ไมโครโฟน 4 ตัว ลำโพง 2 ตัว กล้อง 2 ตัว เพื่อใช้ประโยชน์ในการทำงานวิจัยต่างๆ ตอนนี้ความสามารถของหุ่นยนต์นาโอะที่ทำได้คือ สามารถเทน้ำส้มได้ เดินขึ้นลงบันไดและทางลาดชันได้ ระหว่างการเดินนั้นสามารถวางแผนการวางเท้าได้อย่างรวดเร็ว อีกทั้งยังสามารถที่จะเดินหลบหลีกสิ่งกีดขวางได้ด้วย

Wabot



รูปที่ 2.10: หุ่นยนต์ฮิวมานอยด์วาบอท

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

2.2.2 งานวิจัยการออกแบบระบบหุ่นยนต์ฮิวมานอยด์

2.3 การออกแบบโครงสร้างของหุ่นยนต์

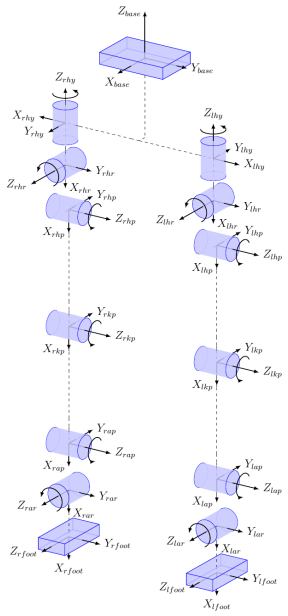
2.3.1 ข้อแตกต่างระหว่างโครงสร้างของมนุษย์กับโครงสร้างของหุ่นยนต์

2.3.1.1 ความแตกต่างขององศาเสรี

เนื่องจากลักษณะข้อต่อของมนุษย์มีความซับซ้อนมากกว่าโครงสร้างของหุ่นยนต์ ทำให้ข้อต่อแต่ละจุดของมนุษย์นั้นสามารถหมุนได้หลายทิศทาง รวมถึงขอบเขตของการหมุนของข้อต่อในแต่จุดก็มีความแตกต่างกัน ในการนำรูปแบบการเดินของมนุษย์ไปใช้กับหุ่นยนต์จึงต้องปรับค่ามุมที่ข้อต่อให้มีความเหมาะสมกับโครงสร้าง และข้อจำกัดเกี่ยวกับการหมุนของข้อต่อจุดต่างๆของหุ่นยนต์ที่จะใช้ทดสอบด้วย

2.3.1.2 ความแตกต่างของอัตราส่วน

นอกจากความแตกต่างขององศาเสรี (DoF) ระหว่างมนุษย์กับหุ่นยนต์แล้ว ความแตกต่างของอัตราส่วนระหว่างโครงสร้างแต่ละส่วนของมนุษย์กับหุ่นยนต์เป็นอีกสาเหตุหนึ่ง ที่ต้องทำการปรับแต่งใหม่มีความเหมาะสม เนื่องจากความยาวของโครงสร้างแต่ละส่วน รวมทั้งระยะห่างระหว่างจุดหมุนแต่ละจุดของมนุษย์กับหุ่นยนต์ที่มีความแตกต่างกัน ดังนั้นจึงต้องกำหนดระบบพิกัดเพื่อใช้อ้างอิงจุดหมุนและความยาวของโครงสร้าง



รูปที่ 2.11: ตัวอย่างตำแหน่งและการหมุนของข้อต่อของหุ่นยนต์เพื่อการอ้างอิง

2.3.1.3 กำลังและประสิทธิภาพของมอเตอร์

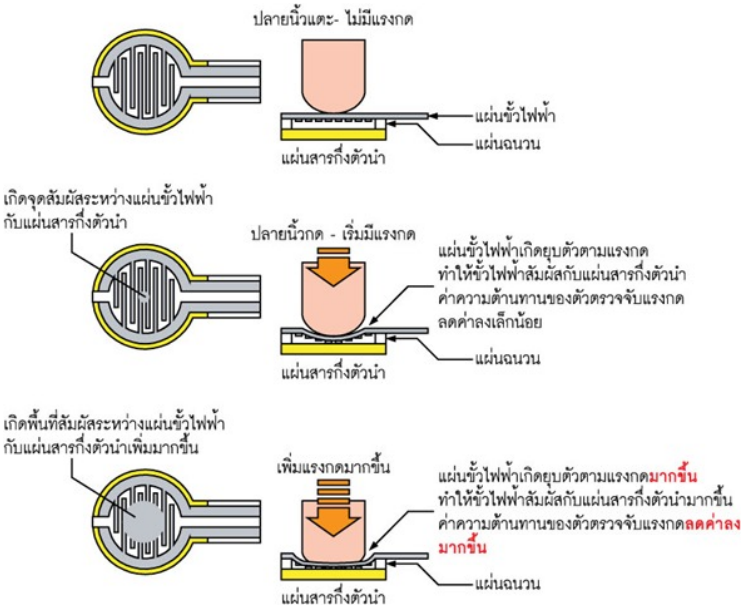
ความสามารถในการรับน้ำหนักของข้อต่อแต่ละจุดมีความแตกต่างกัน การเคลื่อนไหวของมนุษย์นั้นจะมีกล้ามเนื้อ และเส้นเอ็นเป็นตัวออกแรงดึงส่วนต่างๆของร่างกายเพื่อทำให้เกิดการเคลื่อนไหวซึ่งจะมีความยืดหยุ่นและแรงดึงที่มีค่าสูง สำหรับการเคลื่อนไหวของหุ่นยนต์ จะใช้การบิดแกนของเซอร์โวมอเตอร์ (Servo Motor) หรือมอเตอร์ที่ติดอยู่ที่ข้อต่อจุดต่างๆ ทำให้ความสามารถในการรับน้ำหนัก แรงบิดและความยืดหยุ่นที่ข้อต่อขึ้นกับกำลังของมอเตอร์เป็นหลัก การสร้างท่าทางของหุ่นยนต์จึงต้องคำนึงถึงความสามารถในการรับน้ำหนักและกำลังของเซอร์โวมอเตอร์ที่ใช้ด้วยเช่นกัน

