การระบุตำแหน่งของ AGV ในร่มอย่างมีประสิทธิภาพ:โดยใช้อัลกอริทึม EXTEND KALMAN FILTER โดยคำนึงถึงความไม่แน่นอนของเซ็นเซอร์และอัตราการสุ่มข้อมูลที่เปลี่ยนแปลง

นายนพรุจ เขียวนาค

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาปรัชญาดุษฎีบัณฑิต
สาขาวิชาวิศวกรรมเครื่องกลและระบบกระบวนการ
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2568

EFFICIENT LOCALIZATION OF INDOOR AGVS: AN ALGORITHMIC APPROACH UTILIZING EXTEND KALMAN FILTER WITH CONSIDERATION OF SENSOR UNCERTAINTY AND CHANGING SAMPLING RATE

NOPPARUT KHAEWNAK

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy in Mechanical and Process System Engineering
Suranaree University of Technology
Academic Year 2025

การระบุตำแหน่งของ AGV ในร่มอย่างมีประสิทธิภาพ:โดยใช้อัลกอริทึม EXTEND KALMAN FILTER โดยคำนึงถึงความไม่แน่นอนของเซ็นเซอร์ และอัตราการสุ่มข้อมูลที่เปลี่ยนแปลง

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาดุษฎีบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

.....

(รศ. ดร.บัณฑิต กฤตาคม)
ประธานกรรมการ

.....

(รศ. ดร.จิระพล ศรีเสริฐผล)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

.....

(ผศ. ดร.สุรเดช ตัญตรัยรัตน์)
กรรมการ

.....

(ผศ. ดร.ธีทัต ดลวิชัย)
กรรมการ

.....

(ผศ. ดร.โศรฎา แข็งการ)
กรรมการ

.....

(รศ. ดร.ยุพาพร รักสกุลพิวัฒน์)
รักษาการแทนรองอธิการบดีฝ่ายวิชาการและประกันคุณภาพ

.....

(รศ. ดร.พรศิริ จงกล)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

นพรุจ เขียวนาค : การระบุตำแหน่งของ AGV ในร่มอย่างมีประสิทธิภาพ:โดยใช้อัลกอริทึม EXTEND KALMAN FILTER โดยคำนึงถึงความไม่แน่นอนของเซ็นเซอร์และอัตราการสุ่มข้อมูลที่เปลี่ยนแปลง (EFFICIENT LOCALIZATION OF INDOOR AGVS: AN ALGORITHMIC APPROACH UTILIZING EXTEND KALMAN FILTER WITH CONSIDERATION OF SENSOR UNCERTAINTY AND CHANGING SAMPLING RATE) อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร.จิระพล ศรีเสริฐผล, 146 หน้า

คำสำคัญ : รถอัตโนมัติในร่ม/การระบุตำแหน่ง/ตัวกรองคามานแบบขยาย/เซนเซอร์ฟิวชั่น

การระบุตำแหน่งของยานพาหนะนำทางอัตโนมัติ (AGV) เป็นปัจจัยการเพิ่มผลิตภาพในภาคอุตสาหกรรม แต่มีข้อจำกัดจากต้นทุนสูงจากเซนเซอร์ วิทยานิพนธ์นี้นำเสนอการพัฒนาระบบระบุตำแหน่งสำหรับ AGV ในอาคาร ที่มีความแม่นยำและเสถียรภาพ โดยอาศัยการหลอมรวมข้อมูลจากชุดเซนเซอร์ต้นทุนต่ำ เริ่มต้นจากการประเมินประสิทธิภาพของเซนเซอร์ ซึ่งประกอบด้วย กล้อง (Camera) การคำนวณตำแหน่งโดยประมาณ (Dead Reckoning) หน่วยวัดแรงเฉื่อย (IMU) และอัลตร้าไวด์แบนด์ (UWB) พบว่าเซนเซอร์ UWB มีอัตราความคลาดเคลื่อนสูงที่สุดเมื่อเทียบกับเซนเซอร์ชนิดอื่น เพื่อให้การหลอมรวมข้อมูลมีเสถียรภาพและความน่าเชื่อถือ จึงได้ใช้ข้อมูลจากเซนเซอร์หลัก 3 ชนิด ได้แก่ กล้อง, Dead Reckoning และ IMU โดยได้นำเสนอระเบียบวิธีที่ครอบคลุมตั้งแต่การปรับปรุงคุณภาพข้อมูลด้วยเทคนิคการแก้ไขความเบี่ยงเบน (Bias Correction) และการถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting) หัวใจของงานวิจัยอยู่ที่การออกแบบและประเมินประสิทธิภาพของอัลกอริทึม Blended Extended Kalman Filter (Blended EKF) ซึ่งเป็นแนวทางใหม่ที่ทำการผสมผสานค่าการวัดจากเซนเซอร์หลักก่อนนำเข้าสู่กระบวนการอัพเดตสถานะ และนำมาเปรียบเทียบกับตัวกรอง Extended Kalman Filter (EKF) ในรูปแบบมาตรฐาน ผลการทดลองเชิงประจักษ์ยืนยันว่า อัลกอริทึม Blended EKF ที่นำเสนอสามารถประมาณค่าตำแหน่งและทิศทางของ AGV ได้อย่างดีเยี่ยม โดยให้เส้นทางการเคลื่อนที่ที่ราบรื่นและต่อเนื่องกว่ารูปแบบมาตรฐานอย่างมีนัยสำคัญ ข้อค้นพบนี้ได้พิสูจน์ให้เห็นถึงศักยภาพของแนวทางที่นำเสนอในการเป็นโซลูชันที่ทรงประสิทธิภาพ สำหรับการประยุกต์ใช้ระบบ AGV ในสภาพแวดล้อมอุตสาหกรรมจริง ซึ่งต้องการทั้งความแม่นยำ ความเสถียร และความคุ้มค่าในการลงทุน

สาขาวิชาวิศวกรรมเครื่องกล ลายมือชื่อนักศึกษา นพรุจ

ปีการศึกษา 2568 ลายมือชื่ออาจารย์ที่ปรึกษา

NOPPARUT KHAEWNAK : EFFICIENT LOCALIZATION OF INDOOR AGVS: AN ALGORITHMIC APPROACH UTILIZING EXTEND KALMAN FILTER WITH CONSIDERATION OF SENSOR UNCERTAINTY AND CHANGING SAMPLING RATE. ASSOC. PROF. JIRAPHON SRISERTPOL, Ph.D., 146 PP.

Keywords: Indoor Agvs/Localization/Extend Kalman Filter/Fusion

Automated Guided Vehicle (AGV) positioning is a factor in increasing productivity in the industrial sector, but it is limited by the high cost of sensors. This thesis presents the development of an accurate and stable indoor positioning system for AGVs by fusion of data from a set of low-cost sensors. Starting with the evaluation of sensor performance, which includes cameras, dead reckoning, inertial measurement units (IMUs), and ultra-wideband (UWB) sensors, it is found that UWB sensors have the highest error rate among the other sensors. To ensure stable and reliable data fusion, data from three main sensors: cameras, dead reckoning, and IMUs are used. A comprehensive methodology is proposed, starting from data quality improvement with bias correction techniques and inverse-variance weighting. The heart of the research lies in the design and evaluation of the Blended Extended Kalman Filter (Blended EKF) algorithm, a novel approach that combines measurements from primary sensors before inputting them into the state updating process and compares them with a standard Extended Kalman Filter (EKF). Empirical experimental results confirm that the proposed Blended EKF algorithm can excellently estimate the position and orientation of AGVs, providing significantly smoother and more continuous trajectories than standard models. These findings demonstrate the potential of the proposed approach as an effective solution for AGV system applications in real-world industrial environments where accuracy, stability, and cost-effectiveness are required.

School of Mechanical Engineering
Academic Year 2025

Student's Signature.....
Advisor's Signature.....

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงด้วยดี ต้องขอขอบพระคุณ บุคคลและกลุ่มบุคคล องค์กรต่าง ๆ ที่ให้ความกรุณาคำปรึกษา แนะนำ ช่วยเหลือทั้งในด้านวิชาการ ด้านการดำเนินงานวิจัยดังต่อไปนี้

รองศาสตราจารย์ ดร.จิระพล ศรีเสริฐผล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้คำปรึกษา แนะนำ ช่วยแก้ปัญหา ให้กำลังใจ ให้ความเมตตาอารี เป็นอย่างดีในการดำเนินชีวิต

รองศาสตราจารย์ เรืออากาศเอก ดร.กนต์ธร ชำนิประศาสน์ ผู้ช่วยศาสตราจารย์ ดร. สุรเดช ตัญตรับรัตน์ ผู้ช่วยศาสตราจารย์ ดร.เศรษฐวิทย์ ภูฉายา และคณาจารย์ทุกท่าน ที่ให้การสนับสนุนเครื่องมืออุปกรณ์ ให้กำลังใจ ให้ความเมตตาอารี เป็นอย่างดีในการดำเนินชีวิต

อาจารย์ทัศพันธุ์ สุวรรณทัต สำนักวิชาวิศวกรรมศาสตร์และนวัตกรรม มหาวิทยาลัยเทคโนโลยีราชมงคลตะวันออก ที่คอยให้การสนับสนุน ให้กำลังใจ ให้ความเมตตาอารี

นายศิริพงษ์ ปะวะโก พี่ ๆ น้อง ๆ และเพื่อน ๆ ในกลุ่ม System and Control Engineering Laboratory ที่ให้คำปรึกษา แนะนำแนวทาง รวมถึงให้กำลังใจตลอดมา

สำนักวิชาวิศวกรรมศาสตร์และนวัตกรรม มหาวิทยาลัยเทคโนโลยีราชมงคลตะวันออก และมหาวิทยาลัยเทคโนโลยีสุรนารี ที่สนับสนุนในการศึกษาต่อครั้งนี้

ท้ายที่สุด ขอขอบพระคุณ ครอบครัว ที่ให้กำลังใจอย่างต่อเนื่อง สนับสนุนและเข้าใจในการศึกษา ผู้เขียนขอขอบพระคุณมา ณ โอกาสนี้

ขอขอบพระคุณทุกท่าน ที่มีส่วนเกี่ยวข้องในวิทยานิพนธ์ฉบับนี้ และขอขอบคุณ คุณครูและอาจารย์ทุกท่าน รวมถึงผู้มีพระคุณ ที่คอยเป็นกำลังใจให้ตลอดมา และตลอดไป

นพรุจ เขียวนาค

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

คำอธิบายสัญลักษณ์และคำย่อ

A	คือ ค่าความเข้มของสัญญาณที่ระยะ 1 เมตร (dBm)
AGV	คือ รถขนส่งอัตโนมัติ
AMR	คือ หุ่นยนต์เคลื่อนที่อัตโนมัติ
A_{re}	คือ ค่าความผิดพลาดสัมพัทธ์
A_r	คือ พื้นที่หน้าตัดของล้อขวา
A_l	คือ พื้นที่หน้าตัดของล้อซ้าย
BALTY	คือ แพลตฟอร์ม AGV ต้นแบบ
BC	คือ การแก้ไขความเบี่ยงเบนของข้อมูล
CAM	คือ กล้อง
CCTV	คือ กล้องวงจรปิด
CNN	คือ เครือข่ายประสาทเทียมแบบคอนโวลูชัน
DDMR	คือ หุ่นยนต์ขับเคลื่อนสองล้ออิสระ
DR	คือ การคำนวณตำแหน่งจากการวัดการเคลื่อนที่
EKF	คือ ตัวกรองคัลมานแบบขยาย
GPS	คือ ระบบกำหนดตำแหน่งบนโลก
GT	คือ เส้นทางจริงอ้างอิง
IMU	คือ เซนเซอร์วัดแรงเฉื่อย
IoT	คือ อินเทอร์เน็ตของสรรพสิ่ง
L	คือ ระยะห่างระหว่างจุดศูนย์กลางถึงล้อ
LiDAR	คือ เซนเซอร์ตรวจวัดระยะด้วยเลเซอร์
M	คือ จำนวนเซนเซอร์ทั้งหมด
MQTT	คือ โปรโตคอลสื่อสารข้อมูล
n	คือ ค่าสูญเสียของสัญญาณในสภาพแวดล้อม
P Q R	คือ เมทริกซ์ Covariance ของ State Process Measurement
$R(\theta)$	คือ เมทริกซ์การหมุน
R_{blend}	คือ Covariance ของข้อมูลที่ผสม
RMSE	คือ รากของค่าเฉลี่ยกำลังสองของข้อผิดพลาด

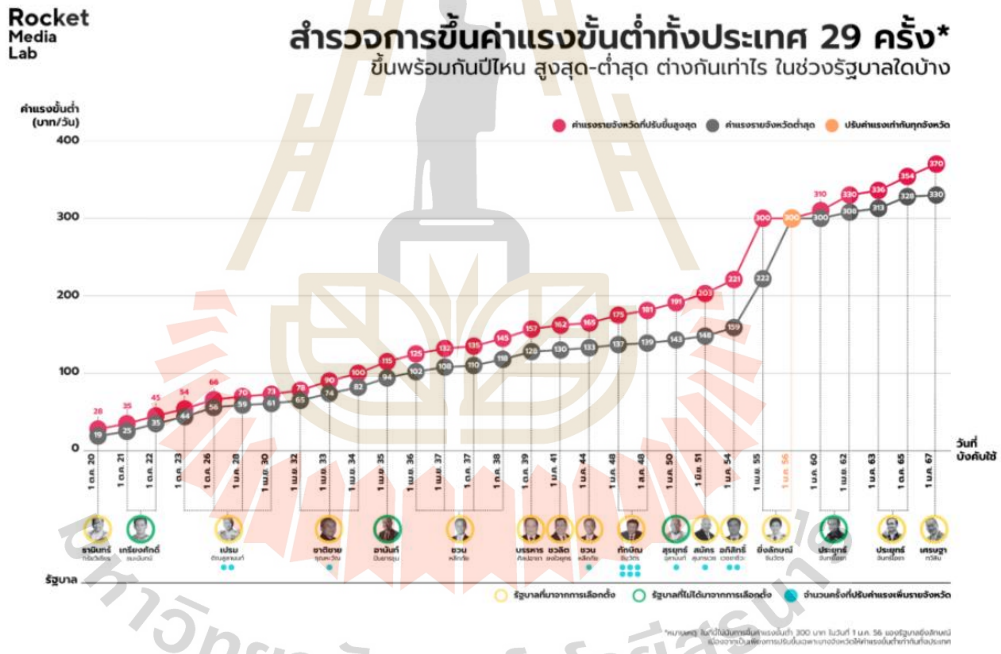
คำอธิบายสัญลักษณ์และคำย่อ (ต่อ)

ROS	คือ ระบบปฏิบัติการสำหรับหุ่นยนต์
SVM	คือ Support Vector Machine
Tag	คือ อุปกรณ์รับสัญญาณในระบบ UWB
Anchor	คือ อุปกรณ์ส่งสัญญาณในระบบ UWB
ToF	คือ Time of Flight (เวลาที่สัญญาณใช้ในการเดินทาง)
u(t)	คือ อินพุตควบคุม
UWB	คือ ระบบสื่อสารย่านความถี่กว้างพิเศษ
u_k	คือ เวกเตอร์ควบคุม (EKF Prediction Step)
v ω	คือ ความเร็วเชิงเส้นและความเร็วเชิงมุม
Var_X Var_Y	คือ ความแปรปรวนของข้อผิดพลาดแกน X Y
Var_θ	คือ ความแปรปรวนของข้อผิดพลาดแกน θ
v_L v_R	คือ ความเร็วของล้อซ้าย–ขวา (m/s)
$_VPL$	คือ ความเร็วจุด P ล้อซ้าย
$_VPR$	คือ ความเร็วจุด P ล้อขวา
WT901C	คือ รุ่น IMU ที่ใช้ในงานวิจัย
x y	คือ พิกัดตำแหน่งในแกน X และ Y (เมตร)
$\hat{x}$	คือ ค่าสถานะที่ประมาณ
X_0 Y_0	คือ พิกัดใน Global Frame
Xr Yr	คือ พิกัดใน Robot Frame
α	คือ ค่าน้ำหนักของข้อมูลการผสม
α_x α_y	คือ ค่าน้ำหนักของ CAM และ DR ตามแกน x และ y
Δt dt	คือ ช่วงเวลาระหว่างการวัด
θ	คือ มุมทิศทางของ AGV (เรเดียน)
σ σ^2	คือ ส่วนเบี่ยงเบนมาตรฐาน และ ความแปรปรวน

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหาในงานวิจัย

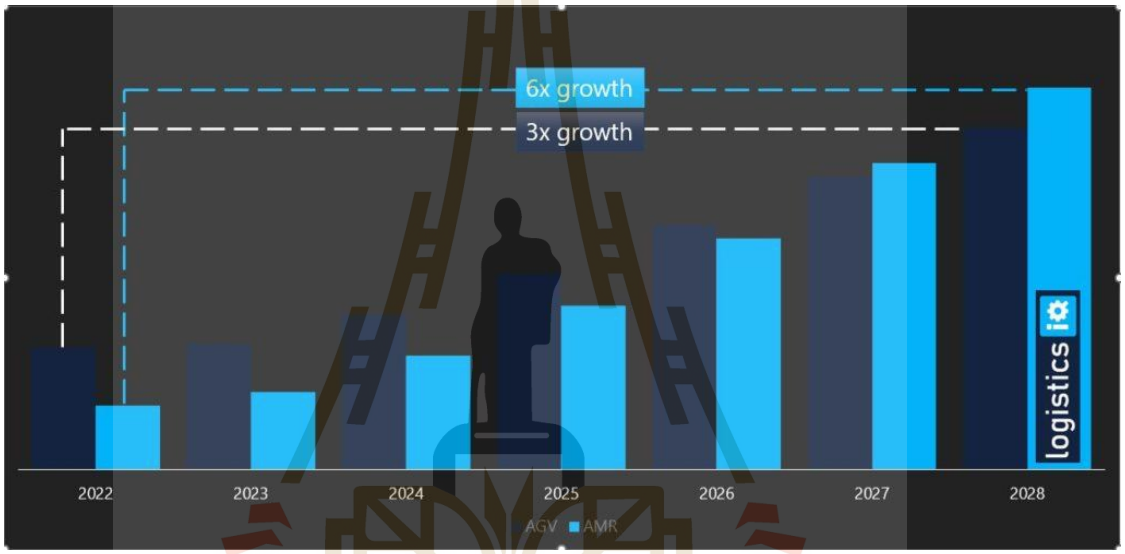
ภาคอุตสาหกรรมกำลังเผชิญกับการแข่งขันที่รุนแรงมากขึ้น จากค่าจ้างแรงงานที่สูงขึ้นอย่างต่อเนื่อง (ดังรูปที่ 1.1) ขณะเดียวกันจำนวนแรงงานที่เข้าสู่ระบบกลับลดลง สถานการณ์ดังกล่าวทำให้ภาคอุตสาหกรรมจำเป็นต้องนำนวัตกรรมมา เพื่อเพิ่มการแข่งขันและระดับประสิทธิภาพกระบวนการผลิต



รูปที่ 1.1 สำรวจค่าแรงขั้นต่ำ (Rocket Media, 2023)

รถนำทางอัตโนมัติ หรือ AGV (Automated Guided Vehicle) เป็นหนึ่งสิ่งที่สำคัญในภาคอุตสาหกรรม ที่ช่วยขนส่งวัสดุในอุตสาหกรรม เพื่อทดแทนแรงงานมนุษย์ ทำให้ AGV เติบโตมากขึ้น (ดังรูป 1.2) อย่างไรก็ตามการใช้งาน AGV ยังมีอุปสรรคสำคัญของต้นทุนที่สูง ในส่วนของระบบการระบุตำแหน่ง (Localization System) ซึ่งเป็นส่วนที่สำคัญในการนำทาง

ในงานวิจัยนี้ คำว่า *“เซนเซอร์ราคาถูก (Low-cost sensors)”* หมายถึง อุปกรณ์ตรวจวัดเชิงพาณิชย์ที่มีต้นทุนต่ำ และเป็นทรัพยากรที่มีอยู่แล้วหรือควรมีอยู่ในระบบอุตสาหกรรมสมัยใหม่ เช่น กล้องวงจรปิด (CCTV), เอนโคดเดอร์ และหน่วยวัดแรงเฉื่อย (IMU) พื้นฐาน โดยมุ่งเน้นแนวคิดการนำทรัพยากรที่มีอยู่มาใช้ให้เกิดประโยชน์สูงสุด (resource reuse) แทนการเพิ่มฮาร์ดแวร์เฉพาะทางที่มีราคาสูง ทั้งนี้ แม้เซนเซอร์ดังกล่าวจะมีความไม่แน่นอนของการวัดสูง แต่สามารถชดเชยข้อจำกัดดังกล่าวได้ด้วยอัลกอริทึมการหลอมรวมข้อมูลและการประมาณสถานะ เช่น Extended Kalman Filter และ Blended Extended Kalman Filter ซึ่งเป็นแนวทางหลักของงานวิจัยนี้



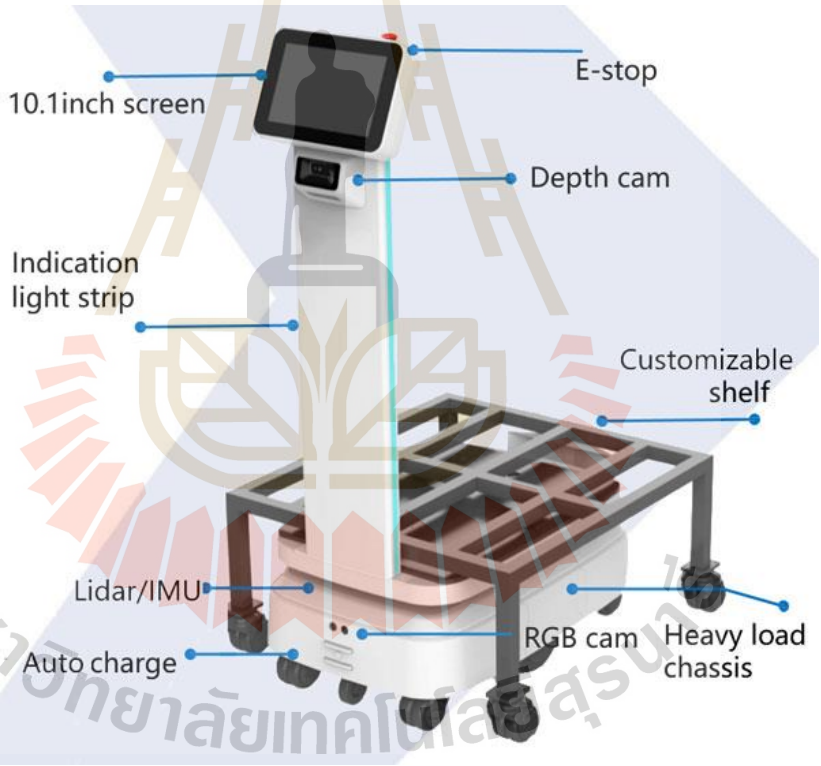
รูปที่ 1.2 AGV-AMR Market (TheLogisticsIq, 2023)

สำหรับราคารถ AGV ในตลาดปัจจุบันขึ้นอยู่กับขนาดของรถและเทคโนโลยี โดยรถ AGV ขนาดเล็กอาจมีราคาสูงถึง 30,000 ถึง 40,000 เหรียญสหรัฐ ในขณะที่รถ AGV ซึ่งนำทางด้วยแถบแม่เหล็กจะมีราคาเริ่มต้นที่ประมาณ 15,000 เหรียญสหรัฐ (Network, 2024) นอกจากการใช้งานในภาคอุตสาหกรรมแล้ว AGV ยังถูกประยุกต์ใช้ในพื้นที่ที่เป็นอันตราย เช่น พื้นที่ในการกู้ภัย (Trevelyan et al., 2008) หรือการปฏิบัติงานในพื้นที่ปนเปื้อนสารเคมีอันตราย (Fragapane et al., 2020; Okin et al., 2019) ในการพัฒนาระบบ AGV โดยทั่วไปมักนิยมใช้ ระบบปฏิบัติการหุ่นยนต์ หรือ ROS (Robot Operating System) เป็นแพลตฟอร์มหลัก เนื่องจากมีแหล่งชุมชนนักพัฒนาขนาดใหญ่ที่ให้การสนับสนุน อย่างไรก็ตาม ROS มีใช้ทรัพยากรประมวลผลที่สูง มีความยุ่งยากสำหรับการเรียนรู้ และไม่มีเสถียรภาพจากการอาศัยแพ็คเกจภายนอก (Robotics, 2023) โดยพื้นฐานแล้ว AGV ที่มีความสามารถในการนำทางอัตโนมัติ (ดังรูป 1.3) จะประกอบด้วยโครงสร้างสำหรับรับน้ำหนัก (Heavy load chassis) ระบบขับเคลื่อน (Drive System) พร้อมแบตเตอรี่ และชุดเซนเซอร์

เช่น เซนเซอร์เลเซอร์ (LiDAR) หน่วยวัดแรงเฉื่อย (IMU) และเซนเซอร์ภาพ (Cameras) ทั้งกล้อง RGB และกล้องวัดความลึก (Depth cam) ยังประกอบไปด้วย เช่น จอแสดงผล ปุ่มหยุดฉุกเฉิน (E-stop) และเซนเซอร์ตรวจจับสิ่งกีดขวาง (Collision Sensors) จากองค์ประกอบดังกล่าว สามารถแบ่ง AGV ออกเป็น 4 ส่วน ได้แก่ สร้างแผนที่ สร้างเส้นทาง การควบคุมตัวรถ และระบุตำแหน่งของตัวรถ

1.1.1 สร้างแผนที่

แผนที่เป็นขั้นตอนแรกในการเริ่มต้นการทำงาน ซึ่ง AGV จะต้องรับรู้ตำแหน่งเริ่มต้นก่อน จากนั้นจึงใช้เซนเซอร์เก็บข้อมูลจากเซนเซอร์เพื่อนำมาสร้างเป็นแผนที่ กระบวนการนี้ รถ AGV จะทำการสร้างแผนที่และระบุตำแหน่งไปพร้อมกัน เรียกว่า SLAM (Simultaneous Localization and Mapping) ในทำนองเดียวกัน หากใช้ กล้อง จะเรียกว่า VSLAM (Visual SLAM) (Tao et al., 2020)



รูปที่ 1.3 อุปกรณ์รถ AGV (Anewtech, 2024)

เมื่อใช้ LiDAR เรียกว่า LiDAR SLAM (LiDAR Simultaneous Localization and Mapping) เมื่อมีการเก็บข้อมูลล่วงหน้าและเก็บข้อมูลของรถจะใช้อัลกอริธึม Support Vector Machine (SVM) ใช้ในการตรวจหารถที่สนใจได้ (Wang & Zhang, 2022) เมื่อสร้างแผนที่จากเซนเซอร์ต่าง ๆ ดัง

ข้างต้นจะเป็นลักษณะในการเรียนรู้แผนที่แล้วสร้างแผนที่ไม่ว่าจะเป็นแผนที่ 2 มิติ หรือ 3 มิติ จะมีการเก็บข้อมูลเหล่านั้นก่อนมาสร้างพื้นที่หรือแผนที่ในการใช้งาน เพื่อที่ใช้สร้างเส้นทางในหัวข้อถัดไป

1.1.2 สร้างเส้นทาง

จากแผนที่ที่ได้จะมีการเก็บพื้นที่ที่เป็นอุปสรรคหรือสิ่งกีดขวางเข้ามาในแผนที่ เมื่อรู้ตำแหน่งของตัวรถ AGV และต้องการไปยังที่สนใจก็จะมีการใช้การสุ่มของเส้นทางและหาเส้นทางที่ใกล้ โดยใช้อัลกอริทึม A* ในการวางแผนเส้นทาง (Guruji et al., 2016) และจะมีการผสมกับวิธีกำลังสองน้อยที่สุดเพื่อให้เส้นทางสั้นและราบรื่น (Liu et al., 2022) เมื่อรู้เส้นทางก็จะส่งข้อมูลไปบังคับการทำงานของตัวรถในหัวข้อถัดไป

1.1.3 ควบคุมตัวรถ

ถ้าเป็นลักษณะที่เคลื่อนที่ตามเส้น ใช้งานกล้องโดยวิธีควบคุมแบบ Fuzzy เพื่อให้เกิดความราบเรียบ (Davoudi & Heidarian, 2020) การทำงานของตัวรถที่ไม่ใช้เส้นต้องรู้เส้นทางการทำงานของตัว AGV และต้องทำการคาดการณ์ตำแหน่งที่รถจะไปก่อน ถ้าการคาดการณ์ตำแหน่งห่างจากตัวรถมากเกินไปจะทำให้ตัวรถไม่อยู่ในเส้นทางที่วางแผนไว้ แต่ถ้าห่างน้อยไปจะทำให้ตัวรถ AGV เคลื่อนที่เหมือนงู ซึ่งวิธีการประมาณตำแหน่งที่ใช้คือ อัลกอริทึม Pure Pursuit Controller (MathWorks, 2024) เมื่อรู้ตำแหน่งที่เคลื่อนที่ จะไปสั่งงานตัวรถ AGV จะมีการคำนวณผ่านแบบจำลองตัวรถ AGV ผ่านตัวควบคุมแบบอนุพันธ์ตามสัดส่วน (PD) ซึ่งสามารถควบคุมบนผิวที่ไม่เหมือนกัน (Li et al., 2022) ในการทำงานของตัวรถ สร้างแผนที่ สร้างเส้นทาง จากกล้องที่ติดกับตัวรถ AGV เมื่อเคลื่อนที่อาจจะทำให้ภาพเอียง จากการสั่นของตัวรถจึงมีการใช้ไจโรสโคปในการปรับภาพ หมุนภาพ ลดการสั่นของกล้อง (Zhao & Ding, 2018) แต่สิ่งที่สำคัญมากสำหรับรถ AGV คือรู้ตำแหน่งของตัวรถ AGV อาจจะทำให้เกิดการผิดพลาดตั้งแต่ต้นทาง จึงในมาอยู่ในส่วนสุดท้ายที่สำคัญ

1.1.4 ระบุตำแหน่งของตัวรถ

ปัจจุบันมีเทคนิคการระบุตำแหน่งหลายรูปแบบที่ใช้ในหลากหลาย ตั้งแต่การนำทางในชีวิตประจำวันไปจนถึงการใช้งานในอุตสาหกรรม ระบบกำหนดตำแหน่ง GPS (Global Positioning System) เป็นระบบดาวเทียมที่ให้ข้อมูลตำแหน่ง โดยใช้ข้อมูลจากสัญญาณจากดาวเทียม ระบบนี้มีการใช้งานอย่างทั่วไป เช่น รถยนต์อัตโนมัติ แต่ใช้ในพื้นที่ปิดไม่ได้ (Firdaus et al., 2023) หรือระบุตำแหน่งภายในอาคาร IPS (Indoor Positioning Systems) มีการใช้เทคโนโลยี เช่น Wi-Fi Bluetooth หรือ RFID โดยมีการติดตั้งอุปกรณ์เพิ่มในการระบุตำแหน่ง เช่น QR code เป็นจุดสังเกตเพื่อยืนยันตำแหน่งของตัวรถให้แม่นยำยิ่งขึ้น (Ang et al., 2020) ส่วน LIDAR หรือกล้องภายในอาคารร่วมกับ YOLO V3 ในการหาตำแหน่งและสถานะของตัวรถได้ (Chen et al., 2019) ซึ่งเหมาะสำหรับหุ่นยนต์อัตโนมัติ หรือยานพาหนะอัตโนมัติแต่ต้องการทรัพยากรประมวลผล

ที่สูง และการระบุตำแหน่งด้วยเซนเซอร์ (Sensor-based Positioning) โดยการใช้เซนเซอร์เช่น IMU (Inertial Measurement Unit) ที่รวม accelerometers และ gyroscopes ช่วยในการหาตำแหน่งในระยะสั้น โดยมักใช้ร่วมกับเทคนิคอื่น ๆ (Jang et al., 2019) เซนเซอร์มีค่าความไม่แน่นอน(Sensor Uncertainty) อัตราการเก็บข้อมูล ความแม่นยำของเซนเซอร์ที่แตกต่างกัน ทำให้เกิดค่าคว่มผิดพลาดของตำแหน่ง (Alharbi & Karimi, 2021; Jiang et al., 2021; Sari, 2020) จึงใช้วิธีการหลอมรวมเซนเซอร์ (Sensor Fusion) เป็นหนึ่งวิธีในการเพิ่มความแม่นยำ (Zong et al., 2020) การหลอมรวมเซนเซอร์ทำให้ช่วยเพิ่มความแม่นยำ แต่ยังช่วยให้ AGV สามารถรับรู้พื้นที่ที่ซับซ้อน โดยใช้ข้อมูลของเซนเซอร์ที่แตกต่างกัน (Wang et al., 2020) การหลอมรวมเซนเซอร์มีหลายวิธี เช่น Kalman Filter (KF) เป็นการหลอมรวมเซนเซอร์ที่มีรูปแบบเชิงเส้น Extended Kalman Filter (EKF) สามารถหลวมรวมเซนเซอร์ได้ที่เป็นแบบไม่เชิงเส้นที่ไม่ซับซ้อน Adaptive Monte Carlo Localization(AMCL) หรือ PF (Particle Filter) และ UKF (Unscented Kalman Filter) สามารถหลวมรวมเซนเซอร์ได้ที่เป็นแบบไม่เชิงเส้น แต่มีการประมวลผลจำนวนมาก (Gupta et al., 2015; Liu et al., 2024; Pastor, 2024; Sentech, 2024; Stelzer et al., 2017; Ullah et al., 2021; Wang et al., 2020; Yuan et al., 2022)

เมื่อพิจารณาภายในอุตสาหกรรม โดยทั่วไปมีการทำงานของ AGV มีเส้นทางที่ชัดเจนและซับซ้อนไม่สูง ทำให้เลือกใช้ Extended Kalman Filter (EKF) เนื่องจากเป็นวิธีที่สามารถใช้งานกับระบบที่ไม่เป็นเชิงเส้น โดยใช้การประมวลผลที่น้อยกว่าเมื่อเทียบกับวิธีการอื่น เช่น Particle Filter (PF) หรือ Unscented Kalman Filter (UKF) ซึ่งเป็นเป้าหมายหลักที่จะทำให้ต้นทุนต่ำ จึงได้หลีกเลี่ยงเซนเซอร์สมรรถนะสูงอย่าง LiDAR หรือ Depth Camera ซึ่งต้องการทรัพยากรประมวลผลสูง และอีกเป้าหมายคือการใช้เซนเซอร์ต้นทุนต่ำมาทำการการหลอมรวม อันได้แก่ Encoder หน่วยวัดแรงเฉื่อย (IMU) อัลตร้าไวด์แบนด์ (UWB) และกล้องวงจรปิด (CCTV) ซึ่งอยู่แล้วในอาคาร ความท้าทายของเซนเซอร์ต้นทุนต่ำ คือความแตกต่างกันอย่างมากของเซนเซอร์ ทั้งในด้านอัตราการสุ่มตัวอย่างข้อมูล ค่าความผิดพลาด และความแปรปรวนของสัญญาณที่ และมีโอกาสเกิดข้อผิดพลาดของสัญญาณได้ตลอดเวลา

นอกเหนือจากการใช้เซนเซอร์บนตัวรถแล้ว ปัจจุบันมีแนวทางที่ได้รับความสนใจมากขึ้น คือการใช้กล้องส่วนกลางที่ติดตั้งจากมุมสูงหรือเพดาน เพื่อให้มุมมองภาพรวมแบบ “God’s-eye view” ครอบคลุมพื้นที่ทั้งหมดของโรงงาน แนวทางนี้ช่วยลดภาระการติดตั้งเซนเซอร์หลายชนิดบน AGV และย้ายภาระการประมวลผลสู่ระบบส่วนกลาง ซึ่งเหมาะสมต่อการพัฒนาโซลูชันต้นทุนต่ำ อย่างไรก็ตาม ระบบกล้องเพียงอย่างเดียวยังมีข้อจำกัดจากการเปลี่ยนแปลงของแสง เงาที่ซับซ้อน และการบดบังในบางช่วง ทำให้ข้อมูลตำแหน่งที่ได้มีความไม่แน่นอนและเกิดความคลาดเคลื่อนได้ง่ายในสภาพการใช้งานจริง

แม้ว่าการประมวลผลภาพขั้นสูง เช่น Deep Learning จะช่วยให้การตรวจจับวัตถุทำได้แม่นยำขึ้น แต่ยังต้องใช้กำลังประมวลผลสูงและมีความไวต่อความแปรปรวนของสภาพแวดล้อม เช่น illumination variation, occlusion และ motion blur จึงยังไม่เหมาะอย่างยิ่งสำหรับการใช้งานจริงในสายการผลิตที่มีสภาวะไม่แน่นอนและเปลี่ยนแปลงตลอดเวลา ความท้าทายเหล่านี้ทำให้การใช้ข้อมูลจากกล้องเพียงอย่างเดียวไม่เพียงพอสำหรับระบบ AGV ที่ต้องการความต่อเนื่องและความเสถียรของการระบุตำแหน่งในทุกสถานการณ์

ดังนั้น การหลอมรวมข้อมูลจากหลายแหล่ง (Multi-Sensor Fusion) เช่น Encoder, IMU, UWB และสัญญาณจากกล้องจึงมีความสำคัญในการเสริมความแม่นยำและลดผลกระทบจาก noise ที่เกิดขึ้นกับเซนเซอร์เดี่ยว การผสานข้อมูลหลายชนิดช่วยลดข้อจำกัดเฉพาะของแต่ละเซนเซอร์ เช่น drift ของระบบ Dead Reckoning หรือ dropout ของข้อมูลภาพ ทำให้ระบบระบุตำแหน่งมีความเชื่อถือได้มากขึ้นภายใต้สภาวะแวดล้อมที่มีความผันผวนสูง

ด้วยเหตุนี้ การหลอมรวมข้อมูลจากหลายแหล่ง (Multi-Sensor Fusion) เช่น Encoder, IMU และสัญญาณภาพจากกล้อง จึงเป็นกลไกสำคัญในการเพิ่มความแม่นยำของตำแหน่งและลดข้อจำกัดของเซนเซอร์เดี่ยว เช่น drift จาก Dead Reckoning หรือ dropout จากกล้อง การผสานข้อมูลหลายชนิดช่วยเพิ่มความเชื่อถือได้และรองรับสภาพแวดล้อมที่มีความผันผวนสูงได้ดีกว่าเดิม

1.2 วัตถุประสงค์งานวิจัย

1.2.1 เพื่อพัฒนาเทคนิคการระบุตำแหน่งของรถ AGV ภายในร่มอาคารที่มีราคาถูก

1.2.2 เพื่อพัฒนาโปรแกรมการระบุตำแหน่งของรถ AGV ภายในร่มให้มีความถูกต้องโดยขึ้นอยู่กับวิธีของ Extended Kalman Filter

1.2.3 เพื่อพัฒนาโปรแกรมที่ผสมผสานข้อมูลการวัดจากเซนเซอร์กับวิธีของ Extended Kalman Filter ในการระบุตำแหน่งของรถ AGV ภายในร่มได้อย่างมีประสิทธิภาพ

1.3 ข้อตกลงงานวิจัย

1.3.1 ใช้ AGV ต้นแบบสำหรับการวิจัย (BALTY AGV) ซึ่งมีขนาดโดยประมาณ ความกว้าง 0.4 เมตร ความยาว 0.3 เมตร และสูง 0.6 เมตร จำกัดความเร็วในการเคลื่อนที่ทดสอบให้อยู่ในช่วง 0.2 ถึง 0.5 เมตรต่อวินาที

1.3.2 โดยมีขนาดของพื้นที่ในการทดลองโดยประมาณ 4 x 8 เมตร

1.3.3 มีระบบขับเคลื่อนแบบสองล้ออิสระ (Differential Drive)

1.3.4 การระบุตำแหน่งอาศัยการหลอมรวมข้อมูลจากชุดเซนเซอร์ต้นทุนต่ำอย่างน้อย 3 ชนิด ซึ่งประกอบด้วย Encoder (สำหรับ Dead Reckoning) หน่วยวัดแรงเฉื่อย (IMU) และ กล้องวงจรปิด (CCTV)

1.3.5 การพัฒนากระบวนการหลอมรวมข้อมูลมีพื้นฐานอยู่บน อัลกอริทึม Extended Kalman Filter (EKF) เป็นหลัก

1.3.6 มุ่งเน้นการจัดการข้อมูลจากเซนเซอร์ที่มีคุณลักษณะแตกต่างกัน ทั้งในด้านความไม่แน่นอน (Uncertainty) และอัตราการสุ่มตัวอย่างข้อมูล (Sampling Rate)

1.3.7 การพัฒนาอัลกอริทึม การจำลองสถานการณ์ และการวิเคราะห์ข้อมูล ใช้โปรแกรม Python เป็นเครื่องมือหลัก แบบออฟไลน์

1.4 ระเบียบวิธีวิจัย

1.4.1 การศึกษาทบทวนวรรณกรรม

ศึกษาและรวบรวมวารสารวิชาการและเอกสารที่เกี่ยวข้อง เพื่อทำความเข้าใจทฤษฎีพื้นฐานของรถนำทางอัตโนมัติ (AGV) เทคนิคการระบุตำแหน่ง คุณลักษณะของเซนเซอร์ชนิดต่าง ๆ และอัลกอริทึม Extended Kalman Filter (EKF)

