การพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติสำหรับอุตสาหกรรม 4.0 ด้วย ROS 2 และ LabVIEW

Development of an Autonomous Mobile Robot for Industry 4.0 Based on ROS 2 and LabVIEW

กังวานไกล ผิผ่วนนอก[1] ธนยศ อริสริยวงศ์[2]*
[1]ศูนย์แห่งความยอดเยี่ยมทางวิศวกรรมเพื่อความยั่งยืน คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ อ.องครักษ์ จ.นครนายก 26120
[2]ห้องปฏิบัติการวิจัยเมคคาทรอนิกส์และหุ่นยนต์ ภาควิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ อ.องครักษ์ จ.นครนายก 26120
Kangwankai Piphounnok[1] Tanayos Arisariyawong[2]*
[1]Exellent Center for Sustainable Engineering, Faculty of Engineering, Ongkharak, Nakhonnayok, 26120
[2]Mechatronics and Robotics Research Laboratory, Mechanical Engineering Department, Faculty of Engineering, Ongkharak, Nakhonnayok, 26120
*Corresponding author Email: Tanayos.Swu@gmail.com
(Received: September 4, 2025; Revised: September 23, 2025 ; Accepted: January 8, 2026)

บทคัดย่อ

หุ่นยนต์เคลื่อนที่อัตโนมัติถือเป็นส่วนสำคัญในกระบวนการผลิตสมัยใหม่ในยุคอุตสาหกรรม 4.0 เนื่องจากสามารถสร้างเส้นทางเดินและหลบหลีกสิ่งกีดวางได้ด้วยตนเอง ทำให้มีความยืดหยุ่นในการใช้งานสูง แต่จะพบว่าหุ่นยนต์เคลื่อนที่อัตโนมัติเชิงพาณิชย์มีราคาสูงและการเชื่อมต่อกับโปรแกรมภายนอกทำได้ยาก เป็นผลให้โรงงานอุตสาหกรรมขนาดเล็กและกลางของไทยไม่สามารถเข้าถึงเทคโนโลยีนี้ได้ ดังนั้นงานวิจัยนี้จึงนำเสนอการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติด้วยซอฟต์แวร์ ROS 2 ที่เป็นโอเพนซอร์สในการพัฒนาระบบควบคุมและนำทางของหุ่นยนต์ ร่วมกับซอฟต์แวร์ LabVIEW ที่เป็นซอฟต์แวร์ทางด้านงานวิศวกรรมเพื่อสร้างส่วนติดต่อกับผู้ใช้งาน โดยโปรแกรมที่พัฒนาขึ้นจาก LabVIEW จะรับคำสั่งจากผู้ใช้งานบนคอมพิวเตอร์และส่งข้อมูลผ่านระบบเครือข่ายไร้สายไปยังโปรแกรม ROS 2 ที่ทำงานอยู่บนคอมพิวเตอร์ของหุ่นยนต์เคลื่อนที่อัตโนมัติ เพื่อควบคุมให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งที่ต้องการ ซึ่งหุ่นยนต์เคลื่อนที่อัตโนมัติที่พัฒนาขึ้นนี้จะมีต้นทุนทางด้านซอฟต์แวร์ที่ลดลงมาก มีความยืดหยุ่นสูง และสามารถขยายต่อได้ในอนาคต จากผลการทดลองพบว่าโปรแกรมที่พัฒนาขึ้นจาก LabVIEW สามารถรับส่งข้อมูลกับโปรแกรม ROS 2 ได้เป็นอย่างดี สามารถสร้างแผนที่จากการควบคุมแบบแมนนวลผ่านโปรแกรม LabVIEW หุ่นยนต์สามารถสร้างเส้นทางเดินไปยังจุดหมายพร้อมหลบหลีกสิ่งกีดขวางได้ด้วยตัวเองอย่างมีประสิทธิภาพ โดยการนำทางในพื้นที่โล่งมีค่าความผิดพลาดเฉลี่ยในแนวแกน x, y, φ เท่ากับ 0.08 m, -0.03 m, -5.19° ตามลำดับ ส่วนกรณีการนำทางที่มีสิ่งกีดขวางมีค่าความผิดพลาดเฉลี่ยในแนวแกน x, y, φ เท่ากับ 0.07 m, -0.03 m, -4.28° ตามลำดับ

คำสำคัญ: หุ่นยนต์เคลื่อนที่อัตโนมัติ การนำทาง การบริการกระจายข้อมูล ระบบปฏิบัติการหุ่นยนต์ แลปวิว

ABSTRACT

Autonomous mobile robots (AMR) are an important part of modern manufacturing processes in the Industry 4.0 era. They can create paths and avoid obstacles on their own, providing high flexibility. However, commercial autonomous mobile robots are expensive and difficult to connect to external programs. As a result, small and medium-sized factories in Thailand cannot access this technology. Therefore, this research presents the development of autonomous mobile robots using the open-source ROS 2 software to develop robot control and navigation systems, along with LabVIEW, an engineering software, to create a user interface. The LabVIEW program receives commands from the user on the computer and transmits data via a wireless network to the ROS 2 program running on the autonomous mobile robot's computer for controlling the robot to the desired location. This autonomous mobile robot offers significantly reduced software costs, is highly flexible, and is expandable in the future. The experimental results show that the LabVIEW program effectively transmits data to the ROS 2 program. A map can be generated from manual control via LabVIEW. This autonomous mobile robot can efficiently create a path to its destination and avoid obstacles. For the open-space case, the average errors in the x, y, φ were 0.08 m, -0.03 m, -5.19°. In the case of navigation with obstacles, the average errors in the x, y, φ were 0.07 m, -0.03 m, -4.28°.

Keyword: Autonomous mobile robot, navigation, data distribution service, ROS, LabVIEW.