2.3.2 อุปกรณ์ที่ใช้ในหุ่นยนต์ฮิวมานอยด์

2.3.2.1 เซนเซอร์ตรวจหน้าสัมผัสที่พื้น

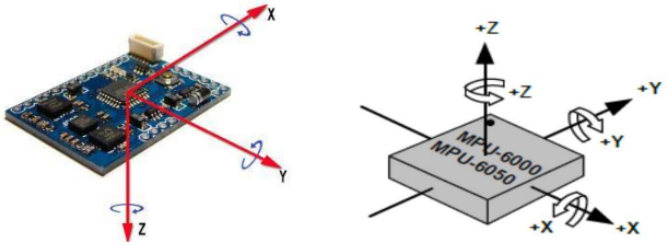
เซนเซอร์ตรวจหน้าสัมผัสที่พื้นเป็นเซนเซอร์ที่ถูกติดตั้งบริเวณฝ่าเท้า เพื่อตรวจสอบการเดินของหุ่นยนต์ฮิวมานอยด์ว่าขณะนี้มีการสัมผัสของฝ่าเท้าของหุ่นยนต์กับพื้นหรือไม่ ซึ่งในงานวิจัยนี้ได้ใช้หลักการตัวตรวจจับแรงกดแบบค่าความต้านทานหรือ Force Sensing Resistor (FSR) ที่ใช้เทคโนโลยีฟิล์มโพลีเมอร์แบบหนาโดยที่เซนเซอร์สามารถเปลี่ยนแรงที่มากระทำให้อยู่ในรูปของการเปลี่ยนแปลงค่าความต้านทานไฟฟ้า ตัวเซนเซอร์มีลักษณะเป็นแผ่น มีโครงสร้าง 5 ชั้น โดยสองชั้นนอกสุดเป็นฟิล์มของโพลีเอสเตอร์ สองชั้นถัดเข้ามาเป็นฟิล์มของโลหะที่เป็นตัวนำไฟฟ้า และชั้นในสุดเป็นหมึกที่มีความไวในการตอบสนองต่อแรงภายนอกที่มากระทำ (Pressure sensitive ink) และโครงสร้างทั้ง 5 ชั้น ถูกรวมเข้าด้วยกันด้วยวิธีลามิเนท จึงทำให้เซนเซอร์วัดแรงนี้มีลักษณะแบนมีความยืดหยุ่นสูง ด้วยเหตุนี้จึงทำให้เซนเซอร์สามารถโค้งงอได้ง่าย แรงดันไฟฟ้าที่ตกคร่อมตัวตรวจจับจะลดลง เมื่อมีแรงกดมากระทำบนแผ่นตรวจจับ มีโครงสร้างของตัวตรวจจับแสดงในรูปที่ 2.12

รูปที่ 2.12: ลักษณะโครงสร้างของตัวตรวจจับแรงกด FSR



รูปที่ 2.13: การทำงานของตัวตรวจจับแรงกด FSR

2.3.2.2 เซนเซอร์วัดความเฉื่อย



รูปที่ 2.14: เซนเซอร์วัดความเฉื่อย

Inertial Measurement Unit (IMU) เป็นส่วนประกอบหลักที่ใช่ในการนำร่องเครื่องบิน ยาน-อวกาศ ดาวเทียม เรือ ขีปนาวุธ ซึ่งในตัวของ IMU ประกอบไปด้วยสองส่วนหลักคือ Accelerometers 3 ทิศทาง ในการรับความเร่งเชิงเส้น และ Gyroscopes 3 ทิศทาง ในการบอกความเร็วเชิงมุม เซนเซอร์ตัวนี้สามารถนำมาใช้ในการหาทิศทางการหมุนของตัวหุ่นยนต์ฮิวมานอยด์ได้

เซนเซอร์วัดความเร็ว (Gyroscope) เป็นอุปกรณ์สำหรับการวัดความเร็ว หรือการรักษาการปรับทิศทาง ขึ้นอยู่กับหลักการของการอนุรักษ์โมเมนตัมเชิงมุม ถ้าไม่มีการเคลื่อนที่ อัตราการเปลี่ยนแปลงมุมจะมีค่าเท่ากับศูนย์

เซนเซอร์วัดความเร่ง (Accelerometer) เป็นอุปกรณ์ที่ใช้วัดความเร่งเชิงเส้น โดยอาศัยการวัดแรงที่กระทำต่อน้ำหนัก อ้างอิงที่เกิดจากแรงโน้มถ่วงโลก ซึ่งแรงโน้มถ่วงของโลกจะเป็นเวกเตอร์ชี้ไปที่แกนกลางโลกเสมอ ตามกฎของนิวตัน

2.3.3 แนวคิดการออกแบบกลไกการเดินของหุ่นยนต์ฮิวมานอยด์

การออกแบบหุ่นยนต์ฮิวมานอยด์ให้สามารถเดินสองขาได้เสมือนมนุษย์โดยใช้จำนวนองศาอิสระให้เท่ากับมนุษย์นั้นพบว่า มีข้อจำกัดทางด้านการออกแบบอยู่มาก เนื่องมาจากอุปกรณ์ที่ใช้ในการขับเคลื่อนข้อต่อต่างๆ มีอยู่อย่างจำกัด รวมถึงข้อจำกัดทางด้านตัวรับรู้ตัวขับของหุ่นยนต์ ดังนั้นผู้จัดทำจึงออกแบบหุ่นยนต์ให้มีองศาอิสระของข้อต่อ ในขาหนึ่งข้าง เท่ากับหกองศาอิสระ ทั้งนี้หุ่นยนต์ยังสามารถเคลื่อนที่ได้ในปริภูมิ และองศาอิสระเพียงพอต่อการใช้งาน