1.4.2 การศึกษาคุณลักษณะของเซนเซอร์

ทำการศึกษาคุณลักษณะเฉพาะของเซนเซอร์ต้นทุนต่ำแต่ละชนิดที่ใช้ในงานวิจัย ได้แก่ Encoder, หน่วยวัดแรงเฉื่อย (IMU), อัลตร้าไวด์แบนด์ (UWB), และกล้องวงจรปิด (CCTV) เพื่อทำความเข้าใจถึงพฤติกรรม

1.4.3 การออกแบบการทดลอง

กำหนดขั้นตอนและกำหนดพื้นที่ในการทดลองเพื่อเก็บข้อมูลเซนเซอร์ทั้งหมด โดยแบ่งเป็น 2 สภาวะ คือ สภาวะหยุดนิ่ง (Static) และสภาวะเคลื่อนที่ (Dynamic)

1.4.4 การทดลองและเก็บข้อมูลในสภาวะหยุดนิ่ง (Static Test)

ดำเนินการทดลองโดยให้ AGV หยุดนิ่ง ณ ตำแหน่งที่ทราบค่าแน่นอน เพื่อรวบรวมข้อมูลสำหรับนำไปวิเคราะห์

1.4.5 การทดลองและเก็บข้อมูลในสภาวะเคลื่อนที่ (Dynamic Test)

ดำเนินการทดลองโดยให้ AGV เคลื่อนที่ไปตามเส้นทาง เพื่อรวบรวมข้อมูลจากเซนเซอร์ทั้งหมดในการใช้งานจริง

1.4.6 พัฒนาโปรแกรมสำหรับปรับปรุงคุณภาพของข้อมูลดิบจากเซนเซอร์

ประกอบด้วย การแก้ไขความเบี่ยงเบน (Bias Correction) และ การถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting)

1.4.7 ทำการระบุตำแหน่งด้วย Standard EKF

โดยใช้โปรแกรมการหลอมรวมข้อมูลจากเซนเซอร์ตามหลักการของอัลกอริทึม Extended Kalman Filter (EKF) ในรูปแบบมาตรฐาน

1.4.8 การพัฒนาการระบุตำแหน่งด้วย Blended EKF

ออกแบบและพัฒนาการหลอมรวมข้อมูลโดยใช้วิธี Blended EKF สำหรับวิจัยนี้เพื่อเปรียบเทียบประสิทธิภาพกับรูปแบบมาตรฐาน

1.4.9 การวิเคราะห์และสรุปผล

นำผลลัพธ์ที่ได้จากการระบุตำแหน่งด้วยอัลกอริทึมทั้งสองรูปแบบมาวิเคราะห์เปรียบเทียบ และอภิปรายสรุปผลการวิจัย

1.4.10 จัดทำเอกสารและรายงานการวิจัย

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ลดต้นทุนของระบบระบุตำแหน่ง

มุ่งพัฒนาความแม่นยำ ภายใต้ต้นทุนต่ำ ในภาคอุตสาหกรรมที่ต้องการระบบระบุตำแหน่งราคาประหยัดและสามารถปรับใช้ได้

1.5.2 ความปลอดภัยและประสิทธิภาพ

ในการปฏิบัติงาน AGV ที่หาตำแหน่งที่ถูกต้องจะช่วยลดการชนหรืออุบัติเหตุในกระบวนการทำงาน อีกทั้งยังช่วยให้การทำงานโดยรวมมีประสิทธิผล

1.5.3 สร้างความรู้และทักษะด้านเทคโนโลยี

การประยุกต์ใช้ Extended Kalman Filter (EKF) และการผสานข้อมูลหลายชนิด (Sensor Fusion) ซึ่งเป็นหัวใจของเทคโนโลยี ทำให้เพิ่มความเข้าใจในด้านการวิเคราะห์ผลข้อมูล การกรองสัญญาณ และใช้ในระบบหุ่นยนต์อัจฉริยะ

1.5.4 เป็นแนวทางในอนาคต

ผลลัพธ์ใช้เป็นข้อมูลการพัฒนาเทคนิคการหาตำแหน่งที่ ที่สามารถปรับใช้กับยานยนต์อัตโนมัติได้

บทที่ 2
ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะนำเสนอการทบทวนเอกสาร งานวิจัย และแนวคิดทางทฤษฎีที่เกี่ยวข้องกับการพัฒนาระบบระบุตำแหน่งสำหรับรถขนส่งอัตโนมัติ (Automated Guided Vehicle: AGV) ซึ่งเป็นพื้นฐานสำคัญของการดำเนินงานวิจัยฉบับนี้ โดยเนื้อหาจะเริ่มจากการอธิบายภาพรวมของระบบ AGV โครงสร้างและองค์ประกอบหลักที่ใช้ในการควบคุมการเคลื่อนที่ จากนั้นจะกล่าวถึงเทคนิคและเซนเซอร์ที่เกี่ยวข้องกับการหาตำแหน่ง รวมถึงการวิเคราะห์ข้อมูลจากแหล่งข้อมูลหลากหลายประเภท นอกจากนี้ยังครอบคลุมทฤษฎีที่เกี่ยวข้องกับการหลอมรวมข้อมูล (Data Fusion) และหลักการของตัวกรองคัลมานแบบขยาย (Extended Kalman Filter: EKF) ซึ่งเป็นกลไกหลักที่พัฒนาระบบหาตำแหน่งในงานวิจัยนี้

2.1 รถขนส่งอัตโนมัติ (Automated Guided Vehicle: AGV) และสถาปัตยกรรมระบบ

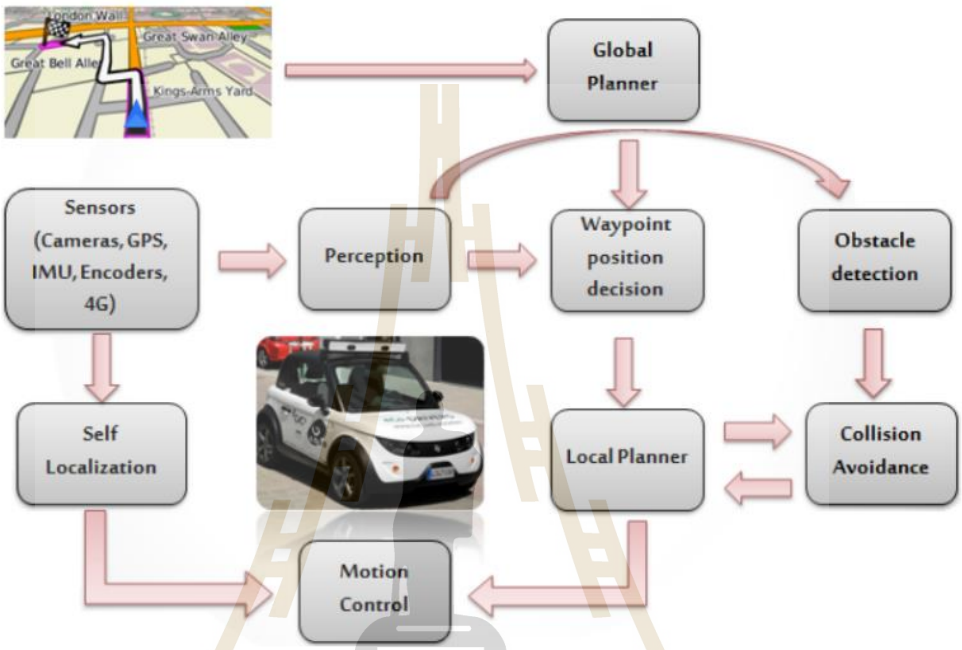
การระบุตำแหน่งของรถ AGV ถือเป็นองค์ประกอบสำคัญของระบบอัตโนมัติในอุตสาหกรรม เนื่องจากส่งผลต่อประสิทธิภาพและความปลอดภัยของการทำงานของ AGV ซึ่งมีหลายประเภท เช่น การย้ายชิ้นงาน (Unit Load Vehicles) การยกสินค้า (Pallet Trucks) และการลากจูง (Towing) ซึ่งต้องมีการความแม่นยำในการนำทาง สำหรับภายในอาคารซึ่งไม่สามารถรับสัญญาณ GPS ได้ การระบุตำแหน่งจึงต้องอาศัยเซนเซอร์ชนิดอื่น ๆ โดยการใช้งาน AGV เพิ่มขึ้นในภาคอุตสาหกรรม แต่ยังคงมีราคาที่สูงสำหรับภาคอุตสาหกรรมขนาดเล็ก ซึ่งเป็นช่องว่างสำคัญในการพัฒนาที่ทำให้ลดต้นทุนจากราคาเซนเซอร์ลง โดยทั่วไป สถาปัตยกรรมของระบบ AGV (ดังรูปที่ 2.1) ประกอบด้วยระบบย่อยที่ทำงานประสานกัน ได้แก่

1) วางแผนเส้นทาง (Global Planner): ทำหน้าที่คำนวณเส้นทางภาพรวมจากจุดเริ่มต้นไปถึงจุดเป้าหมาย

2) การรับรู้และหลีกเลี่ยงสิ่งกีดขวาง (Perception & Obstacle Avoidance): ใช้ข้อมูลจากเซนเซอร์ในการรับรู้ ตรวจจับและหลีกเลี่ยงวัตถุ

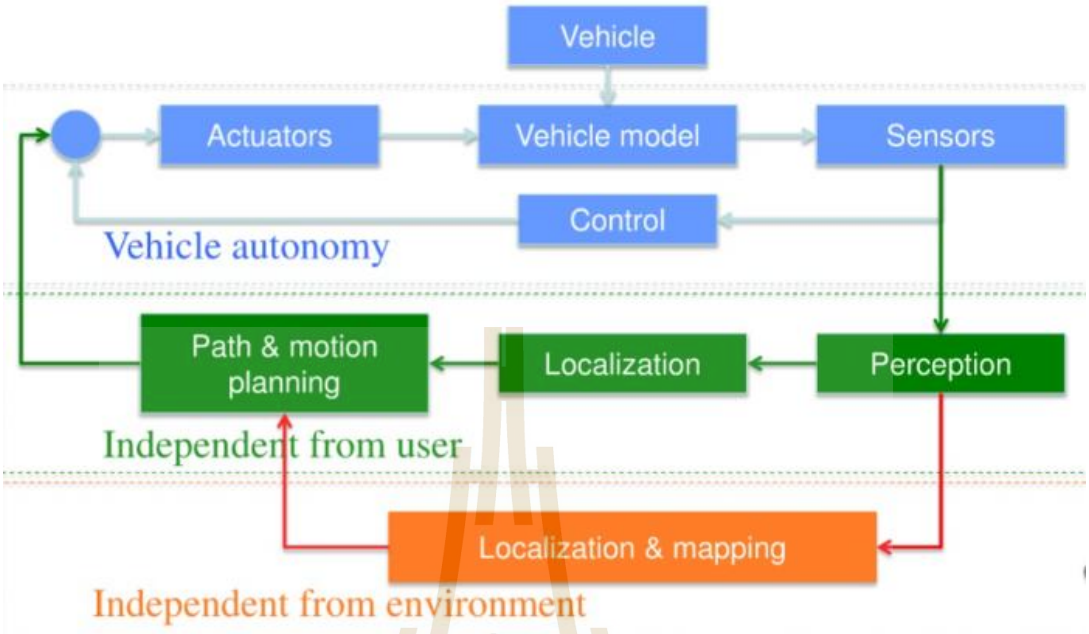
3) ควบคุมตัวรถ (Motion Control): ทำหน้าที่ควบคุมความเร็ว มุมเลี้ยว และการขับเคลื่อนของล้อรถ AGV เคลื่อนที่ไปตามเส้นทางที่กำหนด

4) การระบุตำแหน่งของตัวรถ (Self-Localization): ถือเป็นระบบที่สำคัญที่สุด โดยทำหน้าที่คำนวณหาตำแหน่งและมุมทิศทางของตัวรถของข้อมูลที่ได้จากเซนเซอร์ เพื่อสร้างตำแหน่งที่มีความถูกต้อง



รูปที่ 2.1 สถาปัตยกรรมระบบของรถอัตโนมัติ (Alcalá et al., 2016)

การทำงานของระบบมีความสัมพันธ์ดังรูป 2.2 ซึ่งแสดงให้เห็นถึงความเป็นอิสระของ AGV ใน 3 ระดับ คือ ความเป็นอิสระของตัวรถ (Vehicle Autonomy) ความเป็นอิสระจากผู้ใช้ (Independent from User) และความเป็นอิสระจากสภาพแวดล้อม (Independent from Environment) ซึ่งทั้งหมดนี้ต้องอาศัยข้อมูลตำแหน่งที่แม่นยำเป็นพื้นฐาน



รูปที่ 2.2 การทำงานของ AGV (cgeorge, 2025)

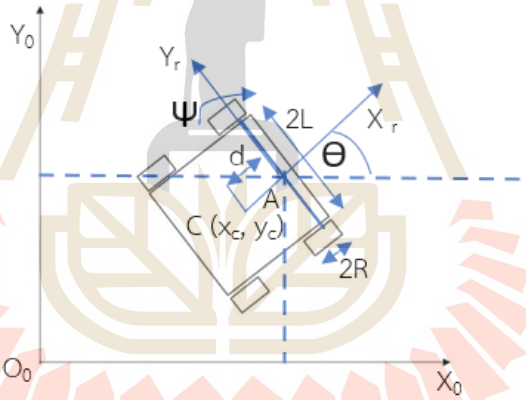
นอกจากสถาปัตยกรรมที่ติดตั้งเซนเซอร์อยู่บนตัวรถแล้ว ปัจจุบันมีแนวโน้มการพัฒนาระบบควบคุมแบบใช้กล้องส่วนกลาง (centralized camera-based control) ที่ติดตั้งกล้องจากมุมมองสูง เช่น เพดานหรือโครงสร้างเหนือพื้นที่ทำงาน เพื่อให้ได้มุมมองภาพรวมของพื้นที่ (God's-eye view) ซึ่งครอบคลุมตำแหน่งของ AGV หลายคันพร้อมกัน วิธีนี้ช่วยลดภาระการติดตั้งเซนเซอร์ราคาแพงบนตัวรถ และย้ายภาระการประมวลผลหลักไปยังเซิร์ฟเวอร์ส่วนกลางที่เชื่อมต่อกับเครือข่ายกล้องวงจรปิด (CCTV network) อย่างไรก็ตาม โครงสร้างแบบใช้กล้องส่วนกลางเพียงอย่างเดียวยังเผชิญความท้าทายจากการเปลี่ยนแปลงสภาพแสง การบดบัง (occlusion) และคุณภาพภาพที่ผันผวน จึงจำเป็นต้องออกแบบวิธีการหลอมรวมข้อมูลร่วมกับเซนเซอร์อื่นเพื่อรักษาความแม่นยำและความเสถียรของการระบุตำแหน่งในสภาพแวดล้อมจริง

2.2 เทคนิคและเซนเซอร์สำหรับการระบุตำแหน่งในอาคาร

การระบุตำแหน่งภายในอาคารเป็นประเด็นสำคัญในการพัฒนาระบบนำทางของยานยนต์อัตโนมัติ เนื่องจากสัญญาณจากระบบดาวเทียมกำหนดตำแหน่งบนพื้นโลก (Global Positioning System: GPS) ไม่สามารถส่งผ่านสิ่งกีดขวาง เช่น ผนังหรือโครงสร้างอาคาร จึงไม่เหมาะสมกับภายในอาคาร จึงได้มีการพัฒนาอุปกรณ์หลากหลายแบบที่สามารถใช้แทนการระบุตำแหน่งด้วย GPS ได้ ซึ่งรวมถึงเทคโนโลยีระบบระบุตำแหน่งภายในอาคาร (Indoor Positioning Systems: IPS) และเซนเซอร์ประเภทต่าง ๆ ที่วัดระยะหรือคำนวณหาตำแหน่งได้ ดังนี้

2.2.1 การระบุตำแหน่งจากเอนโคดเดอร์ (Encoder-based Localization / Dead Reckoning)

การระบุตำแหน่งโดยอาศัยข้อมูลจากเอนโคดเดอร์ (Encoder) หรือที่เรียกโดยทั่วไปว่า Dead Reckoning เป็นวิธีการใช้กันในรถ AGV เพื่อประมาณหาค่าตำแหน่งและค่ามุมทิศทางของรถในขณะเคลื่อนที่ โดยอาศัยการวัดสัญญาณพัลส์ (pulses) จากล้อที่ได้รับจากเซนเซอร์เอนโคดเดอร์ ซึ่งจะแปลงข้อมูลความเร็วเชิงมุมของล้อเป็นระยะทางและมุมทิศทาง หลักการของ Dead Reckoning อ้างอิงจากแบบจำลองเชิงจลน์ศาสตร์ (Kinematic Model) ซึ่งใช้คำนวณหาความเร็วเชิงเส้น ความเร็วเชิงมุม และตำแหน่งของรถในระบบพิกัดอ้างอิง โดยทั่วไป AGV ส่วนใหญ่จะใช้แบบ Differential Drive Mobile Robot (DDMR) ซึ่งใช้ล้อสองข้างที่หมุนอิสระจากกันเพื่อกำหนดมุมทิศทางการเคลื่อนที่ ดังรูป 2.3 การคำนวณหาตำแหน่งเริ่มต้นจากการกำหนดระบบพิกัดขึ้นมาสองระบบ คือ ระบบพิกัดอ้างอิง (Global Frame, (X_0, Y_0)) และ ระบบพิกัดของตัวรถ (Robot Frame, (X_r, Y_r)) โดยมีจุดกำเนิดอยู่ที่จุดกึ่งกลางของแกนล้อ (จุด A)



รูปที่ 2.3 แบบจำลอง AGV ขับเคลื่อนแบบล้อสองอิสระ (ดัดแปลงจาก (Bouzoualegh et al., 2018))

โดยกำหนดตำแหน่งและมุมทิศทางของรถเคลื่อนที่คือ

$$q^g = [x_a \; y_a \; \Theta]^T \tag{2.1}$$

ขั้นตอนในการแปลงระหว่างเฟรมทั้งสองนี้ ตำแหน่งของจุดใดๆ บนตัวรถสามารถกำหนดได้ในเฟรมของรถดังต่อไปนี้

$$q^g = [x^r \; y^r \; \Theta^r \;]^T \tag{2.2}$$

เป็นพิกัดของจุดที่กำหนดในกรอบของตัวรถ จากนั้นพิกัดทั้งสองจะสัมพันธ์กันด้วยการแปลงต่อไปนี้

$$X^I = R(\Theta)X^r \tag{2.3}$$

โดยที่ R(θ) คือ เมทริกซ์การหมุน

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

สมการนี้ใช้ในการแปลงพิกัดของความเร็วจากกรอบตัวรถไปยังพิกัดอ้างอิงได้เป็น

$$\dot{X}^I = R(\Theta)\dot{X}^r \tag{2.5}$$

โดยที่ $\dot{X}^I$ คือ เวกเตอร์ความเร็วในกรอบพิกัดอ้างอิง และ $\dot{X}^r$ คือ เวกเตอร์ความเร็วในกรอบพิกัดของตัวรถ จากความสัมพันธ์นี้ สามารถสร้างแบบจำลองจลนศาสตร์ของ AGV ได้โดยสมมติฐานที่สำคัญคือ ล้อหมุนโดยไม่มีการลื่นไถล (No-slipping condition) ซึ่งทำให้สามารถคำนวณความเร็วเชิงเส้น (v) และความเร็วเชิงมุม (ω) ของตัวรถได้จากความเร็วล้อซ้ายและขวา (v_R, v_L)
ความเร็วเชิงเส้นของตัวรถ v คำนวณจากค่าเฉลี่ยของความเร็วล้อทั้งสองข้างดังสมการ

$$v = \frac{v_R + v_L}{2} = R\frac{(\dot{\varphi}_R + \dot{\varphi}_L)}{2} \tag{2.6}$$

และความเร็วเชิงมุม ω ของระบบแบบ DDMR ดังนี้

$$\omega = \frac{v_R - v_L}{2L} = R\frac{(\dot{\varphi}_R - \dot{\varphi}_R)}{2L} \tag{2.7}$$

เมื่อ L คือ ระยะห่างระหว่างศูนย์กลางล้อถึงล้อ
$\dot{\varphi}_R, \dot{\varphi}_R$ คือ อัตราการหมุนของล้อขวาและซ้าย (rad/s)

ในการหา x y และ v_x v_y เขียนได้เป็นฟังก์ชันของความเร็วของจุดศูนย์กลางของตัวรถ A

$$
\begin{aligned}
x_{pR} &= x_a + vdt\cos\theta \\
y_{pR} &= y_a + vdt\sin\theta \\
\dot{x}_{pR} &= \dot{x}_a + v\cos\theta \\
\dot{y}_{pR} &= \dot{y}_a + v\sin\theta \\
\theta &= \theta_0 + \omega dt
\end{aligned}
\tag{2.8}
$$

2.2.2 การระบุตำแหน่งจากหน่วยวัดแรงเฉื่อย (IMU-based Localization)

หน่วยวัดแรงเฉื่อย (Inertial Measurement Unit: IMU) เป็นตัวตรวจวัดการเคลื่อนที่ที่สำคัญในระบบตัวรถ AGV เนื่องจากสามารถใช้ประเมินค่าการเร่ง ความเร็วเชิงมุม และมุมทิศทางโดยไม่ต้องอาศัยสัญญาณภายนอก ซึ่งประกอบด้วยเซนเซอร์ 2 ชนิดหลักคือ วัดความเร่งสำหรับตรวจวัดความเร่งเชิงเส้นในแกน x, y, z และ Gyroscope สำหรับตรวจวัดอัตราการหมุนเชิงมุมในแต่ละแกน ข้อมูลจาก IMU สามารถคำนวณหา x, y และมุมทิศทางของรถได้

หลักการคำนวณเชิงพลศาสตร์ (Dynamic Model)

การคำนวณหาตำแหน่งด้วย IMU อาศัยแบบจำลองเชิงพลศาสตร์ของการเคลื่อนที่ ซึ่งพิจารณาความสัมพันธ์ระหว่างความเร่งเชิงเส้น (a) และความเร็ว (v) โดยสมมติว่าความเร่งที่วัดได้จาก Accelerometer มีองค์ประกอบของการทำงานตามแกน (x, y) ของตัวรถ ซึ่งสามารถหาความเร็วและตำแหน่งได้จากการอินทิเกรต (Integration) ตามเวลาในรูปแบบต่อเนื่อง และสามารถประมาณค่าตำแหน่ง (x) โดยอาศัยสมการของนิวตัน (Becker, 2023) ดังแสดงในสมการ

$$
x = x_0 + v_0\Delta t + \frac{1}{2}a\Delta t^2 \tag{2.9}
$$

เมื่อ x_0 คือ ตำแหน่งเริ่มต้น (m), v_0 คือ ความเร็วเริ่มต้น(m/s) และ Δt คือ ช่วงเวลา(s)
เขียนสมการของ IMU โดยสถานะของระบบ (x y z) ประกอบไปด้วยตำแหน่ง ความเร็ว และความเร่งในแกนต่าง ๆ ($[x\ y\ z]^T$=[x, y, z, v_{x0}, v_{y0}, v_{z0}, a_x, a_y, a_z]^T) ดังแสดงในสมการ

$$
\begin{aligned}
x &= x_0 + v_{x0}\Delta t + \frac{1}{2}a_x\Delta t^2 \\
y &= y_0 + v_{y0}\Delta t + \frac{1}{2}a_x\Delta t^2 \\
z &= z_0 + v_{z0}\Delta t + \frac{1}{2}a_z\Delta t^2
\end{aligned}
\tag{2.10}
$$

โดย IMU คือ สามารถคำนวณมุมทิศทาง (Heading) ของ AGV ได้โดยตรงจากข้อมูลของไจโรสโคป ซึ่งสำคัญต่อการนำทาง

2.2.3 การระบุตำแหน่งจากอัลตร้าไวด์แบนด์ (UWB-based Localization)

เทคโนโลยี Ultra-Wideband (UWB) เป็นระบบสื่อสารไร้สายที่ใช้คลื่นวิทยุความถี่สูงและมีแบนด์วิดท์กว้างมาก (Che et al., 2023; Dong et al., 2019) คุณสมบัติเด่นของ UWB คือ การส่งสัญญาณพลังงานต่ำในช่วงเวลาสั้น ๆ ทำให้สัญญาณมีการรบกวนน้อยและใช้งานได้ในระบบภายในอาคาร สำหรับระบบหาตำแหน่งของ AGV โดยใช้ UWB ในการคำนวณระยะระหว่าง Tag (ติดตั้งบนยานพาหนะ) และ Anchors (จุดอ้างอิงคงที่ในพื้นที่) โดยทั่วไปมีสองวิธีในการหาระยะ ได้แก่

1) ความเข้มของสัญญาณ (Received Signal Strength Indicator: RSSI) สำหรับการติดตามตำแหน่งโดยใช้ UWB (Ultra-Wideband) จากค่า RSSI และระยะ (d) ที่ได้จากเซนเซอร์ UWB สามตัว(Che et al., 2023; Dong et al., 2019) โดยการคำนวณระยะทางจากความเข้มของสัญญาณ สามารถใช้ในการคำนวณระยะทาง d จากความเข้มของสัญญาณ (RSSI) จากสมการ

$$d = 10^{\frac{(A-RSSI)}{10n}} \tag{2.11}$$

A คือ ความเข้มของสัญญาณที่วัดได้ที่ระยะทาง 1 เมตร (dBm)

RSSI คือ ความเข้มของสัญญาณ (dBm)

n คือ ค่าความสูญเสียของสัญญาณ (path loss exponent)

2) เวลาการเดินทางของสัญญาณ (Time of Flight: ToF) สำหรับการติดตามตำแหน่งโดยใช้ UWB (Ultra-Wideband) จาก Time of Flight (ToF) เพื่อคำนวณตำแหน่งโดยการวัดเวลาของสัญญาณจาก Tag ไปยัง Anchors โดยคำนวณระยะทางจากสมการ d = c x t ซึ่ง d คือ ระยะทาง (เมตร) c คือความเร็วของสัญญาณ (เมตร/วินาที) t คือเวลาเดินทางของสัญญาณ (วินาที)

เมื่อได้ระยะทางจาก Anchors แล้วทั้ง 2 วิธีแล้ว จะแบ่งเป็น 2 วิธีการคำนวณ วิธีแรกเมื่อได้ระยะทาง d_1 d_2 และ d_3 จากเซนเซอร์ UWB สามตัว ตำแหน่ง Anchor 1 อยู่ที่จุดติดตั้ง (x_1, y_1) Anchor 2 อยู่ที่จุดติดตั้ง (x_2, y_2) และ Anchor 3 อยู่ที่จุดติดตั้ง (x_3, y_3) ของเซนเซอร์ UWB สามารถใช้วิธี trilateration ในการคำนวณตำแหน่ง AGV (Mobile Node) สามารถเขียนสมการได้

$$(x - x_1)^2 + (y - y_1)^2 = d_1^2 \tag{2.12}$$

$$(x - x_2)^2 + (y - y_2)^2 = d_2^2 \quad (2.13)$$

$$(x - x_3)^2 + (y - y_3)^2 = d_3^2 \quad (2.14)$$

จากสมการ (2.12), (2.13), และ (2.14) สามารถแก้สมการทั้งสามนี้พร้อมกันจะให้ค่าพิกัดของตำแหน่งแท็กของ x และ y ของรถ AGV ได้

วิธีที่ 2 ใช้สมการตรีโกณมิติ เพื่อหาระยะทาง d_1 และ d_2 จากเซนเซอร์ UWB ทั้งสองตัว (Anchor 1 และ Anchor 2) จากตำแหน่ง (x_1, y_1) และ (x_2, y_2) ของเซนเซอร์ UWB ตามลำดับ และระยะห่างระหว่าง Anchor 1 และ Anchor 2 (d_0) โดยใช้สมการตรีโกณมิติ

2.2.4 การระบุตำแหน่งจากกล้องวงจรปิด (CCTV-based Localization)

หนึ่งในแนวทางสำคัญคือการใช้กล้องวงจรปิด (Closed-Circuit Television: CCTV) เพื่อหา x, y ของตัวรถภายในอาคาร ซึ่งเป็นเทคนิคที่เหมาะกับโรงงานหรือพื้นที่ที่มีติดกล้องวงจรปิดอยู่แล้ว เช่น พื้นที่ติดตั้งกล้องมุมสูง (Overhead Camera) ที่สามารถมองเห็นพื้นที่ทำงานของตัวรถได้ ระบบนี้จะใช้ภาพจากกล้องในการตรวจจับและหา x, y ของตัวรถแบบออนไลน์ โดยอาศัยหลักการของ Absolute Positioning ซึ่งได้ x, y เชิงพิกัดจริงของพื้นที่ ข้อมูลที่ได้จะถูกนำมาใช้เป็นข้อในการคำนวณหา x, y ร่วมกับเซนเซอร์อื่น เช่น Encoder และ IMU เพื่อเพิ่มความแม่นในการประมาณหา x, y ของตัวรถในบางสถาปัตยกรรม กล้องจะไม่ได้ติดตั้งบนตัวรถเพียงตัวเดียว แต่ถูกติดตั้งเป็นเครือข่ายกล้องส่วนกลางที่ครอบคลุมพื้นที่ทำงานทั้งหมดของโรงงาน จากมุมมองในลักษณะ “God’s-eye view” ข้อมูลจากกล้องหลายตัวจะถูกส่งไปยังเซิร์ฟเวอร์กลางเพื่อประมวลผลร่วมกัน ทำให้สามารถติดตามตำแหน่งของ AGV หลายคันได้พร้อมกันในเชิงพื้นที่เดียวกัน แนวทางนี้ช่วยลดต้นทุนเซนเซอร์บนตัวรถ แต่ทำให้ข้อมูลภาพมีความอ่อนไหวต่อสภาพแสง การบดบัง และความถี่ในการอัพเดตตำแหน่ง ดังนั้นจึงเหมาะอย่างยิ่งที่จะนำผลการวัดจากกล้องส่วนกลางนี้มาหลอมรวมกับข้อมูลจาก Encoder และ IMU เพื่อเพิ่มความน่าเชื่อถือของการประมาณตำแหน่งในระบบจริง

หลักการพื้นฐาน: การแปลงพิกัดระหว่างภาพและพิกัดจริง

ขั้นตอนหลักของการใช้กล้องเพื่อระบุตำแหน่งคือการแปลงพิกัดของวัตถุในภาพให้อยู่ในระบบพิกัดจริง ซึ่งต้องผ่านกระบวนการเพื่อปรับแก้ไขค่าความคลาดเคลื่อนของเลนส์และการสอบเทียบกล้อง (Camera Calibration) เพื่อให้คำนวณ x, y

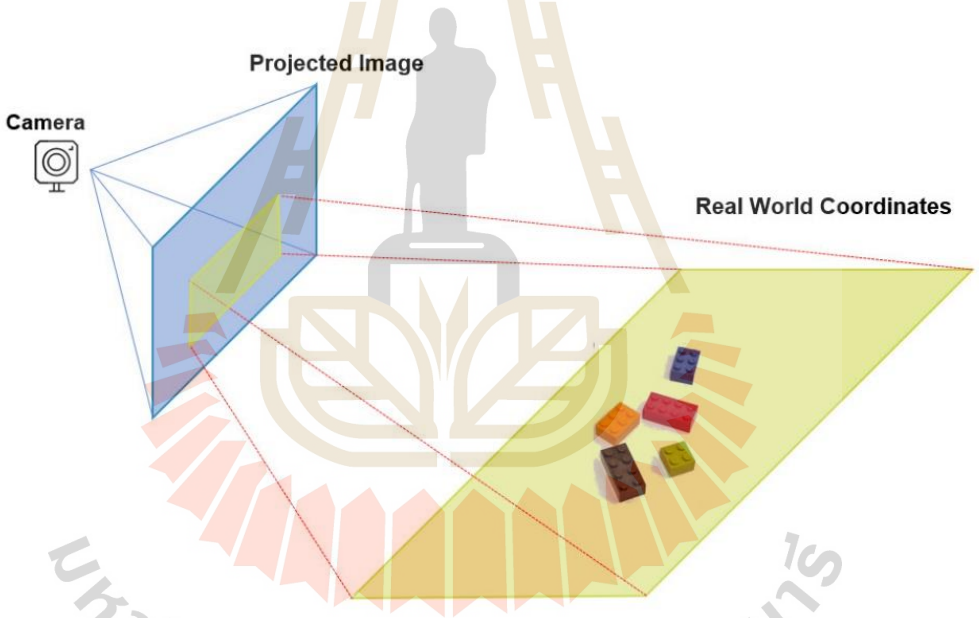
การแก้ไขความบิดเบือนของเลนส์ (Lens Distortion Correction)

ภาพมักมีความบิดเบือนจากลักษณะของเลนส์ โดยเฉพาะกล้องมุมกว้าง ซึ่งอาจทำให้การคำนวณหา x, y เชิงพิกัดมีความคลาดเคลื่อน เพื่อแก้ไขปัญหานี้จะใช้วิธีการความบิดเบือนของเลนส์ในลักษณะรัศมี (Radial Distortion) และเชิงสัมผัส (Tangential Distortion) (ดังรูป 2.4) การสอบ

เทียบกล้องจะทำโดยใช้ลวดลายกระดานหมากรุก (Checkerboard Pattern) เพื่อหาค่าภายใน (Intrinsic Parameters) และค่าภายนอก (Extrinsic Parameters) ของกล้อง ซึ่งทำให้เชื่อมโยงพิกัดระหว่างภาพกับพื้นที่จริงได้อย่างแม่นยำ การสอบเทียบกล้อง (Camera Calibration) เป็นขั้นตอนที่ช่วยให้ระบบหา x, y ด้วยภาพสามารถแปลงพิกัดจากระนาบภาพไปยังพิกัดจริงได้อย่างถูกต้อง การสอบเทียบจะใช้ข้อมูลค่าของกล้อง 2 ประเภท ได้แก่

ค่าภายใน (Intrinsic Parameters) คือคุณลักษณะเฉพาะของตัวกล้องและเลนส์ ซึ่งประกอบด้วย

1) ระยะโฟกัส (f_x , f_y) แทนระยะโฟกัสของเลนส์ในแกน x และ y
2) จุดศูนย์กลางของภาพ (c_x, c_y) จุดตัดของแนวแกนภาพกับระนาบรับภาพ ซึ่งมักไม่ตรงกับกึ่งกลางของภาพจริง
3) ค่าสัมประสิทธิ์ความบิดเบือน (Distortion Coefficients) ค่าที่ใช้อธิบายและแก้ไขความบิดเบือนเชิงรัศมีและเชิงสัมผัส (Radial & Tangential Distortion)



รูปที่ 2.4 การบิดเบือนของภาพจากเลนส์ (Garcia, 2023)

ค่าภายนอก (Extrinsic Parameters) คือ ข้อมูลที่อธิบายตำแหน่งและมุมทิศทางการวางของกรอบพิกัดของกล้องกับกรอบพิกัดจริง ประกอบด้วย

1) เมทริกซ์การหมุน (Rotation Matrix, R) อธิบายมุมทิศทางการหันการหมุนของกล้อง
2) การเลื่อนตำแหน่ง (Translation Vector, t) อธิบายระยะห่าง x, y ของกล้องจากระบบพิกัดอ้างอิง

ในทางปฏิบัติ กระบวนการสอบเทียบกล้องจะทำโดยการถ่ายภาพวัตถุที่มีขนาดและรูปแบบที่ทราบล่วงหน้า เช่น ลวดลายกระดานหมากรุก (**Checkerboard Pattern**) ในหลายมุมมอง เพื่อให้ระบบสามารถคำนวณค่าของกล้องทั้งภายในและภายนอกได้อย่างแม่นยำ โดยใช้ไลบรารี OpenCV ถ้ากำหนดจุดจริง p (p_x, p_y, p_z) และมองจากจุดอ้างอิงของกล้อง $p_{Cam} = T_w^{Cam} p$ โดย T_w^{Cam} คือ เมทริกซ์การแปลงจากพิกัดจริงไปยังระบบแกนพิกัดของกล้อง p_{Cam} หมายถึงจุดที่สังเกตได้ในระบบพิกัดของกล้อง แกน Z คือแกนที่ออกจากจุดกลางเลนส์ในแนวสายตาของกล้อง แกน X จะเป็นแนวนอนในระนาบของกล้อง และแกน Y จะเป็นแนวตั้งในระนาบของกล้องสามารถเขียนได้ดังสมการ (Kala, 2023)

$$p_{Cam} = T_w^{Cam} p = [R_{3x3} t_{3x1}] p_{4x1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \tag{2.15}$$

กล้องจะฉายจุดทั้งหมดจากพิกัดจริงไปบนเลนส์ออปติก ซึ่งจะอยู่ในลักษณะของภาพฉายผ่านเมทริกซ์การปรับเทียบกล้อง C โดยคำนวณของสามเหลี่ยมคล้าย สามารถคำนวณการฉายลงบนภาพ 2 มิติได้ โดยให้ $p_I(u_I', v_I', w)$คือจุดในภาพที่สร้างขึ้น โดยใช้สมการ (Kala, 2023)

$$p_I = C p_{Cam} \tag{2.16}$$

$$\begin{bmatrix} u_I' \\ v_I' \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} p_{Cam} \tag{2.17}$$

โดยที่ f_x และ f_y คือ ระยะโฟกัสในแกน 2 แกน และ (c_x, c_y) คือพิกัดของจุดศูนย์กลางของภาพ รวมสมการ 2.15 และ 2.17 ได้สมการ

$$\begin{bmatrix} u_I' \\ v_I' \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} T_w^{Cam} p = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \tag{2.18}$$

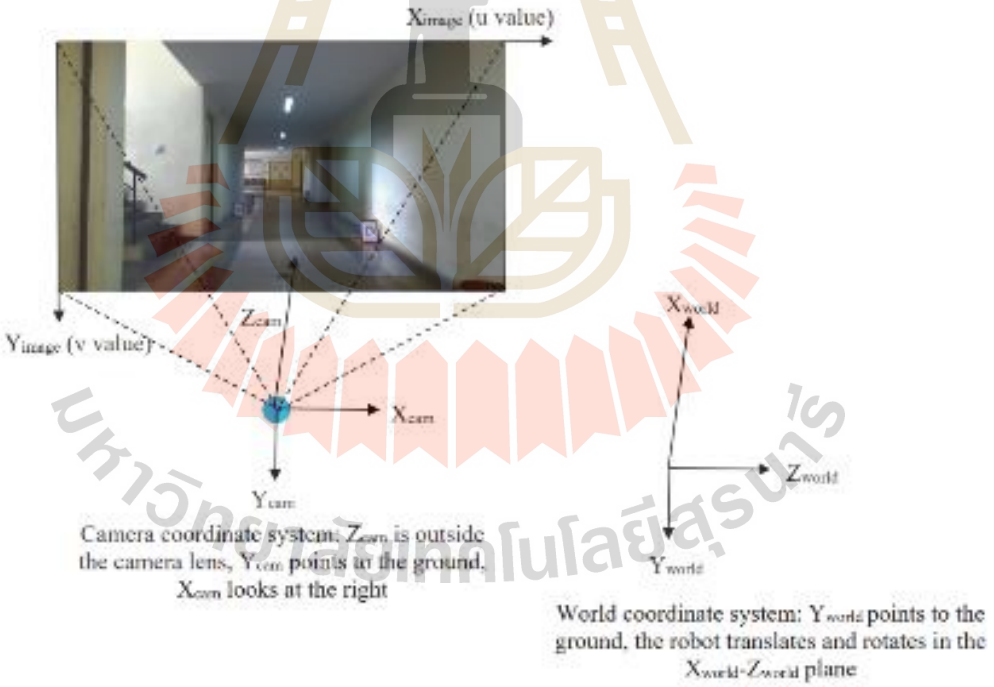
โดยที่ w แสดงถึงการปรับมาตราส่วน ในการฉายภาพบนระนาบภาพ ข้อมูลการปรับมาตราส่วนจะสูญหายไปเนื่องจากกล้องมองเห็นเป็น 2 มิติ จาก (u_I', v_I', w) ถูกแปลงเป็น $(u_I, v_I, 1) =$

$(u_I'/w, v_I'/w, 1)$ จะได้จุดพิกัดของกล้องคือ $(p_{xCam}, p_{yCam}, p_{zCam}, 1)$ จะให้พิกัดในภาพตามที่กำหนดโดยสมการ

$$\begin{bmatrix} u_I' \\ v_I' \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{xCam} \\ p_{yCam} \\ p_{zCam} \end{bmatrix} = \begin{bmatrix} f_x p_{xCam} + c_x p_{zCam} \\ f_y p_{yCam} + c_y p_{zCam} \\ p_{zCam} \end{bmatrix} \tag{2.19}$$

$$\begin{bmatrix} u_I \\ v_I \end{bmatrix} = \begin{bmatrix} (f_x p_{xCam} + c_x p_{zCam})/p_{zCam} \\ (f_y p_{yCam} + c_y p_{zCam})/p_{zCam} \end{bmatrix} = \begin{bmatrix} (f_x p_{xCam}/p_{zCam} + c_x) \\ (f_y p_{yCam}/p_{zCam} + c_y) \end{bmatrix} \tag{2.20}$$

พิกัดของภาพในที่นี้มีจุดกำเนิดที่มุมบนซ้ายของภาพ แกน X ของภาพ (ค่า u_I) อยู่ในแนวนอนจากซ้ายไปขวา และแกน Y ของภาพ (ค่า v_I) อยู่ในแนวตั้งจากบนลงล่าง (ดังรูป 2.5) เมื่อทราบตำแหน่งจริงแล้วจะหาค่าสัมประสิทธิ์ในการแปลงเป็นพิกัดในภาพได้ ในทางกลับกันกล้องสามารถระบุตำแหน่งในรูปภาพได้ทำให้สามารถหาพิกัดจริงได้