1. บทนำ

ปัจจุบันขบวนการผลิตในอุตสาหกรรมได้เปลี่ยนเป็นแบบอุตสาหกรรม 4.0 [1-2] ซึ่งเป็นการผลิตสมัยใหม่มีความยืดหยุ่นสูง สามารถปรับเปลี่ยนรูปแบบผลิตภัณฑ์ได้ตามความต้องการอย่างรวดเร็ว และลดการใช้แรงงานคน มีการนำระบบควบคุมอัตโนมัติมาทำงานร่วมกับหุ่นยนต์อุตสาหกรรมมากขึ้น [3-5] การขนถ่ายวัสดุเป็นกิจกรรมที่มีความสำคัญมากต่อระบบการผลิตซึ่งอาจเป็นค่าใช้จ่ายมากถึง 2 ใน 3 ของการผลิตทั้งหมด [6] ดังนั้นการขนย้ายวัสดุหรืออุปกรณ์จำเป็นจะต้องมีระบบการเคลื่อนย้ายอัตโนมัติมาช่วยทำงาน จึงมีการพัฒนารถขนส่งแบบนำทางอัตโนมัติ (Automated Guided Vehicle: AGV) หรือเรียกว่ารถเอจีวี ขึ้นมาใช้ในสายการผลิต [7] โดยรถเอจีวีจะต้องติดตั้งเส้นทางการวิ่งไว้ที่พื้นเพื่อใช้นำร่องแบบกำหนดเส้นทางเดินที่แน่นอน การปรับเปลี่ยนรูปแบบเส้นทางการวิ่งทำได้ยาก ต่อมามีการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติ (Autonomous Mobile Robot: AMR) ที่มีความสามารถหลบหลีกสิ่งกีดขวางและสร้างแผนที่เพื่อกำหนดเส้นทางการวิ่งได้ด้วยตนเองขึ้นมา [8] แต่การพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติเป็นเรื่องที่มีความยุ่งยากและซับซ้อนเนื่องจากมีองค์ประกอบหลายส่วนที่ทำงานร่วมกัน ทำให้ต้นทุนในการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติมีราคาสูงมาก อีกทั้งการพัฒนาโปรแกรมของหุ่นยนต์ก็ทำได้ลำบากเนื่องจากยังเป็นการพัฒนาแบบรวมทุกฟังก์ชันการทำงานไว้ในโปรแกรมเดียว ต่อมาได้มีการพัฒนาซอฟต์แวร์ ROS (Robot Operating System) ซึ่งเป็นชุดซอฟต์แวร์โอเพนซอร์สสำหรับพัฒนาหุ่นยนต์ โดยมีไลบรารี เครื่องมือ และเฟรมเวิร์คต่างๆ ที่ช่วยให้สามารถสร้างโมดูลการทำงานของหุ่นยนต์เป็นส่วนๆ และสามารถสื่อสารกันได้ ทำให้การพัฒนาระบบหุ่นยนต์มีความยืดหยุ่นและมีต้นทุนที่ต่ำ [9] โดยปัจจุบันได้พัฒนาเป็นซอฟต์แวร์ ROS รุ่นที่ 2 หรือเรียกว่า ROS 2 ที่มี

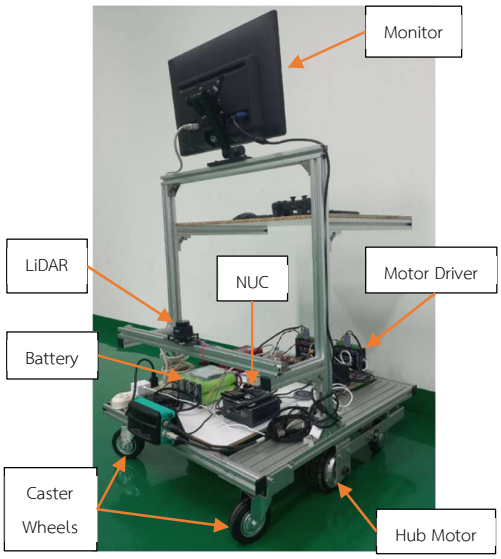
ความสามารถเพิ่มขึ้น ปรับปรุงเฟรมเวิร์คให้เป็นมาตรฐานอุตสาหกรรม เปลี่ยนระบบการสื่อสารระหว่างโหนดเป็น DDS (Data Distribution Service) ซึ่งเป็นมาตรฐานสากลในการรับส่งข้อมูลระหว่างโปรแกรมต่างๆ เป็นต้น [10] ทำให้มีการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติโดยใช้ซอฟต์แวร์ ROS 2 กันอย่างแพร่หลาย [11-12] รวมถึงมีการพัฒนาให้โปรแกรมภายนอกสามารถเชื่อมต่อกับซอฟต์แวร์ ROS ได้ด้วย [13] แต่จะพบว่าซอฟต์แวร์ ROS 2 จะมีเฉพาะเครื่องมือพัฒนาทางด้านหุ่นยนต์เท่านั้นไม่มีเครื่องมือทางด้านวิศวกรรมอื่นๆ รวมถึงไม่มีเครื่องมือสำหรับใช้สร้างส่วนติดต่อกับผู้ใช้งาน ทำให้การนำไปใช้งานจริงมีข้อจำกัดและขยายการใช้งานไปยังงานวิศวกรรมอื่นๆทำได้ยาก

ดังนั้นเพื่อแก้ปัญหาดังกล่าวข้างต้นงานวิจัยนี้จึงนำเสนอการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติโดยใช้ซอฟต์แวร์ ROS 2 พัฒนาระบบควบคุมและนำทางของหุ่นยนต์ ร่วมกับซอฟต์แวร์ LabVIEW [14] ที่ได้รับความนิยมในงานวิศวกรรมเพื่อสร้างส่วนติดต่อกับผู้ใช้งาน สามารถรับส่งข้อมูลผ่านเครือข่ายไร้สายและทำงานข้ามแพลตฟอร์มได้ ทำให้หุ่นยนต์เคลื่อนที่อัตโนมัติที่พัฒนาขึ้นนี้มีความยืดหยุ่นในการใช้งานสูง สามารถขยายการใช้งานไปยังงานวิศวกรรมต่างๆเพิ่มเติมได้ ซึ่งเหมาะกับการนำมาใช้ในอุตสาหกรรม 4.0 เป็นอย่างดี