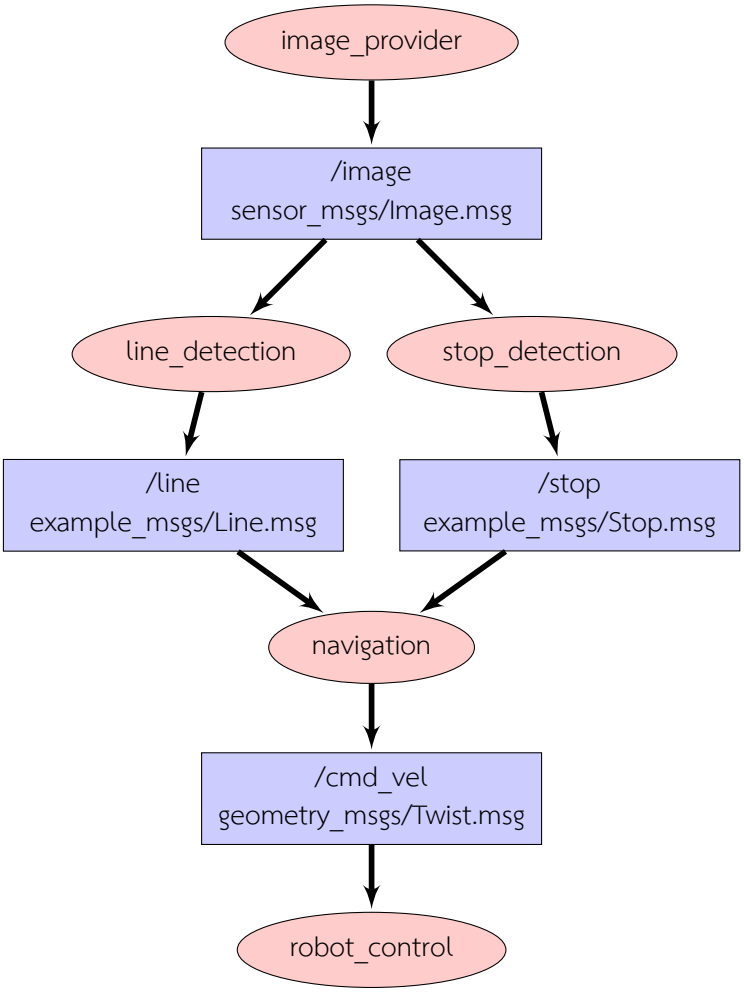
2.4 การออกแบบโปรแกรมด้วย ROS

2.4.1 Robot Operating System

The Robot Operating System (ROS) was developed by Willow Garage, originally for the PR2 robot in 2007 [Quigley et al., 2009]. It is an open source framework for developing software in robotics with a focus on the ability to run parallel on distributed computer systems. It can be run on different operating systems, but only Ubuntu and Debian are officially supported. Its main advantage is a big library of available software modules for common robotics tasks. These are developed, maintained and documented by the ROS community and adding further modules is easy. Using ROS decreases the time for developing software as most of the parts can be taken from the library. In the following section, a short overview of the concepts of ROS is provided. A deeper insight can be found in the online documentation [3].

Nodes

A node is a process in the ROS system. It can communicate with other ROS nodes via topics or services. In doing so, all nodes form a computational graph. Each node has typically a clearly defined subtask. For example one node gets the camera image, a second node detects balls on this image, and a third node computes the ball positions. Nodes can be easily reused in different tasks, for example the node which gets the camera image can be reused in a different task that detects goals. Open source implementations of nodes for most standard subtasks, especially hardware controlling, already exist. Thereby the effort in implementing a new task is drastically decreased. The most important packages used for this thesis are men- tioned in section 2.3.7



รูปที่ 2.15: ตัวอย่างสถาปัตยกรรมของ ROS

Example ROS architecture for a simple wheeled robot with the task to follow a line until it finds a stop marker. The nodes are displayed as ellipses with a name and the topics are shown as rectangles with name and message type. First, an image is provided from the camera. Lines and stop markers are detected on this image. Based on this information, a navigation node computes the necessary movement of the robot and publishes it. The robot control node is

then controlling the motors of the robot accordingly

ตารางที่ 2.1: Max and min temps recorded in the first two weeks of July

Twist.msg	
geometry_msgs/Vector3	linear
geometry_msgs/Vector3	angular

(ก) First Week

Stop.msg	
uint8	RED = 0
uint8	GREEN = 1
uint8	color
float32	distance

(ข) First Week

two example messages. The Twist message (left) is already defined in ROS. The Stop message (right) is newly defined for the special applica- tion case of figure 2.4.