รูปที่ 2.5 ภาพฉายอธิบายพิกัด (Kala, 2023)

2.3 วิธีการวัด (Measurement Methods)

การวัดถือเป็นกิจกรรมพื้นฐานที่เกี่ยวข้อง ทั้งในด้านชีวิต ด้านวิทยาศาสตร์และวิศวกรรมศาสตร์ โดยประเมินค่าของปริมาณที่ศึกษา ตัวอย่างเช่น การวัดอุณหภูมิของวัตถุ การคำนวณปริมาณของเหลว หรือการประเมินแรง การวัดความแม่นยำและมีเชื่อถือสำคัญต่อการควบคุมกระบวนการ งานวิศวกรรมศาสตร์ การวัดสำคัญต่อคุณภาพของผลทดลอง เนื่องจากส่งผลต่อความแม่นที่ใช้กับระบบ (Figliola & Beasley, 2011) การเลือกใช้เครื่องมือ วิธีการวัด เพื่อให้วัดค่าได้แม่นสม่ำเสมอ ตามวัตถุประสงค์

2.3.1 การวัดทั่วไป

สามารถอธิบายได้ว่าเป็นการแปลงปริมาณทางกายภาพให้กลายเป็นข้อมูลเชิงสัญญาณที่สามารถอ่านหรือวิเคราะห์ได้ โดยทั่วไป ระบบการวัดมักประกอบด้วยส่วนหลัก ได้แก่ เซนเซอร์ที่ทำหน้าที่ตรวจจับค่าทางกายภาพ วงจรประมวลผลสัญญาณสำหรับขยายหรือปรับค่า และส่วนแสดงผลสำหรับนำเสนอค่าที่วัดได้ในรูปแบบที่มนุษย์สามารถรับรู้ได้ เช่น ผ่านหน้าจอ ระบบที่ดีควรมีความถูกต้อง เที่ยงตรง และมีเสถียรภาพสูง เพื่อควบคุมกระบวนการทางอุตสาหกรรม

2.3.2 ขั้นตอนการทำงาน

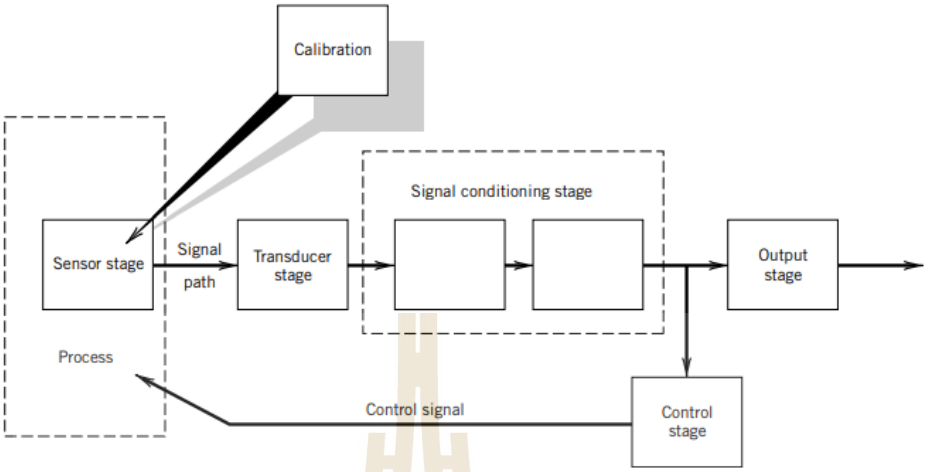
การวัด (ดังรูปที่ 2.6) โดยทั่วไปแบ่ง ดังนี้

1) ขั้นตอนตัวรับรู้และทรานสดิวเซอร์(sensor–transducer stage) ซึ่งจะตรวจจับข้อมูลที่ต้องการวัด เช่น อุณหภูมิ ความดัน หรือความสูง

2) ขั้นตอนปรับแต่งสัญญาณ(signal-conditioning stage) เป็นการปรับขนาดของสัญญาณที่ตรวจจับได้ผ่านการกรองหรือขยาย

3) ขั้นตอนการแสดงผล(output stage) ที่ทำหน้าที่แปลงสัญญาณที่ได้เป็นรูปแบบที่มนุษย์สามารถรับรู้ได้ เช่น หน้าจอแสดงค่า หรือการบันทึกเป็นค่า

4) ขั้นตอนควบคุม(feedback-control stage) เป็นผลจากการวัดเพื่อปรับการทำงานของระบบ เช่น ควบคุมอุณหภูมิในเตาอบ หรือควบคุมในสายการผลิต เพื่อให้ไปตามค่าที่กำหนดไว้



รูปที่ 2.6 ส่วนประกอบของระบบการวัดทั่วไป (Figliola & Beasley, 2011)

2.3.3 การออกแบบการทดลองและการวัด

ต้องคำนึงถึงการควบคุมปัจจัยในระหว่างการทดลอง

1) กำหนดวัตถุประสงค์ของการทดสอบ การเริ่มต้นของการออกแบบ คือ การกำหนดคำถามหรือวัตถุประสงค์ เช่น ในกรณีศึกษาวัดการใช้เชื้อเพลิงของรถยนต์ใหม่ จำเป็นต้องระบุว่าต้องการทราบการใช้น้ำมันเฉลี่ยภายใต้สภาวะการขับขี่ทั่วไปหรือเจาะจง

2) ระบุและควบคุมปัจจัยที่เกี่ยวข้อง มีผลต่อประสิทธิผลในควบคุมให้เหมาะสม ได้แก่

2.1) ตัวแปรอิสระ (Independent Variables) หมายถึงตัวแปรที่สามารถปรับหรือกำหนดค่าได้เอง เช่น เส้นทางการขับขี่ ผู้ขับขี่

2.2) ตัวแปรตาม (Dependent Variables) คือผลที่ขึ้นจากการเปลี่ยนของตัวแปรอิสระ เช่น ระยะทางที่ AGV หรือเวลาการประมวลผลข้อมูล

2.3) ตัวแปรแทรกซ้อน (Extraneous Variables) เป็นตัวแปรที่ควบคุมไม่ได้โดยตรงแต่ส่งผล เช่น ความชื้นในอากาศ หรืออุณหภูมิของพื้นที่ทดสอบ

3) การลดสัญญาณรบกวนและการควบคุมความไม่แน่นอน ในงานวิจัยทางวิศวกรรม การรบกวนของสัญญาณ (Noise) และการแทรกแซง (Interference) มักเกิดขึ้นจากแหล่งภายนอก ดังนั้น จึงจำเป็นต้องลดผลของสัญญาณรบกวน เพื่อสะท้อนผลจริงของตัวแปรที่สนใจ

3.1) สัญญาณรบกวน (Noise) เป็นค่าการสุ่มที่เพิ่มความกระจัดกระจายของข้อมูล

3.2) การรบกวน (Interference) เป็นสิ่งที่ไม่ต้องการที่ถูกแทรกในข้อมูล เช่น การเปลี่ยนความดันบรรยากาศจากจุดเดือดของน้ำ

2.3.4 เทคนิคการลดผลภายนอก

การวัดและความแม่นยำนั้น ต้องลดปัจจัยภายนอก ซึ่งนิยม คือ

1) การสุ่มในการทดสอบ (Randomization)

การสุ่มช่วยลดอคติและผลจากปัจจัยภายนอก จากการทดสอบให้เป็นแบบสุ่ม ช่วยกระจายของผลลัพธ์ ทำให้เป็นกลางมากขึ้น

2) การใช้บล็อกแบบสุ่ม (Randomized Blocks)

สถานการณ์ที่ปัจจัยภายนอกมีแนวโน้มแบบไม่ต่อเนื่อง เช่น อุปกรณ์หรือคนในแต่ละรอบการทดลองไม่เหมือนเดิม โดยจะทำการจัดกลุ่มข้อมูลคล้ายกันให้อยู่ในบล็อกเดียวกัน เพื่อควบคุมปัจจัยภายในบล็อกให้คงที่ และลดปัจจัยภายนอก

2.3.5 การทำซ้ำและการทำสำเนา

การทำซ้ำและการทำสำเนาสามารถทำได้ดังนี้

1) การทำซ้ำ (Repetition) หมายถึงการวัดค่าซ้ำหลายครั้งภายใต้เงื่อนไขเดียวกัน เช่น การทดลองซ้ำในชุดอุปกรณ์เดียวหรือในช่วงเวลาเดียว เพื่อประเมินปัจจัยของข้อมูลที่ได้และตรวจสอบผลในทางปฏิบัติการเดียวกัน

2) การทำสำเนา (Replication) หมายถึงการทำการวัดซ้ำภายใต้สภาวะหรือเงื่อนไขที่ไม่เหมือนกัน เช่น การเปลี่ยนอุปกรณ์หรือคน เพื่อปัจจัยที่คงที่ของระบบและความถูกต้องของกระบวนการวัด

การวัดแบบ Concomitant Methods ซึ่งเป็นการใช้อุปกร์วัดหลายชนิดในการตรวจสอบปัจจัยเดียวกัน วิธีนี้ช่วยตรวจสอบเพิ่มความมั่นใจในผลลัพธ์

2.3.6 การสอบเทียบ (Calibration)

การสอบเทียบคือการนำค่าป้อนที่ทราบค่าแน่นอน (Standard) ใส่เข้าสู่ระบบการวัดเพื่อ สังเกตค่าผลลัพธ์ของระบบ ตามค่ามาตรฐาน (Standard) ที่เป็นที่ยอมรับ เพื่อปรับค่าการตอบสนองให้สอดคล้องกับมาตรฐานอ้างอิง คือ

การสอบเทียบแบบสถิต (Static Calibration)

การสอบเทียบแบบสถิตเป็นการสอบเทียบที่ค่าของปัจจัยที่เกี่ยวข้องยังคงที่ โดยในกระบวนการนี้ ค่าป้อนที่ทราบค่าแน่นอนจะถูกใส่เข้าสู่ระบบ และทำการบันทึกค่าผลลัพธ์ที่วัด ซึ่งสามารถนำค่าป้อนและค่าผลลัพธ์ที่วัดได้มาวาดกราฟเพื่อสร้างกราฟการสอบเทียบแบบตรง จากอุกรณ์มาตรฐาน เพื่อใช้ในการคำนวณฟังก์ชันปรับเทียบ (calibration curve)

สอบเทียบแบบพลวัต (Dynamic Calibration)

สอบเทียบแบบพลวัตจะใช้เมื่อปัจจัยเปลี่ยนตามเวลา การสอบเทียบประเภทนี้มักใช้สัญญาณไซน์หรือสัญญาณสเต็ปเป็นค่าป้อนเพื่อสอบเทียบระบบการวัดแบบพลวัต

2.3.7 ความไวต่อการวัดแบบสถิต (Static Sensitivity)

ค่าความไวต่อการวัดแบบสถิต (Static Sensitivity) หาค่าได้จากความชัน คือ

$$K = \frac{dy}{dx}\Big|_{x=x_1} \tag{2.21}$$

โดย K คือ ความไวต่อการวัดที่ตำแหน่ง x_1

ช่วงการทำงาน (Range) ช่วงการทำงานของระบบการวัดถูกกำหนดโดยขอบเขตของค่าป้อนจาก $x_{\min}$ ถึง $x_{\max}$ ซึ่งสามารถคำนวณช่วงการทำงานได้ดังสมการ

$$r_i = x_{max} - x_{min} \tag{2.22}$$

และสำหรับค่าผลลัพธ์ช่วงการทำงานสามารถคำนวณได้จาก

$$r_o = y_{max} - y_{min} \tag{2.23}$$

ค่าความไวต่อการวัดจะสะท้อนถึงการตรวจจับการเปลี่ยนของค่าป้อนเข้า หากระบบมีความไวสูง จะแยกสัญญาณขาเข้าละเอียดมากยิ่งขึ้น (Yuan et al., 2022)

2.3.8 ความแม่นยำและความคลาดเคลื่อน

ความแม่นยำ (Accuracy) หมายถึงระดับความเหมือนของค่าที่วัดได้กับค่าจริง ค่าที่วัดได้มักมีค่าความคลาดเคลื่อนจากปัจจัยภายนอก

ความคลาดเคลื่อน (Error) คือ

$$\text{e} = \text{ค่าที่วัด} - \text{ค่าความจริง} \tag{2.24}$$

นอกจากนี้ยังนิยมอยู่ในรูปของ ค่าความคลาดเคลื่อนสัมพัทธ์ (Relative Error) เพื่อแสดงสัดส่วนของค่าเทียบกับค่าจริง คือ

$$A_{re} = \left(\frac{|e|}{\text{ค่าอ้างอิง}}\right) x100 \tag{2.25}$$

ความแม่นยำพิจารณาทั้ง ความคลาดเคลื่อนเชิงระบบ (Systematic Error) และ ความคลาดเคลื่อนแบบสุ่ม (Random Error)

1) ความคลาดเคลื่อนเชิงระบบ หมายถึงความผิดพลาดที่เกิดซ้ำได้ภายใต้สภาวะการวัดเดียวกัน เช่น อุณหภูมิแวดล้อมคงที่ไม่เท่ากับค่ามาตรฐาน
2) ความคลาดเคลื่อนแบบสุ่ม หมายถึงความผิดพลาดที่ไม่รู้ได้แน่นอน เช่น การสั่นสะเทือน การเปลี่ยนแรงลม

ทั้งนี้ การลดค่าสามารถใช้งานอุปกรณ์ที่มีความละเอียดเพิ่ม ปรับปรุงวิธีการสอบเทียบ และลดผลกระทบอื่น ๆ

2.3.9 ความไม่แน่นอน (Uncertainty)

ความไม่แน่นอนในการวัดหมายถึงค่าความคลาดเคลื่อนที่ไม่รู้ ว่าเกิดจากสาเหตุใด การประเมินค่าความไม่แน่นอนจึงต้องทำในทางด้านวิศวกรรม เช่น การตั้งค่าการวัด การหาค่าความไม่แน่นอนรวม คือ

$$u_c = \sqrt{u_1^2 + u_2^2 + \cdots + u_M^2} \tag{2.26}$$

โดย u_c คือ ค่าความไม่แน่นอนรวม *M* คือ จำนวนปัจจัย

2.3.10 ฮิสเทรีซิส (Hysteresis)

ฮิสเทอรีซิสหมายถึงผลจากความไม่สามารถกลับมาอยู่ในสถานะเดิม เช่น การเพิ่มค่าปริมาณที่วัดขึ้นกับการลดค่าลง ให้ผลไม่เท่ากันทั้งสองทิศทาง ซึ่งอาจเกิดจากความฝืดของกลไกหรือแรงเฉื่อยภายในระบบ

ค่าความผิดพลาดจากฮิสเทอรีซิสสามารถคำนวณได้จากสมการ

$$u_h = \frac{x_{t1} - x_{t2}}{2} \tag{2.27}$$

ซึ่ง x_{t1} คือ ค่าที่ได้ในขณะเพิ่มค่า และ x_{t2} คือ ค่าที่ได้ในขณะลดค่า ค่าที่เกิดจากฮิสเทอรีซิสรวมเข้ากับค่าความไม่แน่นอนรวม ซึ่งค่าของฮิสเทรีซิส (hysteresis uncertainty) จะถูกระบุเป็นค่าร้อยละของช่วงอินพุตที่วัดได้ทั้งหมด

2.3.11 ความเป็นเชิงเส้นและความผิดพลาดของความไว (Linearity and Sensitivity Errors)

อุปกรณ์วัดช่วงใช้งานควรมีความเป็นเชิงเส้น (linear relationship) ระหว่างอินพุตและเอาต์พุต บางครั้งไม่เป็นเส้นตรง (non-linearity) หมายถึงค่าที่วัดได้ไม่เป็นเชิงเส้นระหว่างค่าอินพุตและเอาต์พุตของระบบ ทำให้จำเป็นต้องหาข้อผิดพลาดจากเส้นตรงตามสมการ

$$u_L(x) = y(x) - y_L(x) \tag{2.28}$$

ซึ่ง y_L(x) คือ ค่าที่คาดว่าจะได้จากสมการเชิงเส้นอ้างอิง และ y(x) คือ ค่าที่วัดได้จริง ค่าสะท้อนถึงการเบี่ยงเบนของระบบจากพฤติกรรมเชิงเส้น ที่ไม่สมบูรณ์ขององค์ประกอบภายในอุปกรณ์ ความผิดพลาดของความไว (Sensitivity Error) หมายถึงค่าความไว (Sensitivity) ของอุปกรณ์ที่วัดไม่คงที่ตลอดช่วงการทำงาน เช่น การเปลี่ยนวัสดุในเซนเซอร์ ความร้อนสะสม หรือการเสื่อมสภาพของวงจรขยายสัญญาณ

2.3.12 ลำดับชั้นของมาตรฐาน (Hierarchy of Standards)

ระบบการวัดที่จำเป็นต้องเทียบกับมาตรฐานอ้างอิงที่ยอมรับ เพื่อให้ผลการวัดเทียบได้ในระดับสากล แม้ว่ามาตรฐานหลักที่สมบูรณ์แบบจะไม่สามารถใช้ได้ในการสอบเทียบ สำหรับการปฏิบัติงาน มีการสร้างลำดับชั้นของมาตรฐานที่ใช้อ้างอิง โดยมีโครงสร้างลำดับชั้นของมาตรฐานดังนี้

1) มาตรฐานหลัก (Primary Standards): เป็นมาตรฐานที่ถือว่าเป็นค่าที่แน่นอนที่สุด
2) มาตรฐานถ่ายโอน (Transfer Standards): ใช้เพื่อสอบเทียบมาตรฐานท้องถิ่น
3) มาตรฐานท้องถิ่น (Local Standards): ใช้เพื่อสอบเทียบมาตรฐานการทำงาน
4) มาตรฐานการทำงาน (Working Standards): เป็นการสอบเทียบอุปกรณ์ในภาคปฏิบัติประจำวัน

การวิเคราะห์เชิงสถิติ (Statistical Analysis of Measurements) เมื่อได้ทำการทดลองซ้ำ การหาค่ากลาง และการกระจายของข้อมูลเพื่อประเมินผลการวัด ตัวชี้วัดเช่น ค่าเฉลี่ย (Mean) ค่าเฉลี่ยคือค่ากลางของผลการวัดทั้งหมด คือ

$$Mean = \frac{\sum_{i=1}^{n} x_i}{n} \tag{2.29}$$

โดยที่ x_i คือ ค่าที่วัดได้ และ n คือ จำนวนค่าทั้งหมดที่วัดได้
ความแปรปรวน (Variance) คือ การกระจายตัวของข้อมูล คำนวณได้จากสมการ

$$Variance = \frac{\sum_{i=1}^{n}(x_i - Mean)^2}{n} \tag{2.30}$$

ค่าที่มากบอกถึงการกระจายตัวกว้าง ขณะที่ค่าที่น้อยบ่งบอกถึงผลคงที่และเชื่อถือได้ ส่วนเบี่ยงเบนมาตรฐาน (Standard Deviation)เป็นรากที่สองของค่าความแปรปรวน ใช้บ่งบอกระดับความแตกต่างของข้อมูลจากค่าเฉลี่ยในหน่วยเดียวกันกับข้อมูลจริง

$$\text{Standard Deviation} = \sqrt{\text{Variance}} \tag{2.31}$$

ค่าที่มี Standard Deviation ต่ำ แสดงว่าการวัดมีเสถียรภาพและความน่าเชื่อถือสูง ซึ่งเป็นพื้นฐานสำคัญในการประเมินความไม่แน่นอนของการวัด (Figliola & Beasley, 2011)

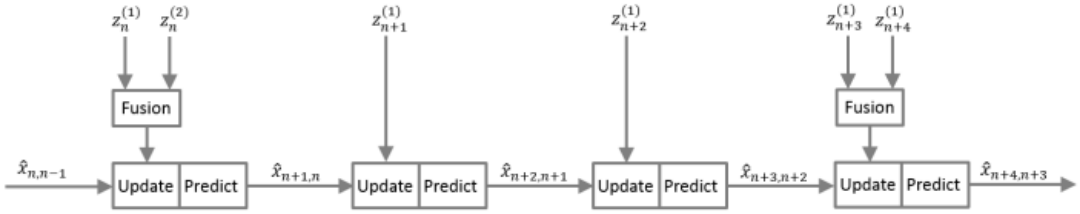
2.4 การหลอมรวมข้อมูลและการจัดการความไม่แน่นอน

ข้อมูลที่ได้รับจากเซนเซอร์เพียงชนิดเดียวมักมีข้อจำกัดทั้งในด้านความละเอียด ความถี่ของสัญญาณ ดังนั้น จึงนำเทคนิคการผสานข้อมูลจากหลายเซนเซอร์ (Multi-Sensor Fusion) มาใช้เป็นการรวมข้อมูล ผสานข้อมูลเพื่อลดค่าความไม่แน่นอนและเพิ่มความแม่นยำของค่าประมาณ เมื่อเทียบกับเซนเซอร์เพียงตัวเดียว

นอกจากนี้ การผสานข้อมูลทำการประมาณค่าตำแหน่ง ความเร็ว หรือมุมทิศทางของตัวรถ เช่น รถยนต์ไร้คนขับ หรือ AGV บางครั้งสัญญาณจากเซนเซอร์บางตัวสูญหายหรือมีสัญญาณรบกวนสูง กระบวนการผสานข้อมูลจะช่วยยเป็นการนำข้อดีของเซนเซอร์ชนิดหนึ่งมาชดเชย ข้อจำกัดของเซนเซอร์อีกชนิดหนึ่ง

2.4.1 ระบบหลายอัตราการเก็บตัวอย่าง (Multi-rate Sensor Systems)

ระบบผสานข้อมูลคือความแตกต่างของอัตราการสุ่มตัวอย่าง (Sampling Rate) เซนเซอร์ แต่ละชนิด เช่น ข้อมูลจาก IMU อาจมีความถี่สูงถึง 100 Hz หรือมากกว่า ในขณะที่ข้อมูลจากเซนเซอร์เชิงภาพอย่างกล้องวงจรปิด อาจมีความถี่เพียง 5–20 Hz (Liu et al., 2024) ความแตกต่างนี้ทำให้เกิด ระบบหลายอัตราการเก็บตัวอย่าง (Multi-rate System) ซึ่งยากต่อการนำข้อมูลที่มาถึงไม่พร้อมกันมาประมวลผลร่วมกันในจังหวะเวลาเดียวกัน เพื่อจัดการกับปัญหานี้ จึงประยุกต์ใช้ Multirate Kalman Filter การปรับปรุงขั้นตอนการทำงานของตัวกรองคัลมานให้สามารถรองรับข้อมูล และใช้การทำงานของ Multirate Kalman Filter ดังรูป 2.7 คือ ภาพการทำนาย (Prediction) สถานะจะเกิดในช่วงเวลา (Δt) อาศัยเซนเซอร์ความถี่สูง (Sensor 1) และแบบจำลอง ในขณะที่ขั้นตอน การอัปเดต (Update) จะมีการอัปเดตย่อยทุกครั้งที่ได้รับข้อมูลจาก Sensor 1 และมีการอัปเดตครั้งสำคัญเมื่อได้รับข้อมูลจากเซนเซอร์ความถี่ต่ำ (Sensor 2) ณ เวลา $3\Delta t$ $6\Delta t$ ... ซึ่งเป็นการนำข้อมูลมาใช้แก้ไขค่าที่เกิดขึ้นระหว่างการทำนาย (Becker, 2023) ระบบผสานข้อมูลแบบหลายอัตราการสุ่ม (Multi-rate Sensor Fusion) เป็นองค์ประกอบสำคัญของระบบนำทางอัตโนมัติยุคใหม่ เนื่องจากช่วยเพิ่มความต่อเนื่อง ความทนทานในกระบวนการคำนวณตำแหน่ง



รูปที่ 2.7 Multirate Kalman Filter (Becker, 2023)

2.4.2 การแก้ไขความเบี่ยงเบน (Bias Correction)

ความเบี่ยงเบน (Bias) คือค่าความผิดพลาดที่มีลักษณะค่อนข้างคงที่ ซึ่งแฝงอยู่ในข้อมูลดิบที่วัดได้จากเซนเซอร์ ในเซนเซอร์เชิงสัมพัทธ์ (Relative Sensors) อย่าง IMU และ Encoder ค่าความเบี่ยงเบนนี้หากไม่ถูกกำจัดออกไป จาก ความคลาดเคลื่อนสะสม (Cumulative Error หรือ Drift) อย่างรวดเร็วในกระบวนการอินทิเกรตเพื่อหาตำแหน่ง โดยเป็นการนำข้อมูลเซ็นเซอร์ดิบในกระบวนการหลอมรวม จะมีการประเมินและแก้ไขค่าความเบี่ยงเบนเชิงระบบของเซ็นเซอร์แต่ละตัว (Perera et al., 2006) วิธีนี้ทำได้โดยการคำนวณค่าเบี่ยงเบนเฉลี่ยระหว่างค่าที่วัดได้ S และจากค่า Ground Truth (GT) ตามสมการ

$$\bar{b}_s = \frac{1}{N}\sum_{t=1}^{N}(S_t - GT_t) \tag{2.32}$$

โดยที่ S_t คือ ค่าที่วัดได้จากเซ็นเซอร์ ณ เวลา t และ GT_t คือ ค่าอ้างอิงจากเส้นทางจริง GT ในเวลาเดียวกัน ค่า S_t' ที่แก้ไขแล้วกำหนดโดย

$$S_t' = S_t - \bar{b}_s \tag{2.33}$$

กระบวนการนี้จะช่วยลดค่า Root Mean Squared Error (RMSE) และปรับปรุงข้อมูล (Khaleghi et al., 2013)

2.4.3 การถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting)

หลังจากทำการแก้ไขความความเบี่ยงเบนแล้ว จะพบว่าค่าเซนเซอร์ต่างชนิดมีระดับความเชื่อมั่นไม่เท่ากัน บางชนิดให้ค่าที่เสถียรกว่า บางตัวมีการกระจายตัวสูง ทำให้ผลมีข้อมูลที่ไม่นิ่ง จึงวัดค่าทางสถิติที่เรียกว่า ความแปรปรวน (Variance, σ^2) แนวคิด คือ เซนเซอร์ที่มีค่าต่ำ หมายถึงการกระจายของค่าผลลัพธ์แคบและมีความเสถียรสูง จึงใช้การถ่วงน้ำหนักด้วยส่วนกลับของความ

แปรปรวน (Inverse-Variance Weighting) เป็นหลักการทางสถิติ (Chen et al., 2025) วิธีนี้สมมติว่าเซ็นเซอร์ที่มีค่าความแปรปรวนต่ำจะได้รับน้ำหนักที่สูงกว่าในระหว่างการรวมข้อมูล น้ำหนัก w_i สำหรับแต่ละเซ็นเซอร์ i คำนวณจากค่า $\boldsymbol{\sigma}_i^2$ ดังนี้

$$w_i = \frac{1}{\sigma_i^2} \tag{2.34}$$

จากนั้นน้ำหนักจะถูกปรับให้ผลรวมของน้ำหนักทั้งหมดเท่ากับ 1 เพื่อให้อยู่ในมาตรฐานเดียวกัน โดยแปลงเป็นรูปแบบ normalized ดังนี้

$$w_i' = \frac{w_i}{\sum_{j=1}^{M} w_j} \tag{2.35}$$

โดยที่ M คือ จำนวนเซ็นเซอร์ทั้งหมด ช่วยให้รวมข้อมูลจากหลายแหล่ง โดยมีความแม่นยำสูงจะมีอิทธิพลต่อค่าประมาณรวมมากกว่าเซนเซอร์ที่มีสัญญาณรบกวนสูง เหมาะกับ AGV เช่น การรวมข้อมูลจาก Encoder IMU และ UWB เพื่อปรับปรุงตำแหน่งและมุมทิศทาง

2.5 ทฤษฎีตัวกรองคัลมาน (Kalman Filter Theory)

การผสานข้อมูลหลายแหล่งเพื่อยกระดับคุณภาพของสัญญาณแล้ว ต่อไปคือการประเมินการทำงานของ AGV ซึ่งต้องอาศัยการคำนวณข้อมูลที่มีความไม่แน่นอนและมีความแปรปรวนจากหลายแหล่งปัญหาการประมาณค่าสถานะระบบลักษณะนี้ คือ ความไม่แน่นอนของสัญญาณแต่ละตัว จากข้อจำกัดของฮาร์ดแวร์ สภาพแวดล้อม และสัญญาณรบกวน (Noise) จึงมีการนำแนวคิดของ “ตัวกรองคัลมาน (Kalman Filter)” มาใช้ประมาณค่าระบบ ตัวกรองคัลมานทำหน้าที่ผสมผสานข้อมูลที่จากแบบจำลองกับข้อมูลจริงที่วัด กล่าวโดยสรุป ตัวกรองคัลมานเป็นเทคนิคที่มีประสิทธิภาพสูงในการประมวลผลข้อมูลที่มีความไม่แน่นอน เป็นรากฐานสำคัญของกระบวนการกรองและผสานข้อมูลในระบบควบคุมอัตโนมัติสมัยใหม่ เช่น AGV หุ่นยนต์เคลื่อนที่ และยานไร้คนขับ

2.5.1 ตัวกรองคัลมาน (Kalman Filter: KF)

ตัวกรองคัลมาน (Kalman Filter: KF) เป็นการประมาณค่าสถานะของ ระบบพลวัตเชิงเส้น (Linear Dynamic System) ที่มีความไม่แน่นอนหรือสัญญาณรบกวน มีกระบวนการทำงาน 2 ขั้นตอนหลัก คือ การทำนาย (Prediction) และ การอัพเดต (Update) (Becker, 2023)

หลักการทำงาน

หัวใจสำคัญของตัวกรองคัลมานคือสถานะของระบบและความไม่แน่นอนที่เกี่ยวข้องด้วย การแจกแจงความน่าจะเป็น (Gaussian Distribution) การแจกแจงนี้ถูกนิยามด้วยสองค่าหลัก

ทางสถิติ คือ ความแปรปรวนร่วม (Covariance Matrix, P) ซึ่งเป็นมาตรวัดเชิงปริมาณของความไม่แน่นอน (Uncertainty) ในการประมาณค่านั้น จากนั้นตัวกรองจะทำงาน โดยใช้กระบวนการทำนายและอัพเดตเพื่อปรับปรุงทั้งค่าประมาณสถานะและความไม่แน่นอนนี้ให้มีความแม่นยำสูงขึ้นอย่างต่อเนื่องเมื่อมีข้อมูลวัดใหม่เข้ามา

กระบวนการทำงานของตัวกรองคัลมานสามารถแบ่งดังนี้

1) ขั้นตอนการทำนาย (Prediction Step) ตัวกรองจะใช้แบบจำลองของระบบเพื่อทำนายสถานะและความไม่แน่นอนของสถานะอนาคต (เวลา k) โดยอาศัยข้อมูลจากสถานะก่อนหน้า (เวลา k−1)

สมการทำนายสถานะ (State Prediction Equation)

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \tag{2.35}$$

โดยที่

$\hat{x}_{k|k-1}$ คือ สถานะที่ทำนาย ณ เวลา k โดยใช้ข้อมูลถึงเวลา k-1

$\hat{x}_{k-1|k-1}$ คือ สถานะที่ประมาณค่าได้ ณ เวลา k-1

F_k คือ เมทริกซ์การเปลี่ยนสถานะ (State Transition Matrix)

B_k คือ เมทริกซ์ควบคุม (Control Matrix)

u_k คือ เวกเตอร์ควบคุม (Control Vector)

สมการทำนายความแปรปรวนร่วม (Covariance Prediction Equation)

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \tag{2.36}$$

โดยที่

$P_{k|k-1}$ คือ ความแปรปรวนร่วมของสถานะที่ทำนาย

$P_{k-1|k-1}$ คือ ความแปรปรวนร่วมของสถานะที่ประมาณค่าได้

Q_k คือ ความแปรปรวนร่วมของสัญญาณรบกวนในกระบวนการ (Process Noise Covariance)

2) ขั้นตอนการอัพเดต (Update Step หรือ Measurement Update) เมื่อมีข้อมูลวัดใหม่ที่ได้จริง (z_k) เข้ามา ตัวกรองจะใช้ข้อมูลนี้เพื่อปรับแก้ค่าที่ได้จากการทำนายให้มีความแม่นยำยิ่งขึ้น

สมการคำนวณอัตราขยายคัลมาน (Kalman Gain Equation)

$$K_k = P_{k|k-1}H_k^T\left(H_k P_{k|k-1}H_k^T + R_k\right)^{-1} \quad (2.37)$$

Kalman Gain (K_k) คือ ค่าน้ำหนักที่ใช้ในการถ่วงดุลระหว่างความน่าเชื่อถือของข้อมูลทำนายและการวัดจริง

H_k คือ เมทริกซ์การวัด (Observation Matrix)

R_k คือ ความแปรปรวนร่วมในการวัด (Measurement Noise Covariance)

สมการอัพเดตสถานะ (State Update Equation)

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_k - H_k\hat{x}_{k|k-1}\right) \quad (2.38)$$

เป็นการปรับแก้ค่าสถานะที่ทำนายไว้

สมการอัปเดตความแปรปรวนร่วม (Covariance Update Equation)

$$P_{k|k} = (I - K_k H_k)P_{k|k-1} \quad (2.39)$$

เป็นการปรับค่าความไม่แน่นอน

2.5.2 ตัวกรองคัลมานแบบขยาย (Extended Kalman Filter: EKF)

เนื่องจากตัวกรองคัลมานแบบปกติสามารถใช้ได้กับระบบเชิงเส้นเท่านั้น แต่ระบบของ AGV ความจริงไม่เป็นเชิงเส้น (Non-linear) (เช่น มีฟังก์ชันตรีโกณมิติในแบบจำลอง) จึงมีการพัฒนา Extended Kalman Filter (EKF) ขึ้นมาเพื่อจัดการ

หลักการทำงาน

EKF คือ การประมาณค่าฟังก์ชันที่ไม่เป็นเชิงเส้นให้เป็นเสมือนฟังก์ชันเชิงเส้น ณ จุดทำงานปัจจุบัน (Local Linearization) โดยใช้ เมทริกซ์จาโคเบียน (Jacobian Matrix) ซึ่งเป็นเมทริกซ์ที่รวบรวมอนุพันธ์ย่อยของฟังก์ชันนั้น ๆ จากนั้นจึงนำระบบที่ถูกแปลงให้เป็นเชิงเส้นแล้วเข้าสู่กระบวนการทำนายและอัพเดตของตัวกรองคัลมานตามปกติ (Becker, 2023)

ขั้นตอนและสมการ

แบบจำลองที่ไม่เป็นเชิงเส้น คือ

$$x_k = f(x_{k-1}, u_k) + w_k \tag{2.40}$$

$$z_k = h(x_k) + v_k \tag{2.41}$$

x_k คือ สถานะของระบบในเวลาที่ k

u_k คือ อินพุตควบคุม

z_k คือ การวัด

w_k และ v_k คือ สัญญาณรบกวน

1) ขั้นตอนการทำนาย (Prediction Step)

การทำนายสถานะ (State Prediction)

$$\hat{x}_k = f\left(\hat{x}_{k-1|k-1}, u_k\right) \tag{2.42}$$

ใช้ฟังก์ชันที่ไม่เป็นเชิงเส้น f ในการทำนาย

การทำนายความแปรปรวนร่วม (Covariance Prediction)

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \tag{2.43}$$

โดยที่ F_k คือ เมทริกซ์จาโคเบียน ของฟังก์ชัน f เทียบกับสถานะ x ณ จุด $\hat{x}_{k-1|k-1}$

$$F_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1|k-1}, u_k} \tag{2.44}$$

2) ขั้นตอนการอัปเดต (Update Step)

คำนวณ Kalman Gain (K_k)

$$K_k = P_{k|k-1} H_k^T \left(H_k P_{k|k-1} H_k^T + R_k\right)^{-1} \tag{2.45}$$

โดยที่ H_k คือ เมทริกซ์จาโคเบียน ของฟังก์ชัน h เทียบกับสถานะ x ณ จุด $\hat{x}_{k-1|k-1}$

$$H_k = \frac{\partial h}{\partial x}\Big|_{\hat{x}_{k|k-1}} \tag{2.46}$$

อัพเดตสถานะ (State Update)

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_k - H_k\hat{x}_{k|k-1}\right) \tag{2.47}$$

ใช้ฟังก์ชันที่ไม่เป็นเชิงเส้น h ในการคำนวณค่าที่คาดว่าจะวัดได้
อัพเดตความแปรปรวนร่วม (Covariance Update)

$$P_{k|k} = (I - K_k H_k)P_{k|k-1} \tag{2.48}$$

EKF จะสามารถทำการประมาณตำแหน่งและค่าความไม่แน่นอนได้ สามารถใช้การหลอมรวมเซนเซอร์เพื่อลดการแจกแจงปกติลงได้

2.6 งานวิจัยที่เกี่ยวข้อง

การระบุตำแหน่งสำหรับยานพาหนะนำทางอัตโนมัติ (AGV) ภายในอาคารได้รับความสนใจอย่างแพร่หลายในการวิจัยและพัฒนา เพื่อยกระดับความแม่น (Accuracy) ความเชื่อถือ (Reliability) และความทนทาน (Robustness) ของการทำงานของ AGV งานนี้มีแนวโน้มมุ่งเน้นการผสานข้อมูลหลากหลายประเภทเพื่อเพิ่มความแม่นในการคาดคะเนตำแหน่งของยานพาหนะในพื้นที่จำกัด เช่น โรงงานหรือคลังสินค้า ซึ่งต้องการความแม่นในระดับสูงและการตอบสนองได้เร็ว ในขณะเดียวกันก็พยายามลดต้นทุนเพื่อให้ใช้ในภาคอุตสาหกรรม สามารถแบ่งกลุ่มได้ดังนี้

การหลอมรวมข้อมูลจากหลายเซนเซอร์สำหรับ AGV

แนวคิดของการหลอมรวมข้อมูลจากหลายเซนเซอร์ (Multi-Sensor Fusion) เพื่อลดข้อด้อยของการใช้ข้อมูลชนิดเดียว เนื่องจากข้อมูลแต่ละชนิดมีข้อเด่น และข้อด้อยแตกกัน การนำข้อมูลหลายประเภทมาประมวลผลชดเชยซึ่งกันและกัน งานวิจัยจำนวนมากได้นำเสนอการหลอมรวมข้อมูลระหว่างเซนเซอร์เชิงสัมพัทธ์ (Relative Sensors) ตัวอย่างเช่น เซนเซอร์เชิงสัมพัทธ์ (Relative Sensors) อย่าง Encoder และ IMU ซึ่งให้ข้อมูลที่มีความถี่สูงแต่ประสบปัญหาความคลาดเคลื่อนสะสม (Drift) เข้ากับเซนเซอร์เชิงสัมบูรณ์ (Absolute Sensors) ตัวอย่างเช่น กล้อง LiDAR หรือ UWB ซึ่งให้ข้อมูลตำแหน่งอ้างอิงกับสภาพแวดล้อมแต่มีความถี่ต่ำกว่า อาจมีจุดอับสัญญาณ ไวต่อ

สัญญาณรบกวนและสภาพแสง โดยการผสานข้อมูลจากเซนเซอร์ (B. Cheng et al., 2024; Guan et al., 2024; Teng et al., 2024; Zhang et al., 2024)

วิธีที่เกี่ยวข้องกับทบทวนวรรณกรรม

ในด้านวิธี ประกอบด้วย Kalman Filter กรองข้อมูลจากหลายแหล่งในการประมาณหาค่าตำแหน่ง และ Extended Kalman Filter (EKF) สำหรับการคาดคะเนและปรับปรุงค่าสถานะจากค่าความไม่แน่นอน ขณะเดียวกัน วิธีอื่น ๆ ตัวอย่างเช่น Particle Filter สำหรับการกรองข้อมูลที่ซับซ้อน และเทคนิคการเรียนรู้เชิงลึก (Neural Networks) นอกจากนี้ยังมีวิธี ตัวอย่างเช่น LS-AKF (Least Squares-Adaptive Kalman Filter) และ IUPF (Implicit Unscented Particle Filter) ยังถูกพัฒนาขึ้นเพื่อเพิ่มความแม่นในการหาตำแหน่ง (B. Cheng et al., 2024; L. Cheng et al., 2024; Song et al., 2024; Wan et al., 2024; Xue & Liang, 2024; Zhang et al., 2024; Zhou et al., 2024)

วิธี Extended Kalman Filter (EKF)

EKF ได้กลายเป็นอุปกรณ์วิธีการมาตรฐานที่ใช้ในการหลอมรวมข้อมูลสำหรับ AGV เนื่องจากสามารถจัดการกับแบบจำลองที่ไม่เป็นเชิงเส้นและสัญญาณรบกวน เช่น การติดตามเส้นทางการของ AGV

แนวทางที่นำเสนองานวิจัย

การหลอมรวมข้อมูลสำหรับ AGV แต่ยังมีสองประการสำคัญ คือ หนึ่ง การขาดกระบวนการประเมินและคัดเลือกเซนเซอร์ต้นทุนต่ำอย่างในสภาพเดียวกัน และ สอง มีการพัฒนาวิธีการหลอมรวมข้อมูลโดยอาจละเลยความสำคัญของขั้นตอนการประมวลผลข้อมูลเบื้องต้น (Pre-processing) มีประสิทธิภาพ วิทยานิพนธ์นี้ เริ่มต้นจากการทดลองเทียบประสิทธิภาพแต่ละชนิด สำหรับการคัดเลือกโดยวิเคราะห์ การแก้ไขความเบี่ยงเบน (Bias Correction) และ การถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting) มีผลต่อข้อมูลตั้งต้นและส่งผลกับระบบ