2. วิธีดำเนินการวิจัย

2.1 ส่วนประกอบและโครงสร้างของหุ่นยนต์เคลื่อนที่อัตโนมัติ

หุ่นยนต์เคลื่อนที่อัตโนมัติที่พัฒนาขึ้นสามารถแสดงได้ดังรูปที่ 1 โดยโครงสร้างทำจากอลูมิเนียมโปรไฟล์ มีขนาด กว้างxยาวxสูง เท่ากับ 670x720x870 mm. ใช้การขับเคลื่อนแบบดิฟเฟอเรนเชียลไดร์ฟ (Differential Drive) ด้วยล้อขับจำนวน 2 ล้อ ที่เป็นอิสระต่อกัน โดยใช้มอเตอร์ฮับ (Hub Motor) เป็นต้นกำลัง ติดตั้งที่กลางตัวหุ่นยนต์ มีลูกล้อ (Caster Wheel) ด้านหน้าและด้านหลังอย่างละ 2 ล้อ สำหรับรักษาสมดุลของตัวหุ่นยนต์ มีการออกแบบกลไกให้ล้อขับและลูกล้อด้านหลังสามารถขึ้นลงได้ตามระดับพื้นที่ไม่เท่ากัน เพื่อช่วยให้ล้อขับติดพื้นตลอดเวลา ติดตั้งเซนเซอร์ไลดาร์ (LiDAR Sensor) เพื่อทำหน้าที่ตรวจจับสิ่งกีดขวางและสร้างแผนที่ ใช้คอมพิวเตอร์ขนาดเล็ก รุ่น Intel NUC Core i5-7260U เป็นตัวควบคุมหลัก ขับเคลื่อนมอเตอร์ฮับด้วยบอร์ดขับมอเตอร์ (Motor Driver) ใช้แบตเตอรี่ชนิด $LiFePO_4$ ขนาด 24 V 15 AH เป็นแหล่งพลังงาน ติดตั้งอินเวอร์เตอร์ (Inverter) ขนาด 1,600 W เพื่อแปลงไฟฟ้ากระแสตรงเป็นกระแสสลับจ่ายให้กับคอมพิวเตอร์และจอแสดงผลที่ติดตั้งด้านบนตัวหุ่นยนต์

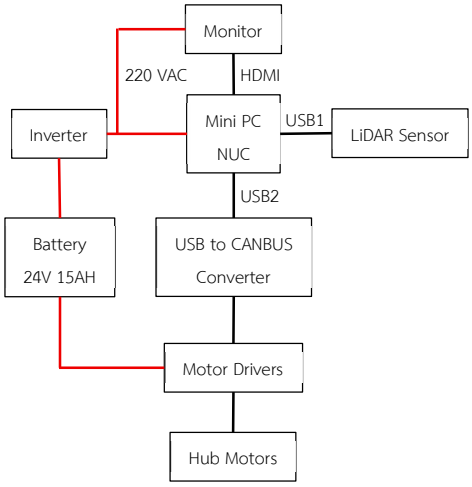


รูปที่ 1 ส่วนประกอบของหุ่นยนต์เคลื่อนที่อัตโนมัติ

2.2 ไดอะแกรมการทำงานของหุ่นยนต์เคลื่อนที่อัตโนมัติ

ไดอะแกรมการทำงานของหุ่นยนต์เคลื่อนที่อัตโนมัติสามารถแสดงได้ดังรูปที่ 2 โดยเริ่มจากแบตเตอรี่จ่ายไฟให้กับบอร์ดขับมอเตอร์เพื่อใช้ในการขับมอเตอร์ฮับ และจ่ายไฟให้กับอินเวอร์เตอร์เพื่อแปลงไฟฟ้ากระแสตรงเป็นกระแสสลับจ่ายให้กับคอมพิวเตอร์และจอแสดงผล จากนั้นจอแสดงผลต่อกับคอมพิวเตอร์ผ่านทางพอร์ต HDMI ส่วนเซนเซอร์ไลดาร์ต่อกับคอมพิวเตอร์ผ่านทาง

พอร์ต USB1 และเนื่องจากบอร์ดขับมอเตอร์รับคำสั่งควบคุมผ่านทางพอร์ต CANBUS เท่านั้น แต่คอมพิวเตอร์ NUC ไม่มีพอร์ต CANBUS ดังนั้นจึงต้องใช้อุปกรณ์แปลงสัญญาณจาก USB ไปเป็นสัญญาณ CANBUS เพื่อให้คอมพิวเตอร์สามารถสื่อสารกับบอร์ดขับมอเตอร์ได้ โดยต่อกับคอมพิวเตอร์ผ่านทางพอร์ต USB2



รูปที่ 2 ไดอะแกรมการทำงานของหุ่นยนต์เคลื่อนที่อัตโนมัติ

2.3 จลนศาสตร์ของหุ่นยนต์เคลื่อนที่อัตโนมัติ

การควบคุมการเคลื่อนที่ของหุ่นยนต์เคลื่อนที่อัตโนมัติทำได้โดยการควบคุมความเร็วเชิงมุมของล้อขับทั้งสองข้างเพื่อให้หุ่นยนต์เคลื่อนที่ไปยังทิศทางต่างๆตามที่ต้องการ แต่คำสั่งที่ใช้ในการควบคุมการเคลื่อนที่ของหุ่นยนต์ในซอฟต์แวร์ ROS 2 จะอยู่ในรูปความเร็วเชิงเส้นและความเร็วเชิงมุมของตัวหุ่นยนต์ โดยความสัมพันธ์ระหว่างความเร็วเชิงเส้นและความเร็วเชิงมุมของตัวหุ่นยนต์กับความเร็วเชิงมุมของล้อขับทั้งสองข้างสามารถพิจารณาได้จากรูปที่ 3 และจลนศาสตร์ของหุ่นยนต์แบบดิฟเฟอเรนเชียลไดร์ฟสามารถแสดงได้ดังสมการที่ 1

$$\omega_L = \frac{V - \omega \times \frac{b}{2}}{r}$$
$$\omega_R = \frac{V + \omega \times \frac{b}{2}}{r} \quad (1)$$