Topics and Messages

Messages are the main communication method between ROS nodes. Messages are always published on a topic, which is identified by a name and can only transmit one type of message. A node can subscribe to a topic and will get the messages other nodes publish to it. Each node can subscribe to and publish on any number of topics and will not know with whom it communicates. Communication can happen between nodes running on the same computer as well as nodes distributed over different computers, as long as they are connected with an TCP/IP network. This is not only useful for parallelization but also eases the visualization of the robots state on a separate desktop computer. Each message has a type which is either predefined in the ROS system or defined by a user package. The interface of a node is mainly defined by the types of the messages it subscribes and publishes. These types can be either predefined in ROS or be newly created by a developer, cp. figure 2.5.

roscore

The roscore is the most central part of a running ROS system. It consists of the rosmaster , the ROS parameter server and the rosout logging node. The ROS master is registering which topics are published by the nodes and on which topics nodes want to subscribe. If one node wants to subscribe to a topic which another node is publishing, a peer-to-peer connection is established from one node to the other. Thereby the master does not become a bottleneck since it only connects the nodes but does not need to handle the messages. To do this, a subscribing node asks the master for a list of nodes which publish on this named topic. The master holds a table of all publishers and sends their names to the subscribing nodes. It also remembers which nodes are subscribing to this topic, if a new publisher is started later. This process is shown in figure 2.6. The ROS parameter server is a global key-value store. It is mainly designed to be used for static configuration parameters. Transmission of data between nodes should be done via topics to prevent making the parameter server a bottleneck. All parameters are globally visible. If a parameter should be able to be changed during runtime, dynamic reconfigure can be used. It

provides the possibility to state on compile time which parameter values should be changeable and provides an rqt plug-in for it, see section 2.3.9. The rosout logging node subscribes to the /rosout topic which is the standard topic for logging. ROS has built in methods to send data to this topic which are displayed on runtime and written in a log file

Services

Services can be seen as remote procedure calls (RPC). In contrast to messages which are an unidirectional stream of data, service calls are blocking and waiting for a response. They have defined types which consist of a request and a response message. The node providing the service is called server and the node calling the service is called client. Services are useful for fast tasks, but should not be used when getting the result can take a long time, because the calling node is blocked until completion. For longer tasks, actions should be used (section 2.3.5). A possible service for the example described in figure 2.4 would be a manual stop service. It would be advertised by the navigation node and would stop the robot even without markers. As this would take not a lot of time, basically just alternating the state of the navigation node, this would be possible to do with a blocking service call.

Actions

Actions are used when a task which takes a long time should be called asyn- chronously, in contrast to the synchronous service calls. Each action has a specified type which consists of three messages: goal, feedback, and result. A node providing the action, the action server, is called by another node, the action client, by sending a goal. The action server will now try to achieve this goal, while the action client is not blocked and can perform other tasks. The server will constantly send feedback messages to the client to inform it about the status of the process towards the goal. The server will send a result message when the goal was reached or if the action was interrupted. Actions can be interrupted by sending new goals which the server considers more important or by request of the action client, for example, if the sent goal is not useful anymore. A possible action for the example described in 2.4 would be driving a certain distance on the line. This needs some time because the robot has to move. Therefore a blocking call is not feasible. The action runs asynchronously and can also be interrupted, for example, if the line ends before reaching the desired distance.

Code Organization

The smallest and main unit for organizing software for ROS is a package. A package has a package.xml 2.16 which describes different metadata of this package, e.g. the package name, the author, the license and dependencies on other packages. A package can build on its own if all dependencies are met. The content can, among other things, be ROS nodes, visualization plug ins, libraries or datasets. It can be distinguished between dependencies on build and runtime. Different packages can be grouped in meta-packages, which hold no content of their own but