นอกจากกระบวนการวิจัย ประเด็นหลักที่นำเสนอคือการออกแบบ และเทียบสถาปัตยกรรมการหลอมรวมข้อมูลภายใน EKF โดยมีแนวคิดใหม่ คือ Blended EKF ซึ่งต่างจาก Standard EKF โดย Blended EKF ทำการหลอมรวมข้อมูลจาก Dead Reckoning และกล้องเข้าด้วยกันก่อนที่จะป้อนเข้าการอัพเดต และได้พิสูจน์ให้เห็นว่าสามารถสร้างความ ราบรื่นและต่อเนื่องกว่า ดังนั้น จึงประยุกต์ใช้เทคนิคที่มีอยู่เดิม แต่เป็นการนำเสนอกระบวนการและนวัตกรรม ซึ่งระบบหาตำแหน่งสำหรับ AGV ต้นทุนต่ำ

2.7 สรุป

ในบทนี้ได้ทำการศึกษาการพัฒนาระบบหาตำแหน่งสำหรับ AGV อย่างครอบคลุม โดยอธิบายภาพรวม สถาปัตยกรรม และองค์ประกอบระบบ AGV ซึ่งชี้ให้เห็นว่า การระบุตำแหน่ง (Localization) คือหัวใจสำคัญของระบบย่อยอื่น ๆ ทั้งหมด

ได้เขียนเทคนิคและวิธีการต่าง ๆ ที่ใช้ในการหาตำแหน่งภายในอาคาร สำหรับเซนเซอร์ต้นทุนต่ำ ได้แก่ Encoder หน่วยวัดแรงเฉื่อย (IMU) อัลตร้าไวด์แบนด์ (UWB) และกล้องวงจรปิด (CCTV) หลักการทำงานและข้อด้อยของเซนเซอร์แต่ละชนิด เช่นปัญหา ความคลาดเคลื่อนสะสม (Cumulative Error) ในเซนเซอร์เชิงสัมพัทธ์ (Encoder, IMU) ซึ่งเป็นปัญหาสำคัญ

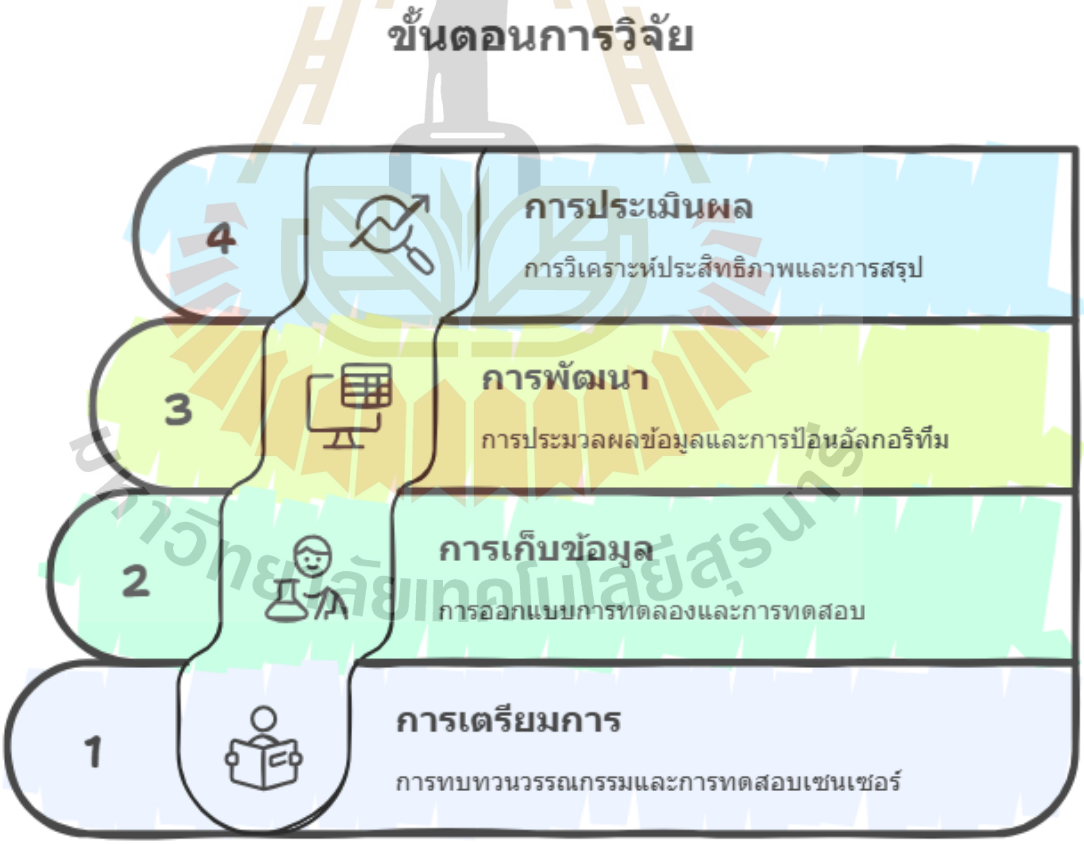
เพื่อจัดการกับข้อด้อยดังกล่าว นำเสนอ การหลอมรวมข้อมูลจากหลายเซนเซอร์ (Multi-Sensor Fusion) และได้อธิบายถึงความท้าทายที่เกี่ยวข้องอย่างละเอียด ไม่ว่าจะเป็น ปัญหาระบบหลายอัตราการเก็บตัวอย่าง (Multi-rate System) และพัฒนากรอบการทำงานเพื่อ จัดการความไม่แน่นอนของข้อมูล ซึ่งใช้ขั้นตอนจัดการข้อมูลเบื้องต้นคือ การแก้ไขความเบี่ยงเบน (Bias Correction) และ การถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting)

ท้ายที่สุด ได้มีการอธิบายทฤษฎีของ ตัวกรองคัลมานแบบขยาย (Extended Kalman Filter: EKF) ซึ่งเป็นอุปกรณ์ที่ใช้ในการหลอมรวมข้อมูลสำหรับ AGV ระบบที่ไม่เป็นเชิงเส้นของ AGV เพื่อนำไปใช้ในวิธีวิจัยและพัฒนากรอบการทำงานสำหรับการหลอมรวมข้อมูล AGV ในบทต่อไป ในการหาตำแหน่งสำหรับ AGV ที่ ต้นทุนต่ำ และใช้ได้จริง

บทที่ 3
ระเบียบวิธีวิจัย

3.1 บทนำ

การดำเนินการวิจัยครั้งนี้เป็นไปตามลำดับขั้นตอนที่กำหนดไว้อย่างเป็นระบบ ดังรูป 3.1 โดยแต่ละขั้นประกอบด้วยกระบวนการตั้งแต่การเตรียมความพร้อมก่อนการทดลอง การพัฒนาแบบจำลองและระบบที่ใช้ในการผสานข้อมูล จนถึงการประเมินประสิทธิภาพของวิธีที่พัฒนาขึ้น แนวทางการดำเนินงานดังกล่าวถูกออกแบบให้สอดคล้องกับวัตถุประสงค์ของการวิจัย เพื่อให้สามารถทดสอบและตรวจสอบความถูกต้องของระบบได้อย่างเป็นขั้นตอน และสะท้อนผลการพัฒนาในแต่ละช่วงของงานวิจัยอย่างเป็นรูปธรรม



รูปที่ 3.1 ภาพรวมขั้นตอนวิจัย

กระบวนการวิจัยถูกแบ่งเป็น 3 ส่วน

1) การออกแบบและเตรียมการทดลอง (Experimental Setup and Design)

ซึ่งจะอธิบายแพลตฟอร์มการออกแบบระบบต้นแบบ AGV การเลือกและติดตั้งเซนเซอร์ตามคุณลักษณะและต้นทุนต่ำที่ใช้ทั้งหมด และสภาวะในการทดลอง รวมถึงการเก็บข้อมูลทั้งในสภาวะหยุดนิ่ง (Static Test) เพื่อวิเคราะห์คุณลักษณะเฉพาะของเซ็นเซอร์ และในสภาวะเคลื่อนที่ (Dynamic Test) เพื่อรวบรวมชุดข้อมูลสำหรับการประเมินผล

2) การพัฒนาการทำงานสำหรับการหลอมรวมข้อมูล (Sensor Fusion)

โดยนำเสนอคณิตศาสตร์และวิธีการ Extended Kalman Filter (EKF) ถูกปรับปรุงให้อยู่ในสองแบบ คือ

2.1) Standard EKF สำหรับทั่วไป

2.2) Blended EKF ออกแบบให้สามารถปรับน้ำหนักข้อมูลตามคุณภาพของแต่ละชนิด ได้มีการเพิ่มขั้นตอนใช้เทคนิคการคำนวณข้อมูลเบื้องต้น เช่น การแก้ไขค่าความเบี่ยงเบน (Bias Correction) และแนวคิดการผสมข้อมูลการวัดด้วยการถ่วงน้ำหนัก (Inverse-Variance Weighting)

3) เกณฑ์การประเมินประสิทธิภาพ (Performance Evaluation)

ซึ่งจะกำหนดวิธีการวิเคราะห์และเทียบผลลัพธ์ที่ได้จากวิธีสองในเชิงปริมาณ โดยใช้ ค่าความคลาดเคลื่อนเฉลี่ยรากกำลังสอง (Root Mean Square Error: RMSE) เพื่อสรุปและยืนยันเทียบความแม่นของ EKF ทั้งสองรูปแบบและระบบนำทางเดิม

โดยภาพรวม ขั้นตอนการดำเนินงานทั้งหมดได้รับการจัดเรียงให้ต่อเนื่องกันเป็นลำดับ ตั้งแต่การเตรียมข้อมูล การประมวลผล จนถึงการประเมินผล เพื่อให้สามารถตรวจสอบและทำซ้ำได้

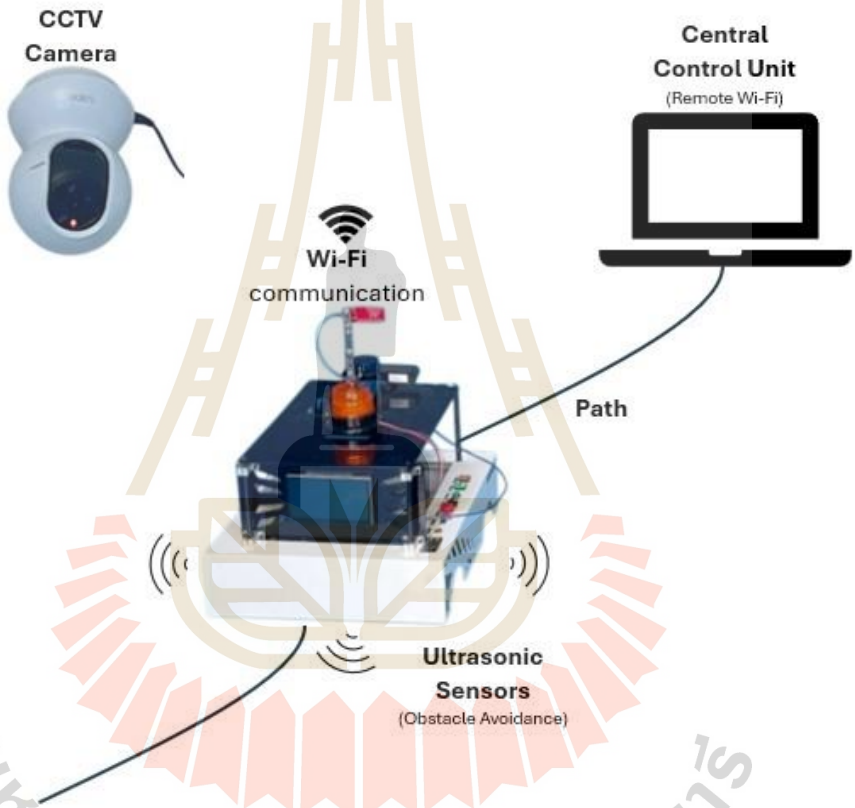
3.2 สถาปัตยกรรม อุปกรณ์และซอฟต์แวร์

แนวคิดการพัฒนาระบบหาตำแหน่งสำหรับ AGV ภายใต้สถาปัตยกรรม "การควบคุมจากส่วนกลางโดยใช้ระบบกล้อง (Centralized Control with Vision System)" ซึ่งเป็นสถาปัตยกรรมที่เน้นการประสานข้อมูลจากหลายแหล่ง เพื่อสนับสนุนการรับรู้ตำแหน่งและการเคลื่อนไหวของ AGV ในพื้นที่จำกัด เน้นลดต้นทุนและลดความซับซ้อนของรถ AGV แต่ละคัน แนวคิดดังกล่าวมุ่งให้การประมวลผลหลักเกิดขึ้นภายนอกตัวรถ โดยมีหน่วยควบคุมส่วนกลางทำหน้าที่รวบรวมและคำนวณข้อมูล ก่อนส่งค่าตำแหน่งที่ประเมินได้กลับไปควบคุม AGV แบบออนไลน์ ระบบนี้ช่วยลดภาระการคำนวณบนตัวรถและเพิ่มความสามารถในการทำงานร่วมกันระหว่างหลายคัน การย้ายการคำนวณไว้ส่วนกลางแบ่งออกเป็นสองส่วน ได้แก่

1) โครงสร้างพื้นฐาน (Infrastructure) ซึ่งประกอบด้วยกล้องตรวจจับวัตถุที่ติดตั้งอยู่ในพื้นที่ทดสอบ เซิร์ฟเวอร์สำหรับคำนวณภาพ และระบบเครือข่ายที่เชื่อมโยงอุปกรณ์ทั้งหมดเข้าด้วยกัน

2) รถ AGV ซึ่งติดตั้งเซนเซอร์พื้นฐาน เช่น Encoder IMU และ UWB สำหรับใช้ในการระบุตำแหน่งในตัวรถ

ความสัมพันธ์ระหว่างส่วนประกอบทั้งสองถูกออกแบบให้ทำงานร่วมกัน ดังรูป 3.2 โดยรายละเอียดของอุปกรณ์ ฮาร์ดแวร์ และซอฟต์แวร์ที่ใช้ในการวิจัยจะแสดงในหัวข้อถัดไป



รูปที่ 3.2 สถาปัตยกรรมระบบควบคุมจากส่วนกลาง

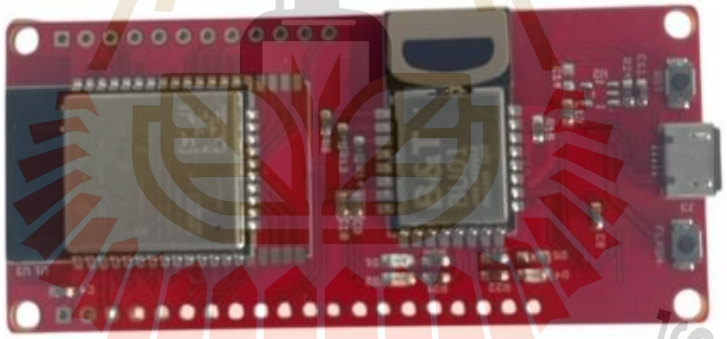
3.2.1 ส่วนรับรู้จากภายนอก: ดวงตาของระบบ (Eyes: The Perception Layer)

ส่วนนี้เสมือนดวงตาของระบบทั้งหมด โดยใช้โครงสร้างพื้นฐานที่รับข้อมูลภาพรวมของพื้นที่ปฏิบัติการทั้งหมดจากมุมสูง (God's-eye view)

1) ระบบกล้องวงจรปิด (CCTV System): ใช้กล้อง TP-Link Tapo C200 ดังแสดงในรูปที่ 3.3ติดตั้งไว้บนเพดานเพื่อครอบคลุมพื้นที่การทำงานทั้งหมด หน้าที่หลักของส่วนนี้คือการส่งข้อมูลภาพวิดีโอต่อเนื่อง (Video Stream) ไปยังส่วนควบคุมกลางเพื่อทำการประมวลผลในลำดับถัดไป

รูปที่ 3.3 กล้องวงจรปิด TP-Link Tapo C200

2) ระบบอัลตร้าไวด์แบนด์ (UWB System): ประกอบด้วยอุปกรณ์ส่งสัญญาณ (Anchors) รุ่น ESP32 UWB DW1000 ดังรูป 3.4 จำนวน 4 ตัว ติดตั้งตามมุมของพื้นที่ปฏิบัติการ และอุปกรณ์รับสัญญาณ (Tag) ที่ติดตั้งบนตัว AGV ระบบนี้ทำหน้าที่วัดระยะห่างระหว่าง Tag และ Anchors เพื่อคำนวณหา x, y ของ AGV และส่งข้อมูล x, y (Position Data) ไปยังส่วนควบคุมกลาง



รูปที่ 3.4 อัลตร้าไวด์แบนด์ ESP32 UWB DW1000

3.2.2 ส่วนควบคุมกลาง: สมองของระบบ (Brain: The Control Layer)

ส่วนควบคุมกลางทำหน้าที่เสมือนศูนย์ประมวลผลหลักของระบบ โดยมีรับข้อมูลที่ติดตั้งบนตัวรถและอยู่ในพื้นที่ทดสอบมาผ่านกระบวนการคำนวณแบบเป็นลำดับขั้น (Processing Pipeline) เพื่อสร้างข้อมูลเชิงสรุปที่ใช้ในการตัดสินใจควบคุมของ AGV แต่ละคัน

กระบวนการทำงานภายในของส่วนควบคุมกลางสามารถแบ่ง ดังนี้

1) การประมวลผลภาพ (Computer Vision)
   ระบบจะรับข้อมูลภาพจากกล้องจากส่วนรับรู้ (Perception Layer) เพื่อนำมาคำนวณด้วยเทคนิคการมองเห็นของคอมพิวเตอร์ เช่น การตรวจจับตำแหน่งและวัตถุที่สนใจภายในพื้นที่ปฏิบัติการ ซึ่งรวมถึงการระบุขอบเขตของ AGV หรือสิ่งกีดขวาง เพื่อเตรียมข้อมูลสำหรับขั้นตอนการวิเคราะห์ในลำดับถัดไป
2) การหาตำแหน่งและการจัดการรถ (Localization & Fleet Management)
   ข้อมูลจากการคำนวณจะถูกนำมารวมกันเพื่อคำนวณ x, y เชิงพื้นที่ที่แม่นยำของ AGV แต่ละคันในระบบ โดยใช้กระบวนการหลอมรวมข้อมูล (Sensor Fusion) ผ่านวิธี Extended Kalman Filter (EKF) หรือ Blended EKF จากนั้นระบบจะจัดการการตัว AGV ทั้งหมดให้สัมพันธ์กันเพื่อลดการชนและระยะของเส้นทางรวม
3) การวางแผนเส้นทาง (Path Planning)
   เมื่อรู้ x, y ปัจจุบันของรถแล้ว ระบบจะทำการคำนวณและสร้างเส้นทางที่เหมาะสมสำหรับ AGV แต่ละคัน โดยพิจารณาปัจจัยด้านระยะทาง สิ่งกีดขวาง และเป้าหมายการของ AGV จากนั้นคำสั่งการเคลื่อนไหวจะถูกส่งผ่านระบบสื่อสารแบบไร้สาย (เช่น MQTT Protocol) ไปยังส่วนปฏิบัติการบนตัวรถเพื่อดำเนินการตามคำสั่งแบบเรียลไทม์

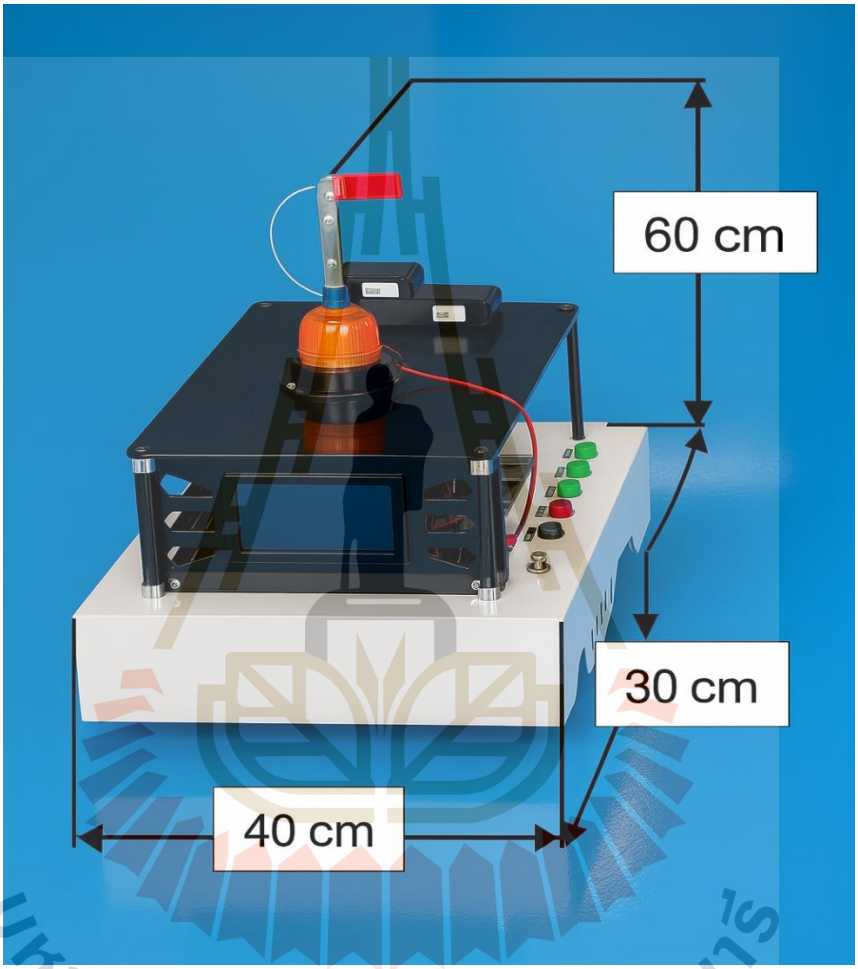
3.2.3 **ส่วนปฏิบัติการ: ตัวรถ** AGV

ส่วนปฏิบัติการเป็นชั้นที่ทำหน้าที่เชื่อมต่อคำสั่งจากระบบควบคุมกลางเข้าสู่ AGV จริง โดยแต่ละคันของ AGV ได้รับการออกแบบให้มีโครงสร้างเรียบง่าย น้ำหนักเบา และสามารถตอบสนองคำสั่งได้อย่างแม่นยำในเวลาจริง เพื่อเพิ่มความคล่องตัวและลดความซับซ้อนในการควบคุม ตัวรถทำหน้าที่หลักในการรับข้อมูลจากหน่วยควบคุมกลาง เช่น คำสั่งความเร็วและมุมทิศทาง จากนั้นจึงคำนวณภายในไมโครคอนโทรลเลอร์ของตัวรถเพื่อขับเคลื่อนตามคำสั่งดังกล่าว ระบบยังสามารถส่งข้อมูลสถานะการเคลื่อนที่กลับไปยังศูนย์ควบคุม เพื่อใช้ในการปรับปรุงหรือแก้ไขเส้นทางได้อย่างต่อเนื่อง

1) AGV ที่ใช้งาน เป็นต้นแบบ ดังรูป 3.5 (BALTY ROBOT) ซึ่งถูกออกแบบให้ขับเคลื่อนด้วยระบบ ล้ออิสระแบบสองล้อ (Differential Drive System) โดยสามารถหมุนได้รอบตัวเอง และเคลื่อนที่ได้อย่างแม่นยำในพื้นที่จำกัด ทั้งนี้โครงสร้างถูกพัฒนาให้รองรับการติดตั้งเซนเซอร์หลายชนิดเพื่อใช้ในการหลอมรวมข้อมูล เช่น Encoder IMU และ CCTV

2) หน่วยควบคุมออนบอร์ด (Onboard Controller)

ใช้ NVIDIA Jetson Nano ทำหน้าที่เป็นสมองกลบนตัวรถ รับคำสั่งระดับสูงจากเซิร์ฟเวอร์ และใช้ข้อมูลจากเซ็นเซอร์บนตัวรถเพื่อควบคุมมอเตอร์ผ่านบอร์ด Teensy 4.1 ให้เคลื่อนที่ตามคำสั่ง



รูปที่ 3.5 AGV BALTY ROBOT

3) ชุดเซนเซอร์บนตัวรถ (Onboard Sensors)

เพื่อให้ระบบ AGV สามารถเคลื่อนที่ให้สอดคล้องกับคำสั่งจากส่วนควบคุมกลาง จึงได้ติดตั้ง ชุดเซนเซอร์บนตัวรถ เพื่อใช้ในการตรวจสอบและยืนยันความถูกต้องของการตามจริง เซนเซอร์เหล่านี้มีบทบาทสำคัญในการให้ข้อมูลเชิงปริมาณ เช่น ระยะ ความเร็วเชิงมุม และมุมทิศทางการหมุน ซึ่งถูกนำไปใช้ในกระบวนการหลอมรวมข้อมูล (Sensor Fusion) เพื่อเพิ่มความเที่ยงตรงของการหา x, y

Encoder ใช้สำหรับวัดระยะทางและความเร็วเชิงเส้นของล้อ โดยคำนวณจากจำนวนพัลส์ที่เกิดขึ้นขณะล้อหมุน ตัวอย่างอุปกรณ์ที่ใช้ ดังรูป 3.6 ซึ่งช่วยให้ระบบสามารถประเมินการเคลื่อนที่ในแนวเส้นตรง

IMU (Inertial Measurement Unit) ใช้สำหรับตรวจจับการหมุนและการเอียงของตัวรถในเชิงมุม โดยวัดค่าอัตราเร่งและอัตราการหมุนรอบแกน x, y และ z ของยานพาหนะ ดังรูป 3.7 ข้อมูลจาก IMU จะถูกใช้ร่วมกับข้อมูลจาก Encoder เพื่อคำนวณมุมทิศทางและความเร็วเชิงมุมของ AGV

รูปที่ 3.6 มอเตอร์พร้อม Encoder

รูปที่ 3.7 IMU

4) การส่งข้อมูลสถานะกลับ (Status Feedback)

AGV จะมีระบบส่งข้อมูลสถานะของการทำงานกลับไปยังเซิร์ฟเวอร์กลาง เพื่อใช้ในการติดตามและประเมินประสิทธิภาพการทำงานแบบออนไลน์ ข้อมูลดังกล่าวอาจประกอบด้วย

x, y ปัจจุบัน ความเร็ว มุมทิศทาง หรือสถานะของภารกิจ (เช่น “ถึงจุดหมายแล้ว” หรือ “ภารกิจเสร็จสมบูรณ์”)

การส่งข้อมูลใช้รูปแบบการสื่อสารผ่านเครือข่าย MQTT (Message Queuing Telemetry Transport) ซึ่งเป็นโปรโตคอลการสื่อสารแบบ publish–subscribe ที่เหมาะกับระบบ IoT และเครือข่ายที่มีแบนด์วิดท์จำกัด MQTT ช่วยให้การส่งข้อมูลมีความรวดเร็ว น้ำหนักเบา และสามารถทำงานได้อย่างมีเสถียรภาพแม้ในสภาวะที่เครือข่ายไม่คงที่

การทดลองใช้ AGV ต้นแบบ BALTY AGV ซึ่งเป็นรถขับเคลื่อนแบบ Differential Drive โดยมีหน่วยคำนวณหลักเป็น NVIDIA Jetson Nano ทำหน้าที่สื่อสารระหว่างตัวรถกับระบบภายนอก และควบคุมการทำงานของระบบหลักทั้งหมด ร่วมกับ ไมโครคอนโทรลเลอร์ Teensy 4.1 ซึ่งทำหน้าที่เชื่อมต่อและควบคุมการอ่านค่าจากเซนเซอร์ เช่น Encoder และ IMU
การพัฒนาวิธีการหลักดำเนินการด้วยภาษา Python เนื่องจากมีความยืดหยุ่นสูง ใช้งานง่าย และรองรับไลบรารีทางวิศวกรรมและวิทยาการคอมพิวเตอร์ เช่น OpenCV, NumPy และ paho-mqtt เพื่อใช้ในการประมวลผลภาพ การคำนวณเชิงตัวเลข และการสื่อสารข้อมูลระหว่างอุปกรณ์
ระบบการสื่อสารระหว่าง AGV และ เซิร์ฟเวอร์กลาง ใช้ โปรโตคอล MQTT (Message Queuing Telemetry Transport) ซึ่งเป็นมาตรฐานการสื่อสารแบบ publish–subscribe ที่มีน้ำหนักเบาและประสิทธิภาพสูง เหมาะสำหรับการเชื่อมโยงข้อมูลในเครือข่าย IoT (Internet of Things) ที่มีอุปกรณ์หลายตัวทำงานร่วมกัน โดยอุปกรณ์และโปรแกรมดังตารางที่ 3.1

ตารางที่ 3.1 รายการอุปกรณ์และซอฟต์แวร์

หมวดหมู่	รายการ	ข้อมูลจำเพาะ/หน้าที่
รถ AGV	AGV ต้นแบบ (BALTY ROBOT)	- ขนาด: 0.4x0.3x0.6 ม. - ระบบขับเคลื่อน: Differential Drive - หน่วยประมวลผลหลัก: NVIDIA Jetson Nano - บอร์ดควบคุมมอเตอร์: Teensy 4.1
เซนเซอร์	มอเตอร์ พร้อม Encoder (ติดตั้งในตัว)	- รุ่น: JGB37-520 high-torque - หน้าที่: วัดระยะทางและทิศทางการเคลื่อนที่
	IMU	- รุ่น: WitMotion WT901C - หน้าที่: วัดค่าความเร่งเชิงเส้น อัตราเร็วเชิงมุม และมุม

ตารางที่ 3.1 รายการอุปกรณ์และซอฟต์แวร์ (ต่อ)

หมวดหมู่	รายการ	ข้อมูลจำเพาะ/หน้าที่
	อัลตร้าไวด์แบนด์ (UWB)	- รุ่น: ESP32 UWB DW1000 - หน้าที่: วัดระยะห่างระหว่าง AGV (Tag) และจุดอ้างอิง (Anchors)
	กล้องวงจรปิด (CCTV)	- รุ่น: TP-Link Tapo C200 - หน้าที่: ตรวจจับและหา x, y ของ AGV จากมุมมองด้านบน
ซอฟต์แวร์	ภาษาโปรแกรม	- Python: ภาษาหลักในการพัฒนาวิธี EKF และวิเคราะห์ข้อมูล
	โปรโตคอลสื่อสาร	- MQTT: ใช้เป็น Broker สำหรับรับ-ส่งข้อมูลระหว่างเซนเซอร์และหน่วยประมวลผลกลาง

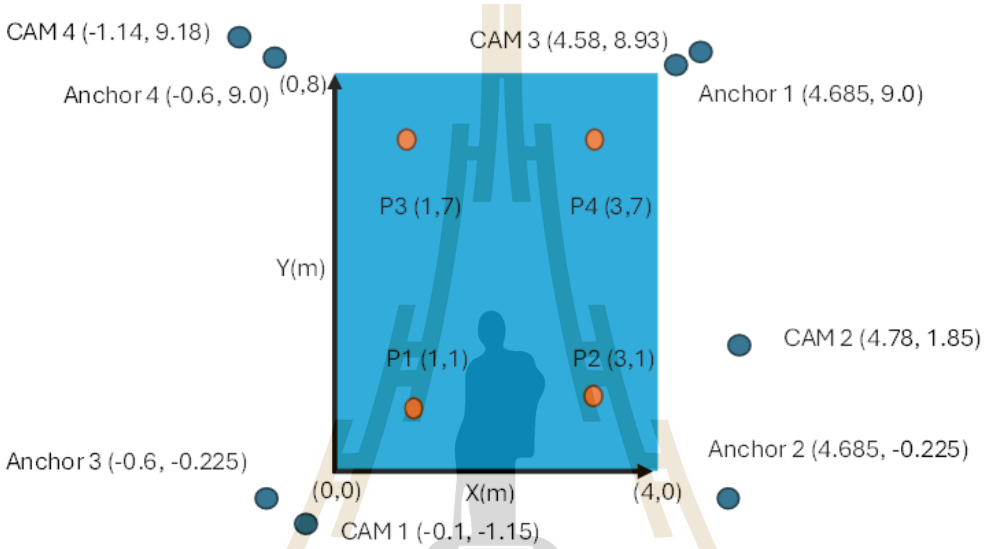
จากโครงสร้างสถาปัตยกรรมของระบบและชุดอุปกรณ์ที่ได้อธิบายไว้ข้างต้น งานวิจัยนี้ได้พัฒนากระบวนการสำหรับการทดลองและการเก็บข้อมูลหลายประเภท ทั้งจาก ระบบพื้นฐาน (CCTV และ UWB) และ บนตัวรถ (Encoder และ IMU) ซึ่งแต่ละชนิดแตกต่างกัน เพื่อใช้ร่วมกันในกระบวนการหลอมรวมข้อมูล (Sensor Fusion Algorithm)

กระบวนการหลอมรวมข้อมูลในงานวิจัยนี้ใช้ วิธี Extended Kalman Filter (EKF) ซึ่งมีบทบาทสำคัญในการคำนวณข้อมูลจากค่าที่มีความไม่แน่นอนและอัตราการสุ่มตัวอย่างแตกต่างกัน โดยเน้นการพัฒนาแบบจำลองและกระบวนการคำนวณที่เหมาะสม เพื่อเพิ่มความเที่ยงตรงของการหา x, yสำหรับ AGV ภายใต้สถาปัตยกรรมการควบคุมแบบรวมศูนย์ (Centralized Vision-Based Control Architecture)

3.3 การออกแบบการทดลอง (Experiment Design)

การทดลองนี้มีวัตถุประสงค์เพื่อจัดเก็บและรวบรวมข้อมูลเชิงทดลอง (Dataset) สำหรับใช้ในการประเมินคุณสมบัติและประสิทธิภาพของวิธีหลอมรวมข้อมูล (Sensor Fusion Algorithm) ที่ได้พัฒนา การออกแบบการทดลองดำเนินการภายในพื้นที่ควบคุมสภาวะ เพื่อจำลองสภาพการทำงานจริงของระบบควบคุม AGV โดยมีขนาดพื้นที่ทดสอบเท่ากับ 4 × 8 เมตร ดังรูป 3.8 มุมมองจากกล้อง CCTV ทั้ง 4 ตัวในพื้นที่ทดสอบสำหรับการติดตามตำแหน่งของ AGV ดังรูป 3.9

บริเวณพื้นที่ดังกล่าวได้ติดตั้ง โครงสร้างพื้นฐานภายนอก (External Sensor Infrastructure) เพื่อรองรับการตรวจจับและเก็บข้อมูล โดยใช้ กล้องวงจรปิด (CCTV) ติดตั้งในลักษณะมุมมองจากด้านบน (Top-Down View) เพื่อใช้ในการติดตาม x, y ของรถ AGV และ อุปกรณ์ Ultra-Wideband (UWB Anchors) จำนวน 4 จุด ติดตั้งในตำแหน่งที่เหมาะสมรอบพื้นที่ทดสอบที่ระดับความสูงประมาณ 0.6 เมตร เพื่อใช้ในการวัดระยะและคำนวณตำแหน่งของ AGV ในเชิงพื้นที่



รูปที่ 3.8 ตำแหน่งติดตั้งกล้องวงจรปิดและ Anchor

รูปที่ 3.9 มุมมองจากกล้อง CCTV ทั้ง 4 ตัวในพื้นที่ทดสอบสำหรับการติดตามตำแหน่งของ AGV

กระบวนการทดลองในงานวิจัยนี้ดำเนินการในลักษณะของการ ประมวลผลแบบออฟไลน์ (Offline Processing) ซึ่งหมายถึง การจัดเก็บและรวบรวมข้อมูลดิบ (Raw Data) จากเซนเซอร์ระหว่างการทดลองจริงให้ครบถ้วนและสมบูรณ์ที่สุด เพื่อใช้เป็นฐานข้อมูลสำหรับการคำนวณภายหลังในโปรแกรมวิเคราะห์ผลการทำงานของระบบ

ข้อมูลที่ได้จะถูกนำเข้าสู่โปรแกรมเพื่อดำเนินการวิเคราะห์เชิงสถิติและเทียบประสิทธิภาพของวิธีที่พัฒนาขึ้น โดยแบ่งออก ดังนี้

3.3.1 ทดสอบในสภาวะหยุดนิ่ง (Static Test)

การทดสอบในสภาวะหยุดนิ่งถูกออกแบบขึ้นเพื่อประเมินความแม่นและเสถียรภาพของระบบหา x, y จากภายนอก (External Localization Systems) ซึ่งในงานนี้ใช้ UWB และ CCTV เป็นอุปกรณ์หลักในการตรวจจับหา x, y ของจุดทดสอบที่กำหนดไว้ล่วงหน้าวัตถุประสงค์ของการทดสอบคือเพื่อตรวจสอบความถูกต้องเชิงตำแหน่งของแต่ละตัว รวมถึงเทียบความแตกต่างของเทคนิคการคำนวณ x, y ในรูปแบบต่าง ๆ ก่อนที่จะนำผลลัพธ์ไปใช้ในการวิเคราะห์และหลอมรวมข้อมูลต่อไป

วัตถุประสงค์

1) เทียบประสิทธิภาพของวิธีการคำนวณหา x, y ด้วย UWB ระหว่างวิธี Trilateration (3P method) ซึ่งใช้ Anchor 3 ตัว และวิธี Triangular (2P method) ซึ่งใช้ Anchor 2 ตัว

2) เทียบความแม่นของข้อมูลของ x, y ที่ได้หาจากวิธีอ่านค่าของ UWB ระหว่างวิธี ToF และวิธี RSSI

3) เพื่อประเมินความแม่นในการหา x, y ของระบบกล้อง CCTV หลังจากผ่านกระบวนการแก้ไขความบิดเบี้ยวของเลนส์ (Lens Distortion Correction)

4) เพื่อเปรียบเทียบความแม่นยำโดยรวมระหว่างระบบ UWB (ด้วยวิธีที่ดีที่สุด) และระบบ CCTV

ขั้นตอนและเงื่อนไข

1) กำหนดตำแหน่งทดสอบ กำหนดจุดทดสอบที่ทราบค่าพิกัดแน่นอน (Ground Truth) ไว้ 4 ตำแหน่งภายในพื้นที่ทดลองขนาด 4x8 เมตร ได้แก่พิกัด (1,1), (3,1), (1,7), และ (3,7) ดังรูป 3.8

2) กำหนดมุมทิศทางของ AGV ณ แต่ละตำแหน่งทดสอบ จะทำการวาง AGV ให้หันหน้าไปตามมุมทิศทาง 4 ทิศทาง คือ 0, 90, 180, และ 270 องศา เทียบกับแกน x เพื่อตรวจสอบว่าทิศทางของตัวรถมีผลต่อความแม่นในการวัด

3) เก็บข้อมูล ในแต่ละ x, y และแต่ละมุมทิศทาง จะทำการบันทึกข้อมูล x, y ที่อ่านได้จากระบบ UWB และ CCTV

4) การทดลองซ้ำ กระบวนการทั้งหมดจะถูกทำซ้ำจำนวน 3 ครั้งในทุกเงื่อนไข เพื่อให้ได้ข้อมูลที่นำไปวิเคราะห์

5) นำชุดข้อมูลที่รวบรวมได้ทั้งหมดมาประมวลผลเพื่อเทียบค่า Error ของแต่ละเทคนิคตามตั้งไว้

3.3.2 ทดสอบในสภาวะเคลื่อนที่ (Dynamic Test)

การทดสอบในสภาวะเคลื่อนที่เป็นขั้นตอนสำคัญของกระบวนการรวบรวมข้อมูล (Dataset Collection) เพื่อใช้ในการวิเคราะห์สมรรถนะของระบบภายใต้เงื่อนไขการทำงานจริง โดยมุ่งเน้นการประเมินพฤติกรรมของเซนเซอร์และการคาดคะเนตำแหน่งขณะเคลื่อนที่ ซึ่งไม่สามารถสังเกตได้ในสภาวะหยุดนิ่ง เช่น การเกิด ความคลาดเคลื่อนสะสม (Drift Error) อันเนื่องมาจากความต่อเนื่องของเส้นทาง ความเร็ว และการเปลี่ยนมุมทิศทาง

การทดสอบในส่วนนี้จึงแบ่งออก ได้แก่

1) ทดสอบเพื่อวิเคราะห์คุณลักษณะ

2) ทดสอบเพื่อรวบรวมข้อมูลตัวรถ AGV

3.3.2.1 ทดสอบเพื่อวิเคราะห์คุณลักษณะเส้นทางเส้นตรง

การทดสอบในส่วนนี้มุ่งเน้นไปที่การเก็บข้อมูลในสภาวะเส้นทางจริงของ AGV เพื่อวิเคราะห์คุณลักษณะพื้นฐานของแต่ละชนิด เช่น ความไว การตอบสนองต่อความเร่ง และความผิดพลาดเชิงมุมที่อาจเกิดขึ้น

ข้อมูลที่ได้จะถูกนำไปใช้ในการปรับปรุงแบบจำลองทางคณิตศาสตร์และขั้นตอนการหลอมรวมข้อมูล (Sensor Fusion) เพื่อให้ระบบสามารถคำนวณตำแหน่งได้อย่างแม่นยำภายใต้สภาวะจริง