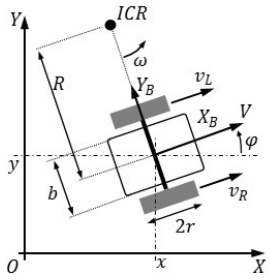
โดย ω_L, ω_R คือ ความเร็วเชิงมุมของล้อขับด้านซ้ายและขวาของหุ่นยนต์

V คือ ความเร็วเชิงเส้นของหุ่นยนต์

ω คือ ความเร็วเชิงมุมของหุ่นยนต์

b คือ ความกว้างของตัวหุ่นยนต์

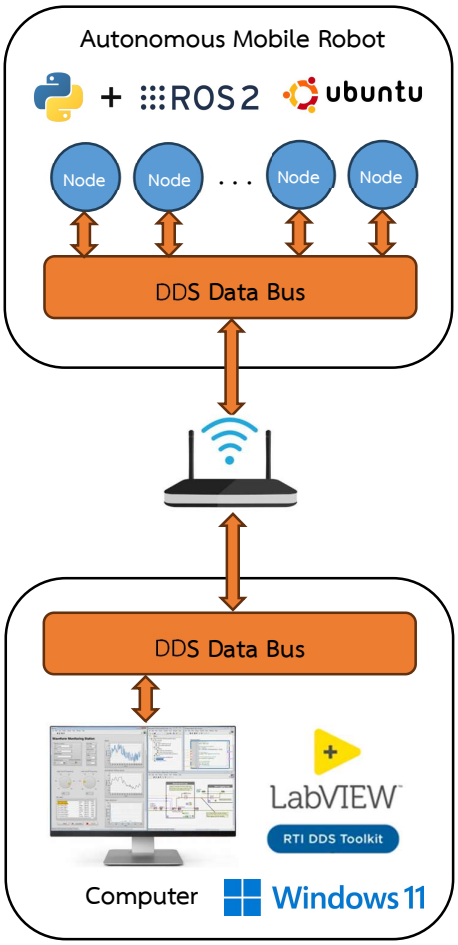
r คือ รัศมีของล้อขับ



รูปที่ 3 จลนศาสตร์ของหุ่นยนต์แบบดิฟเฟอเรนเชียลไดร์ฟ [15]

2.4 การออกแบบระบบการทำงานร่วมกันระหว่าง ROS 2 และ LabVIEW ของหุ่นยนต์เคลื่อนที่อัตโนมัติ

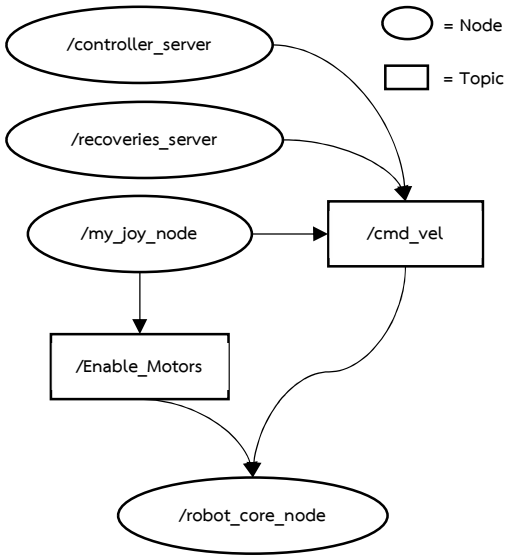
หุ่นยนต์เคลื่อนที่อัตโนมัติจะใช้ซอฟต์แวร์ ROS 2 เวอร์ชัน Galactic ในการพัฒนาระบบควบคุมและนำทาง โดยทำงานบนระบบปฏิบัติการ Ubuntu เวอร์ชัน 20.04 ส่วนซอฟต์แวร์ LabVIEW จะใช้สำหรับพัฒนาโปรแกรมติดต่อกับผู้ใช้งานโดยทำงานบนระบบปฏิบัติการ Windows 11 ดังนั้นเพื่อให้ซอฟต์แวร์ทั้งสองสามารถสื่อสารกันได้ข้ามแพลตฟอร์มจึงออกแบบระบบการทำงานร่วมกันดังรูปที่ 4 โดยในซอฟต์แวร์ ROS 2 จะประกอบไปด้วยโหนด (Node) ทำหน้าที่เหมือนโปรแกรมย่อย แต่ละโหนดจะทำหน้าที่แตกต่างกัน และสามารถสื่อสารกันได้ทั้งภายในระบบ ROS 2 เองหรือกับโปรแกรมภายนอกก็ได้ผ่านบัสข้อมูล DDS ในส่วนของโปรแกรม LabVIEW ที่พัฒนาขึ้นจะใช้ RTI DDS Toolkit [16] ในการรับส่งข้อมูลผ่านบัสข้อมูล DDS เพื่อให้เป็นมาตรฐานเดียวกันกับซอฟต์แวร์ ROS 2 และทั้งโปรแกรม LabVIEW กับ ROS 2 จะรับส่งข้อมูลกันผ่านระบบเครือข่ายไร้สาย Wi-Fi



รูปที่ 4 ระบบการทำงานร่วมกันระหว่าง ROS 2 และ LabVIEW ของหุ่นยนต์เคลื่อนที่อัตโนมัติ