only collect packages which belong together.

```xml
<package>
    <name>example_package</name>
    <version>1.0.0</version>
    <description>Short example for a package.xml.</description>
    <maintainer email="ex@ample.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 2.16: ตัวอย่างไฟล์ package.xml แต่ละ tags สามารถใช้ในการบอกข้อมูลของ package นี้ ใครเป็นเจ้าของ ใครเป็นคนเขียน รวมไปถึง dependencies ที่จำเป็นต้องใช้ของ package นี้ด้วย

Code Distribution

การที่จะนำ Nodes กลับมาใช้ใหม่หรือเอาออกมาแชร์ได้นั้น จะต้องมีการทำเอกสารของ Packages นั้นๆ ด้วย โดยปกติแล้วจะถูกนำไปเก็บไว้ที่ GitHub และ package dependencies จะบอกไว้ในไฟล์ package.xml เรียบร้อยแล้ว เพื่อให้ง่ายต่อการนำไปติดตั้ง หากผู้ที่นำไปใช้พัฒนาต่อหรือแก้ไขข้อผิดพลาดก็สามารถที่จะช่วยกันได้ โดยการ Pull request หรือ Report issues ได้

ROS Packages ที่ใช้ในงานวิจัย

ในส่วนนี้จะอธิบายคร่าวๆถึง ROS standard packages ที่จะเอามาใช้ในงานวิจัยครั้งนี้

rosbag rosbag เป็นแพกเกจที่สามารถบันทึก message ที่ส่งหากันในระหว่างที่ ROS กำลังทำงานได้ ไฟล์ที่บันทึกจะเรียกว่า rosbag ประโยชน์ของมันคือเราสามารถเอาเข้ามาใช้ในการตรวจสอบ หรือนำมาเล่นซ้ำได้ อีกทั้งยังง่ายต่อการค้นหาข้อผิดพลาดอีกด้วย

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

robot_state_publisher แพกเกจที่ subscribe JointState message เพื่อที่จะนำตำแหน่งของของข้อต่อ และแปลงให้อยู่ในรูปข้อมูลของ tf2, tf2 สามารถเรียกจาก Node ใดๆก็ได้เพื่อที่จะหา Coordinate frame ที่ต้องการได้

URDF Unified Robot Description Format (URDF) เป็นไฟล์ XML ที่เอาไว้อธิบายลักษณะของหุ่นยนต์ ใน ROS มีแพกเกจที่ใช้สำหรับการอ่านไฟล์ คือ urdf_parser แต่ไฟล์นี้ก็มีการใช้งานโดย tf2 เช่นกัน

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

Visualization

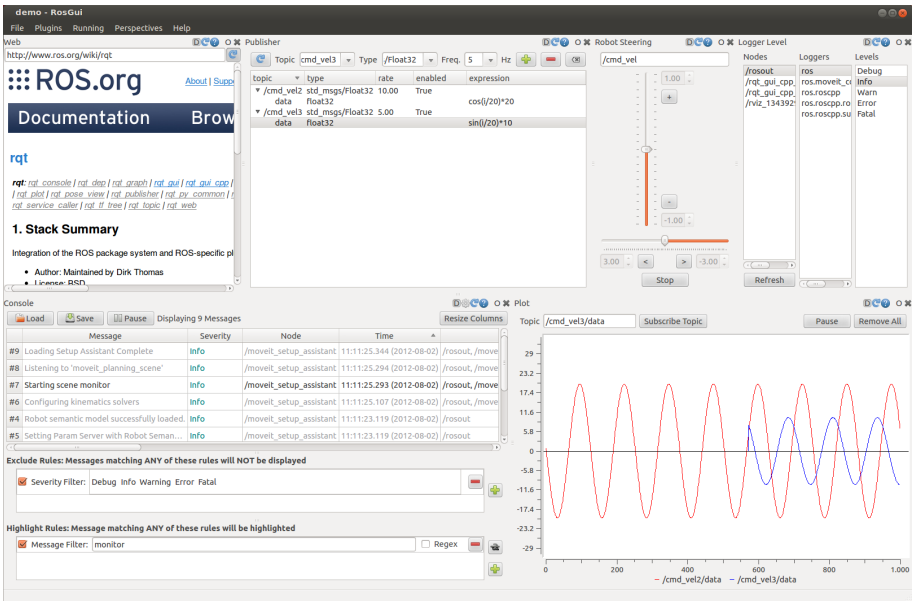
One of ROS' core strengths is providing different tools for visualization. Due to its publisher-subscriber architecture, it is very easy to establish a data stream from the program to the visualization. Either the visualization can subscribe on topics that are already being in use or additional topics can be provided from the software to deliver more information to the visualization. Messages in ROS are only published if there is a subscriber on the topic, therefore publishing additional topics for visu- alization reasons does not cost performance when the visualization is not running. ROS provides two main tools for visualization which can be extended by plug-ins. It is also possible to implement own tools which are independent from these.

rqt rqt is a QT based interface with ROS connection 2.17 Plug-ins can be launched and provide a QtWidget . Multiple widgets can be displayed at the same time. They can be resized and positioned by drag and drop. ROS provides a set of plug-ins but writing an own plug-in is possible too. These plug-ins can be used to visualize data in 2D but also to provide a controlling interface. In the following, some examples of important plug-ins are shown. The node graph plug-in shows all currently running nodes and topics. Published and subscribed topics are connected to their nodes and thereby it is easy to see the flow of data. This plug-in is especially helpful to get an overview of the running software and to find misconnected nodes. The topic monitor lists all current topics. It is possible