วัตถุประสงค์

1) เพื่อรวบรวมชุดข้อมูลที่สมบูรณ์ซึ่งประกอบด้วยข้อมูลจากเซ็นเซอร์ทุกชนิด (UWB CCTV Encoder และ IMU) พร้อมกับเวลา (Timestamp) ที่ตรงกัน ภายใต้สภาวะควบคุม

2) เพื่อประเมินความแม่นยำ (Accuracy) และศึกษาลักษณะของ การระบุ x, y แต่ละเซ็นเซอร์แยกกัน โดยเปรียบเทียบกับเส้นทางอ้างอิงจริง (Ground Truth Path) ได้แก่

การหา x, y ด้วย UWB

การหา x, y ด้วย CCTV

การหา x, y ค่าตำแหน่งด้วย Dead Reckoning (จาก Encoder และ IMU)

3) เพื่อจัดทำชุดข้อมูลอ้างอิงพื้นฐาน (Baseline Dataset) ที่ใช้ในการประเมินสมรรถนะของเซนเซอร์แต่ละประเภท โดยชุดข้อมูลนี้มีความสำคัญต่อการวิเคราะห์ลักษณะการทำงานและข้อจำกัดเฉพาะของแต่ละเซนเซอร์ เพื่อใช้เป็นข้อมูลตั้งต้นในการพัฒนาและปรับปรุงขั้นตอนการหลอมรวมข้อมูลในภายหลัง

ขั้นตอนและเงื่อนไข

1) กำหนดเส้นทาง การทดลองจะใช้เส้นทางที่เป็น เส้นทางตรง (Linear Motion) เพื่อวิเคราะห์คุณลักษณะของเซ็นเซอร์

2) การควบคุมความเร็ว การทดลองจะดำเนินการโดยให้ AGV เคลื่อนที่ด้วยความเร็วเฉลี่ย 2 ระดับ คือ 0.27 เมตร/วินาที และ 0.4 เมตร/วินาที

3) การบันทึกข้อมูล ระหว่างที่ AGV เคลื่อนที่ ทำการบันทึกข้อมูลดิบทั้งหมดพร้อมกันอย่างต่อเนื่อง ซึ่งประกอบด้วย

ข้อมูลตำแหน่งจาก UWB (Position data from UWB)

ข้อมูลตำแหน่งจาก CCTV (Location data from CCTV)

ข้อมูลการหมุนของล้อจาก Encoder (Encoder data)

ข้อมูลทิศทางจาก IMU (direction data from the IMU)

4) คำนวณเบื้องต้น นำข้อมูลแต่ละชนิดมาคำนวณเส้นทางการเคลื่อนที่ (Path Trajectory) แยกตามแต่ละระบบเซ็นเซอร์

5) วิเคราะห์คุณลักษณะ นำเส้นทางที่คำนวณได้แต่ละระบบมาเปรียบเทียบกับเส้นทางอ้างอิงจริง (Ground Truth Path) เพื่อวิเคราะห์หาค่า RMSE และศึกษารูปแบบเฉพาะตัวของเซ็นเซอร์แต่ละชนิดในสภาวะเคลื่อนที่

3.3.2.2 ทดสอบเพื่อรวบรวมข้อมูลตัวรถ AGV สำหรับวิธีการหลอมรวมข้อมูล

การทดสอบในส่วนนี้ได้รับการออกแบบโดยมีวัตถุประสงค์เพื่อจัดเก็บและรวบรวมข้อมูล (Dataset) สำหรับนำไปใช้ในการพัฒนาและประเมินประสิทธิภาพของอัลกอริทึมการหลอมรวมข้อมูล ซึ่งจะนำเสนออย่างละเอียดในหัวข้อ 3.4 การพัฒนาอัลกอริทึมสำหรับการหลอมรวมข้อมูล ภายหลัง

การทดลองนี้จำลองสภาวะการเคลื่อนที่จริงของ AGV โดยเฉพาะในกรณีของระบบ Dead Reckoning ที่อาศัยข้อมูลจากเซนเซอร์ Encoder และ IMU ซึ่งมีแนวโน้มที่จะเกิดความคลาดเคลื่อนสะสม (Accumulated Drift Error) เมื่อเวลาผ่านไป ส่งผลให้ตำแหน่งที่คำนวณได้เบี่ยงเบนจากค่าจริง

ข้อมูลที่รวบรวมจากการทดลองนี้จึงถูกนำมาใช้ในการประเมินสมรรถนะของอัลกอริทึมการหลอมรวมข้อมูล (Sensor Fusion Algorithm) ที่ได้พัฒนาขึ้นในงานวิจัยนี้ เพื่อ

ทดสอบความสามารถในการลดข้อผิดพลาดสะสมและเพิ่มความแม่นยำของการระบุตำแหน่งในสภาวะการเคลื่อนที่จริงของ AGV

วัตถุประสงค์

1) เพื่อรวบรวมชุดข้อมูลที่สมบูรณ์ในเส้นทางซับซ้อนและมีการเลี้ยว ซึ่งเป็นเหตุให้เกิด Drift Error ในการคำนวณแบบ Dead Reckoning

2) ข้อมูลการทดสอบ (Test Dataset) ใช้เป็น ข้อมูลนำเข้า (Input) สำหรับวิธี Standard EKF และ Blended EKF หัวข้อถัดไป เพื่อเทียบการระบุตำแหน่งภายใต้สภาวะจริง

ขั้นตอนการดำเนินงาน

2.1) เส้นทางการเคลื่อนที่ AGV จะเคลื่อนที่ไปตามเส้นทางที่กำหนด ประกอบด้วย เส้นตรง (Straight Section) และ เลี้ยว/หมุน (Rotating Section) โดยมีลำดับ ดังนี้

เริ่มต้นที่พิกัด (x=0, y=7)

เคลื่อนที่เป็นเส้นตรงไปยังพิกัด (x=1, y=7) แล้วหยุด

ณ ตำแหน่งดังกล่าว ทำการ หมุนเปลี่ยนมุมทิศทาง

จากนั้น เส้นทางเป็นเส้นตรงอีกครั้ง และหยุดการเคลื่อนที่

2.2) การบันทึกข้อมูล: ทำการบันทึกข้อมูลจากเซ็นเซอร์ที่เกี่ยวข้องทั้งหมดที่จะใช้ในกระบวนการหลอมรวมข้อมูล ได้แก่ CCTV, Encoder, และ IMU อย่างต่อเนื่องตลอด

2.3) ข้อมูลที่ได้จากการทดลองทั้งหมดจะถูกจัดเก็บอย่างเป็นระบบ เพื่อใช้เป็นข้อมูลต้นทางในขั้นตอนการพัฒนา โครงร่างการทำงานของวิธีการหลอมรวมข้อมูล (Sensor Fusion Framework) ซึ่งจะนำเสนอในหัวข้อถัดไป โดยมีจุดประสงค์เพื่อใช้ในการวิเคราะห์และเปรียบเทียบประสิทธิภาพของวิธีแบบ Standard EKF และ Blended EKF อย่างเป็นลำดับขั้นตอน

3.4 การพัฒนากรอบการทำงานสำหรับการหลอมรวมข้อมูล (Sensor Fusion Framework)

งานวิจัยนี้ คือ การพัฒนากรอบการทำงาน (Framework) สำหรับการหลอมรวมข้อมูลจากเซ็นเซอร์ต้นทุนต่ำหลายชนิด โดยมีอัลกอริทึม Extended Kalman Filter (EKF) ในการประมาณค่าสถานะของ AGV กรอบการทำงานเพื่อเทียบประสิทธิภาพระหว่างการหลอมรวมข้อมูล คือ Standard EKF และ Blended EKF ดังนี้

3.4.1 คำนวณข้อมูลเบื้องต้น (Data Pre-processing)

ก่อนนำข้อมูลจากเซนเซอร์แต่ละประเภทเข้าสู่กระบวนการหลอมรวมข้อมูล จำเป็นต้องผ่านขั้นตอนการประมวลผลเบื้องต้น เพื่อปรับปรุงคุณภาพของข้อมูลให้มีความสอดคล้อง

ลดความคลาดเคลื่อน และเพิ่มความน่าเชื่อถือของผลการวัด ทั้งนี้กระบวนการดังกล่าวประกอบด้วยสองขั้นตอนหลัก ดังต่อไปนี้

3.4.1.1 การแก้ไขความเบี่ยงเบน (Bias Correction)

เพื่อลด error ของเซ็นเซอร์แต่ละตัว จะทำการประเมินและแก้ไขค่าความเบี่ยงเบนโดยการคำนวณค่าเบี่ยงเบนเฉลี่ยระหว่างค่าที่วัดได้จากเซ็นเซอร์ (S_t) กับค่าอ้างอิงจริง (GT_t) ตามสมการ

$$\bar{b}_s = \frac{1}{N}\sum_{t=1}^{N}(S_t - GT_t) \tag{3.1}$$

โดยที่ St คือค่าที่วัดได้จากเซ็นเซอร์ ณ เวลา t และ GT_t คือค่าอ้างอิงจากเส้นทางจริง GT ในเวลาเดียวกัน ค่า St′ ที่แก้ไขแล้วกำหนดโดย

$$S_t' = S_t - \bar{b}_s \tag{3.2}$$

กระบวนการนี้จะช่วยลดค่า RMSE ของข้อมูลดิบ

3.4.1.2 ถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting)

หลังจากแก้ไขความเบี่ยงเบนแล้ว ข้อมูลแต่ละตัวคำนวณน้ำหนักทางสถิติตามหลักการที่ว่า เซ็นเซอร์ที่มีค่าความแปรปรวน (σ^2) ต่ำกว่า ย่อมเชื่อถือสูงกว่าและควรมีน้ำหนัก (w_i) ที่มากกว่า โดยคำนวณตามสมการ

$$w_i = \frac{1}{\sigma_i^2} \tag{3.3}$$

จากนั้นจึงทำการปรับมาตรฐานน้ำหนัก (Normalize) เพื่อให้ผลรวมของน้ำหนักทั้งหมดเท่ากับ 1

$$w_i' = \frac{w_i}{\sum_{j=1}^{M} w_j} \tag{3.4}$$

โดยที่ M คือ จำนวนเซ็นเซอร์ทั้งหมด ค่าน้ำหนักที่ปรับมาตรฐานแล้วนี้ใช้ในวิธีการ ต่อไป

3.4.2 ตัวกรอง Standard EKF (Normal Series Fusion EKF)

ในขั้นตอนนี้ ตัวกรอง EKF จะใช้แบบจำลอง (Motion Model) ของรถ AGV ร่วมกับข้อมูลจากเซนเซอร์วัดการเคลื่อนที่ เช่น IMU และ Encoder เพื่อคาดการณ์สถานะของระบบ (เช่น ตำแหน่งและมุมทิศทาง) ในช่วงเวลาถัดไป ตัวกรองนี้ทำงานเป็นกระบวนการเชิงลำดับ (Sequential

Process) พร้อมทั้งประเมินค่าความไม่แน่นอนของการประมาณสถานะในขั้นตอนนี้ด้วย และทำวนซ้ำ 2 ขั้นตอนได้แก่ ขั้นตอนการทำนาย (Prediction Step) และขั้นตอนการปรับปรุง (Update Step) ได้ทำไว้ที่บทที่ 2 ดังนี้

3.4.2.1 ขั้นตอนการทำนาย (Prediction Step)

ในขั้นตอนนี้ ตัวกรอง EKF จะใช้แบบจำลอง (Motion Model) ของรถ AGV ร่วมกับข้อมูลควบคุม (u_k) จากเซนเซอร์วัดการเคลื่อนที่ เช่น Dead Reckoning (IMU และ Encoder) เพื่อคาดการณ์สถานะของระบบ (เช่น ตำแหน่งและทิศมุมทาง) ในช่วงเวลาถัดไป พร้อมทั้งประเมินค่าความไม่แน่นอนของการทำนายที่เพิ่มขึ้น

3.4.2.2 ขั้นตอนการปรับปรุง (Update Step)

เมื่อวัด (z_k) จากเซ็นเซอร์ภายนอก (ในที่นี้คือกล้อง CCTV) เข้ามา ตัวกรอง EKF จะใช้ข้อมูลดังกล่าวเพื่อ จะปรับค่าการประมาณสถานะให้ใกล้เคียงกับค่าจริงมากขึ้น โดยคำนวณค่า Kalman Gain เพื่อถ่วงน้ำหนักระหว่างค่าที่ได้จากแบบจำลองการเคลื่อนที่และค่าที่ได้จากการวัดจริง แล้วนำมาปรับแก้สถานะของ AGV ให้มีความแม่นยำสูงขึ้น

ในรูปแบบ EKF มาตรฐานนี้ การปรับปรุงค่าจะเกิดขึ้น แบบลำดับ (Sequentially) กล่าวคือ จะใช้ข้อมูลจากการวัดของแต่ละตัวในแต่ละรอบของตัวกรอง

3.4.3 การออกแบบตัวกรอง Blended EKF (นวัตกรรมหลักของงานวิจัย)

Blended EKF ถูกพัฒนาขึ้นเพื่อแก้ไขปัญหาเส้นทางโดยประมาณที่ไม่ราบรื่น (Unsmooth Trajectory) ซึ่งเกิดจากการที่ Standard EKF ต้องสลับการอัพเดตระหว่างข้อมูลจากกล้อง (CAM) ที่รบกวนสูง และ Dead Reckoning ที่มี error สะสม

แนวคิด คือ การ ผสม (Blend) ข้อมูลการวัดจากกล้อง (z_{CAM}) และข้อมูลจาก Dead Reckoning (z_{DR}) เข้าด้วยกัน นำเข้าในขั้นตอนการปรับปรุง (Update Step) ของ EKF การผสมนี้จะสร้างข้อมูลการวัดใหม่ (z_{blend}) ซึ่งจะช่วยลดการสั้นของค่าประมาณและทำให้ได้เส้นทางที่ราบรื่นยิ่งขึ้น โดยคำนวณตามสมการ

$$z_{blend} = \alpha_{CAM} z_{CAM} + \alpha_{dr} z_{dr} \tag{3.5}$$

โดยที่ α_{CAM} และ α_{dr} คือ ค่าน้ำหนักที่กำหนดให้กับข้อมูลจากกล้องและ Dead Reckoning ตามลำดับซึ่งต้องเป็นไปตามเงื่อนไข

$$\alpha_{CAM} + \alpha_{dr} = 1 \tag{3.6}$$

ค่าความไม่แน่นอนของข้อมูลที่ผสมแล้ว R_{blend} สามารถประมาณได้จาก

$$R_{blend} = \alpha_{CAM}^2 R_{CAM} + \alpha_{dr}^2 R_{dr} \quad (3.7)$$

จากนั้นจึงนำค่า z_{blend} และ R_{blend} ไปใช้ในการปรับปรุง (Update Step) ของ EKF การเลือกค่าน้ำหนัก α สามารถทำได้โดยการทดลองหรือปรับจูนการใช้งานจริง

เหตุผลและแรงจูงใจในการพัฒนา Blended EKF

การผสมข้อมูลก่อนเข้าสู่ขั้นตอนการอัปเดตของ EKF มีความจำเป็นเนื่องจากข้อมูลจากกล้อง (CAM) มักมีความผันผวนสูงจากปัจจัยด้านแสง เงา การบดบังบางส่วน และความล่าช้าในการประมวลผลภาพ ส่งผลให้เกิดการสั่นของค่าตำแหน่ง (jitter) และความไม่ต่อเนื่องของเส้นทางที่ประมาณค่า ในขณะที่ข้อมูลจาก Dead Reckoning (DR) แม้จะมีความต่อเนื่องดี แต่มีข้อจำกัดสำคัญคือความผิดพลาดสะสม (drift) ที่เพิ่มขึ้นตามระยะเวลาและระยะทาง การใช้ Standard EKF ซึ่งอัปเดตด้วยข้อมูลจากเซนเซอร์เพียงชนิดเดียวในแต่ละครั้ง ทำให้เส้นทางที่ได้มีความกระโดด (jump) หรือมีการแกว่งในช่วงที่ CAM มี noise สูง ดังนั้น Blended EKF จึงถูกพัฒนาขึ้นเพื่อลดผลกระทบของ noise ที่เกิดขึ้นเฉพาะช่วง และเพิ่มความราบรื่นของข้อมูลก่อนเข้าสู่กระบวนการ Estimate

ความแตกต่างระหว่าง Standard EKF และ Blended EKF

Standard EKF ทำการอัปเดตสถานะด้วยข้อมูลการวัดจากแหล่งเดียว เช่น CAM หรือ DR ตามที่เข้ามาในเวลานั้น ซึ่งอาจทำให้ผลการประมาณตำแหน่งตอบสนองต่อ noise มากเกินไป ในขณะที่ Blended EKF จะสร้างข้อมูลการวัดใหม่ z_{blend} จากการรวมข้อมูล CAM และ DR ก่อนการอัปเดต ทำให้ข้อมูลที่ใช้แก้ไขสถานะมีลักษณะสมดุลระหว่างความต่อเนื่อง (จาก DR) และความแม่นยำ (จาก CAM) ส่งผลให้เส้นทางที่ประมาณมีความราบรื่นและสม่ำเสมอกว่า Standard EKF อย่างชัดเจน

การเลือกค่าน้ำหนัก αตามความน่าเชื่อถือของเซนเซอร์

ค่าของ α_{CAM}และ α_{DR}สามารถกำหนดตามความน่าเชื่อถือของข้อมูลที่เกิดขึ้นจริง โดยอ้างอิงค่าความแปรปรวน (Variance) ของข้อมูลหลังการทำ Bias Correction ซึ่งสะท้อนระดับความเสถียรของข้อมูลเซนเซอร์แต่ละชนิด เซนเซอร์ที่มีความแปรปรวนต่ำจะได้รับค่าน้ำหนักสูงกว่า แนวทางนี้ช่วยให้ Blended EKF ปรับระดับการเชื่อข้อมูลของแต่ละเซนเซอร์ให้เหมาะสมกับสภาวะการทำงานจริง และช่วยลดความคลาดเคลื่อนโดยรวมของระบบ

รูปแบบเมทริกซ์ของข้อมูลที่ผสมแล้ว (ใช้ในการอัปเดต)

เพื่อให้ Blended EKF สามารถทำงานร่วมกับโครงสร้างเดิมของ Measurement Model ได้อย่างสมบูรณ์ สามารถเขียนข้อมูลที่ผสมแล้วในรูปเมทริกซ์ดังนี้

$$z_{blend} = \begin{bmatrix} x_{blend} \\ y_{blend} \end{bmatrix} = \alpha_{CAM} \begin{bmatrix} x_{CAM} \\ y_{CAM} \end{bmatrix} + \alpha_{DR} \begin{bmatrix} x_{DR} \\ y_{DR} \end{bmatrix} \quad (3.8)$$

และความเชื่อมั่นของค่าที่ผสมแล้วเป็น

$$R_{blend} = \alpha_{CAM}^2 R_{CAM} + \alpha_{DR}^2 R_{DR} \quad (3.9)$$

ซึ่งทำให้สามารถแทนที่ค่าการวัดเดิมในสมการอัปเดตของ EKF ได้โดยตรง

ประโยชน์เชิงพฤติกรรมของ Blended EKF

จากการทดลอง พบว่าเส้นทางที่ได้จาก Blended EKF มีความราบรื่นและลดการสั่นของเส้นทางได้ดีกว่า Standard EKF โดยเฉพาะในช่วงเลี้ยวโค้งหรือช่วงที่กล้องมี noise สูง การผสมข้อมูลก่อนเข้าสู่ EKF ยังช่วยลดความผิดพลาดที่เกิดจาก drift ของ DR และลดการแกว่งของค่าตำแหน่งจาก CAM ทำให้ผลการประมาณตำแหน่งมีความต่อเนื่องและเสถียรมากขึ้น เหมาะสำหรับงานระบุตำแหน่ง AGV ที่ใช้เซนเซอร์ต้นทุนต่ำในสภาวะแวดล้อมจริง

3.5 การค้นหาค่าสัมประสิทธิ์การผสม (αx, αy) ด้วย Optuna และ TPE Sampler

ค่าสัมประสิทธิ์การผสม α_xและ α_yเป็นองค์ประกอบสำคัญที่กำหนดความสัมพันธ์ระหว่างข้อมูลจากกล้อง (CAM) และข้อมูลจากระบบ Dead Reckoning ในขั้นตอนการสร้างค่าการวัดที่ผสม (Blended Measurement) สำหรับ Blended EKF การเลือกค่าที่เหมาะสมทำให้ตัวกรองสามารถลดความคลาดเคลื่อนของตำแหน่งที่ประมาณได้และช่วยให้เส้นทางมีความต่อเนื่องมากยิ่งขึ้น ในทางกลับกัน หากกำหนดค่าไม่เหมาะสมอาจทำให้ตำแหน่งที่คำนวณมีความผิดพลาดเพิ่มขึ้น โดยเฉพาะในสภาพแวดล้อมที่สัญญาณภาพจากกล้องมีความแปรปรวนสูง

งานวิจัยนี้จึงใช้ **Optuna** ซึ่งเป็นเฟรมเวิร์กสำหรับค้นหาค่าพารามิเตอร์เชิงตัวเลขโดยอัตโนมัติ และออกแบบมาสำหรับงานที่ต้องการเพิ่มประสิทธิภาพเชิง Bayesian Optimization (Akiba et al., 2019) Optuna ใช้กลไก **Tree-structured Parzen Estimator (TPE)** ในการเสนอค่าพารามิเตอร์ใหม่ โดยออกแบบให้ค้นหาบริเวณที่มีแนวโน้มว่าจะให้ค่าผลลัพธ์ที่ดีกว่าอย่างเป็นระบบ อัลกอริทึมนี้เป็นแนวคิดที่ได้รับการพัฒนาโดย (Bergstra et al., 2011) และได้รับการพิสูจน์ว่าให้ผลดีกว่าวิธีสุ่มทั่วไปและวิธีการไล่ค่าทีละจุดแบบดั้งเดิม

3.5.1 ฟังก์ชันวัตถุประสงค์ (Objective Function)

การประเมินความเหมาะสมของชุดพารามิเตอร์ใช้ตัวชี้วัด RMSE_{XY} ซึ่งพิจารณาความคลาดเคลื่อนเชิงตำแหน่งบนแกน X และ Y ระหว่างผลลัพธ์จาก Blended EKF และตำแหน่งจริง (Ground Truth) โดยกำหนดปัญหาให้อยู่ในรูปแบบการหาค่าต่ำสุด ดังนี้

$$\min_{\alpha_x,\alpha_y} \; RMSE_{XY}(\alpha_x, \alpha_y) \tag{3.10}$$

Optuna จะทดลองค่าพารามิเตอร์แต่ละคู่ แล้วคำนวณค่าการวัดผสมและรัน Blended EKF ตลอดช่วงเวลาของการทดลอง จากนั้นจึงส่งค่าความคลาดเคลื่อน RMSE ให้ตัวค้นหาพารามิเตอร์เพื่อใช้ปรับปรุงโมเดลของ Bayesian Optimization ต่อไปตามหลักการของ (Akiba et al., 2019)

3.5.2 หลักการของ TPE Sampler

TPE เป็นอัลกอริทึมที่สร้างแบบจำลองการกระจายตัวของพารามิเตอร์จากผลลัพธ์ที่เคยทดลองมาแล้ว โดยแบ่งข้อมูลออกเป็นสองกลุ่ม ได้แก่

1) **กลุ่มค่าที่ให้ผลลัพธ์ดี** คือค่าที่ให้ค่า RMSE ต่ำกว่าเกณฑ์
2) **กลุ่มค่าที่ให้ผลลัพธ์ไม่ดี** คือค่าที่ RMSE สูงกว่าเกณฑ์

โดยพิจารณาตามแนวคิดของ (Bergstra et al., 2011)
จากนั้น TPE จะสร้างแบบจำลอง density สองฟังก์ชัน คือ

$l(x)$: ความน่าจะเป็นของพารามิเตอร์ที่อยู่ในกลุ่มผลลัพธ์ที่ดี

$g(x)$: ความน่าจะเป็นของพารามิเตอร์ที่อยู่ในกลุ่มผลลัพธ์ที่ไม่ดี

โดย $x = (\alpha_x, \alpha_y)$
อัลกอริทึมจะเลือกค่าพารามิเตอร์ใหม่จากบริเวณที่ทำให้อัตราส่วน

$$\frac{l(x)}{g(x)} \tag{3.11}$$

มีค่าสูงที่สุด ซึ่งหมายถึงบริเวณที่มีแนวโน้มให้ค่าความผิดพลาดต่ำกว่าบริเวณอื่น กลไกนี้ช่วยให้ TPE สำรวจพื้นที่การค้นหาเฉพาะส่วนที่มีความหวังสูง ช่วยลดจำนวนรอบที่ต้องทดลองลงอย่างมีนัยสำคัญ และเข้ากับหลักการของ Bayesian Optimization ที่ (Snoek et al., 2012) ได้เสนอไว้ในงานด้านการเพิ่มประสิทธิภาพพารามิเตอร์เชิงสถิติ

3.5.3 การตั้งค่าการค้นหา

งานวิจัยนี้กำหนดเงื่อนไขในการค้นหาพารามิเตอร์ดังนี้

ช่วงของพารามิเตอร์

$$\alpha_x, \alpha_y \in [0.0, 1.0] \tag{3.12}$$

วัตถุประสงค์การค้นหา: ลดค่า RMSE (minimize)

จำนวนรอบทดลอง: 100–300 รอบ เพื่อให้ TPE มีข้อมูลเพียงพอในการปรับปรุงแบบจำลอง

ขั้นตอนของ Optuna ในแต่ละรอบประกอบด้วย

1) เลือกค่าพารามิเตอร์ (α_x, α_y)จากแบบจำลองของ TPE
2) คำนวณค่าการวัดผสม

$$z_{\text{blend}} = \alpha_x z_{\text{CAM}} + \alpha_y z_{\text{DR}} \tag{3.13}$$

3) รัน Blended EKF ตลอดเส้นทางทดลอง
4) คำนวณ RMSE_{XY} และส่งให้ Optuna
5) แบบจำลองของ TPE ถูกอัปเดตด้วยผลลัพธ์ใหม่
6) ทำซ้ำจนกว่าจะครบจำนวนรอบที่กำหนด

เมื่อกระบวนการค้นหาสิ้นสุดลง ระบบจะรายงานค่าที่เหมาะสมที่สุด

$$(\alpha_x^{\backslash *}, \alpha_y^{\backslash *}) \tag{3.14}$$

ซึ่งถูกนำไปใช้กับ Blended EKF

3.6 เกณฑ์การประเมินประสิทธิภาพ (Performance Evaluation)

เพื่อทำการเทียบประสิทธิภาพของแนวทางการหา x, y แต่ละแบบ ทั้งในส่วนประเมินเซ็นเซอร์และการประเมินวิธีการหลอมรวมข้อมูล ซึ่งได้กำหนดเชิงปริมาณ เพื่อให้สามารถสรุปผลได้อย่างน่าเชื่อถือ

เกณฑ์หลักที่ใช้ในการวัดความแม่นยำของระบบ คือ ค่าความคลาดเคลื่อนรากของค่าเฉลี่ยกำลังสอง (Root Mean Square Error: RMSE) ซึ่งเป็นตัวชี้วัดทางสถิติที่นิยมใช้เพื่อประเมินความแตกต่างระหว่างค่าที่ระบบคาดคะเนได้กับค่าจริง โดย RMSE

3.6.1 นิยามและการคำนวณค่า RMSE

ค่า RMSE จะถูกคำนวณ ระหว่างตำแหน่งจริงบนเส้นทางอ้างอิง (Ground Truth) กับตำแหน่งที่ประมาณได้จากวิธี ณ ทุกๆ จุดเวลาตลอดเส้นทางการทดลอง ค่า RMSE ที่น้อยกว่าจะหมายถึงวิธีนั้นมีความแม่นสูงกว่า มีสมการในการคำนวณดังนี้

$$RMSE_{xy} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2)} \quad (3.15)$$

$$RMSE_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((x_i - \hat{x}_i)^2)} \quad (3.16)$$

$$RMSE_y = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((y_i - \hat{y}_i)^2)} \quad (3.17)$$

โดยที่

N คือ จำนวนจุดข้อมูลทั้งหมด

$(x_i,\ y_i)$ คือ พิกัดของตำแหน่งอ้างอิงจริง (Ground Truth) ณ เวลาที่ i

$(\hat{x}_i,\ \hat{y}_i$ คือ พิกัดของตำแหน่งที่ประมาณได้จากวิธี ณ เวลาที่ i

$RMSE_{xy}$ $RMSE_x$ $RMSE_y$ คือ Root Mean Square Error ของทั้ง 2 แกน (XและY) ของ X ของ Y ตามลำดับ

3.6.2 นิยามและการคำนวณค่า RMSE

ค่า RMSE จะถูกนำมาเทียบประสิทธิภาพดังนี้

1) การประเมินประสิทธิภาพของเซ็นเซอร์

ข้อมูล "การทดลองเส้นตรง" (หัวข้อ 3.3.2.1) มาคำนวณค่า RMSE ของเส้นทางที่ได้จากระบบระบุตำแหน่งแต่ละระบบแยกกัน (UWB, CCTV, และ Dead Reckoning) โดยเทียบกับเส้นทางอ้างอิงจริง ในการประเมินประสิทธิภาพตั้งต้นของเซ็นเซอร์แต่ละชนิด

2) การประเมินวิธีการหลอมรวมข้อมูล

ข้อมูล "การทดลองการหลอมรวมข้อมูล" (หัวข้อ 3.3.2.2) มาคำนวณด้วยวิธี Standard EKF และ Blended EKF

คำนวณค่า RMSE ของเส้นทางโดยประมาณ (Estimated Path) ที่ได้จากวิธีทั้งสอง โดยเทียบกับเส้นทางอ้างอิงจริง

เทียบค่า RMSE เพื่อสรุปว่าวิธีรูปแบบใด (Standard EKF หรือ Blended EKF) สามารถให้ผลลัพธ์การหา x, y ที่มีความแม่นสูงกว่าภายใต้สภาวะการทดลอง

3.7 สรุป

บทที่ 3 ได้นำเสนอระเบียบวิธีวิจัยอย่างเป็นระบบ ตั้งแต่โครงสร้างสถาปัตยกรรมแบบควบคุมจากส่วนกลางด้วยระบบกล้อง การกำหนดอุปกรณ์และซอฟต์แวร์ที่ใช้ (CCTV, UWB, Encoder, IMU, MQTT และ Python) ไปจนถึงการออกแบบการทดลองเพื่อเก็บข้อมูลทั้งแบบหยุดนิ่งและแบบเคลื่อนที่ในพื้นที่ทดสอบขนาด 4×8 เมตร นอกจากนี้ ยังได้อธิบายกรอบการทำงานสำหรับการหลอมรวมข้อมูล โดยเริ่มจากการปรับปรุงคุณภาพข้อมูลด้วย Bias Correction และ Inverse-Variance Weighting จากนั้นพัฒนาและเปรียบเทียบอัลกอริทึม Standard EKF กับ Blended EKF ซึ่งเป็นแนวทางหลักของงานวิจัย พร้อมทั้งนำ Optuna (TPE Sampler) มาค้นหาค่าน้ำหนักการผสมที่เหมาะสม สุดท้ายกำหนดเกณฑ์การประเมินผลด้วยค่า RMSE เพื่อใช้ยืนยันประสิทธิภาพของระบบและวิธีที่พัฒนาขึ้นในบทถัดไปอย่างเป็นรูปธรรม

บทที่ 4
ผลการวิจัยและอภิปรายผล

บทนี้นำเสนอผลลัพธ์ที่ได้จากการดำเนินการทดลองตามระเบียบวิธีวิจัยที่อธิบายไว้ในบทที่ 3 โดยเริ่มจากการประเมินประสิทธิภาพของเซนเซอร์แต่ละชนิดแยกกัน ตามด้วยผลการปรับปรุงคุณภาพข้อมูล และผลการประมวลผลด้วยอัลกอริทึมการหลอมรวมข้อมูลทั้งสองแนวทางที่พัฒนา สุดท้ายเป็นการวิเคราะห์และเปรียบเทียบผล เพื่อสรุปองค์ความรู้และข้อค้นพบสำคัญ

4.1 ทดสอบประสิทธิภาพรายเซ็นเซอร์ (ก่อนการหลอมรวม)

เพื่อประเมินคุณลักษณะและประสิทธิภาพพื้นฐานของเซ็นเซอร์แต่ละชนิด ก่อนที่จะนำไปประยุกต์ใช้ในกระบวนการหลอมรวมข้อมูล (Data Fusion) ได้มีการออกแบบการทดลองแบ่ง คือ การทดสอบในสภาวะหยุดนิ่ง เพื่อหาค่าพื้นฐาน การปรับเทียบ และวิธีการวัดผลที่เหมาะสมที่สุด และการทดลองแบบพลวัต เพื่อประเมินประสิทธิภาพของเซ็นเซอร์ในขณะที่ยานพาหนะนำทางอัตโนมัติ (AGV) กำลังเคลื่อนที่จริง

4.1.1 การทดสอบในสภาวะหยุดนิ่ง (Static Test)

การทดลองเพื่อคัดเลือกวิธีและคำนวณ x, y ที่ดีที่สุดสำหรับ UWB และเพื่อเปรียบเทียบความแม่นยำตั้งต้นระหว่าง UWB และ CCTV ในสภาวะหยุดนิ่ง โดยทำการทดลอง ณ ตำแหน่งที่กำหนดไว้ 4 จุด และให้ AGV หันหน้าไป 4 ทิศทาง ทำการทดลองซ้ำ 3 ครั้ง ดังตาราง 4.1

1) เทียบสัญญาณ UWB (ToF เทียบกับ RSSI) และวิธีการคำนวณ เทียบการวัดระยะทางด้วยสัญญาณ 2 รูปแบบ คือ ToF และ RSSI ร่วมกับวิธีการคำนวณ x, y 2 แบบ คือ Trilateration (ใช้ Anchor 3 จุด) และ Triangular (ใช้ Anchor 2 จุด) พบว่า

สัญญาณ RSSI มีค่า error สูงมาก โดยมีค่า error สูงสุดในแกน X ถึง 9.2825 เมตร และในแกน Y สูงถึง 7.6216 เมตร ซึ่งไม่เสถียร

สัญญาณ ToF มีความแม่นสูงกว่า เทียบกันแล้ว ToF ให้ค่า error ต่ำกว่า RSSI มาก จึงเลือกวิธีการแบบ ToF

2) เทียบวิธีคำนวณตำแหน่ง UWB (Trilateration เทียบกับ Triangular) เมื่อใช้สัญญาณ ToF พบว่าวิธีคำนวณ Trilateration (3-point) ให้ค่า error สูงสุดในแกน X ที่ 1.4464 เมตร และแกน Y ที่ 1.0464 เมตร ซึ่งมีความแม่นมากกว่าวิธี Triangular (2-point) จึงได้เลือกใช้ Trilateration

3) เทียบความแม่นระหว่าง UWB และ CCTV สำหรับ UWB แล้วใช้ ToF และ Trilateration เทียบกับระบบ CCTV ปรากฏว่า CCTV ให้ความแม่นสูงกว่า ดังตาราง 2 โดย CCTV มีค่า error สูงสุดในแกน X เพียง 0.087 เมตร และในแกน Y ที่ 0.214 เมตร

ตารางที่ 4.1 ตารางการทดสอบ error แบบหยุดนิ่ง

Type	Reading	Method	x (m)	y (m)
UWB	RSSI	Trilateration	9.2825 (Max error)	—
	RSSI	Triangular	—	7.6216(Max error)
	ToF	Triangular	1.4828	1.8054
	ToF	Trilateration	1.4464	1.0464
CCTV	—	—	0.087	0.214

สรุปผลการทดสอบในสภาวะหยุดนิ่ง ระบบ CCTV มีความแม่นในการหา x,y ในสภาวะหยุดนิ่งสูงที่สุด และสำหรับ UWB ควรเลือกใช้การวัดระยะด้วยสัญญาณ ToF กับคำนวณ Trilateration เลือก Anchor ที่ใกล้รถ AGV

4.1.2 ทดลองแบบพลวัต (Dynamic Test)

การทดสอบในสภาวะเคลื่อนที่จัดทำขึ้นเพื่อประเมินประสิทธิภาพของเซนเซอร์ภายใต้เงื่อนไขการใช้งานจริง โดยได้ออกแบบรูปแบบการทดลองเชิงพลวัตจำนวนสองลักษณะ ซึ่งแต่ละลักษณะมีรูปแบบการเคลื่อนที่และวัตถุประสงค์ในการเก็บข้อมูลที่แตกต่างกัน เพื่อใช้วิเคราะห์ความแม่น ความคงที่ และความเสถียรของสัญญาณที่ได้จากเซนเซอร์ในสถานการณ์ที่ใกล้เคียงกับการทำงานจริงของรถ AGV

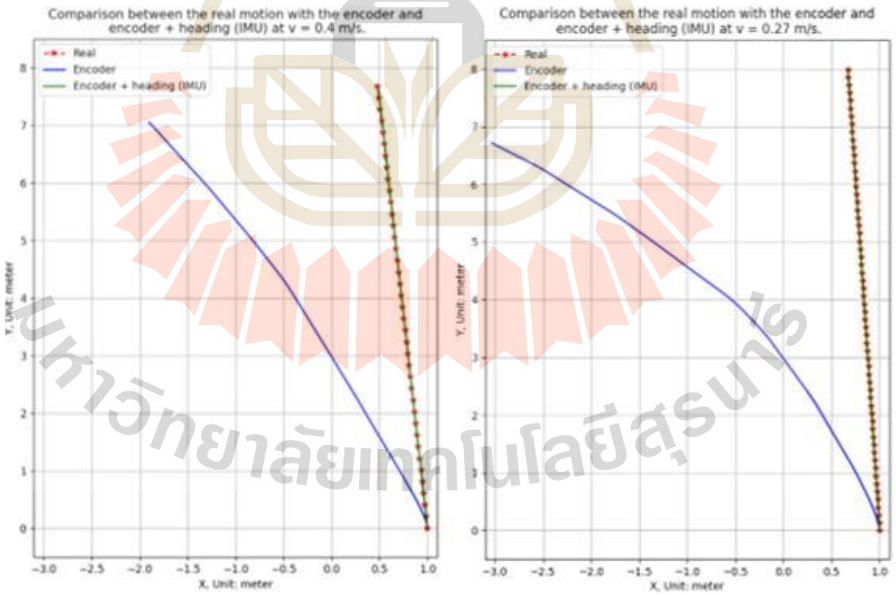
4.1.2.1 การทดลองที่ 1 เส้นตรง

การทดลองมีวัตถุประสงค์หลักเพื่อ ประเมินและเทียบความแม่น ของระบบระบุตำแหน่งแต่ละชนิดแบบแยกส่วน อันได้แก่ UWB CCTV และ Dead Reckoning (ซึ่งเป็นการทำงาน Encoder และ IMU) ภายใต้การเคลื่อนที่ของ AGV ในแนวเส้นตรง การทดลองแบ่งออกเป็นสองลักษณะ โดยกำหนดความเร็วเฉลี่ยของ AGV ที่ 0.4 m/s และ 0.27 m/s ตามลำดับ ในการเก็บข้อมูล บันทึกต่อเนื่องและทันท่วงที เซ็นเซอร์ UWB Encoder และ IMU ถูกตั้งค่าให้บันทึกข้อมูลทุกๆ ประมาณ 0.1 วินาที ในขณะที่ระบบ CCTV ซึ่งทำงานผ่านการประมวลผลภาพ จะบันทึกค่าตำแหน่งล่าสุดที่คำนวณได้ ณ เวลานั้นๆ

ผลการทดลอง พบว่า ข้อมูล x, y ที่ได้จากเซ็นเซอร์แต่ละตัวถูกนำมาเทียบกับ x, y อ้างอิงจริง (Ground Truth) เพื่อคำนวณหาค่า RMSE ซึ่งเป็นตัวชี้เพื่อประเมินความแม่นเชิงปริมาณ พบว่า ระบบ Dead Reckoning (Encoder + IMU) มีความแม่นยำสูงสุดอย่างมีนัยสำคัญ ดังตาราง 4.2

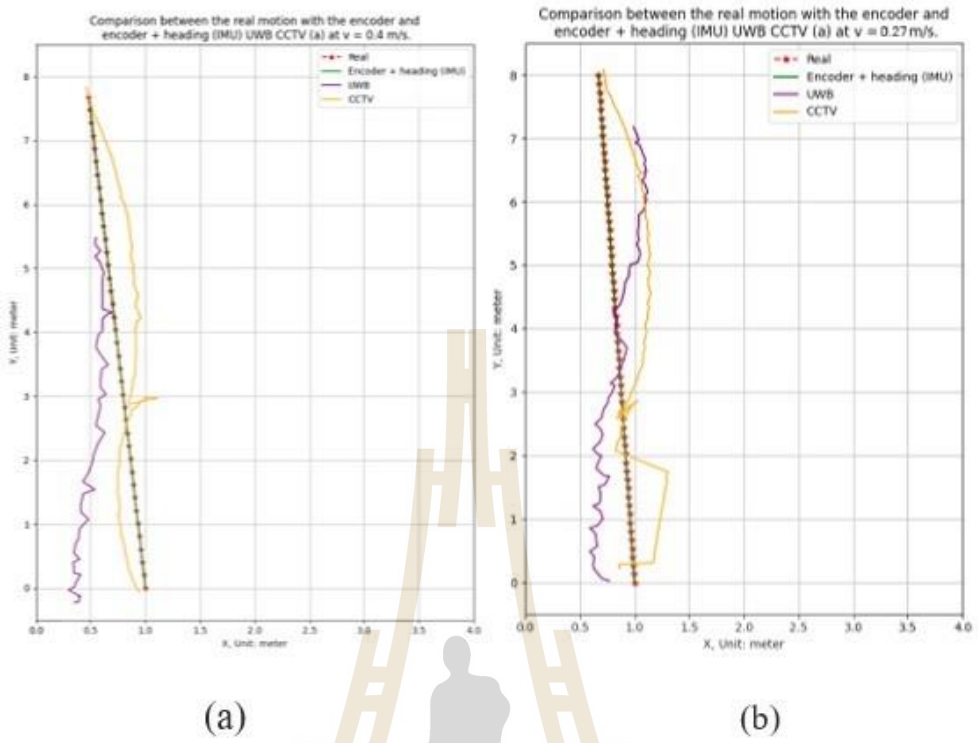
ตารางที่ 4.2 ตารางค่า RMSE การทดลองที่ 1 เส้นตรง

Type	Speed	Max Error (m)	RMSE	RMSE (x, y)	RMSE(x)	RMSE(y)
encoder	0.4 (m/s)	0.0805	0.1226	0.0526	0.0707	0.0881
Encoder + IMU		0.0101	0.048	0.0035	0.0731	0.0732
UWB		0.6562	2.2	0.3317	1.1858	1.2313
CCTV		0.3449	0.7537	0.1769	0.3951	0.4329
encoder	0.27 (m/s)	0.0979	0.107	0.0637	0.0642	0.0904
Encoder + IMU		0.0034	0.1086	0.0021	0.0678	0.0678
UWB		0.4053	0.82	0.2591	0.5156	0.577
CCTV		0.3725	0.4794	0.2414	0.2628	0.3568



(a) (b)

รูปที่ 4.1 กราฟเทียบเส้นทางจริงกับเส้นทาง Encoder และ Encoder + IMU ที่ความเร็ว 0.4 m/s และ 0.27 m/s



(a) (b)

รูปที่ 4.2 กราฟเทียบเส้นทางจริงกับเส้นทาง Encoder + IMU UWB และ CCTV ที่ความเร็ว 0.4 m/s และ 0.27 m/s.