2.5 การออกแบบระบบควบคุมและนำทางของหุ่นยนต์เคลื่อนที่อัตโนมัติด้วย ROS 2

ในซอฟต์แวร์ ROS 2 จะแบ่งโปรแกรมการทำงานออกเป็นโปรแกรมย่อยๆที่เรียกว่าโหนด แต่ละโหนดสามารถสื่อสารกันได้บนบัสข้อมูล DDS ผ่านสิ่งที่เรียกว่าหัวข้อ (Topic) ในงานวิจัยนี้ใช้ภาษา Python สำหรับการพัฒนาโปรแกรมของหุ่นยนต์เคลื่อนที่อัตโนมัติ และใช้เพจเกจชื่อว่า Nav2 [17] ในการนำทางและหลบหลีกสิ่งกีดขวาง เลือกใช้วิธีการกำหนดตำแหน่งของหุ่นยนต์ในแผนที่ด้วยวิธี AMCL (Adaptive Monte Carlo Localization) การสร้างเส้นทางเดินแบบ Global Path Planner ด้วยวิธี A* Algorithm และการสร้างเส้นทางเดินแบบ Local Path Planner ด้วยวิธี DWA (Dynamic Window Approach) ตัวอย่างแผนภาพของโหนดและหัวข้อในโปรแกรม ROS 2 ของหุ่นยนต์เคลื่อนที่อัตโนมัติสามารถแสดงได้ดังรูปที่ 5

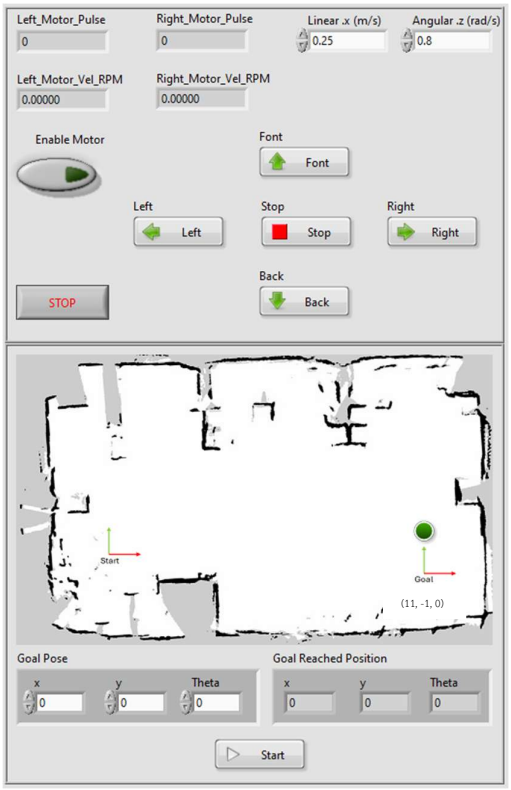


รูปที่ 5 ตัวอย่างแผนภาพของโหนดและหัวข้อในโปรแกรม ROS 2 ของหุ่นยนต์เคลื่อนที่อัตโนมัติ

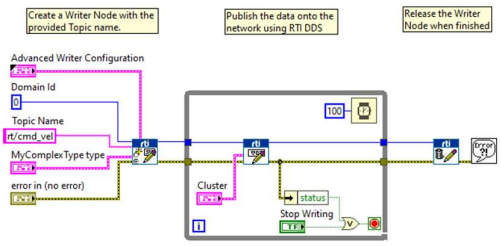
2.6 การออกแบบส่วนติดต่อกับผู้ใช้งานด้วย LabVIEW

โปรแกรมที่ทำหน้าที่ติดต่อกับผู้ใช้งานจะพัฒนาด้วยซอฟต์แวร์ LabVIEW และใช้ RTI DDS Toolkit ในการรับส่งข้อมูลผ่านบัสข้อมูล DDS เพื่อให้เป็นมาตรฐานเดียวกันกับโปรแกรม ROS 2 ที่ทำงานบนหุ่นยนต์เคลื่อนที่อัตโนมัติ โดย RTI DDS Toolkit จะทำหน้าที่แปลงข้อมูลชนิดต่างๆใน LabVIEW ให้เป็นข้อความและหัวข้อตามมาตรฐานบัสข้อมูล DDS ซึ่งจะต้องตรงกันกับที่ได้กำหนดไว้ในโปรแกรม ROS 2 เช่นจากรูปที่ 5 หัวข้อที่ชื่อว่า /cmd_vel จะรับข้อมูลของความเร็วเชิงเส้นและความเร็วเชิงมุมที่ต้องการให้หุ่นยนต์เคลื่อนที่เป็นข้อความชนิด Float64 ดังนั้นเมื่อผู้ใช้ป้อนข้อมูลของความเร็วเชิงเส้นและความเร็วเชิงมุมที่ต้องการในโปรแกรม LabVIEW แล้ว RTI DDS Toolkit ก็จะแปลงข้อมูลดังกล่าวในโปรแกรม

LabVIEW ไปเป็นข้อความชนิด Float64 แล้วส่งข้อความไปที่หัวข้อ /cmd_vel ผ่านทางบัสข้อมูล DDS จากนั้นโปรแกรม ROS 2 ในหุ่นยนต์เคลื่อนที่ก็จะรับข้อมูลนี้แล้วสั่งให้หุ่นยนต์เคลื่อนที่ตามที่ต้องการ โดยข้อมูลทั้งหมดสื่อสารกันผ่านระบบเครือข่ายไร้สาย Wi-Fi หน้าต่างของโปรแกรม LabVIEW ที่พัฒนาขึ้น และตัวอย่างโค้ดสามารถแสดงได้ดังรูปที่ 6 และ รูปที่ 7 ตามลำดับ



รูปที่ 6 โปรแกรม LabVIEW สำหรับติดต่อกับผู้ใช้งาน



รูปที่ 7 ตัวอย่างโค้ดโปรแกรม LabVIEW ที่ใช้ส่งข้อมูลความเร็วไปที่หัวข้อ /cmd_vel ผ่าน RTI DDS Toolkit