to subscribe to them and to get the current transmitted values. Further on, statistics about the publishing rate and the used bandwidth can be shown for each topic. The rqt plot plug-in enables live plotting of data, using matplotlib . The dynamic reconfigure plug-in provides an interface to change parameter values pre- viously defined to be reconfigurable. This is useful when tuning parameter values, because changing it is possible on the fly and does not require a restart of the software

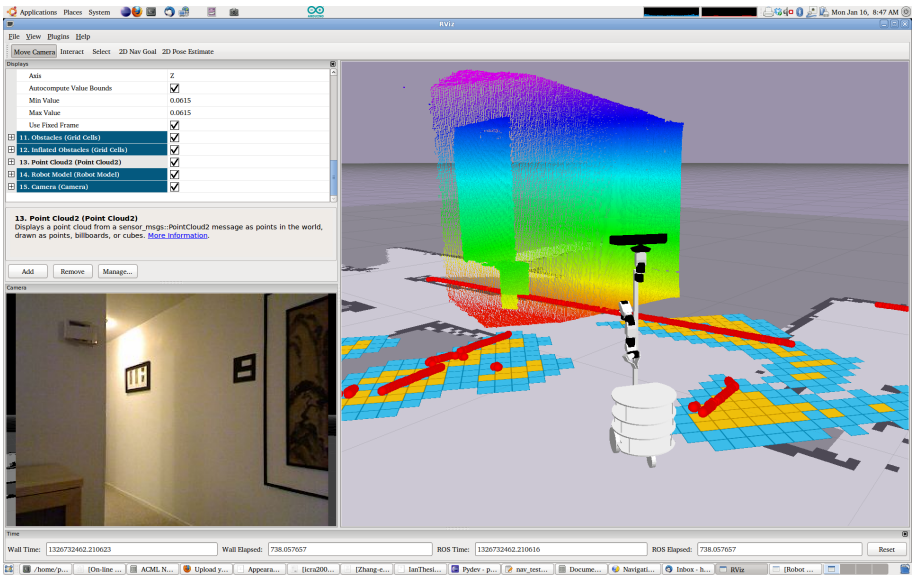
RViz RViz provides a 3D visualization of the robots state and its environment. The standardized URDF format is used to get a visual robot model, which is then used to show the current positions of the robots joints. Furthermore, sensor data can be displayed using marker messages. These messages can be published by any node and define three dimensional states which are displayed in RViz. This is for example helpful to get a visualization of recognized objects. Furthermore, a lot of different standard ROS messages can directly be visualized in RViz, e.g. camera images, depth clouds, laser scans and point clouds. Thereby RViz provides visualization without additional effort, if the standard messages are used. It is especially used for localization and mapping, because it is possible to see the current sensor inputs of the robot as well as its map in the same window 2.18

Simulation

Simulation is a crucial part when developing robot software, since it gives the developer the possibility to run his software without using hardware. This can prevent hardware damages because bugs can be found before running it on the robot and it can be used to



รูปที่ 2.17: ตัวอย่างการแสดงผลใน rqt ในรูปเป็นการนำ rqt มาเขียนเป็น GUI ให้ผู้ใช้สามารถใช้งานได้ง่าย และสามารถที่จะปรับแต่ง parameters ต่างๆได้เรียลไทม์



รูปที่ 2.18: ตัวอย่างการแสดงผลใน RViz ในรูปนี้เป็นเคสของหุ่นยนต์เคลื่อนที่ด้วยล้อ และทำแผนที่ด้วยข้อมูลความลึกที่ได้มาจาก Kinect

accelerate development, e.g. by testing in faster than real time. While ROS can generally use any simulator, Gazebo is normally preferred, since it has a good ROS integration and was also originally developed for ROS. In order to use Gazebo, an URDF of the robot is required. This URDF is used to display the robot in the simulator and to compute its collisions with itself and the environment. The simulator can provide sensor data in the corresponding standard messages. To actuate the servos of the robot, dif- ferent controllers are available which work with the corresponding messages, e.g. JointTrajectory

2.5 การออกแบบระบบพื้นฐาน

2.5.1 ข้อแตกต่างระหว่าง Open platform กับ Non-open platform

หุ่นยนต์ Open platform คือ การออกแบบระบบพื้นฐานของหุ่นยนต์ที่เปิดให้ผู้ที่ต้องการศึกษาหรือผู้ใช้ทั่วไปสามารถเข้าถึงข้อมูลต่างๆที่เกี่ยวข้องกับหุ่นยนต์นั้นๆได้ ผู้ใช้สามารถที่จะนำข้อมูลเหล่านั้นมาแก้ไข ปรับปรุง แต่งเติม หรือเรียนรู้และพัฒนาตามได้ด้วยตนเอง ซึ่งข้อมูลที่กล่าวมานั้นสามารถหาได้จากเว็บไซต์ของผู้พัฒนาหุ่นยนต์ ปัจจุบันมีหุ่นยนต์ฮิวมานอยด์ที่เป็นเปิดให้เข้าถึงหลายรูปแบบแตกต่างกันไป

ส่วนหุ่นยนต์ Non-open source platform คือหุ่นยนต์ที่สร้างมาเฉพาะเจาะจงสำหรับการวิจัย การสำรวจ หรือการแข่งขันโดยเฉพาะ ไม่เปิดให้บุคคลภายนอกเข้าศึกษาหรือแก้ไขปรับปรุง ซึ่งทำให้หุ่นยนต์ประเภทนี้ไม่เหมาะสำหรับผู้วิจัยที่จะเรียนรู้และศึกษาด้วยตนเอง เพราะมีขนาดใหญ่ ใช้ทรัพยากรมาก และการออกแบบมีความซับซ้อน เรียนรู้ยากกว่าหุ่นยนต์แบบ Open platform