รูปที่ 4.1 และ 4.2 เมื่อเทียบการใช้ Encoder อย่างเดียว กับ Encoder + IMU จะเห็นว่า IMU ช่วยลดค่า RMSE รวมลง ซึ่ง IMU ช่วยแก้ไขมุมทิศทางและลด error สะสม (Drift Error) ได้ ดังรูป 4.1 เส้นทางใกล้กับค่าจริง และ รูป 4.2 เส้นทาง Encoder + IMU (สีเขียว) มีความแม่นยำและเกาะกลุ่มกับเส้นทางจริงมากที่สุด ตามมาด้วย CCTV (สีเหลือง) ที่มีความแม่นยำรองลงมา และ UWB (สีม่วง) ที่แสดงการเบี่ยงเบนออกจากเส้นทางจริงมากที่สุด ซึ่งสอดคล้องกับค่า RMSE ที่สูงที่สุดในตารางที่ 4.2 โดยเฉพาะที่ความเร็ว 0.4 m/s การหลอมรวมข้อมูลระหว่าง Encoder และ IMU เป็นวิธีการหา x, y ที่มีประสิทธิภาพ เชื่อถือ ในการทดลองนี้ และต่อไปจะทำการตัด UWB ที่มีการเบี่ยงเบนสูงออกจากการทดสอง

4.1.2.2 การทดลองที่ 2 เส้นตรงและเลี้ยว

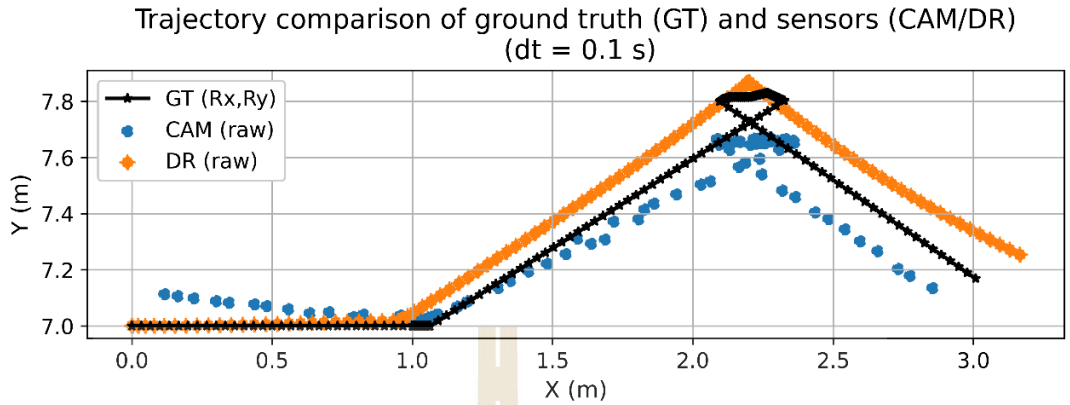
การทดลองนี้วิเคราะห์ของข้อมูลดิบ (Raw Data) จาก Camera (CCTV) และ Dead Reckoning (Encoder + IMU) ขณะรถ AGV เคลื่อนที่บนเส้นทางที่มียุ่งยากขึ้น ซึ่งประกอบด้วย ทั้งส่วนที่เป็นเส้นตรงและส่วนที่เป็นเส้นโค้ง โดยการทดลองดำเนินภายใต้สองสภาวะของอัตราการเก็บตัวอย่างข้อมูล (Sampling Interval, dt) ที่แตกต่างกัน ได้แก่ (1) dt = 0.1 วินาที โดยข้อมูลจาก DR ถูกเก็บทุก ๆ 0.1 วินาที และข้อมูลจากกล้อง CCTV ถูกบันทึกจากการส่งภาพ

ล่าสุด ซึ่งมีอัตราการเก็บประมาณ 0.1–0.2 วินาที และ (2) dt = 0.2 วินาที ซึ่งทั้ง DR และ CCTV ถูกตั้งค่าให้เก็บข้อมูลในอัตราเดียวกัน การวิเคราะห์ก่อนนำข้อมูลไปผ่านกระบวนการปรับปรุงคุณภาพเพิ่มเติม ผลการคำนวณค่า RMSE ตารางที่ 4.3 ซึ่งสะท้อนถึงระดับ error ของข้อมูลดิบภายใต้สภาวะเส้นทางลักษณะเส้นตรงและเลี้ยว

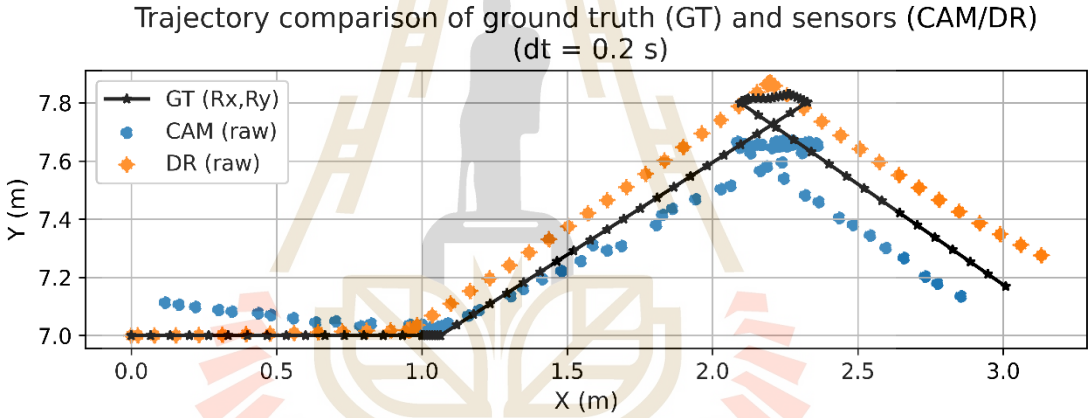
ตารางที่ 4.3 สรุปค่า RMSE ของข้อมูลดิบ (การทดลองที่ 2 เส้นตรงและเลี้ยว)

dt(s)	Sensor	RMSE(X)	RMSE(Y)	RMSE (X,Y)	RMSE (rad)
0.1	CCTV	0.06650	0.1002	0.1203	
	DR	0.0934	0.0503	0.1061	
	IMU heading				1.1959
0.2	CCTV	0.0601	0.0993	0.1161	
	DR	0.0898	0.0561	0.1059	
	IMU heading				1.2039

ตารางที่ 4.3 ข้อมูลดิบทั้งสองชนิดมีค่า error โดยค่า RMSE ของ DR ในแกน X จะสูงกว่า CAM แต่ในแกน Y จะต่ำกว่า ซึ่งแสดงถึงลักษณะ error ที่แตกต่างกันของเซ็นเซอร์แต่ละชนิด เพื่อให้เห็นภาพ error ของข้อมูลดิบ นำข้อมูลมาพล็อตเป็นกราฟเทียบเส้นทางจริง (Ground Truth) กับเส้นทางที่ได้จากข้อมูลดิบของ CCTV และ DR ดังรูป 4.3 และ 4.4 เส้นทางที่ได้จากข้อมูลดิบของทั้ง CCTV และ DR มีการเบี่ยงจากเส้นทางจริง ช่วงเลี้ยวโค้ง จะเห็นการเบี่ยงเบน รูป 4.3 และ 4.4 แสดงให้เห็นถึง error ที่มีอยู่ในข้อมูลดิบ ซึ่งสอดคล้องกับค่า RMSE ในตาราง 4.3 การสร้าง ข้อมูลพื้นฐาน (Baseline) ของประสิทธิภาพเซ็นเซอร์บนเส้นทางที่ยุ่งยาก และนำขั้นตอน การปรับปรุงคุณภาพข้อมูล (Data Pre-processing) เช่น การแก้ไขความเบี่ยงเบน (Bias Correction) และ การหลอมรวมข้อมูล (Data Fusion) มาใช้ในขั้นตอนต่อไป



รูปที่ 4.3 กราฟเทียบเส้นทางของข้อมูลดิบจาก CAM (จุดสีส้ม) และ DR (จุดสีน้ำเงิน) เทียบกับเส้นทางจริง (เส้นสีดำ) ที่อัตราการเก็บตัวอย่างข้อมูล dt = 0.1 วินาที



รูปที่ 4.4 กราฟเทียบเส้นทางของข้อมูลดิบจาก CAM (จุดสีส้ม) และ DR (จุดสีสีน้ำเงิน) เทียบกับเส้นทางจริง (เส้นสีดำ) ที่อัตราการเก็บตัวอย่างข้อมูล dt = 0.2 วินาที

4.2 ผลของการปรับปรุงคุณภาพข้อมูล (Bias Correction)

ในหัวข้อ 4.1.2.2 เป็นข้อมูลดิบ (Raw Data) จาก CCTV และ DR มีค่า error และเบี่ยงออกจากเส้นทางจริง โดยเฉพาะเลี้ยวโค้ง ดังนั้น เพื่อเพิ่มความแม่นของข้อมูลก่อนนำไปใช้กับการหลอมรวมข้อมูล (Data Fusion)

4.2.1 วิเคราะห์ค่าเบี่ยงเบนเฉลี่ย

ในการแก้ไข ได้มีการหาค่าเบี่ยงเบนเฉลี่ย (Average Bias Errors) ของเซ็นเซอร์แต่ละชนิด เพื่อทำความเข้าใจถึงลักษณะและทิศทางของความคลาดเคลื่อนพื้นฐาน ตาราง 4.4 ได้สรุปค่าเบี่ยงเบนเฉลี่ยของเซ็นเซอร์ที่อัตราการเก็บตัวอย่างข้อมูล (dt) 0.1 และ 0.2 วินาที จะเห็นว่า

เซ็นเซอร์แต่ละชนิดมีลักษณะความเบี่ยงเบนที่ต่างกัน ทั้งด้านมุมทิศทาง (ค่าบวก/ลบ) และขนาด ข้อมูลเหล่านี้ได้นำไปใช้เพื่อพัฒนาและแก้ไขค่าเบี่ยงเบนในข้อมูลดิบ

ตารางที่ 4.4 ค่าเบี่ยงเบนเฉลี่ย

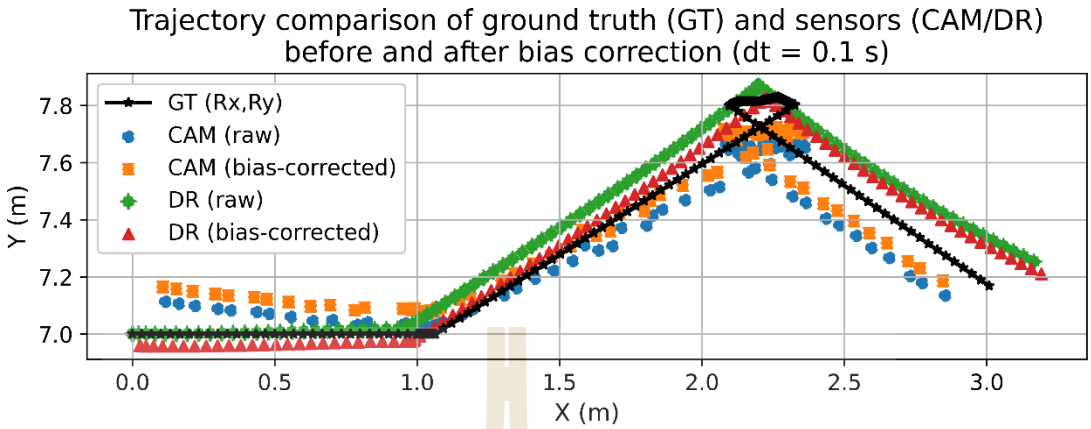
dt (s)	Sensor	Mean Error X (m)	Mean Error Y (m)	Mean Error θ (rad)
0.1	CAM	0.0084	-0.0515	-
	DR	-0.0258	0.0423	-
	IMU	-	-	0.2176
0.2	CAM	0.0058	-0.05	-
	DR	-0.0263	0.0463	-
	IMU	-	-	0.2492

4.2.2 ประสิทธิภาพหลังการแก้ไขความเบี่ยงเบน

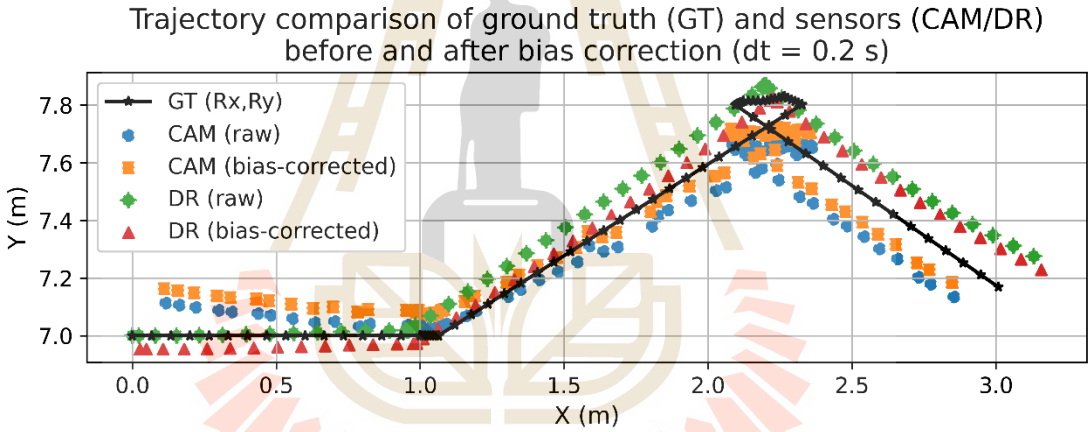
เพื่อประเมินประสิทธิภาพของกระบวนการ Bias Correction ได้เทียบค่า RMSE ของข้อมูล ก่อนแก้ไขข้อมูลดิบ (Raw) และ หลังการแก้ไขค่าเบี่ยงเบน (Bias Corrected, BC) ดังตาราง 4.5 ค่า RMSE ทุกตัวลดลง หลังจากผ่านกระบวนการ Bias Correction โดย DR ที่ค่า RMSE ในแกน X และ Y ลดลงอย่างมาก

ตารางที่ 4.5 เทียบค่า RMSE ของเซ็นเซอร์ก่อนและหลังการทำ Bias Correction

dt (s)	Sensor	X_{raw} (m)	X_{BC} (m)	Y_{raw} (m)	Y_{BC} (m)	Θ_{raw} (rad)	Θ_{BC} (rad)
0.1	CAM	0.06649	0.06595	0.10021	0.08595	–	–
	DR	0.09339	0.08974	0.05033	0.02724	–	–
	IMU	–	–	–	–	1.1959	1.1524
0.2	CAM	0.06010	0.05981	0.09934	0.08582	–	–
	DR	0.08979	0.08582	0.05612	0.03166	–	–
	IMU	–	–	–	–	1.2039	1.1497



รูปที่ 4.5 กราฟเทียบเส้นทางก่อน (Raw) และหลัง (Bias-Corrected) การแก้ไขความเบี่ยงเบน ที่ dt = 0.1 วินาที



รูปที่ 4.6 กราฟเทียบเส้นทางก่อน (Raw) และหลัง (Bias-Corrected) การแก้ไขความเบี่ยงเบน ที่ dt = 0.2 วินาที

รูปที่ 4.5 และ 4.6 เส้นทางหลังการแก้ไข (CAM-BC: จุดสีส้ม และ DR-BC: จุดสีแดง) สามารถเคลื่อนที่เข้าใกล้และสอดคล้องกับเส้นทางจริง (GT: เส้นสีดำ) ได้ดีกว่า เส้นทางของข้อมูลดิบ (CAM-Raw: จุดสีน้ำเงิน และ DR-Raw: จุดสีเขียว) การเบี่ยงเบนออกจากเส้นทางจริงลดลง โดยเฉพาะในช่วงเลี้ยวโค้ง

ผลคือ การทำ Bias Correction เป็นขั้นตอนคำนวณพื้นฐาน (Pre-processing) สำคัญ ในการปรับปรุงข้อมูลดิบให้มีความแม่นยำสูงสุดตั้งแต่ต้นทาง เป็นการสร้างรากฐานข้อมูล ที่ส่งผลต่อวิธีการหลอมรวมข้อมูล

4.3 คำนวณน้ำหนักด้วยหลักความแปรปรวนผกผัน (Inverse-Variance Weighting)

หลังจากข้อมูลที่ได้จากเซนเซอร์แต่ละชนิดผ่านการปรับปรุงคุณภาพด้วยกระบวนการแก้ไขความเอนเอียง (Bias Correction) แล้ว ขั้นตอนถัดไปคือการประเมินระดับความน่าเชื่อถือของข้อมูลแต่ละแหล่ง เพื่อให้กระบวนการหลอมรวมข้อมูลในขั้นตอนถัดไปมีประสิทธิภาพมากยิ่งขึ้น งานวิจัยนี้เลือกใช้หลักการ ถ่วงน้ำหนักด้วยส่วนกลับของความแปรปรวน (Inverse-Variance Weighting) ซึ่งเป็นแนวทางทางสถิติที่ให้น้ำหนักข้อมูลมากขึ้นกับเซนเซอร์ที่มีค่าความแปรปรวนต่ำ (มีความแม่นยำสูง) ในขณะที่ข้อมูลที่มีความแปรปรวนสูงจะได้รับน้ำหนักน้อยกว่า

ผลการคำนวณพบว่า หลังจากผ่านการแก้ไข Bias Correction แล้ว ค่าความแปรปรวน (Variance) และค่าน้ำหนัก (Weight) ที่คำนวณได้มีความสัมพันธ์ในทิศทางตรงข้าม กล่าวคือ เซนเซอร์ที่ให้ข้อมูลมีเสถียรภาพมากกว่าจะได้รับค่าน้ำหนักสูงกว่า ผลลัพธ์จากการทดลองที่อัตราการเก็บข้อมูล (Sampling Time; dt) เท่ากับ 0.1 s และ 0.2 s สรุปได้ว่า (ตารางที่ 4.6 และ 4.7) ระบบ Dead Reckoning (Encoder + IMU) ให้ค่าความแปรปรวนในแนว Y ต่ำที่สุด (ประมาณ 0.000744 m^2) เมื่อเทียบกับเซนเซอร์อื่น

นอกจากนี้ ยังพบว่าเมื่อเปรียบเทียบระหว่างค่าความแปรปรวนในแกน X และ Y ค่าความแปรปรวนในแกน Y มีค่าน้อยกว่าอย่างมีนัยสำคัญ (ช่วง 0.0007–0.001) ซึ่งสะท้อนถึงความแม่นยำของการเคลื่อนที่ในแนวตรง ขณะที่ข้อมูลจาก CCTV มีแนวโน้มเกิดความเบี่ยงเบนมากกว่า

เมื่อนำข้อมูลที่ผ่านการปรับปรุงแล้วไปใช้ในการหลอมรวมข้อมูล (Fused Correction) พบว่า ค่าความคลาดเคลื่อน RMSE ของผลลัพธ์จาก EKF ลดลงอย่างมีนัยสำคัญ (ตารางที่ 4.8) เมื่อเทียบกับกรณีข้อมูลดิบ (Fused Raw) ทั้งในแกน X และ Y ซึ่งแสดงให้เห็นว่าการแก้ไข Bias Correction ร่วมกับการถ่วงน้ำหนักแบบ Inverse-Variance มีผลชัดเจนในการเพิ่มความแม่นยำของข้อมูลที่หลอมรวมได้

ตารางที่ 4.6 ค่าแปรปรวน (Variance) และน้ำหนัก (Weight) ของเซ็นเซอร์ที่ dt = 0.1s

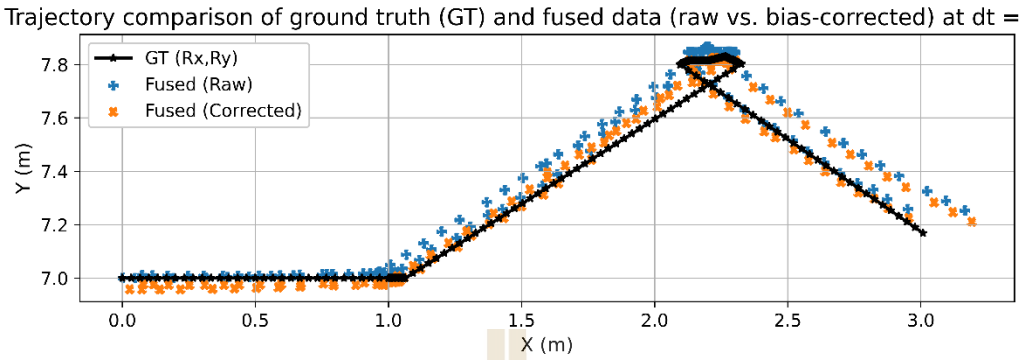
Data	Sensor	Var_X (m^2)	Var_Y (m^2)	Var_θ (rad^2)	w_X	w_y	w_θ
Raw	CAM	0.004374	0.007429	--	0.648666	0.091047	--
	DR	0.008076	0.000744	--	0.351344	0.908953	--
	IMU	--	--	1.320644	--	--	1
Bias Corrected	CAM	0.004374	0.007429	--	0.648666	0.091047	--
	DR	0.008076	0.000744	--	0.351344	0.908953	--
	IMU	--	--	1.320644	--	--	1

ตารางที่ 4.7 ค่าแปรปรวน (Variance) และน้ำหนัก (Weight) ของเซ็นเซอร์ที่ dt = 0.2s

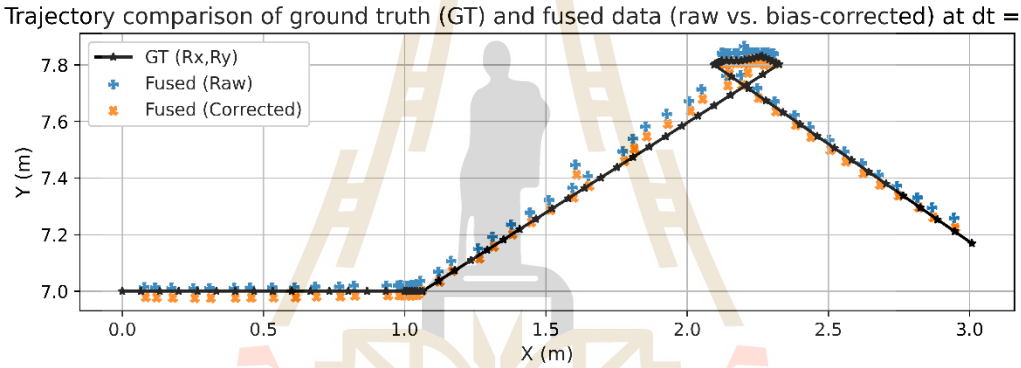
Data	Sensor	Var_X (m^2)	Var_Y (m^2)	Var_θ (rad^2)	w_x	w_y	w_θ
Raw	CAM	0.003599	0.007408	--	0.673066	0.119823	--
	DR	0.007409	0.001009	--	0.326934	0.880177	--
	IMU	--	--	1.320571	--	--	1
Bias Corrected	CAM	0.003599	0.007408	--	0.673066	0.119823	--
	DR	0.007409	0.001009	--	0.326934	0.880177	--
	IMU	--	--	1.320571	--	--	1

ตารางที่ 4.8 สรุป RMSE ของข้อมูล ก่อนและหลังการทำ Bias Correction

dt (s)	Sensor Version	RMSE X (m)	RMSE Y (m)
0.1	Fused (Raw)	0.0718	0.0447
	Fused (Corrected)	0.0695	0.0234
0.2	Fused (Raw)	0.0410	0.0414
	Fused (Corrected)	0.0411	0.0223



รูปที่ 4.7 กราฟเทียบเส้นทางระหว่าง Ground Truth (GT) และข้อมูลหลังการหลอมรวม (Raw vs Bias-Corrected) ที่ dt = 0.1 s



รูปที่ 4.8 กราฟเทียบเส้นทางระหว่าง Ground Truth (GT) และข้อมูลหลังการหลอมรวม (Raw vs. Bias-Corrected) ที่ dt = 0.2 s

รูปที่ 4.7 และ 4.8 ข้อมูลที่ผ่านการแก้ไข (Fused Corrected) สามารถ เคลื่อนที่เข้าใกล้และสอดคล้องกับเส้นทางจริง (Ground Truth) ได้ดีกว่า เส้นทางของข้อมูลดิบ (Fused Raw) อย่างเห็นได้ชัด การเบี่ยงเบนออกจากเส้นทางจริง โดยช่วงเลี้ยวโค้ง เทคนิค Inverse-Variance Weighting ในการหลอมรวมข้อมูลจากหลายเซ็นเซอร์ พิสูจน์ให้เห็นว่า การทำ Bias Correction เป็นขั้นตอนการคำนวณพื้นฐาน (Pre-processing) มีความสำคัญ เพื่อให้ได้ผลลัพธ์การหา x, y ที่มีความแม่น

4.4 ผลการหาตำแหน่งด้วย Standard EKF

หลังจากที่ข้อมูลจากเซนเซอร์แต่ละชนิดผ่านการปรับปรุงคุณภาพด้วยกระบวนการแก้ไขความเบี่ยงเบน (Bias Correction) แล้ว ขั้นตอนต่อไปคือการหลอมรวมข้อมูลเพื่อเพิ่มความแม่นยำของการประมาณค่าตำแหน่ง โดยใช้ตัวกรองคาลมานแบบขยายมาตรฐาน (Standard EKF) เป็น

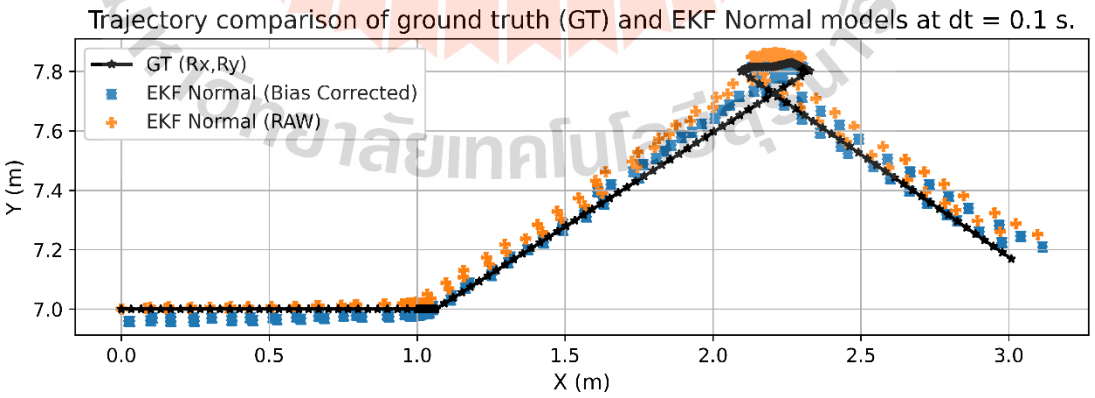
กระบวนการหลักในการประมวลผลข้อมูลดังกล่าว ซึ่งผลลัพธ์จากการหลอมรวมนี้ถูกเรียกว่า Normal Bias-Corrected EKF

ในกระบวนการนี้ ข้อมูลจากระบบ Dead Reckoning ซึ่งมีอัตราการส่งข้อมูลที่ต่อเนื่อง ถูกนำมาใช้ในขั้นตอนการทำนาย (Prediction Step) เพื่อประมาณค่าการเคลื่อนที่ของ AGV ตามแบบจำลองการเคลื่อนที่ของระบบ จากนั้นข้อมูลจากกล้อง (CCTV) ซึ่งให้ตำแหน่งที่มีความแม่นยำสูง จะถูกนำมาใช้ในขั้นตอนการปรับปรุง (Update Step) เพื่อปรับแก้ค่าตำแหน่งที่คาดการณ์ไว้ให้มีความถูกต้องมากยิ่งขึ้น และลดข้อผิดพลาดที่เกิดจากการสะสมของความคลาดเคลื่อน (Drift Error) จากระบบ Dead Reckoning

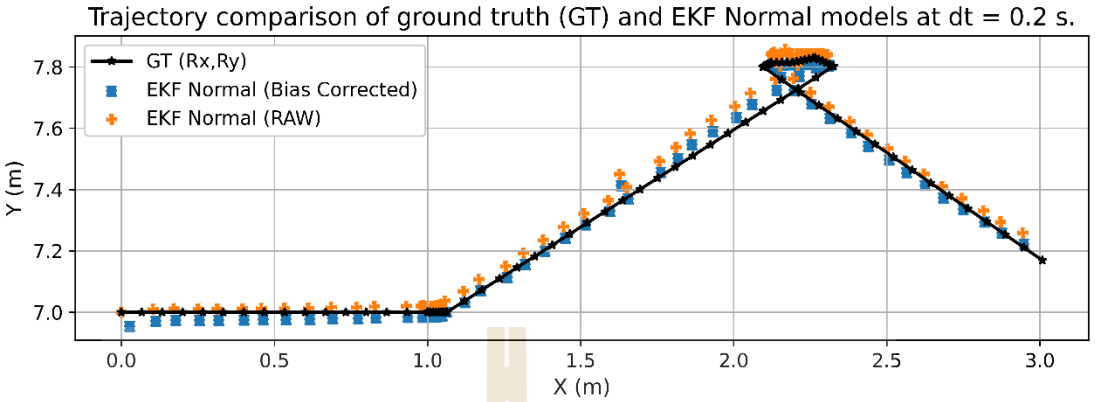
ตารางที่ 4.9 ตารางสรุปค่า RMSE ของ Standard EKF

dt (s)	EKF Mode	RMSE X(m)	RMSE Y(m)	RMSE XY(m)
0.1	Normal (RAW)	0.055219259	0.044184317	0.070720722
	Normal (Bias Corrected)	0.05358052	0.022927812	0.058279985
0.2	Normal (RAW)	0.039453961	0.041265908	0.057091945
	Normal (Bias Corrected)	0.039536428	0.022022628	0.045256218

จากตารางที่ 4.9 การใช้ Standard EKF กับข้อมูลที่ผ่านการแก้ไขความเบี่ยงเบน (Bias Corrected) สามารถ ลดค่าความคลาดเคลื่อนรวม (RMSE XY) ลงได้อย่างมีนัยสำคัญ เมื่อเทียบกับการใช้ข้อมูลดิบ (RAW) โดยตรง ที่อัตราการเก็บตัวอย่างข้อมูล dt = 0.1s ค่า RMSE XY ลดลงจาก 0.0707 เมตร เหลือเพียง 0.0582 เมตร



รูปที่ 4.9 กราฟเปรียบเทียบเส้นทางการเคลื่อนที่ของ Standard EKF ที่ dt = 0.1 s



รูปที่ 4.10 กราฟเทียบเส้นทางของ Standard EKF ที่ dt = 0.2 s

จากรูปที่ 4.9 และ 4.10 เส้นทางของ Standard EKF (Normal Bias Corrected) สามารถเกาะกลุ่มและตามเส้นทางจริง (Ground Truth - GT) ได้ดีกว่า เส้นทางที่ได้จากการหลอมรวมข้อมูลดิบ (Normal RAW) อย่างเห็นได้ชัด อย่างไรก็ตาม เมื่อสังเกตพบข้อจำกัดที่สำคัญของวิธีนี้ คือ เส้นทางยังคงมีลักษณะ การแกว่งหรือกระตุกเล็กน้อย (Slight Oscillation) ปรากฏการณ์นี้เป็นลักษณะเฉพาะของการปรับแก้ค่าตำแหน่งอย่างกะทันหัน เมื่อได้รับข้อมูลใหม่จากเซ็นเซอร์ภายนอก (CCTV) ทำให้เส้นทางการเคลื่อนที่ขาดความราบรื่น (Smoothness)

4.5 ผลการระบุตำแหน่งด้วย Blended EKF

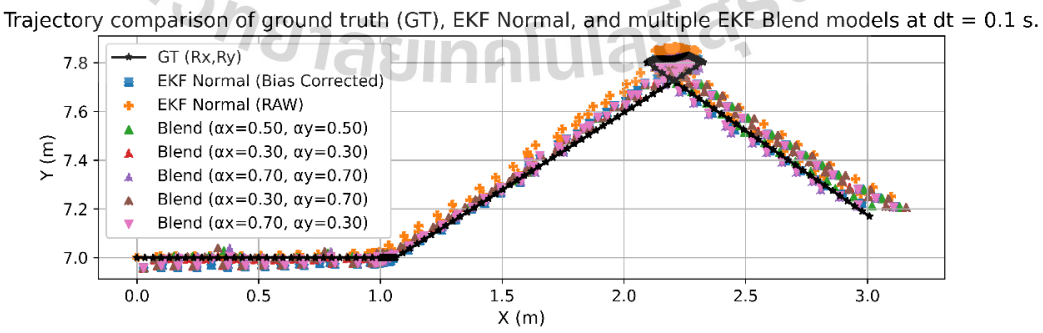
เพื่อปรับปรุงประสิทธิภาพและแก้ไขข้อจำกัดด้านความราบรื่นของเส้นทาง (Smoothness) ใน EKF แบบมาตรฐาน งานวิจัยนี้จึงได้นำเสนอ Blended EKF ซึ่งเป็นนวัตกรรมหลักของงานวิจัยนี้ แนวคิดสำคัญของวิธีนี้คือ แทนที่ใช้ข้อมูลจากเซ็นเซอร์ CAM เพื่อ "แก้ไข" ค่าที่ทำนายจาก DR เพียงอย่างเดียว วิธีการ ผสม (Blend) ของข้อมูล CAM และ DR ร่วมกัน ก่อน ป้อนในขั้นตอนการปรับปรุง (Update Step) ตัวกรอง

กระบวนการผสมนี้จะใช้ ค่าสัมประสิทธิ์การผสม (α_x, α_y) ซึ่งเป็นตัวกำหนดสัดส่วนความสำคัญของข้อมูลแต่ละตัวในแต่ละแกน ค่าสัมประสิทธิ์เหล่านี้กำหนดจากข้อมูล RMSE และค่าน้ำหนัก (Inverse-Variance Weighting) ที่ได้ทำไว้ในหัวข้อ 4.3

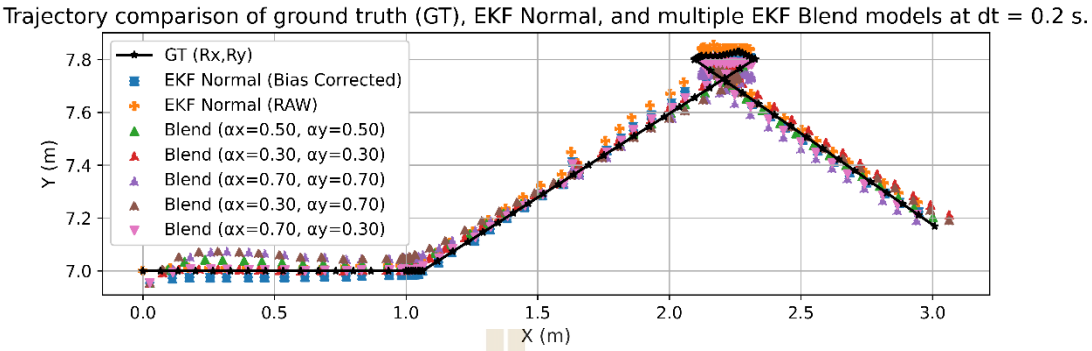
ตารางที่ 4.10 เทียบระหว่าง Normal EKF และ Blended EKF

dt (s)	EKF Mode	RMSE X(m)	RMSE Y(m)	RMSE XY(m)
0.1	Normal (RAW)	0.055219259	0.044184317	0.070720722
	Normal (Bias Corrected)	0.05358052	0.022927812	0.058279985
	Blend (αx=0.50, αy=0.50)	0.059065446	0.01733388	0.061556399
	Blend (αx=0.30, αy=0.30)	0.070483646	0.01756962	0.072640456
	Blend (αx=0.70, αy=0.70)	0.052599105	0.01764171	0.055478787
	Blend (αx=0.30, αy=0.70)	0.070489354	0.017659442	0.072667771
	Blend (αx=0.70, αy=0.30)	0.052559833	0.01757374	0.055419964
0.2	Normal (RAW)	0.039453961	0.041265908	0.057091945
	Normal (Bias Corrected)	0.039536428	0.022022628	0.045256218
	Blend (αx=0.50, αy=0.50)	0.042735705	0.032831886	0.05389131
	Blend (αx=0.30, αy=0.30)	0.056508568	0.018965989	0.059606434
	Blend (αx=0.70, αy=0.70)	0.039983346	0.051770278	0.065412764
	Blend (αx=0.30, αy=0.70)	0.056461919	0.051781832	0.076611399
	Blend (αx=0.70, αy=0.30)	0.03996311	0.018911546	0.044211953

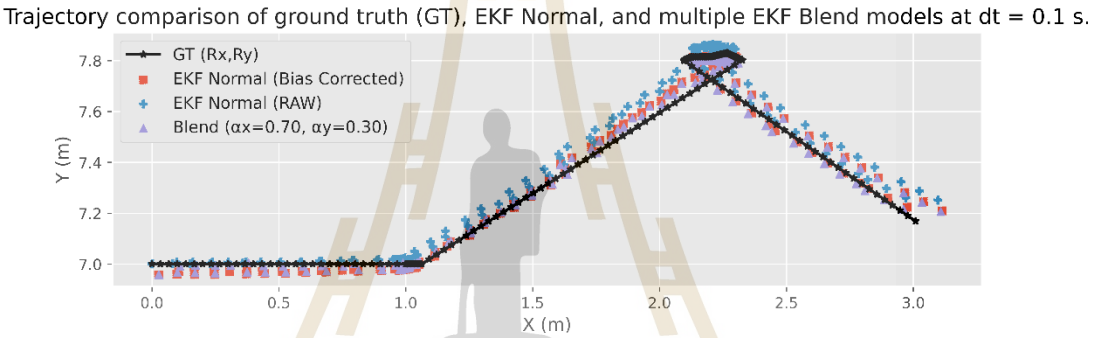
จากตารางที่ 4.10 พบว่า Blended EKF ให้ผลลัพธ์ที่แม่นสูงกว่าเมื่อเทียบ Standard EKF อย่างชัดเจน ในสภาวะที่การเก็บตัวอย่างข้อมูล dt = 0.2s เมื่อกำหนดค่าสัมประสิทธิ์เป็น Blend (α_x=0.70, α_y=0.30) สามารถลดค่า RMSE รวม (RMSE XY) ลงเหลือเพียง 0.04424 เมตร ต่ำที่สุดจากทั้งหมด การผสมข้อมูลด้วยอัตราส่วนที่เหมาะสมช่วยเพิ่มความแม่นยำโดยรวมของระบบได้อย่างมีนัยสำคัญ



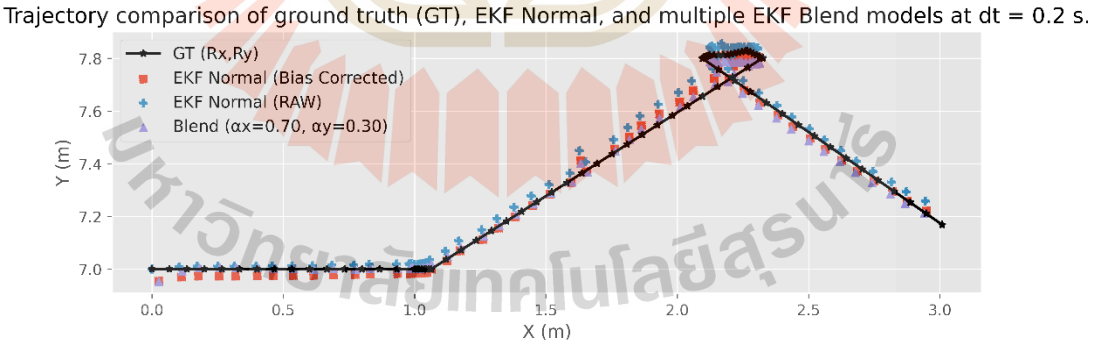
รูปที่ 4.11 กราฟเทียบเส้นทางของ EKF รูปแบบต่าง ๆ ที่ dt = 0.1 s.