จากรูปที่ 6 หน้าต่างโปรแกรมจะแบ่งออกเป็น 2 ส่วน คือส่วนกรอบบนใช้สำหรับสั่งควบคุมหุ่นยนต์เคลื่อนที่แบบแมนนวล (Manual) พร้อมทั้งแสดงข้อมูลของตำแหน่งและความเร็วขณะหุ่นยนต์กำลังเคลื่อนที่ ใช้สำหรับการทำแผนที่บริเวณที่ต้องการทดสอบ ส่วนกรอบล่างใช้สำหรับทดสอบการนำทางและหลบหลีกสิ่งกีดขวาง โดยจะแสดงแผนที่บริเวณที่ต้องการทดสอบ จากนั้นกำหนดพิกัดจุดหมายปลายทาง คือ ระยะตามแนวแกน x ระยะตามแนวแกน y และมุมที่ต้องการรอบแกน z (Theta) แล้วกดปุ่ม Start หุ่นยนต์จะเริ่มเคลื่อนที่ด้วยตัวเอง เมื่อถึงจุดหมายหลอดไฟบนแผนที่สีเขียวจะติดพร้อมแสดงข้อมูลพิกัดของตำแหน่งปลายทางที่เกิดขึ้นจริง ส่วนรูปที่ 7 เป็นการแสดงตัวอย่างโค้ดโปรแกรม LabVIEW ที่ใช้ส่งข้อมูลความเร็วที่ต้องการไปที่หัวข้อ /cmd_vel ผ่าน RTI DDS Toolkit เพื่อให้โปรแกรม ROS 2 บนหุ่นยนต์เคลื่อนที่นำไปเป็นข้อมูลสำหรับสั่งให้หุ่นยนต์เคลื่อนที่ตามความเร็วที่ได้รับมา

3. ผลการวิจัย

3.1 การทดลองควบคุมแบบแมนนวล

ในการทดลองนี้จะเป็นการทดสอบการควบคุมแบบแมนนวลเพื่อทำแผนที่สำหรับใช้ทดสอบการนำทางและหลบหลีกสิ่งกีดขวางแบบอัตโนมัติ จากคุณสมบัติของมอเตอร์ที่ใช้เป็นล้อขับ จึงกำหนดให้หุ่นยนต์เคลื่อนที่ด้วยความเร็วเชิงเส้นเท่ากับ 0.25 m/s และความเร็วเชิงมุมเท่ากับ 0.8 rad/s โดยใช้โปรแกรม LabVIEW ที่พัฒนาขึ้นดังรูปที่ 6 เริ่มจากกดปุ่ม Enable Motor เพื่อให้มอเตอร์ที่ล้อขับทำงาน จากนั้นกดปุ่มลูกศรกำหนดทิศทางเพื่อให้หุ่นยนต์เคลื่อนที่ไปรอบๆบริเวณที่ต้องการทำแผนที่จนรอบ โดยการทำแผนที่จะใช้เทคนิค Cartographer [18] โดยขนาดพื้นที่สำหรับใช้ทำแผนที่มีขนาด กว้างxยาว เท่ากับ 11x15 m.

จากผลการทดลองพบว่าการควบคุมหุ่นยนต์เคลื่อนที่แบบแมนนวลด้วยโปรแกรม LabVIEW ที่พัฒนาขึ้น ผ่านเครือข่ายไร้สาย Wi-Fi สามารถทำได้อย่างมีประสิทธิภาพ

ไม่รู้สึกถึงการหน่วงเวลาในการรับส่งข้อมูลระหว่างคอมพิวเตอร์ของผู้ใช้งานกับคอมพิวเตอร์บนตัวหุ่นยนต์เคลื่อนที่ ภาพแผนที่บริวณพื้นที่ทดสอบที่เก็บได้สามารถแสดงได้ดังรูปที่ 8



รูปที่ 8 แผนที่บริวณพื้นที่ทดสอบที่เก็บได้

3.2 การทดลองการนำทางและหลบหลีกสิ่งกีดขวางแบบอัตโนมัติ

ในการทดลองนี้จะนำแผนที่จากการทดลองที่ 3.1 มาใช้ทดสอบการนำทางและหลบหลีกสิ่งกีดขวางของหุ่นยนต์เคลื่อนที่อัตโนมัติ โดยจะแบ่งการทดลองออกเป็น 2 การทดลองย่อย คือ การทดลองการนำทางในพื้นที่โล่ง และการนำทางที่มีสิ่งกีดขวาง ซึ่งทั้งสองการทดลองจะใช้ระบบพิกัดฉากและอ้างอิงตัวแปรตามรูปที่ 3 โดยมีพิกัดจุดเริ่มต้น (x, y, φ) เท่ากับ (0, 0, 0) และพิกัดจุดหมายปลายทางเท่ากับ (11, -1, 0) กำหนดให้ระยะทางตามแนวแกน x และ y มีหน่วยเป็นเมตร ส่วนมุม φ มีหน่วยเป็นองศา ตั้งค่าตัวแปรที่สำคัญในเพจเกจ Nav2 ได้แก่ ความเร็วสูงสุดในการเคลื่อนที่เชิงเส้นและเชิงมุมเท่ากับ 0.40 m/s และ 0.55 rad/s ตามลำดับ เพื่อรักษาความความผิดพลาดให้อยู่ในช่วงที่กำหนดไว้ พื้นที่สำหรับทำ Local Cost Map เท่ากับ 3x3 m ซึ่งเป็นค่าที่เหมาะสมกับหน่วยความจำที่มีและประสิทธิภาพของ CPU ค่าความผิดพลาดที่ยอมรับได้ตามระนาบ xy และมุม φ เท่ากับ 0.1 m และ 5.73° ตามลำดับ และขนาด กว้างxยาว ของ Robot Footprint เท่ากับ 0.680x0.730 m. ซึ่งมีขนาดใหญ่กว่าตัวหุ่นยนต์เล็กน้อย โดยจะทำการทดลองจำนวน 5 รอบ บันทึกผลค่าความผิดพลาดของตำแหน่งตามแนวแกน x, y, φ รวมถึงเวลาที่ใช้ในการเคลื่อนที่

3.2.1 การทดลองนำทางในพื้นที่โล่ง

สำหรับการทดลองนี้จะสั่งให้หุ่นยนต์เคลื่อนที่จากตำแหน่งเริ่มต้นไปยังจุดหมายปลายทางผ่านทางโปรแกรม LabVIEW ที่พัฒนาขึ้นดังรูปที่ 6 โดยจะไม่มีสิ่งกีดขวางเพื่อทดสอบความสามารถในการนำทางเป็นหลัก ผลการทดลองสามารถแสดงได้ดังตารางที่ 1 ส่วนเส้นแนวทางเดินที่สร้างจาก Nav2 สามารถแสดงได้ดังรูปที่ 9 และการเคลื่อนที่ของหุ่นยนต์จริงสามารถแสดงได้ดังรูปที่ 10