บทที่ 3

การดำเนินงานวิจัย

3.1 หน้าที่ความรับผิดชอบ

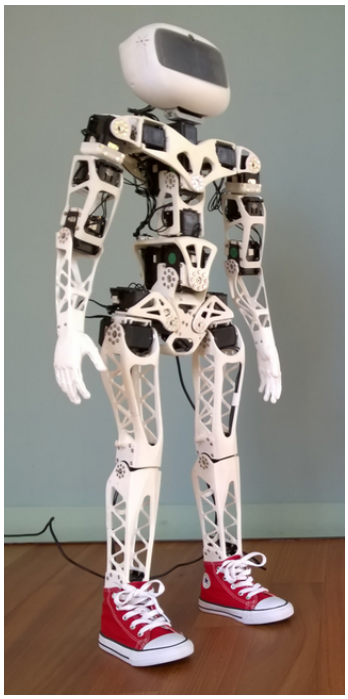
List are really easy to create

- One entry in the list
- Another entry in the list

3.2 การออกแบบโครงสร้างของหุ่นยนต์

3.2.1 การเชื่อมต่อหุ่นยนต์ฮิวมานอยด์

โครงสร้างแพลตฟอร์มหุ่นยนต์อุทัยจะประกอบไปด้วยสองขา เพื่อทำให้เกิดองศาอิสระเป็น 12 องศาอิสระ (DOFs) ใช้ Dynamixel servos 12 ตัว มอเตอร์ Dynamixel ทั้งหมดเชื่อมต่อกันแบบ daisy chain ข้างนึงของมอเตอร์ตัวแรกเชื่อมต่อกับแบตเตอร์รี่ 12V และอีกข้างต่อกับ USB2Dynamixel ทั้งหมดนี้เป็นการเชื่อมต่อ Odroid เข้ากับหุ่นยนต์ ดังรูปที่ 5.1



รูปที่ 3.1: การเชื่อมต่อระหว่าง Odroid กับ Dynamixel servos

หุ่นยนต์ฮิวมานอยด์อุทัยใช้เซอร์โวมอเตอร์ 12 ตัว ทำให้เกิดเป็น 12 องศาอิสระ USB2Dynamixel ใช้เพื่อที่จะสั่งการเซอร์โวมอเตอร์ Dynamixel ผ่าน Odroid ตำแหน่งของเซอร์โวมอเตอร์ Dynamixel EX-106+ นั้นมาจากเอนโคดเดอร์ที่อยู่ภายใน เซนเซอร์ Gyro/Accelerometer ติดอยู่กับตัวของหุ่นยนต์ เพื่อช่วยในการทรงตัวของหุ่นยนต์ เซนเซอร์ Accelerometer จะอัพเดตค่าของตัวเองเรื่อยๆ ฟังก์ชั่นส่วนเสริมจะมาจาก ROS และ Odroid เชื่อมต่อกับคอมพิวเตอร์ภายนอกผ่าน Wi-Fi

3.2.2 อุปกรณ์ที่ใช้ในหุ่นยนต์ฮิวมานอยด์อุทัย

Odroid

Dynamixel servo EX-106+

Dynamixel EX-106+ เป็นตัวขับเคลื่อนที่นิยมใช้ในปัจจุบัน โดยความสามารถของมันคือ สามารถที่จะอ่านค่าความเร็ว แรงดันไฟฟ้า กระแสไฟฟ้า อุณหภูมิ ตำแหน่ง และแรงบิด มอเตอร์แต่ละตัวจะมีบอร์ดควบคุมของตัวเอง

USB2Dynamixel connector

ตัวนี้เป็นอุปกรณ์สำหรับเชื่อมต่อ Odroid กับ Dynamixel โดยจะเชื่อมต่อผ่านพอร์ท USB ของ Odroid ไปยัง Dynamixel motor ผ่านสายทั้งหมด 4 เส้น เป็นการเชื่อมต่อแบบ RS-485

Accelerometer

Accelerometer ที่ใช้เป็น MPU-9250 Accelerometer+Gyro+Magnito เพื่อเอาไว้ใช้หามุมเอียงของหุ่นยนต์ เทียบกับโลก

Ground contact sensor

Wi-Fi Adapter

3.3 การออกแบบโปรแกรมด้วย ROS

3.4 การออกแบบระบบพื้นฐาน

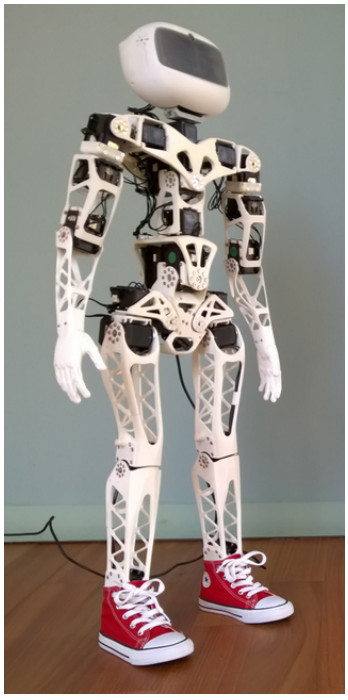
บทที่ 4

ผลการวิจัย

4.1 การออกแบบโครงสร้างของหุ่นยนต์

4.1.1 การเชื่อมต่อหุ่นยนต์ฮิวมานอยด์

โครงสร้างแพลตฟอร์มหุ่นยนต์อุทัยจะประกอบไปด้วยสองขา เพื่อทำให้เกิดองศาอิสระเป็น 12 องศาอิสระ (DOFs) ใช้ Dynamixel servos 12 ตัว มอเตอร์ Dynamixel ทั้งหมดเชื่อมต่อกันแบบ daisy chain ข้างนึงของมอเตอร์ตัวแรกเชื่อมต่อกับแบตเตอร์รี่ 12V และอีกข้างต่อกับ USB2Dynamixel ทั้งหมดนี้เป็นการเชื่อมต่อ Odroid เข้ากับหุ่นยนต์ ดังรูปที่ 5.1



รูปที่ 4.1: การเชื่อมต่อระหว่าง Odroid กับ Dynamixel servos

หุ่นยนต์ฮิวมานอยด์อุทัยใช้เซอร์โวมอเตอร์ 12 ตัว ทำให้เกิดเป็น 12 องศาอิสระ USB2Dynamixel ใช้เพื่อที่จะสั่งการเซอร์โวมอเตอร์ Dynamixel ผ่าน Odroid ตำแหน่งของเซอร์โวมอเตอร์ Dynamixel EX-106+ นั้นมาจากเอนโคดเดอร์ที่อยู่ภายใน เซนเซอร์ Gyro/Accelerometer ติดอยู่กับตัวของหุ่นยนต์ เพื่อช่วยในการทรงตัวของหุ่นยนต์ เซนเซอร์ Accelerometer จะอัพเดตค่าของตัวเองเรื่อยๆ ฟังก์ชั่นส่วนเสริมจะมาจาก ROS และ Odroid เชื่อมต่อกับคอมพิวเตอร์ภายนอกผ่าน Wi-Fi

4.1.2 อุปกรณ์ที่ใช้ในหุ่นยนต์ฮิวมานอยด์อุทัย

Odroid

Dynamixel servo EX-106+

Dynamixel EX-106+ เป็นตัวขับเคลื่อนที่นิยมใช้ในปัจจุบัน โดยความสามารถของมันคือ สามารถที่จะอ่านค่าความเร็ว แรงดันไฟฟ้า กระแสไฟฟ้า อุณหภูมิ ตำแหน่ง และแรงบิด มอเตอร์แต่ละตัวจะมีบอร์ดควบคุม

ของตัวเอง

USB2Dynamixel connector