รูปที่ 4.12 กราฟเทียบเส้นทางของ EKF รูปแบบต่าง ๆ ที่ dt = 0.2 s.



รูปที่ 4.13 กราฟเทียบเส้นทางของ Blend EKF (α_x=0.70, α_y=0.30) ที่ dt = 0.1 s.



รูปที่ 4.14 กราฟเทียบเส้นทางของ Blend EKF (α_x=0.70, α_y=0.30) ที่ dt = 0.2 s.

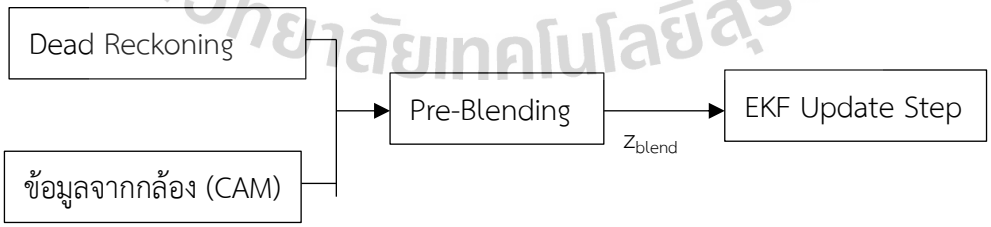
รูปที่ 4.11 4.12 4.13 และ 4.14 เส้นทางของ Blended EKF ไม่เพียงแต่จะ แนบชิดกับเส้นทางจริง (Ground Truth - GT) แต่ยังมีความ ราบรื่น (Smoother) และ ต่อเนื่อง (Continuous) มากกว่า เส้นทางของ EKF แบบมาตรฐาน การแกว่งหรือกระตุกของเส้นทางที่เคยปรากฏใน Normal

EKF ได้ลดลง ที่ทำให้เส้นทางราบรื่นขึ้นนั้น ซึ่งมาจากการที่วิธีการผสมข้อมูลก่อน ทำให้การปรับปรุงค่าสถานะของตัวกรองเป็นไปอย่าง นุ่มนวลและมีเสถียรภาพมากกว่า ไม่ใช่การปรับแก้ค่าแบบกะทันหัน สิ่งนี้ส่งผลดีในงานจริง เพราะควบคุมมอเตอร์ของ AGV ได้นุ่มนวล ลดการสึกหรอของอุปกรณ์ เพิ่มความปลอดภัย การผสมข้อมูล CAM–DR ก่อนขั้นตอนการปรับปรุง (Pre-Update Blending) เพื่อปรับปรุงความแม่นยำของการประมาณตำแหน่งและลดความผันผวนของข้อมูลการวัด งานวิจัยนี้ได้ใช้แนวคิดการผสมข้อมูลจากกล้อง (CAM) และข้อมูล Dead Reckoning ล่วงหน้า ก่อนเข้าสู่ขั้นตอนการปรับปรุงของ Extended Kalman Filter (Update Step) แนวคิดดังกล่าวสอดคล้องกับรายละเอียดในงานวิจัยต้นฉบับที่อธิบายกลไก “pre-blended measurement vector” ซึ่งมุ่งลดความแปรปรวนสูงของข้อมูล CAM และความลื่นไหลเกินจริงของ DR เมื่อใช้งานแยกกัน

แนวคิดนี้ คือ การสร้างเวกเตอร์การวัดใหม่ที่ผสมข้อมูลทั้งสองแหล่งด้วยน้ำหนัก α_xและ α_yที่ได้จากกระบวนการเพิ่มประสิทธิภาพของ Optuna ก่อนนำเข้าสู่สมการปรับปรุงของ EKF โดยมีรูปแบบดังนี้

$$z_{\text{blend}} = \begin{bmatrix} x_{\text{blend}} \\ y_{\text{blend}} \end{bmatrix} = \begin{bmatrix} \alpha_x\, x_{CAM} + (1-\alpha_x)\, x_{DR} \\ \alpha_y\, y_{CAM} + (1-\alpha_y)\, y_{DR} \end{bmatrix} \tag{4.1}$$

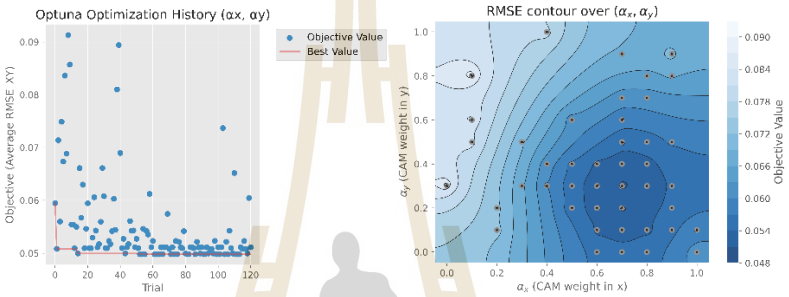
เวกเตอร์ z_{blend} นี้จะถูกนำเข้าในขั้นตอน Update ตามสมการมาตรฐานของ EKF ซึ่งประกอบด้วยการคำนวณ Kalman Gain, การปรับปรุงสถานะ และการอัพเดตเมทริกซ์ความไม่แน่นอน โดยลักษณะสำคัญคือ ไม่มีการใช้ค่า CAM หรือ DR โดยตรง แต่ใช้ค่าที่ผสมแล้วเท่านั้นในทุกกรณี ดังรูป ที่ 4.15 กลไกนี้ช่วยลดข้อจำกัดของการสลับแหล่งข้อมูลแบบเดิม และให้ผลเส้นทางที่ราบรื่นกว่าอย่างมีนัยสำคัญ ทั้งยังช่วยลดอิทธิพลของข้อมูลที่มีลักษณะ noise สูง เช่น CAM ในสภาพแสงไม่สม่ำเสมอ และในขณะเดียวกันลดผลสะสมความคลาดเคลื่อนของ DR ในช่วงเคลื่อนที่ระยะยาว



รูปที่ 4.15 แผนภาพขั้นตอนการผสมข้อมูล CAM–DR ก่อนเข้าสู่กระบวนการปรับปรุงของ EKF

4.6 ผลลัพธ์การหาค่าสัมประสิทธิ์การผสมด้วย Optuna

การค้นหาค่าสัมประสิทธิ์การผสม (α_x, α_y)ด้วย Optuna มีจุดประสงค์เพื่อให้ได้ค่าที่ช่วยลดความคลาดเคลื่อนของตำแหน่งที่ประมาณจาก Blended EKF ให้ต่ำที่สุด โดยใช้ค่า $\mathbf{RMSE}_{XY}$ เป็นตัวชี้วัดหลัก กระบวนการค้นหาใช้ TPE Sampler ในการเสนอค่าพารามิเตอร์แต่ละรอบ ผลลัพธ์ที่ได้สามารถแสดงให้เห็นถึงประสิทธิภาพของวิธีการเพิ่มประสิทธิภาพนี้ได้อย่างชัดเจน ดังรูปที่ 4.16



(a) (b)

รูปที่ 4.16 ผลการค้นหาค่าสัมประสิทธิ์การผสมด้วย Optuna

รูปที่ 4.16(a) แสดงการลดลงของค่า RMSE เมื่อจำนวนรอบการค้นหาเพิ่มขึ้น จะพบว่าช่วงแรก RMSE มีการเปลี่ยนแปลงค่อนข้างกว้าง แต่เมื่อจำนวนรอบมากขึ้น ค่า RMSE ค่อย ๆ ลดลงและมีแนวโน้มทรงตัว แสดงถึงความสามารถของ TPE Sampler ในการมุ่งค้นหาบริเวณที่มีค่าพารามิเตอร์ที่มีแนวโน้มให้ผลลัพธ์ที่ดีขึ้นเรื่อย ๆ รูป 4.16(b) เป็นกราฟสามมิติแบบฉายบนระนาบที่แสดงความสัมพันธ์ของคู่พารามิเตอร์ (α_x, α_y)กับค่าความผิดพลาด RMSE จะเห็นได้ชัดเจนว่ามีบางบริเวณที่ให้ค่า RMSE ต่ำกว่าบริเวณอื่น ซึ่งบ่งชี้ว่าโซนดังกล่าวเป็นโซนที่เหมาะสมสำหรับการกำหนดค่าสัมประสิทธิ์การผสม โดยจุดที่มีสีอ่อนที่สุดในแผนภาพคือค่าที่ให้ผลลัพธ์ดีที่สุดในเชิงการลด RMSE

จากผลการค้นหาโดยรวม พบว่าคู่พารามิเตอร์ที่ให้ผลลัพธ์ดีที่สุด คือ

$$\alpha_x^* = 0.7, \alpha_y^* = 0.3$$

ค่าทั้งสองถูกนำไปใช้เป็นค่าสัมประสิทธิ์อ้างอิงในการคำนวณ Blended EKF สำหรับการเปรียบเทียบในตารางผลลัพธ์ RMSE และตัวชี้วัดความราบรื่นของเส้นทาง ทั้งนี้ ผลการค้นหาจาก Optuna มีส่วนสำคัญในการทำให้ Blended EKF สามารถสร้างเส้นทางที่ราบรื่นกว่าและมีความคลาดเคลื่อนเชิง

ตำแหน่งลดลงอย่างมีนัยสำคัญเมื่อเทียบกับการกำหนดค่าสัมประสิทธิ์แบบสุ่มหรือเลือกค่าคงที่ด้วยตนเอง

4.7 ตัวชี้วัดความราบรื่นของเส้นทาง (Trajectory Roughness Index: TRI)

การประเมินคุณภาพของเส้นทางการเคลื่อนที่ไม่ได้ขึ้นอยู่เพียงแค่ค่าความคลาดเคลื่อนเชิงตำแหน่ง เช่น RMSE เท่านั้น แต่ยังต้องคำนึงถึง “ความราบรื่นของการเคลื่อนที่” (trajectory smoothness) ซึ่งมีความสำคัญอย่างยิ่งสำหรับระบบนำทางของ AGV งานวิจัยล่าสุดหลายชิ้นแสดงให้เห็นว่าตัวชี้วัดความราบรื่นของเส้นทางสามารถใช้ระบุลักษณะการเคลื่อนที่ที่ผิดปกติหรือพฤติกรรมเฉพาะได้ เช่น การแยกแยะประเภทของโดรนด้วยข้อมูล track จากเรดาร์ (Roy et al., 2022) รวมถึงงานด้าน representation learning ที่ชี้ให้เห็นว่าเส้นทางที่ราบรื่นกว่าจะให้ latent feature ที่มีความคงตัวและเหมาะสมต่อกระบวนการเรียนรู้มากกว่า (Desai et al., 2024)

ด้วยเหตุนี้ งานวิจัยนี้ได้นำเสนอ “Trajectory Roughness Index” เป็นตัวชี้วัดเสริมสำหรับประเมินคุณภาพเส้นทางที่ได้จากการประมวลผลของ EKF โดย TRI จะสะท้อนระดับความสั่น ความไม่ต่อเนื่อง หรือการเปลี่ยนทิศทางอย่างกะทันหันของเส้นทาง ซึ่ง RMSE ไม่สามารถอธิบายได้

นิยามทางคณิตศาสตร์ของ TRI

ค่า TRI นิยามด้วยอนุพันธ์เชิงจำกัดของตำแหน่งในแต่ละช่วงเวลา เพื่อสะท้อนระดับความราบรื่นของการเคลื่อนที่:

$$TRI = \frac{1}{N-1}\sum_{k=2}^{N} \left\| \begin{bmatrix} x_k - x_{k-1} \\ y_k - y_{k-1} \end{bmatrix} \right\|^2 \tag{4.2}$$

หรือในรูปแบบขยาย

$$TRI = \frac{1}{N-1}\sum_{k=2}^{N} \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2} \tag{4.3}$$

ซึ่งรูปแบบนี้สอดคล้องกับแนวคิดด้าน trajectory smoothness ที่ใช้ในงานด้านการติดตามวัตถุและการประเมินเส้นทางการเคลื่อนที่ (Roy et al., 2022) ค่าของ TRI ที่ต่ำหมายถึงเส้นทางที่เคลื่อนที่ได้ราบรื่น มั่นคง และปราศจากการแกว่งหรือการเปลี่ยนทิศฉับพลัน ในขณะที่ค่าที่สูงขึ้นบ่งชี้ถึงความสั่นหรือการเคลื่อนที่ที่เปลี่ยนแปลงรุนแรง

ผลลัพธ์ของ TRI จากการทดลอง

ตารางที่ 4.11 แสดงผลลัพธ์ค่า TRI จากการประเมินเส้นทางที่ได้จาก EKF Normal (RAW), EKF Normal (Bias Corrected) และ EKF Blend ในสองเงื่อนไขการสุ่มข้อมูลคือ dt = 0.1 s และ dt = 0.2 s ซึ่งดึงมาจากบทความต้นฉบับของงานวิจัยนี้

ตารางที่ 4.11 ค่า Trajectory Roughness Index สำหรับ dt = 0.1 และ 0.2 s

dt (s)	Mode	TRI
0.1	EKF Normal (RAW)	0.0405
	EKF Normal (BC)	0.0371
	EKF Blend (αx=0.50, αy=0.50)	0.0368
	EKF Blend (αx=0.30, αy=0.30)	**0.0266**
	EKF Blend (αx=0.70, αy=0.70)	0.0455
	EKF Blend (αx=0.30, αy=0.70)	0.0301
	EKF Blend (αx=0.70, αy=0.30)	0.0428
0.2	EKF Normal (RAW)	0.0235
	EKF Normal (BC)	0.0235
	EKF Blend (αx=0.50, αy=0.50)	0.0227
	EKF Blend (αx=0.30, αy=0.30)	**0.0219**
	EKF Blend (αx=0.70, αy=0.70)	0.0237
	EKF Blend (αx=0.30, αy=0.70)	0.0221
	EKF Blend (αx=0.70, αy=0.30)	0.0236

จากตารางที่ 4.11 พบว่า

1) โมเดล **EKF Blend** ให้ค่า TRI ต่ำกว่า EKF Normal อย่างสม่ำเสมอ
2) ค่าที่ดีที่สุดคือ $(\alpha_x = 0.30, \alpha_y = 0.30)$ สำหรับทั้งสองค่า dt
3) ค่า TRI ให้มุมมองเสริมจาก RMSE โดยสะท้อนความราบรื่นของเส้นทางการเคลื่อนที่ ซึ่งเป็นสิ่งจำเป็นต่อการใช้งาน AGV ในพื้นที่จำกัด

4.8 วิเคราะห์เปรียบเทียบและอภิปรายผล

ส่วนนี้นำเสนอการสังเคราะห์ผลการทดลองทั้งหมด เพื่อประเมินและเปรียบเทียบประสิทธิภาพของระบบระบุตำแหน่งและขั้นตอนการหลอมรวมข้อมูลในแต่ละวิธีอย่างเป็นระบบ พร้อมทั้งอภิปรายถึงข้อค้นพบที่สำคัญและแนวทางพัฒนาในอนาคตของงานวิจัย

4.8.1 การวิเคราะห์เชิงลึกระหว่าง Standard EKF และ Blended EKF

การประเมินประสิทธิภาพของระบบประมาณตำแหน่งจำเป็นต้องวิเคราะห์ผลลัพธ์จากหลายมิติ ได้แก่ RMSE, ความราบรื่นของเส้นทาง และความสอดคล้องของสัญญาณจากเซนเซอร์ต่าง ๆ ในงานนี้ Standard EKF และ Blended EKF ถูกเปรียบเทียบภายใต้สองอัตราการประมวลผลของตัวกรอง (dt = 0.1 s และ dt = 0.2 s) ขณะที่กล้อง (CAM) มีอัตราการวัดคงที่ที่ 0.2 s เสมอ ผลเชิงลึกที่ได้สามารถสรุปเป็นหัวข้อต่อไปนี้

1) ผลของ Bias Correction ต่อประสิทธิภาพของ Standard EKF

ข้อมูลกล้องมีโอกาสเกิดอคติคงที่ (systematic bias) จากการแปลงภาพ การปรับเทียบเลนส์ หรือความคลาดเคลื่อนจากระบบมาร์กเกอร์ โดยอคตินี้ไม่ได้เกิดแบบสุ่ม และจะถูกส่งเข้าสู่ EKF ทุกครั้งที่มีการอัปเดต ส่งผลให้ค่า RMSE สูงกว่าแม้โมเดลการเคลื่อนที่จะถูกต้อง การทำ Bias Correction ช่วยลดความผิดพลาดแบบคงที่ ทำให้ตำแหน่ง CAM เข้าใกล้ตำแหน่งจริงมากขึ้น หลังการแก้ bias ค่า RMSE ของ Standard EKF ลดลงทันทีในทั้งสอง sampling time สะท้อนว่าการแก้ bias เป็นขั้นตอนสำคัญก่อนเข้าสู่ EKF Update แม้ยังไม่แก้ปัญหา noise ความถี่สูงของภาพได้ทั้งหมด

2) บทบาทของ Sampling Time (dt) ต่อการทำงานของ EKF

แม้กล้องจะส่งข้อมูลทุก 0.2 วินาที แต่ความถี่ของ EKF สามารถเลือกให้เร็วกว่าได้ โดยในงานนี้ทดสอบที่สองค่า

2.1) dt = 0.1 s (10 Hz)

2.1.1) EKF ทำงานถี่กว่าสัญญาณกล้อง

2.1.2) ระหว่างรอบที่กล้องยังไม่ออกข้อมูล จะใช้ DR และ IMU เป็นตัวทำนายสถานะ

2.1.3) ข้อมูล CAM จะเข้ามาปรับทีละ 0.2 s และถูก interpolate ให้ตรงเวลา

2.1.4) ทำให้ trajectory มีความละเอียดสูง แต่เสี่ยงเกิดการสั่น (oscillation) เมื่อ ข้อมูล CAM ผันผวน

2.2) dt = 0.2 s (5 Hz)

2.2.1) เป็นความถี่เดียวกับ CAM

2.2.2) ทุกเซนเซอร์ถูกซิงโครไนซ์แบบตรงเวลา

2.2.3) เส้นทางมีความราบรื่นกว่า แต่รายละเอียดการเปลี่ยนแปลงเล็ก ๆ จะลดลง

2.2.4) DR มี drift มากขึ้นเมื่อช่วงเวลาระหว่างรอบยาวขึ้น

จากทั้งสองกรณี พบว่า dt = 0.1 s ให้ความละเอียดของการเคลื่อนที่สูงกว่า แต่ความผันผวนของ CAM ส่งผลมากกว่า ในขณะที่ dt = 0.2 s ให้ความราบรื่นที่ดีกว่า โดยเฉพาะเมื่อใช้ Blended EKF

3) เหตุผลที่ Blended EKF ให้ผลลัพธ์เหนือกว่า Standard EKF

ความแตกต่างหลักระหว่างสองวิธีคือ “ตำแหน่งของการผสมข้อมูล” Blended EKF ทำการผสมข้อมูล CAM–DR ล่วงหน้าก่อนเข้าสู่ขั้นตอน Update ซึ่งช่วยลด noise ของการวัดอย่างมีนัยสำคัญ

เวกเตอร์การวัดที่ผสมถูกกำหนดดังนี้:

$$z_{\text{blend}} = \begin{bmatrix} \alpha_x x_{CAM} + (1-\alpha_x) x_{DR} \\ \alpha_y y_{CAM} + (1-\alpha_y) y_{DR} \end{bmatrix} \tag{4.4}$$

ข้อดีสำคัญของการผสมก่อน Update Step ได้แก่:

ลดผลกระทบของ noise ความถี่สูงจากภาพ

ลด drift ของ DR ในช่วงเวลานาน

ช่วยให้ Kalman Gain ไม่ถูกครอบงำโดยข้อมูลที่ผันผวน

ทำให้ผลการประมาณนิ่งขึ้นทั้งในแกน X และ Y

ผลลัพธ์จาก RMSE และ TRI ในบทความแสดงชัดว่า Blended EKF ให้เส้นทางที่มีความเสถียรสูงกว่า และมีค่าความคลาดเคลื่อนต่ำกว่า

4) การวิเคราะห์ผล RMSE เมื่อใช้ค่าที่ได้จาก Optuna

จากรูป Optimization History และ Contour Plot ค่าพารามิเตอร์ที่ให้ RMSE ต่ำที่สุด คือ:

$$\alpha_x^* = 0.70, \alpha_y^* = 0.30$$

สาเหตุเชิงโครงสร้าง:

ในแกน X ข้อมูลจาก CAM มีความแม่นยำกว่า DR ⟶ จึงควรเพิ่มน้ำหนัก CAM

ในแกน Y สัญญาณจาก DR มีเสถียรภาพสูงกว่า CAM ⟶ จึงลดน้ำหนัก CAM ลง

ผลลัพธ์นี้สอดคล้องกับรูป contour ที่แสดงให้เห็นทางด้านขวาของภาพว่า พื้นที่รอบ (0.70, 0.30) เป็นพื้นที่ที่ RMSE ต่ำที่สุดอย่างชัดเจน

5) การตีความผล TRI (ความราบรื่นของเส้นทาง)

เมื่อใช้ตัวชี้วัด TRI พบค่าที่ดีที่สุด คือ:

$$\alpha_x = 0.30, \alpha_y = 0.30$$

ซึ่งต่างจากค่าที่ให้ RMSE ต่ำที่สุดอย่างชัดเจน เนื่องจาก:

การให้น้ำหนัก CAM และ DR ใกล้เคียงกันช่วยลดการแกว่งของเส้นทาง

DR ช่วยเก็บทิศทางในภาพรวม ส่วน CAM ช่วยดึงตำแหน่งให้ไม่ drift

เมื่อสัดส่วนใกล้กัน ผลรวมทำให้เส้นทางราบรื่นกว่าการเน้น CAM หรือ DR ด้านใดด้านหนึ่งมากเกินไป

นี่แสดงถึง “Trade-off” สำคัญระหว่างความแม่นยำเชิงตำแหน่ง (RMSE) และคุณภาพเส้นทาง

6) บทสรุปเชิงลึกของ EKF ทั้งสองรูปแบบ

Standard EKF

ไวต่อ noise ของ CAM

ไวต่อ drift ของ DR

ผลขึ้นกับ dt อย่างมาก

การแก้ bias ช่วยได้ แต่ไม่ขจัดความผันผวน

Blended EKF

ผสมข้อมูลก่อน Update ⟶ ลด variance ของสัญญาณวัด

รองรับ noise ของกล้องได้ดีกว่า

ลด drift ของ DR ได้มีประสิทธิภาพ

ให้ RMSE ต่ำกว่าในทุกเงื่อนไข

ให้ TRI ต่ำกว่า ทำให้เส้นทางราบรื่นกว่า

ปรับแต่งความสำคัญของเซนเซอร์ได้ตามสภาพแวดล้อม

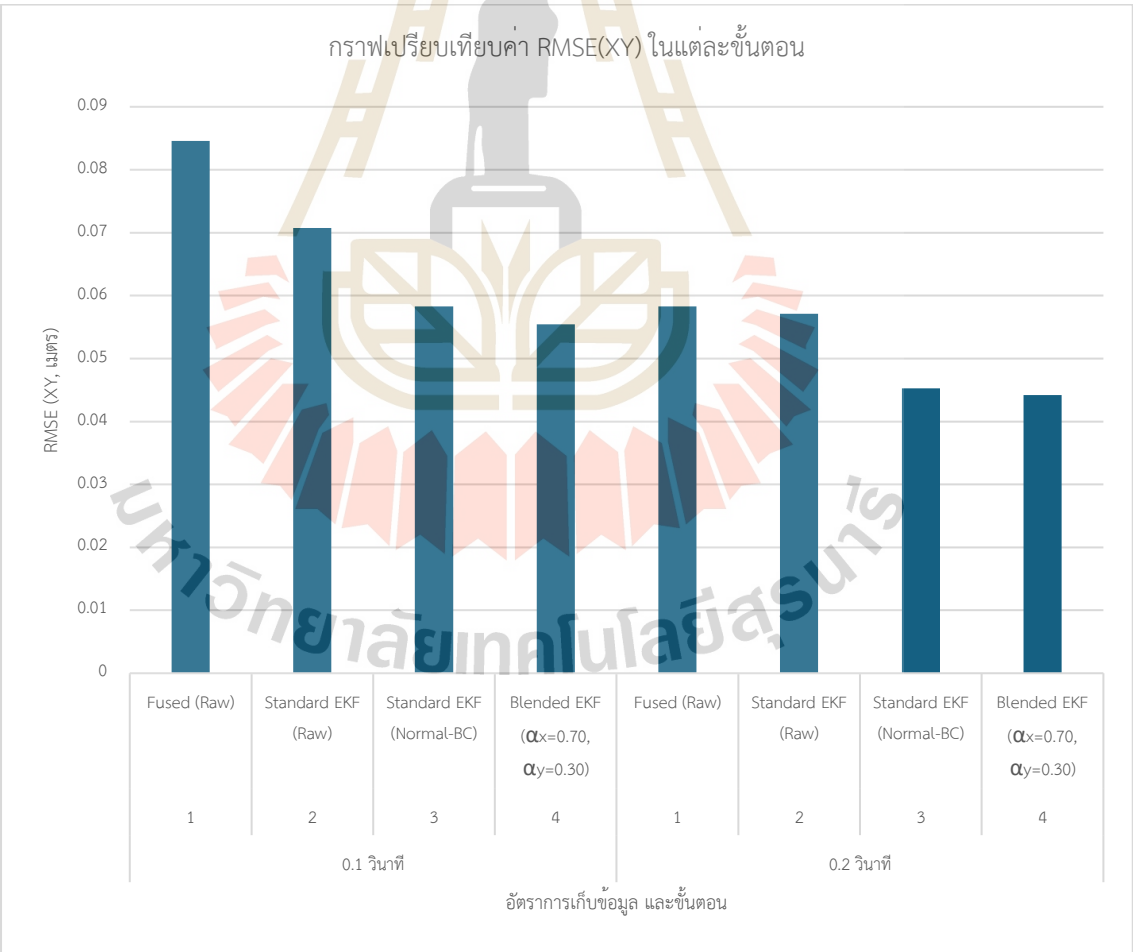
โดยรวม Blended EKF เหมาะสมกับงาน AGV ที่ต้องการทั้งความแม่นยำและความราบรื่นของเส้นทาง โดยเฉพาะในสภาพแวดล้อมที่เซนเซอร์มีคุณภาพแตกต่างกันชัดเจน

4.8.2 สรุปและเปรียบเทียบผลการทดลอง

การวิเคราะห์เชิงเปรียบเทียบมุ่งเน้นการประเมินประสิทธิภาพของการระบุตำแหน่งในแต่ละขั้นตอน ตั้งแต่ข้อมูลดิบก่อนการประมวลผลไปจนถึงการหลอมรวมข้อมูล โดยอ้างอิงค่าความคลาดเคลื่อนเชิงรากที่สองเฉลี่ย (RMSE XY) ของแต่ละวิธีการตามที่แสดงในตารางที่ 4.12 เพื่อใช้เป็นเกณฑ์ในการวัดประสิทธิภาพเชิงปริมาณของแต่ละระบบภายใต้สภาพการทดลองเดียวกัน

ตารางที่ 4.12 เทียบค่า RMSE XY ของแต่ละวิธีที่ dt = 0.1s และ dt = 0.2s

อัตราการเก็บข้อมูล (dt)	ลำดับขั้นตอน	วิธีการ	RMSE XY (เมตร)	% การพัฒนาจากขั้นตอนก่อนหน้า
0.1 วินาที	1	ข้อมูล Fused (Raw)	0.0846	(Baseline)
	2	Standard EKF (Raw)	0.07072	16.43%
	3	Standard EKF (Normal-BC)	0.05828	17.57%
	4	Blended EKF (α_x=0.70, α_y=0.30)	0.05542	4.87%
0.2 วินาที	1	ข้อมูล Fused (Raw)	0.0583	(Baseline)
	2	Standard EKF (Raw)	0.05709	2.07%
	3	Standard EKF (Normal-BC)	0.04526	20.72%
	4	Blended EKF (α_x=0.70, α_y=0.30)	0.04421	2.31%



รูปที่ 4.17 กราฟเทียบค่าความคลาดเคลื่อน (RMSE) ในแต่ละขั้นตอนการพัฒนา

จากผลการทดลองที่แสดงในตาราง 4.11 และรูป 4.15 สามารถสรุปได้ว่า Blended EKF เป็นแนวทางที่ให้ประสิทธิภาพสูงที่สุดในการระบุตำแหน่งของ AGV โดยสามารถยกระดับความแม่นยำได้อย่างต่อเนื่องในทุกขั้นตอนของกระบวนการ ตั้งแต่การแก้ไขความเอนเอียงของข้อมูล (Bias Correction) ซึ่งช่วยลดความคลาดเคลื่อนพื้นฐาน ได้อย่างมีนัยสำคัญ ไปจนถึงการประมวลผลด้วย Standard EKF ที่เพิ่มความแม่นยำขึ้นอีกระดับหนึ่ง อย่างไรก็ตาม ผลการทดลองชี้ให้เห็นอย่างชัดเจนว่า Blended EKF มีความเหนือกว่า Standard EKF โดยเฉพาะในกรณีที่ใช้ค่าสัมประสิทธิ์ α_x=0.70 และ α_y=0.30 ซึ่งให้ค่า RMSE รวม (RMSE_{XY}) ต่ำที่สุดในทุกกรณีทดสอบ

การวิเคราะห์เพิ่มเติมจากตาราง 4.3 ยังพบว่า ระบบ Dead Reckoning มีความน่าเชื่อถือในแนวแกน Y สูงกว่ากล้อง (CAM) แต่กลับมีความคลาดเคลื่อนมากกว่าในแกน X ส่งผลให้เมื่อทำการผสมข้อมูล (Fusion) ระหว่าง CAM และ DR ด้วยวิธี Blended EKF สามารถชดเชยจุดอ่อนของแต่ละเซนเซอร์ได้อย่างเหมาะสม ทำให้ค่าความคลาดเคลื่อนรวมลดลงอย่างมีนัยสำคัญทางสถิติ และสะท้อนให้เห็นถึงประสิทธิภาพของวิธีการหลอมรวมข้อมูลแบบถ่วงน้ำหนักที่ออกแบบไว้

4.8.3 อภิปรายผล ข้อจำกัด และข้อเสนอแนะ

อัลกอริทึม Blended EKF ที่พัฒนาขึ้นในงานวิจัยนี้ มีศักยภาพในการเพิ่มความเที่ยงตรงของการระบุตำแหน่งได้ดีกว่า **Standard EKF** อย่างมีนัยสำคัญ นอกจากความแม่นยำที่สูงขึ้น วิธีการดังกล่าวยังช่วยสร้างเส้นทางการเคลื่อนที่ของ AGV ให้มีความต่อเนื่องและคงเสถียรมากยิ่งขึ้น ซึ่งคุณลักษณะนี้มีความสำคัญต่อระบบควบคุมการขับเคลื่อนแบบอัตโนมัติ

ผลลัพธ์ดังกล่าวส่งผลให้ระบบสามารถลดภาระในการควบคุมมอเตอร์ ลดอัตราการสึกหรอของส่วนประกอบทางกล และเสริมระดับความปลอดภัยระหว่างการทำงานในพื้นที่อุตสาหกรรม จึงนับเป็นแนวทางที่มีศักยภาพสูงในการประยุกต์ใช้กับระบบ AGV สมัยใหม่ ซึ่งต้องการทั้งความเสถียรและความแม่นยำในระดับสูงสำหรับการปฏิบัติงานจริง

4.9 สรุป

บทที่ 4 ได้นำเสนอผลการทดลองและอภิปรายผลอย่างเป็นลำดับ เริ่มจากการประเมินประสิทธิภาพของเซนเซอร์รายตัว พบว่า CCTV มีความแม่นยำสูงกว่า UWB และสำหรับ UWB ควรเลือกใช้ ToF ร่วมกับ Trilateration ขณะที่ในสภาวะเคลื่อนที่ Dead Reckoning (Encoder+IMU) ให้ความแม่นยำดีและมีความต่อเนื่องของเส้นทาง แต่ยังมีโอกาสเกิด drift เมื่อเคลื่อนที่นานหรือมีการเลี้ยว จากนั้นผลการปรับปรุงข้อมูลด้วย Bias Correction และ Inverse-Variance Weighting ช่วยลดความคลาดเคลื่อนของข้อมูลก่อนหลอมรวมอย่างชัดเจน เมื่อประมวลผลด้วย Standard EKF พบว่าสามารถลด RMSE ได้ แต่ยังมีอาการแกว่งของเส้นทางจากความผันผวนของข้อมูลกล้อง ขณะที่ Blended EKF ซึ่งผสมข้อมูล CAM–DR ก่อนขั้นตอน Update ให้ผลลัพธ์ที่ดีที่สุด โดยเฉพาะเมื่อใช้

ค่าที่ได้จาก Optuna (α_x=0.70, α_y=0.30) ซึ่งทำให้ RMSE รวมต่ำสุด และเส้นทางมีความต่อเนื่องมากขึ้น พร้อมทั้งใช้ TRI เป็นตัวชี้วัดเสริมเพื่อสะท้อนความราบรื่นของเส้นทางร่วมกับ RMSE

โดยสรุป ผลการทดลองยืนยันว่าแนวทาง Blended EKF สามารถยกระดับความแม่นยำและความเสถียรของการระบุตำแหน่งได้เหนือกว่า Standard EKF ภายใต้เงื่อนไขเซนเซอร์ต้นทุนต่ำ และเหมาะสำหรับการประยุกต์ใช้กับระบบ AGV ในสภาพแวดล้อมจริงที่ข้อมูลมีความไม่แน่นอนและอัตราการสุ่มตัวอย่างแตกต่างกัน

บทที่ 5
สรุปและข้อเสนอแนะ

งานวิจัยนี้มุ่งพัฒนาระบบระบุตำแหน่งของยานพาหนะขับเคลื่อนอัตโนมัติ (AGV) ภายในอาคาร โดยเน้นเพิ่มความเที่ยงตรง ความเสถียร และประสิทธิภาพในการประมวลผลภายใต้ต้นทุนที่เหมาะสม หัวใจสำคัญของงานคือการพัฒนาอัลกอริทึม Blended Extended Kalman Filter (Blended EKF) ซึ่งออกแบบให้สามารถจัดการกับความไม่แน่นอนของสัญญาณจากเซ็นเซอร์หลายประเภทได้อย่างมีประสิทธิภาพ อัลกอริทึมนี้ถูกออกแบบเพื่อรองรับความแปรผันของคุณภาพข้อมูลที่เกิดจากความไม่แน่นอนของเซ็นเซอร์ (Sensor Uncertainty) รวมถึงอัตราการเก็บข้อมูลที่แตกต่างกันของแต่ละอุปกรณ์ (Variable Sampling Rate) โดยใช้แนวคิดการผสมผสานข้อมูลจากหลายแหล่งอย่างเหมาะสม เพื่อเพิ่มความแม่นยำของการประมาณตำแหน่งของ AGV ภายใต้สภาพแวดล้อมภายในอาคารที่มีข้อจำกัดด้านสัญญาณและสิ่งกีดขวาง

จากการทดลองและประเมินผล พบว่า Blended EKF สามารถลดค่าความคลาดเคลื่อนโดยรวมได้ดีกว่าแบบมาตรฐาน (Standard EKF) ทั้งในสภาวะการทดสอบแบบนิ่ง (Static Test) และแบบเคลื่อนที่ (Dynamic Test) โดยให้ค่าความผิดพลาดเฉลี่ยต่ำที่สุดในกรณีที่ใช้การผสมค่าพารามิเตอร์ $\alpha x = 0.7$ และ $\alpha y = 0.3$ แสดงให้เห็นถึงศักยภาพของการรวมข้อมูลเชิงสถิติที่เหมาะสมต่อการเพิ่มความเที่ยงตรงของระบบในเชิงปฏิบัติการ

5.1 สรุปและอภิปรายผลการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนาระบบระบุตำแหน่งภายในอาคารสำหรับยานพาหนะขับเคลื่อนอัตโนมัติ (AGV) โดยเน้นการเพิ่มความแม่นยำ ความเสถียร และความทนทานต่อสภาวะของสัญญาณเซนเซอร์ที่มีความไม่แน่นอนสูง อัลกอริทึม Blended Extended Kalman Filter (Blended EKF) ถูกเสนอขึ้นเพื่อแก้ไขข้อจำกัดของ Standard EKF ในสถานการณ์ที่ข้อมูลจากกล้อง (CAM) มีความผันผวน และข้อมูลจาก Dead Reckoning มีแนวโน้มเกิด drift ตามระยะเวลา

จากการทดลองเปรียบเทียบในหลายเงื่อนไข ทั้งในกรณีการเคลื่อนที่แบบนิ่ง (static) และแบบเคลื่อนที่จริง (dynamic) พบว่าอัลกอริทึม Blended EKF ช่วยลดค่าความคลาดเคลื่อนเชิงตำแหน่ง (RMSE) ได้อย่างมีนัยสำคัญ โดยเฉพาะเมื่อใช้ค่าสัมประสิทธิ์ผสมที่ได้จาก Optuna ได้แก่

$$\alpha_x = 0.70, \alpha_y = 0.30$$

ซึ่งเป็นค่าที่ให้ RMSE ต่ำที่สุดตามผลการค้นหาจาก Optimization History และ Contour Plot เมื่อพิจารณาเปรียบเทียบกับข้อมูล Fused (Raw) ซึ่งได้จากการผสานข้อมูลดิบด้วยวิธี Inverse-Variance Weighting พบว่าอัลกอริทึม Blended EKF สามารถลดค่า RMSE ลงได้ 34.49% ที่อัตราการเก็บข้อมูล 0.1 วินาที และ 24.17% ที่อัตราการเก็บข้อมูล 0.2 วินาที ขณะเดียวกัน เมื่อเปรียบเทียบกับ Standard EKF (Raw) ค่า RMSE ลดลง 21.64% และ 22.56% ตามลำดับ ผลลัพธ์ดังกล่าวแสดงให้เห็นว่าแนวคิดการผสมข้อมูลก่อนขั้นตอนการอัปเดตสถานะของ EKF ช่วยเพิ่มความแม่นยำของการประมาณตำแหน่งได้อย่างมีนัยสำคัญ ขณะเดียวกัน การประเมินคุณภาพเส้นทางด้วย Trajectory Roughness Index พบว่าค่า

$$\alpha_x = 0.30, \alpha_y = 0.30$$

ให้เส้นทางที่ราบรื่นที่สุด สะท้อนให้เห็นความสัมพันธ์แบบ trade-off ระหว่าง “ความแม่นยำของตำแหน่ง” และ “ความต่อเนื่องของเส้นทาง” ซึ่งเป็นประเด็นสำคัญสำหรับระบบควบคุม AGV ในพื้นที่จำกัด

การใช้เทคนิค Bias Correction ก่อนหลอมรวมข้อมูลช่วยลดอคติคงที่ในข้อมูลจากกล้อง ทำให้ค่าที่วัดได้มีความใกล้เคียงค่าจริงมากขึ้น และยังช่วยให้ EKF ปรับปรุงสถานะได้มีเสถียรภาพมากกว่าเดิม ส่วนการใช้ EKF ที่มี sampling time ต่างกัน (dt = 0.1 s และ dt = 0.2 s) ชี้ให้เห็นว่าการเพิ่มความถี่ในการประมาณ (dt = 0.1 s) ช่วยเพิ่มรายละเอียดของเส้นทาง แต่ทำให้ระบบไวต่อ noise ของกล้องมากขึ้น จึงต้องอาศัยการผสมข้อมูลที่เหมาะสมเพื่อปรับสมดุล

โดยสรุป อัลกอริทึม Blended EKF ที่พัฒนาขึ้นสามารถเพิ่มความแม่นยำ ความเสถียร และคุณภาพเส้นทางของ AGV ได้อย่างมีประสิทธิภาพ เหมาะสมต่อการนำไปใช้งานในพื้นที่ภายในอาคารที่มีข้อจำกัดด้านสัญญาณและสิ่งกีดขวาง

5.2 ข้อจำกัดของงานวิจัย

แม้ว่างานวิจัยนี้จะสามารถพัฒนาอัลกอริทึมที่เพิ่มคุณภาพการระบุตำแหน่งได้อย่างมีประสิทธิผล แต่ยังมีข้อจำกัดบางประการที่ควรพิจารณาเพื่อประเมินศักยภาพในการนำไปใช้งานจริง ดังนี้

5.2.1 ข้อจำกัดของสภาพแวดล้อมในการทดลอง (Controlled Indoor Environment)

การทดลองทั้งหมดดำเนินการภายในห้องปฏิบัติการที่มีการควบคุมความสว่าง ลักษณะพื้นผิว และตำแหน่งการติดตั้งกล้อง ส่งผลให้ข้อมูลมีความเสถียรสูงและ noise ต่ำกว่าใน