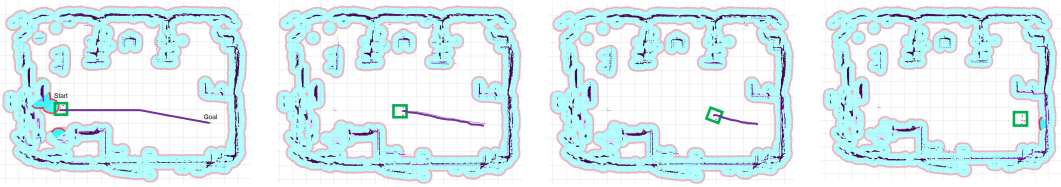
ตารางที่ 1 ผลการทดลองนำทางในพื้นที่โล่ง

ลำดับ	ค่าความผิดพลาด			เวลา (s)
	x (m)	y (m)	φ (deg)	
1	0.08	-0.03	-5.35	44.14
2	0.09	-0.04	-4.91	44.64
3	0.09	-0.03	-5.36	44.42
4	0.07	-0.03	-5.80	44.53
5	0.08	-0.03	-4.51	44.72
เฉลี่ย	0.08	-0.03	-5.19	44.49

จากตารางที่ 1 จะพบว่าค่าความผิดพลาดเฉลี่ยในแนวแกน x, y, φ เท่ากับ 0.08 m, -0.03 m, -5.19° ตามลำดับ และเวลาเฉลี่ยที่ใช้ในการเคลื่อนที่เท่ากับ 44.49 s ซึ่งจะพบว่าค่าความผิดพลาดที่ได้ไม่เกินค่าไว้ใน Nav2

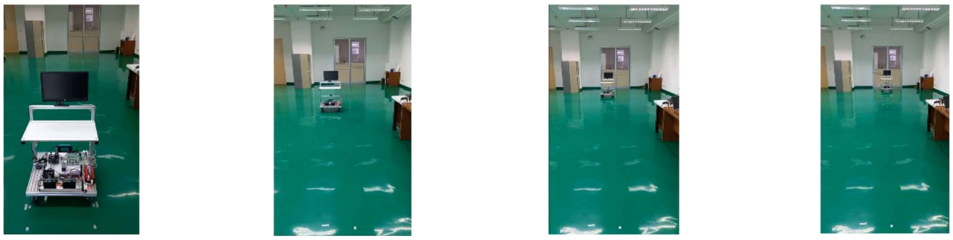
3.2.2 การทดลองนำทางที่มีสิ่งกีดขวาง

สำหรับการทดลองนี้จะสั่งให้หุ่นยนต์เคลื่อนที่จากตำแหน่งเริ่มต้นไปยังจุดหมายปลายทางเช่นเดียวกับหัวข้อ 3.2.1 แต่จะมีสิ่งกีดขวางเพื่อทดสอบความสามารถในการนำทางและหลบหลีกสิ่งกีดขวาง ผลการทดลองสามารถแสดงได้ดังตารางที่ 2 ส่วนเส้นแนวทางเดินที่สร้างจาก Nav2 สามารถแสดงได้ดังรูปที่ 11 และการเคลื่อนที่ของหุ่นยนต์จริงสามารถแสดงได้ดังรูปที่ 12



(ก) (ข) (ค) (ง)

รูปที่ 9 เส้นแนวทางเดินของหุ่นยนต์จากการทดลองนำทางในพื้นที่โล่ง โดยเริ่มจากรูป 9 (ก) ไปยัง 9 (ง)



(ก) (ข) (ค) (ง)

รูปที่ 10 การเคลื่อนที่ของหุ่นยนต์จริงจากการทดลองนำทางในพื้นที่โล่ง โดยเริ่มจากรูป 10 (ก) ไปยัง 10 (ง)

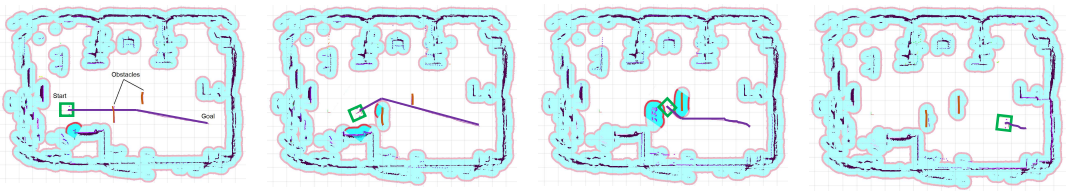
ตารางที่ 2 ผลการทดลองนำทางที่มีสิ่งกีดขวาง

ลำดับ	ค่าความผิดพลาด			เวลา (s)
	x (m)	y (m)	φ (deg)	
1	0.08	-0.03	-4.62	47.26
2	0.03	-0.03	-4.40	48.10
3	0.08	-0.04	-4.20	47.53
4	0.09	-0.04	-4.32	48.15
5	0.08	-0.03	-3.87	47.13
เฉลี่ย	0.07	-0.03	-4.28	47.63

จากตารางที่ 2 ค่าความผิดพลาดเฉลี่ยในแนวแกน x, y, φ เท่ากับ 0.07 m, -0.03 m, -4.28° และเวลาเฉลี่ยที่ใช้ในการเคลื่อนที่เท่ากับ 47.63 s ซึ่งจะพบว่าค่าความผิดพลาดที่ได้ไม่เกินค่าที่ตั้งไว้ใน Nav2 ส่วนเวลาที่ใช้ในการเคลื่อนที่กรณีมีสิ่งกีดขวางจะมากกว่ากรณีพื้นที่โล่ง