ตัวนี้เป็นอุปกรณ์สำหรับเชื่อมต่อ Odroid กับ Dynamixel โดยจะเชื่อมต่อผ่านพอร์ท USB ของ Odroid ไปยัง Dynamixel motor ผ่านสายทั้งหมด 4 เส้น เป็นการเชื่อมต่อแบบ RS-485

Accelerometer

Accelerometer ที่ใช้เป็น MPU-9250 Accelerometer+Gyro+Magnito เพื่อเอาไว้ใช้หามุมเอียงของหุ่นยนต์ เทียบกับโลก

Ground contact sensor

Wi-Fi Adapter

4.2 การออกแบบโปรแกรมด้วย ROS

4.2.1 Simulation Gazebo

ต้องติดตั้ง package ต่อไปนี้

1 joint_state_controller

2 effort_controller

3 controller_manager*

4 gazebo_ros_control*

4.3 การออกแบบระบบพื้นฐาน

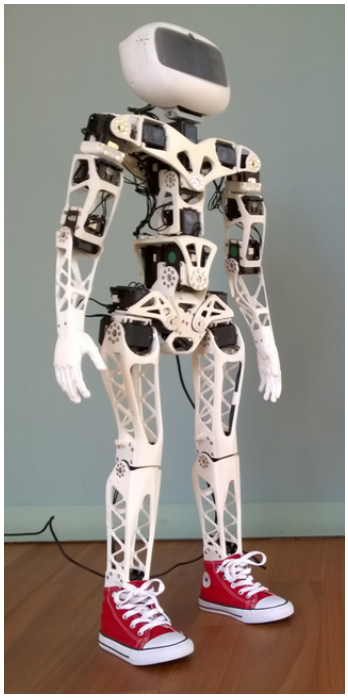
บทที่ 5

สรุปผลการทดลองและข้อเสนอแนะ

5.1 การออกแบบโครงสร้างของหุ่นยนต์

5.1.1 การเชื่อมต่อหุ่นยนต์ฮิวมานอยด์

โครงสร้างแพลตฟอร์มหุ่นยนต์อุทัยจะประกอบไปด้วยสองขา เพื่อทำให้เกิดองศาอิสระเป็น 12 องศาอิสระ (DOFs) ใช้ Dynamixel servos 12 ตัว มอเตอร์ Dynamixel ทั้งหมดเชื่อมต่อกันแบบ daisy chain ข้างนึงของมอเตอร์ตัวแรกเชื่อมต่อกับแบตเตอร์รี่ 12V และอีกข้างต่อกับ USB2Dynamixel ทั้งหมดนี้เป็นการเชื่อมต่อ Odroid เข้ากับหุ่นยนต์ ดังรูปที่ 5.1



รูปที่ 5.1: การเชื่อมต่อระหว่าง Odroid กับ Dynamixel servos

หุ่นยนต์ฮิวมานอยด์อุทัยใช้เซอร์โวมอเตอร์ 12 ตัว ทำให้เกิดเป็น 12 องศาอิสระ USB2Dynamixel ใช้เพื่อที่จะสั่งการเซอร์โวมอเตอร์ Dynamixel ผ่าน Odroid ตำแหน่งของเซอร์โวมอเตอร์ Dynamixel EX-106+ นั้นมาจากเอนโคดเดอร์ที่อยู่ภายใน เซนเซอร์ Gyro/Accelerometer ติดอยู่กับตัวของหุ่นยนต์ เพื่อช่วยในการทรงตัวของหุ่นยนต์ เซนเซอร์ Accelerometer จะอัพเดตค่าของตัวเองเรื่อยๆ ฟังก์ชั่นส่วนเสริมจะมาจาก ROS และ Odroid เชื่อมต่อกับคอมพิวเตอร์ภายนอกผ่าน Wi-Fi

5.1.2 อุปกรณ์ที่ใช้ในหุ่นยนต์ฮิวมานอยด์อุทัย

Odroid

Dynamixel servo EX-106+

Dynamixel EX-106+ เป็นตัวขับเคลื่อนที่นิยมใช้ในปัจจุบัน โดยความสามารถของมันคือ สามารถที่จะอ่านค่าความเร็ว แรงดันไฟฟ้า กระแสไฟฟ้า อุณหภูมิ ตำแหน่ง และแรงบิด มอเตอร์แต่ละตัวจะมีบอร์ดควบคุม

ของตัวเอง

USB2Dynamixel connector

ตัวนี้เป็นอุปกรณ์สำหรับเชื่อมต่อ Odroid กับ Dynamixel โดยจะเชื่อมต่อผ่านพอร์ท USB ของ Odroid ไปยัง Dynamixel motor ผ่านสายทั้งหมด 4 เส้น เป็นการเชื่อมต่อแบบ RS-485

Accelerometer

Accelerometer ที่ใช้เป็น MPU-9250 Accelerometer+Gyro+Magnito เพื่อเอาไว้ใช้หามุมเอียงของหุ่นยนต์ เทียบกับโลก

Ground contact sensor

Wi-Fi Adapter

5.2 การออกแบบโปรแกรมด้วย ROS

5.3 การออกแบบระบบพื้นฐาน

5.4 สรุปภาพรวม

ทำได้ดีนะจ๊ะ

ภาคผนวก

ภาคผนวก ก

ผลงานตีพิมพ์เผยแพร่จากวิทยานิพนธ์

ก.1 บทความวิจัยเสนอในที่ประชุมวิชาการและมีการพิมพ์รวมเล่ม

ก.2 บทความวิชาการ

ภาคผนวก ข

แหล่งข้อมูล Latex

ข.1 แหล่งข้อมูลออนไลน์

ประวัติผู้เขียน

ชื่อ สกุล	นายจิรัฏฐ์ ศรีรัตนอาภรณ์
รหัสนักศึกษา	57340500067
วุฒิการศึกษา	วิศวกรรมศาสตรบัณฑิต(วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ)
ชื่อสถาบัน	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
ปีที่สำเร็จการศึกษา	2560
ทุนการศึกษา	กินกันตายเทค
	กิดาฟนนนนหกดาา

ประวัติผู้เขียน

ชื่อ สกุล	นายเจษฎากร ทาไชยวงค์
รหัสนักศึกษา	57340500067
วุฒิการศึกษา	วิศวกรรมศาสตรบัณฑิต(วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ)
ชื่อสถาบัน	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
ปีที่สำเร็จการศึกษา	2560
ทุนการศึกษา	กินกันตายเทค
	กิดาฟนนนนหกดาา

ประวัติผู้เขียน

ชื่อ สกุล	นายวุฒิภัทร โชคอนันตทรัพย์
รหัสนักศึกษา	57340500067
วุฒิการศึกษา	วิศวกรรมศาสตรบัณฑิต(วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ)
ชื่อสถาบัน	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
ปีที่สำเร็จการศึกษา	2560
ทุนการศึกษา	กินกันตายเทค
	กิดาฟนนนนหกดาา