4. สรุปและอภิปรายผล

งานวิจัยนี้ได้นำเสนอการพัฒนาหุ่นยนต์เคลื่อนที่อัตโนมัติด้วยซอฟต์แวร์ ROS 2 ที่เป็นโอเพนซอร์สร่วมกับซอฟต์แวร์ LabVIEW ที่เป็นซอฟต์แวร์ทางด้านงานวิศวกรรมที่ได้รับความนิยม เพื่อออกแบบระบบควบคุมและนำทางหุ่นยนต์รวมถึงส่วนติดต่อกับผู้ใช้งาน โดยแสดงให้เห็นถึงความสามารถในการทำงานข้ามแพลตฟอร์ม สื่อสารผ่านเครือข่ายไร้สาย การลดต้นทุนด้านซอฟต์แวร์ และความสามารถในการขยายต่อได้ จากผลการทดลองจะพบว่าโปรแกรมที่พัฒนาขึ้นจาก LabVIEW สามารถรับส่งข้อมูลกับโปรแกรม ROS 2 ได้เป็นอย่างดี หุ่นยนต์สามารถสร้างเส้นทางไปยังจุดหมายพร้อมหลบหลีกสิ่งกีดขวางได้ด้วยตัวเองอย่างมีประสิทธิภาพ โดยมีค่าความผิดพลาดเฉลี่ยไม่เกินค่าที่ได้ตั้งไว้



(ก) (ข) (ค) (ง)

รูปที่ 11 เส้นแนวทางเดินของหุ่นยนต์จากการทดลองนำทางที่มีสิ่งกีดขวาง โดยเริ่มจากรูป 11 (ก) ไปยัง 11 (ง)



(ก) (ข) (ค) (ง)

รูปที่ 12 การเคลื่อนที่ของหุ่นยนต์จริงจากการทดลองนำทางที่มีสิ่งกีดขวาง โดยเริ่มจากรูป 12 (ก) ไปยัง 12 (ง)

5. กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนอุดหนุนการวิจัยจากงบประมาณเงินรายได้คณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ ประจำปีงบประมาณ 2567 หมายเลขโครงการ 201/2567

6. เอกสารอ้างอิง

[1] M. A. M. S. Lemstra and M. A. de Mesquita, “Industry 4.0: A tertiary literature review,” *Technol. Forecast. Soc. Change*, vol. 186, p. 122204, Jan. 2023.

[2] S. Rane, P. Shah and R. Sekhar, “Survey of technologies for Industry 4.0,” in *Proc. 6th Int. Conf. Comput., Commun., Control Autom. (ICCUBEA)*, Pune, India, 2022, pp.1–6.

[3] S. Eskilander, *Product Design for Automatic Assembly – A Method for Product Design (DFA2)*. Stockholm, Sweden: Royal Institute of Technology (KTH), 2001.

[4] E. Oztemel and S. Gursev, “Literature review of Industry 4.0 and related technologies,” *J. Intell. Manuf.*, vol. 31, pp. 127–182, 2020.

[5] R. Y. Zhong, X. Xu, E. Klotz and S. T. Newman, “Intelligent manufacturing in the context of Industry 4.0: A review,” *Engineering*, vol. 3, no. 5, pp. 616–630, Oct. 2017.

[6] P. Chutima, *Flexible Manufacturing System*. Bangkok, Thailand: Chulalongkorn University Press, 2001.

[7] R. C. Arkin and R. R. Murphy, “Autonomous navigation in a manufacturing environment,” *IEEE Trans. Robot. Autom.*, vol. 6, no. 4, pp. 445–454, Aug. 1990.

[8] X. Zhao and T. Chidambareswaran, “Autonomous mobile robots in manufacturing operations,” in *Proc. 19th Int. Conf. Autom. Sci. Eng. (CASE)*, Auckland, New Zealand, 2023, pp. 1–7.

[9] L. Joseph and J. Cacace, *Mastering ROS for Robotics Programming: Best Practices and Troubleshooting Solutions When Working with ROS*, 3rd ed. Birmingham, U.K.: Packt Publishing, 2021.

[10] D. Thomas, W. Woodall, and E. Fernandez, “Next-generation ROS: Building on DDS,” in *Proc. ROSCon 2014*, Chicago, IL, USA, Sept. 2014. Mountain View, CA: Open Robotics.

[11] P. Phueakthong and J. Varagul, “A development of mobile robot based on ROS2 for navigation application,” in *Proc. Int. Electron. Symp. (IES)*, Surabaya, Indonesia, 2021, pp. 517–520.

[12] Y. Liu, Y. Lu, C. Peng, Q. Bu, Y. C. Liang and J. Sun, “Autonomous vehicle based on ROS2 for indoor package delivery,” in *Proc. 28th Int. Conf. Autom. Comput. (ICAC)*, Birmingham, U.K., 2023, pp. 1–7.

[13] C. R. Pozna, E. Horváth, and J. Kovács, “Developing rapid prototype-capable applications for industrial mobile robot platforms,” in *Proc. IEEE 18th Int. Conf. Intell. Eng. Syst. (INES)*, Tihany, Hungary, 2014, pp. 203–207.

[14] H. W. Picot, M. Ateeq, B. Abdullah, and J. Cullen, “Industry 4.0 LabVIEW-based industrial condition monitoring system for industrial IoT,” in *Proc. 12th Int. Conf. Develop. eSystems Eng. (DeSE)*, Kazan, Russia, 2019, pp. 1020–1025.

[15] ROS.org. (2023, April 13). Diff drive controller [Online]. Available: https://wiki.ros.org/diff_drive_controller

[16] Real-Time Innovations, Inc. (2024, Nov 18). Data Distribution Service Toolkit [Online]. Available: www.ni.com/en/support/downloads/tools-network/download.data-distribution-service-toolkit.html

[17] Open Navigation LLC. (2025, Aug 20). Navigation2 (Nav2) Framework [Online]. Available: https://docs.nav2.org/

[18] X. Zhang, J. Lai, D. Xu, H. Li, and M. Fu, “2D lidar-based SLAM and path planning for indoor rescue using mobile robots,” *J. Adv. Transp.*, vol. 2020, p. 8867937, 2020.