สถานการณ์จริง เช่น โรงงานที่มีการเคลื่อนย้ายวัตถุตลอดเวลา หรือพื้นที่ที่มีการบดบังภาพ (occlusion) โดยไม่คาดคิด เงื่อนไขที่ควบคุมมากเกินไปอาจทำให้ผลลัพธ์สะท้อนประสิทธิภาพที่ “ดีกว่าสภาพจริง” ดังนั้น จึงยังไม่สามารถสรุปได้ว่าวิธี Blended EKF จะคงประสิทธิภาพเดิมในสภาพแวดล้อมที่ dynamic และไม่สามารถควบคุมได้อย่างสมบูรณ์

5.2.2 ค่าความแปรปรวนของเซนเซอร์ถือว่าเป็นค่าคงที่ (Sensor Statistics Assumed Constant)

ตัวกรอง EKF ในงานนี้ใช้ค่าความแปรปรวนของสัญญาณ (noise covariance) แบบคงที่สำหรับทั้ง CAM และ DR แม้ในการใช้งานจริง noise จะเปลี่ยนตลอดเวลา เช่น

1) กล้องมีความผันผวนสูงขึ้นเมื่อ AGV เร่งความเร็ว
2) encoder error เพิ่มขึ้นเมื่อพื้นมีการลื่นไถล
3) IMU noise เปลี่ยนตามแรงสั่นสะเทือนของตัวรถ

ข้อสมมติเช่นนี้อาจทำให้ตัวกรองประมวลผลไม่สอดคล้องกับสภาพการทำงานจริงตลอดเวลา การใช้เทคนิคการเรียนรู้ค่าความไม่แน่นอนแบบเวลาต่อเนื่อง เช่น **Gaussian Process Regression (GPR)** หรือ **Adaptive Noise Estimation** อาจเป็นทางเลือกที่ช่วยให้การประมาณสถานะมีความยืดหยุ่นและแม่นยำยิ่งขึ้น

5.2.3 การใช้ค่าน้ำหนักผสมแบบคงที่ (Fixed-Weight Blending)

แม้ Optuna จะช่วยค้นหาคู่พารามิเตอร์ (α_x, α_y)ที่เหมาะสมที่สุดสำหรับ RMSE แต่ค่าดังกล่าวยังคงเป็น “ค่าคงที่” สำหรับทั้งเส้นทาง ทั้งที่ในสถานการณ์จริง:

1) ความน่าเชื่อถือของ CAM และ DR เปลี่ยนไปตามสภาพแสงหรือพื้นผิว
2) noise ของเซนเซอร์แต่ละตัวไม่คงที่
3) การเกิด drift หรือ outlier อาจทำให้บางช่วงต้องใช้น้ำหนักต่างจากช่วงอื่น

การใช้ค่าคงที่ตลอดเวลาจึงอาจไม่เหมาะสมในสภาพที่มีความผันผวนสูง จึงควรพัฒนาวิธีการประมวลผลแบบปรับค่าอัตโนมัติ เช่น

1) **Adaptive Blending** ที่ปรับ (α_x, α_y)ตาม uncertainty จริงของแต่ละเซนเซอร์
2) **Reinforcement Learning** เพื่อให้ระบบเรียนรู้วิธีเลือกน้ำหนักที่เหมาะสมในแต่ละสถานการณ์
3) **Gaussian Process Regression (GPR)** สำหรับประมาณระดับความเชื่อมั่นของสัญญาณแบบเรียลไทม์

ข้อจำกัดนี้เป็นประเด็นสำคัญที่ควรได้รับการแก้ไขในงานวิจัยถัดไป

5.3 ข้อเสนอแนะสำหรับงานวิจัยในอนาคต

จากข้อจำกัดที่กล่าวมา สามารถต่อยอดงานวิจัยในทิศทางต่อไปนี้

5.3.1 ทดสอบระบบในสภาพแวดล้อมจริง (Real-World Validation)

ควรทดลองระบบในพื้นที่ที่ไม่สามารถควบคุมได้ เช่น โรงงานอัตโนมัติ คลังสินค้า หรือพื้นที่ที่มีการเคลื่อนไหวสูง เพื่อทดสอบความทนทานของอัลกอริทึมต่อการเปลี่ยนแปลงของแสง พื้นผิว การบดบังภาพ และความแปรปรวนทางกายภาพอื่น ๆ

5.3.2 พัฒนาโมเดลปรับตัวอัตโนมัติสำหรับการเลือกน้ำหนักเซนเซอร์ (Adaptive Online Learning)

ควรออกแบบโมเดลที่สามารถปรับน้ำหนัก (α_x, α_y)แบบอัตโนมัติตามสถานการณ์ เช่น

RL-based weight selection

Neural network ที่ประเมินความเชื่อมั่นของเซนเซอร์

Online bias estimation

เพื่อให้ Blended EKF ปรับตัวตาม noise จริงของเซนเซอร์ได้ต่อเนื่อง

5.3.3 ขยายไปสู่ระบบหลอมรวมข้อมูลหลายแหล่ง (Advanced Multi-Sensor Fusion)

สามารถรวมสัญญาณจาก Depth Camera, LiDAR หรือระบบ Marker-based เพิ่มเติม เพื่อเพิ่มความแม่นยำของตำแหน่งในแกน Z หรือเพื่อเพิ่มความทนทานในพื้นที่ที่กล้องถูกบดบัง

5.3.4 พัฒนาระบบสำหรับหลายยานพาหนะ (Multi-AGV Collaboration)

ควรทดสอบการประสานงานของ Blended EKF ในระบบที่มี AGV หลายคัน เพื่อให้สามารถแลกเปลี่ยนข้อมูลตำแหน่งระหว่างกันแบบกระจาย (Distributed Fusion) เพื่อเพิ่มความปลอดภัยและการทำงานร่วมกันในพื้นที่จำกัด

5.3.5 การประยุกต์ใช้งานในอุตสาหกรรมจริง

ควรทดลองใช้อัลกอริทึมบนแพลตฟอร์ม AGV ที่ใช้งานจริงในโรงงาน เพื่อประเมินประสิทธิภาพภายใต้ภาระงานยาวนาน การสั่นสะเทือนจริง และสภาพการรบกวนแบบ non-stationary ซึ่งเป็นข้อมูลสำคัญสำหรับการพัฒนาสู่ระบบที่พร้อมใช้งานเชิงพาณิชย์

รายการอ้างอิง

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). *Optuna* Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining,

Alcalá, E., Sellart, L., Puig, V., Quevedo, J., Saludes, J., Vázquez, D., & López, A. (2016, 21-24 June 2016). Comparison of two non-linear model-based control strategies for autonomous vehicles. 2016 24th Mediterranean Conference on Control and Automation (MED),

Alharbi, M., & Karimi, H. A. (2021). Context-Aware Sensor Uncertainty Estimation for Autonomous Vehicles. *Vehicles,* 3(4), 721-735. https://doi.org/10.3390/vehicles3040042

Anewtech. (2024). AMR Robot Lift Solutions. https://www.anewtech.net/solutions/ai-robotics/amr-robot-lift-solutions

Ang, J. L. F., Lee, W. K., Ooi, B. Y., & Ooi, T. W. M. (2020, 9-11 March 2020). Location Sensing using QR codes via 2D camera for Automated Guided Vehicles. 2020 IEEE Sensors Applications Symposium (SAS),

Becker, A. (2023). *Kalman Filter from the Ground Up*. KilmanFilter.NET. https://books.google.co.th/books?id=_2fa0AEACAAJ

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). *Algorithms for Hyper-Parameter Optimization* Advances in Neural Information Processing Systems, https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf

Birkl, C. R., Roberts, M. R., McTurk, E., Bruce, P. G., & Howey, D. A. (2017). Degradation diagnostics for lithium ion cells. *Journal of Power Sources,* 341, 373-386. https://doi.org/10.1016/j.jpowsour.2016.12.011

Bouzoualegh, S., Guechi, E.-H., & Kelaiaia, R. (2018). Model Predictive Control of a Differential-Drive Mobile Robot. *Acta Universitatis Sapientiae Electrical and Mechanical Engineering,* 10(1), 20-41. https://doi.org/10.2478/auseme-2018-0002

cgeorge. (2025). Autonomous Mobile Robotics: An Introduction to Theory and Practice. https://www.slideserve.com/cgeorge/autonomous-mobile-robotics-powerpoint-ppt-presentation

Che, F., Ahmed, Q. Z., Lazaridis, P. I., Sureephong, P., & Alade, T. (2023, Jun 19). Indoor Positioning System (IPS) Using Ultra-Wide Bandwidth (UWB)-For Industrial Internet of Things (IIoT). *Sensors (Basel),* 23(12). https://doi.org/10.3390/s23125710

Chen, J., Liu, X., Liu, Y., Chen, L., & Lu, W. (2019, 19-21 Oct. 2019). The Method of AGV Detection and Cargo Status Recognition Based on Globe Vision. 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI),

Chen, Y., Cui, Q., & Wang, S. (2025, May 12). Fusion Ranging Method of Monocular Camera and Millimeter-Wave Radar Based on Improved Extended Kalman Filtering. *Sensors (Basel),* 25(10). https://doi.org/10.3390/s25103045

Cheng, B., He, X., Li, X., Zhang, N., Song, W., & Wu, H. (2024, Aug 2). Research on Positioning and Navigation System of Greenhouse Mobile Robot Based on Multi-Sensor Fusion. *Sensors (Basel),* 24(15). https://doi.org/10.3390/s24154998

Cheng, L., Zhao, Z., Shi, Y., & Lu, Y. (2024, 2024/05/01/). Implicit unscented particle filter based indoor fusion positioning algorithms for sensor networks. *Alexandria Engineering Journal,* 94, 104-119. https://doi.org/https://doi.org/10.1016/j.aej.2024.03.039

Davoudi, M., & Heidarian, A. (2020, 02/24). Real-time Fuzzy Image Processing Method for Visual feedback of a Follower Robot. *WSEAS TRANSACTIONS ON CIRCUITS AND SYSTEMS,* 19, 28-35. https://doi.org/10.37394/23201.2020.19.4

Desai, S., Shrivastava, A., D'Elia, M., Najm, H. N., & Dingreville, R. (2024, 2024/01/15/). Trade-offs in the latent representation of microstructure evolution. *Acta Materialia,* 263, 119514. https://doi.org/https://doi.org/10.1016/j.actamat.2023.119514

Dong, Z. Y., Xu, W. M., & Zhuang, H. (2019, 2019/01/01/). Research on ZigBee Indoor Technology Positioning Based on RSSI. *Procedia Computer Science,* 154, 424-429. https://doi.org/https://doi.org/10.1016/j.procs.2019.06.060

Figliola, R., & Beasley, D. (2011). *Theory and design for mechanical measurements, Fifth edition*.

Firdaus, A. R., Hutagalung, A., Syahputra, A., & Analia, R. (2023). Indoor Localization Using Positional Tracking Feature of Stereo Camera on Quadcopter. *Electronics,* 12(2). https://doi.org/10.3390/electronics12020406

Fragapane, G., Hvolby, H.-H., Sgarbossa, F., & Strandhagen, J. O. (2020). Autonomous Mobile Robots in Hospital Logistics. In *Advances in Production Management Systems. The Path to Digital Transformation and Innovation of Production Management Systems* (pp. 672-679). https://doi.org/10.1007/978-3-030-57993-7_76

Garcia, P. (2023). Calculate X, Y, Z Real World Coordinates from Image Coordinates using OpenCV. https://medium.com/@pacogarcia3/calculate-x-y-z-real-world-coordinates-from-image-coordinates-using-opencv-from-fdxlabs-0adf0ec37cef

Guan, M., Hao, Z., Zhang, S., & Chen, S. (2024, 2024/07/19). A multi-sensor fusion framework for localization using LiDAR, IMU and RGB-D camera. *Measurement Science and Technology,* 35(10), 105112. https://doi.org/10.1088/1361-6501/ad601f

Gupta, S. D., Yu, J. Y., Mallick, M., Coates, M., & Morelande, M. (2015, 6-9 July 2015). Comparison of angle-only filtering algorithms in 3D using EKF, UKF, PF, PFF, and ensemble KF. 2015 18th International Conference on Information Fusion (Fusion),

Guruji, A. K., Agarwal, H., & Parsediya, D. K. (2016). Time-efficient A* Algorithm for Robot Path Planning. *Procedia Technology,* 23, 144-149. https://doi.org/10.1016/j.protcy.2016.03.010

Jang, B., Kim, H., & Kim, J. W. (2019, Jun 29). IPSCL: An Accurate Indoor Positioning Algorithm Using Sensors and Crowdsourced Landmarks. *Sensors (Basel),* 19(13). https://doi.org/10.3390/s19132891

Jiang, P., Chen, L., Guo, H., Yu, M., & Xiong, J. (2021, 2021/03/01). Novel indoor positioning algorithm based on Lidar/inertial measurement unit integrated system. *International Journal of Advanced Robotic Systems,* 18(2), 17298 81421999923. https://doi.org/10.1177/1729881421999923

Kala, R. (2023). *Autonomous Mobile Robots: Planning, Navigation and Simulation*. Academic Press. https://books.google.co.th/books?id=KIusEAAAQBAJ

Khaleghi, B., Khamis, A., Karray, F. O., & Razavi, S. N. (2013, 2013/01/01/). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion,* 14(1), 28-44. https://doi.org/https://doi.org/10.1016/j.inffus.2011.08.001

Li, K., Bao, L., Han, C., Shin, K., & Kim, W. (2022, 27 Nov.-1 Dec. 2022). IMU-based Online Kinematic Model Parameter Estimation for Four-Wheeled Skid-Steering Mobile Robot Desired Motion Tracking on Different Terrains. 2022 22nd International Conference on Control, Automation and Systems (ICCAS),

Liu, L., Wang, B., & Xu, H. (2022). Research on Path-Planning Algorithm Integrating Optimization A-Star Algorithm and Artificial Potential Field Method. *Electronics,* 11(22). https://doi.org/10.3390/electronics11223660

Liu, Y., Wang, S., Xie, Y., Xiong, T., & Wu, M. (2024, Feb 14). A Review of Sensing Technologies for Indoor Autonomous Mobile Robots. *Sensors (Basel),* 24(4). https://doi.org/10.3390/s24041222

MathWorks. (2024). Pure Pursuit Controller. https://www.mathworks.com/help/nav/ug/pure-pursuit-controller.html

Network, A. G. V. (2024). AGV vs AMR: The Truth - 12 Unbiased Differences and Comparison. https://www.agvnetwork.com/agv-vs-amr

Okin, P., Devi, M., Agus, S., & Wynd, R. (2019). Hazardous Waste Handling At Hospital. *ATLR,* 2, 633-639. https://doi.org/10.25292/ATLR.V2I0.218

Pastor, A. (2024). AGV Sensors – The eyes and ears of mobile robots. https://www.agvnetwork.com/mobile-robot-sensors-agv-amr

Perera, L. D. L., Wijesoma, W. S., & Adams, M. D. (2006, 2006/07/01). The Estimation Theoretic Sensor Bias Correction Problem in Map Aided Localization. *The International Journal of Robotics Research,* 25(7), 645-667. https://doi.org/10.1177/0278364906066755

Robotics, A. W. E. (2023). The Benefits of Using ROS in Robotics. https://www.awerobotics.com/the-benefits-of-using-ros-in-robotics/

Rocket Media, L. (2023, 2023/12/29). ปีใหม่ ค่าแรงใหม่ : ค่าแรงขั้นต่ำไทยยุติธรรมหรือไม่ ขึ้นยังไง ขึ้นช่วงไหน ทำไมขึ้นไม่เท่ากัน. https://rocketmedialab.co/minimum-wage-2023/

Roy, S., Petrizze, D., Dihel, L., Xue, M., Dolph, C., & Holbrook, H. (2022, 2022-06-01). *Using Trajectory Smoothness Metrics to Identify Drones in Radar Track Data* Aviation 2022 Forum, Chicago, IL. https://ntrs.nasa.gov/api/citations/20220006977/downloads/aiaa_bird_drone_differentiation_paper_v2_cvd_sr.pdf

Sari, O. (2020). *Sensor Fusion for Localization of Automated Guided Vehicles* Aalto University]. Espoo, Finland. https://aaltodoc.aalto.fi/items/e0 3 3 ade3 - 9 4 aa-4dda-a6ac-2125a6431ea1

Sentech. (2024). Lidar, Autonomous Driving and AGV. https://e.sentech.nl/en/news/lidar-autonomous-driving-and-agv

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms Advances in Neural Information Processing Systems https://proceedings. neurips.cc/paper_files/paper/2 0 1 2 / file/0 5 3 1 1655a15b75fab 86956663e1819cd-Paper.pdf

Song, J., Mei, W., Xu, Y., Fu, Q., & Bu, L. (2024). Practical Implementation of KalmanNet for Accurate Data Fusion in Integrated Navigation. *IEEE Signal Processing Letters,* 31, 1890-1894. https://doi.org/10.1109/LSP.2024.3431443

Stelzer, I. V., Kager, J., & Herwig, C. (2017). Comparison of Particle Filter and Extended Kalman Filter Algorithms for Monitoring of Bioprocesses. In A. Espuña, M. Graells, & L. Puigjaner (Eds.), *Computer Aided Chemical Engineering* (Vol. 40, pp. 1483-1488). Elsevier. https://doi.org/https://doi.org/10.1016/B978-0-444-63965-3.50249-X

Tao, C., Jin, Y., Cao, F., Zhang, Z., Li, C., Gao, H., & Zabalza, J. (2020). 3D Semantic VSLAM of Indoor Environment Based on Mask Scoring RCNN. *Discrete Dynamics in Nature and Society,* 2020, 1-14. https://doi.org/10.1155/2020/5916205

Teng, X., Shen, Z., Huang, L., Li, H., & Li, W. (2024, 2024/06/01/). Multi-sensor fusion based wheeled robot research on indoor positioning method. *Results in Engineering,* 22, 102268. https://doi.org/https://doi.org/10.1016/j.rineng.2024.102268

TheLogisticsIq. (2023). Automated Guided Vehicles (AGV) Market. https://www.thelogisticsiq.com/research/automated-guided-vehicles-agv-market

Trevelyan, J. P., Kang, S.-C., & Hamel, W. R. (2008). Robotics in Hazardous Applications. In B. Siciliano & O. Khatib (Eds.), *Springer Handbook of Robotics* (pp. 1101-1126). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-30301-5_49

Ullah, I., Qian, S., Deng, Z., & Lee, J.-H. (2021). Extended Kalman Filter-based localization algorithm by edge computing in Wireless Sensor Networks. *Digital Communications and Networks,* 7(2), 187-195. https://doi.org/10.1016/j.dcan.2020.08.002

Wan, D., Xu, Y., Li, C., & Zhang, Y. (2024, 2024//). LS-SVM Assisted Multi-rate INS UWB Integrated Indoor Quadrotor Localization Using Kalman Filter. Multimedia Technology and Enhanced Learning, Cham.

Wang, H., & Zhang, X. (2022, 2022/05/01). Real-time vehicle detection and tracking using 3D LiDAR. *Asian Journal of Control,* 24(3), 1459-1469. https://doi.org/10.1002/asjc.2519

Wang, T.-c., Tong, C.-s., & Xu, B.-l. (2020, 2020/02/01). AGV navigation analysis based on multi-sensor data fusion. *Multimedia Tools and Applications,* 79(7), 5109-5124. https://doi.org/10.1007/s11042-018-6336-3

Xue, R., & Liang, Z. (2024, 2024/06/01/). A simulated fusion localization algorithm with adaptive error covariance matrix for closed corridor seamless positioning. *Digital Signal Processing,* 149, 104495. https://doi.org/10.1016/j.dsp.2024.104495

Yuan, C., Wang, Y., & Liu, J. (2022, 2022/12/01). Research on multi-sensor fusion-based AGV positioning and navigation technology in storage environment. *Journal of Physics: Conference Series,* 2378(1), 012052. https://doi.org/10.1088/1742-6596/2378/1/012052

Zhang, L., Zhang, L., Gao, R., Pan, L., Xu, C., & Cheng, K. (2024). Indoor Mobile Robot Localization Applying IMU/Stereo Camera/LiDAR and Graph-Based Optimization. *IEEE Sensors Journal,* 24(13), 21466-21478. https://doi.org/ 10.1109/JSEN.2024.3400269

Zhao, L., & Ding, Y. (2018, 16-17 July 2018). Design of AGV Low-Delay Visual Servo System. 2018 International Conference on Audio, Language and Image Processing (ICALIP).

Zhou, P., Wang, H., Gravina, R., & Sun, F. (2024). WIO-EKF: Extended Kalman Filtering-Based Wi-Fi and Inertial Odometry Fusion Method for Indoor Localization. *IEEE Internet of Things Journal,* 11(13), 23592-23603. https://doi.org/10.1109/JIOT.2024.3386889

Zong, C., Ji, Z., Yu, Y., & Shi, H. (2020). Research on Obstacle Avoidance Method for Mobile Robot Based on Multisensor Information Fusion. *Sensors and Materials,* 32(4). https://doi.org/10.18494/sam.2020.2540

ภาคผนวก ก

รายชื่อบทความทางวิชาการที่ได้รับการตีพิมพ์เผยแพร่ในระหว่างศึกษา

รายชื่อบทความทางวิชาการที่ได้รับการตีพิมพ์เผยแพร่ในระหว่างศึกษา

Nopparut Khaewnak, Siripong Pawako, Akkharachai Kosiyanurak, Suradet Tantrairatn and Jiraphon Srisertpol (2025). “Low-Cost Multi-Sensor Localization for Indoor AGVs” September 19,2025.

ASPG Publishing, LLC
Editorial Office
e-mail: support@americaspg.com

September 19, 2025

Paper Acceptance letter

Paper Title:

Low-Cost Multi-Sensor Localization for Indoor AGVs

Nopparut Khaewnak[1], Siripong Pawako[1], Akkharachai Kosiyanurak[1], Suradet Tantrairatn[1], Jiraphon Srisertpol [1]•

[1]School of Mechanical Engineering, Institute of Engineering, Suranaree University of Technology), Nakhon Ratchasima, Thailand

Emails: d6500412@g.sut.ac.th; siripongpawako@gmail.com; m6501891@g.sut.ac.th; suradetj@sut.ac.th, jiraphon@sut.ac.th

Dear Authors,

We are delighted to send you this acceptance letter to inform you that your paper with the above details has been accepted for publication at Journal of Intelligent Systems and Internet of Things (JISIoT), Congratulations!
Your paper will be published in the coming issue.

For more information, please visit the journal website using this link
https://www.americaspg.com/journals/show/18

Best Regards,

A. Mark

ASPG Editorial Office,
www.americaspg.com
editorialoffice@americaspg.com

Journal of Intelligent Systems and Internet of Things
Vol. 17, No. 02, PP. 295-310-, 2025
DOI: https://doi.org/10.54216/JISIoT.170219

Low-Cost Multi-Sensor Localization for Indoor AGVs

Nopparut Khaewnak [1], Siripong Pawako[1], Akkharachai Kosiyanurak[1], Suradet Tantrairatn[1], Jiraphon Srisertpol[1,*]

[1]School of Mechanical Engineering, Institute of Engineering, Suranaree University of Technology), Nakhon Ratchasima, Thailand

Emails: d6500412@g.sut.ac.th; siripongpawako@gmail.com; m6501891@g.sut.ac.th; suradetj@sut.ac.th; jiraphon@sut.ac.th

Abstract

Indoor transportation systems are a key area of development, where Automated Guided Vehicles (AGVs) help to increase efficiency and reduce labor costs. However, high-precision positioning technologies such as LiDAR and GNSS are expensive, making them unsuitable for widespread use. This research has developed a low-cost positioning system for indoor AGVs using multiple sensors, including CCTV, UWB, inertial measurement units (IMUs), and encoders. The experiment was carried out under both static and dynamic conditions. In static tests, Trilateration distance measurements show a lower positioning error than the triangular method, with a maximum error of 1.4464 m (x-axis) and 1.0464 m (y-axis) in dynamic tests. The integrated Encoder and IMU sensor data yielded the lowest error (RMSE = 0.0732 m at 0.4 m/s, 0.0678 at 0.27 m/s), Next is CCTV, while UWB has the highest error rate. The application of a Parallel Sensor Fusion architecture optimized using a Generalized Reduced Gradient (GRG) nonlinear algorithm, significantly reduced localization errors. The RMSE values decreased to 0.0623 m (0.4 m/s) and 0.0411 m (0.27 m/s). The results, in a controlled environment laboratory, indicate that combining multiple sensors will improve the positioning accuracy. Combining the encoder and IMU effectively reduces accumulated errors and increases system stability. While Adjust the weight of the sensor offline, this proposed system offers a cost-effective positioning solution for indoor AGVs, which contributes to the development of affordable and accurate AGV navigation systems.

Received: January 01, 2025 Revised: March 01, 2025 Accepted: June 05, 2025

Keywords: Automatic Guided Vehicles (AGVs); Positioning Technology; Low-cost Sensors; Multi-Sensor Fusion; Localization

1. Introduction

Freight transportation is fundamental to the manufacturing industry. Starting with human and animal labor, assistive devices such as trolleys and conveyors have been developed to increase transportation efficiency [1]. Modern smart factory transportation systems, a key component of Industry 4.0, are designed to operate automatically. Automated Guided Vehicles (AGVs) are intelligent systems that increase accuracy and reduce costs. These AGVs act as "delivery vehicles", communicating within the Internet of Things (IoT) system by collecting and transmitting data through a centralized control system. However, AGVs used in industrial factories operate indoors.

AGVs used in industrial facilities often operate in indoor environments where satellite positioning systems such as the Global Positioning System (GPS) or Global Navigation Satellite System (GNSS) are not accurately operational because the signal is interfered with by obstacles [2, 3]. This challenge has driven advances in indoor positioning technologies, including the Affordable Indoor Navigation Solution (AINS) [4], which has different types of sensors for positioning. Automated guided vehicles (AGVs) rely on various sensors for precise positioning and navigation. Among these, LiDAR (Light Detection and Ranging) is the most widely used technology due to its high accuracy and accuracy in environmental mapping [5]. The high cost is still a major disadvantage [6, 7]. Cameras are also often used to create maps and detect objects. However, conventional monocular cameras cannot

measure depth [8, 9], so accessories such as ArUco markers [10] or light-emitting diode panels [11] are required. (LED) for better spatial perception.

Inertial units (IMUs) are used to track the movement of AGVs, but their accuracy decreases over time due to accumulated errors [12]. While these technologies can improve AGV navigation efficiency, the high cost of LiDAR and camera limitations have prompted research into cost-effective alternative positioning solutions [13]. New approaches such as Ultra-Wideband (UWB), closed-circuit television (CCTV), IMUs, and encoders are being considered to replace LiDAR sensors in the indoor AGV, Which provides a reasonable price for accuracy

Ultra-wideband (UWB) technology is a reliable solution for AGV positioning, however, the performance of this technology depends on optimal transmitter positioning to ensure accurate line-of-sight (LoS) transmission [14]. The accuracy of this technology can be further improved by incorporating Kalman filters to reduce noise and improve state estimation [15]. In addition, closed-circuit television (CCTV), a technology that is widely used in industrial environments, is widely used in industrial environments. AGV navigation can be supplemented by route monitoring and identifying areas with a high density of obstacles, which improves overall performance [16].

Sensor integration is an important technique to reduce errors and increase the accuracy of low-cost sensors in AGV navigation. The sensor data is processed sequentially, and the processing is parallel, where multiple sensors operate simultaneously. A hybrid approach that combines the two methods can further optimize the sensor by taking advantage of the strengths of each [17–19].

The integration of multiple sensors greatly improves the positioning accuracy of AGVs. For example, combining UWB and LiDAR yields an average squared error (RMSE) of 36.3 cm, while combining Odometry and LiDAR yields an RMSE of 43.4 cm [20].

While numerous techniques have been developed for indoor AGV localization through sensor fusion, many existing studies still rely on expensive sensors like LiDAR or RGB-D cameras, making them unsuitable for cost-sensitive AGV platforms. Additionally, there has been relatively little attention paid to how real-world factors—such as sensor noise and inconsistent sampling rates—impact localization accuracy when using affordable, mixed-sensor setups like UWB, CCTV, IMU, and encoders. Therefore, the objective of this research is to develop a low-cost, multi-sensor localization framework for indoor AGVs. The proposed system integrates UWB, CCTV, IMU, and encoder data using a parallel sensor fusion structure. A Generalized Reduced Gradient (GRG) optimization algorithm is employed to dynamically determine the optimal sensor fusion weights that minimize positioning Root Mean Square Error (RMSE) in varying speeds. This document is structured as follows: Part II describes in detail the positioning methods and functions of each sensor used in this study. Part III describes the characteristics of the sensor. Part IV presents the results of the experiment. Analyse positioning accuracy for fixed conditions and move at different speeds. In addition, Generalized Reduced Gradient (GRG) techniques are used to optimize calculations in determining the optimal sensor set. Although Kalman Filter is popular, this research uses GRG to reduce the computational resources for optimum sensor weighting configuration. Part V summarizes the findings and key conclusions of this research.

2. Material and Methods

In this section, we describe the methodology for determining the position and function of each type of sensor used in this research. The study employs the BALTY AGV, an autonomous mobile robot equipped with multiple sensors, designed to determine its position within the experimental area, as illustrated in Figure 2. To minimize costs, the system utilizes low-cost sensors instead of LiDAR, which is commonly used in traditional positioning but is expensive.

The developed positioning system integrates internal and external sensor data to achieve high accuracy. The AGV positioning process can be divided into two main approaches: (1) internal positioning, which is inside the AGV such as encoders and IMUs, and (2) external positioning, which uses external sensors such as UWB and CCTV. By combining data from multiple sensors.

2.1 System Architecture and IoT Integration

The presented AGV operational concept is semi-autonomous. The core of the processing is the Jetson Nano board connected to the teensy board to control the JGB37-520 DC motor for navigation and obstacle avoidance. The system uses a Camera (Tapo C200) as the main sensor for image processing to allow the robot to move along the specified path, along with Ultrasonic Sensors installed to detect and avoid obstacles, as shown in Figure 1.

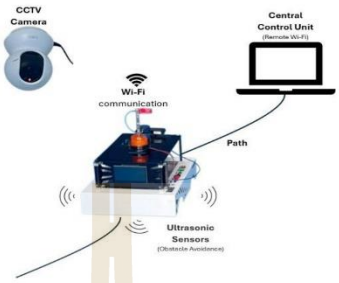


Figure 1. Conceptual Diagram of AGV Operation with CCTV, Ultrasonic Sensors, and Central Wi-Fi Control.

The BALTY AGV robot (as shown in Figure 2) is used for localization. The system integrates both Internal and external sensors. Internal sensors include a motor encoder for estimating position from PWM signals and an IMU (WitMotion WT901C) for measuring movement direction. External sensors include a UWB module (ESP32 UWB DW1000) used to measure the distance between tags mounted on the AGV and anchors placed around the test area. Additionally, a CCTV camera (Tapo C200) is used to process images to determine the AGV's location. Data from all sensors is transmitted over a Wi-Fi network using the MQTT protocol for real-time communication. The data is then forwarded to the Central Control Unit (CCU) (as shown in Figure 3), which is crucial for data fusion, route planning, and traffic management. Moving the processing load to the CCU significantly reduces hardware costs and complexity on each robot, while increasing accuracy and coordination capabilities across multiple AGVs.

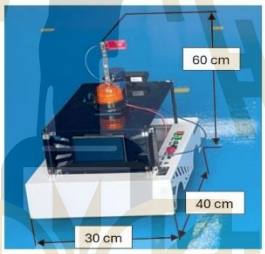


Figure 2. AGV BALTY ROBOT.

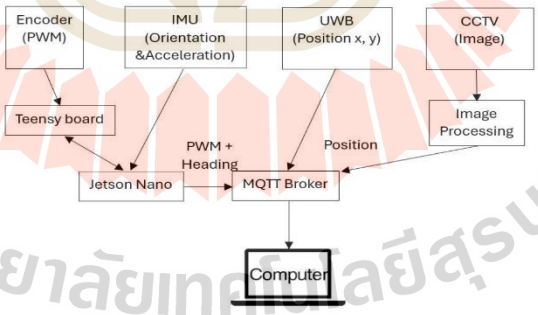


Figure 3. Communications of sensors.

2.2 Sensor Specifications

2.2.1 Encoder Sensor

The encoder is a sensor attached to the DC Motor (JGB37-520) located on the back of the motor shaft. This encoder uses a Double Magnetic Hall Encoder Disc that provides an AB-phase (Quadrature Signal) output signal. This makes it possible to analyze the direction and number of revolutions of rotation. An operating voltage of 3.3–5 V DC provides a frequency signal of 90 pulses per revolution of the motor shaft, as shown in Figure 4.

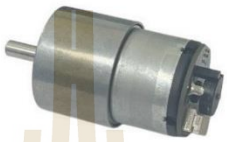


Figure 4. DC Motor with Encoder Sensor.

2.2.2 Inertial Measurement Unit Sensor

IMU (Inertial Measurement Unit) WitMotion WT901C is a 9-axis sensor (9 DOF – Degrees of Freedom) that can measure a variety of values, including linear acceleration, rotational rate, and other factors. The WT901C supports measurement of acceleration in the range of ±16g and rotational rates of up to ±2000 degrees per second (°/s). In this study, the output rate of the IMU is configured to 10 Hz, as shown in Figure 5.

Figure 5. IMU Sensor.

2.2.3 UWB Sensor

UWB consists of an ESP32 UWB board equipped with a DW1000 module, which can transmit and receive UWB wave signals for distance measurement with Time of Flight (ToF) techniques and distance estimation based on signal strength or RSSI (Received Signal Strength Indicator), as shown in Figure 6. The Anchor installation is defined at the 4 points of a 4×8 meter rectangular area, fixed to the bracket perpendicular to the ground, so that the signal can be received evenly from all directions, as shown in Figure 8. The tag mounted on the AGV measures the distance from the anchor at 4 points to calculate the 2D coordinates by trilateration or triangulation technique, and then sends the location coordinates back to the central processor via WiFi using the MQTT protocol.

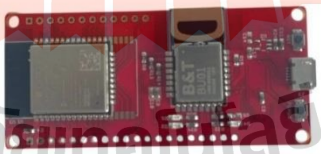


Figure 6. UWB Sensor.

2.2.4 CCTV

There are 4 CCTVs used in the system, that is TP-Link Tapo C200, which is a wireless IP camera (WiFi Camera) that can be connected via a wireless network to transmit images directly to the central processing system. The CCTV also supports low-light operation with infrared at a wavelength of 850 nm (850 nm IR LED), which can be seen at a distance of up to 30 feet. This camera comes with a lens with an f/NO aperture of 2.4 and a focal length of 4 mm. This is ideal for indoor medium-distance photography. The camera can record video at the highest resolution of Full HD (1080p) at 30 frames per second. It uses the H.264 video compression standard to efficiently transmit real-time image data over a WiFi network, as shown in Figure 7.The CCTV is installed at four CCTV cameras of the indoor experimental area (4 m × 8 m), as shown in Figure 8.

Figure 7. CCTV Sensor.

2.3 Localization Techniques

AGV localization has both outdoor localization that can use GPS sensors, CCTV, and Lidar, which is the main sensor for localization, and indoor localization cannot use GPS sensors. Most of the users will use LiDAR CCTV for localization but the price of LiDAR is high so it is the origin of using low-cost sensors, which will be localized in 2 forms according to the reading of the sensor.

2.3.1 Internal Localization (Encoder and IMU)

Localization determination from inside the AGV is done by using sensors to measure values from the AGV itself. The internal sensors measure values from the encoder and the IMU. AGV localization determination from the encoder sensor and its integration with heading Localization determination of an AGV using a two-wheel differential drive system can be efficiently achieved by combining information from the initial encoders with a mathematical model of the AGV, as shown in Figure 9(a).

The position of the AGV can be found by defining two coordinate systems. The first coordinate system is the global coordinate system on the frame (X_0, Y_0) and the second system is the robot coordinate system (X_r, Y_r) as shown in Figure 9(a). It can be seen that the two frames are defined, with the origin of the robot frame set to be the midpoint A on the axis between the wheels, the center of mass C of the robot is considered to be on the axis of symmetry, at a distance d from the origin A, with the radius of the wheel R and the distance from the center of the wheel L.

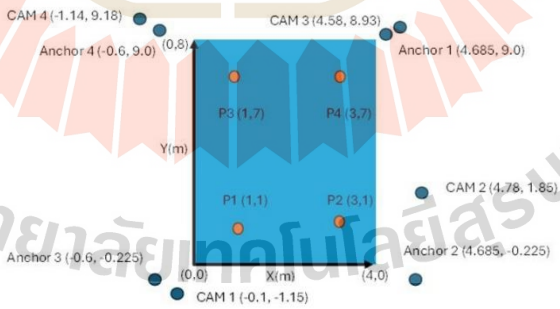
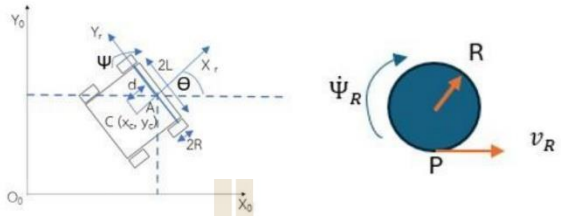


Figure 8. CCTV Sensor.



(a) Differential drive [21] (b) Right wheel [22]

Figure 9. Two components of the robot.

The kinematics and dynamics modelling of a two-wheeled self-driving AGV (DDMR, Differential Drive Mobile Robots) is designed since this equation describes the relationship between the velocity and the robot frame. The motion of the DDMR is characterized by non-holonomic constraint equations based on a set of assumptions [22, 23]. The first assumption is that there is no sliding motion. This constraint only means that the robot can move in a curved manner (forward and backward), but not sideways. In the robot frame, this assumption means that the velocity of center A is zero. The second assumption is the rotation of the wheels. There is one point of contact P with the ground as shown in Figure 9(b). There is no wheel slip in the longitudinal axis (X_r) and no slip in the perpendicular axis (Y_r). The velocity of the contact point in the robot frame is related to the velocity of the driven wheels in conjunction with the robot geometry, where the linear velocity of each driven wheel is the average of the linear velocities of the two wheels.

$$v = \frac{v_R + v_L}{2} = R\frac{\dot{\varphi}_R + \dot{\varphi}_L}{2} \quad (1)$$

Where v_R is the speed of the right wheel (m/s), v_L is the speed of the left wheel (m/s), $\dot{\varphi}_R$ is the angular speed of the right wheel (rad/s), $\dot{\varphi}_L$ is the angular speed of the left wheel (rad/s), And the angular velocity of DDMR, i.e.,

$$\omega = \frac{v_R + v_L}{2L} = R\frac{\dot{\varphi}_R - \dot{\varphi}_L}{2L} \quad (2)$$

To determine the position and velocity, it can be calculated as a function of velocity from the center of A, i.e.,

$$\begin{aligned} X &= X_A + vdt\cos\theta \\ Y &= Y_A + vdt\sin\theta \\ \dot{X} &= \dot{X}_A + v\cos\theta \\ \dot{Y} &= \dot{Y}_A + v\sin\theta \\ \theta &= \theta_0 + \omega dt \end{aligned} \quad (3)$$

Where X is the position in the x-axis, Y is the position in the y-axis, $\dot{X}$ is the velocity in the x-axis of the AGV, $\dot{Y}$ is the velocity in the y-axis of the AGV, $\dot{X}_A$ is the velocity in the x-axis of point A, $\dot{Y}_A$ is the velocity in the y-axis of point A, θ_0 is the initial angle of the AGV concerning the x-axis, dt is the change in time (Delta-time), θ is the angle of the AGV or heading of the AGV. and read the heading from the IMU sensor as an angle and use to replace the angle that occurs.

2.3.2 External Localization (UWB and CCTV)

External positioning is the use of external sensors to measure the position of an object, which is often already in use and is a low-cost sensor. Positioning from the sensor has

2.3.2.1 Localization from UWB sensor

UWB (Ultra-Wideband) position tracking from the Received Signal Strength Indicator (RSSI) and distance (d) obtained from 3 UWB sensors [24]. The distance d can be calculated from the RSSI using the equation.

$$d = 10^{\frac{A-RSSI}{10n}} \quad (4)$$

Where A is the measured signal strength at a 1-meter distance (dBm), RSSI is the measured signal strength (dBm), and n is the path loss exponent.

Another method is to use the distance from the UWB Time of Flight (ToF) to calculate the location by measuring the distance based on the time the signal travels from the Tag to the Anchors, which uses Ultra-Wideband (UWB).

From both methods, distance can be measured, which is divided into 2 calculation methods. When the distances d_1, d_2, and d_3 from the 3 UWB sensors (Anchor 1, Anchor 2, Anchor 3) from the positions (X_1, Y_1), (X_2, Y_2) and (X_3, Y_3) of the UWB sensors, respectively, the trilateration principle can be used to calculate the position (X, Y).

$$(X - X_1)^2 + (Y - Y_1)^2 = d_1^2 \quad (5)$$

$$(X - X_2)^2 + (Y - Y_2)^2 = d_2^2 \quad (6)$$

$$(X - X_3)^2 + (Y - Y_3)^2 = d_3^2 \quad (7)$$

From Eq. (5, 6 and 7) we can solve this system of equations to find the x and y positions of the AGV.

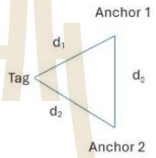


Figure 10. The triangular method.

Method 2 uses the trigonometry principle, using trigonometry equations, to find the distances d1 and d2 from the two UWB sensors (Anchor 1 and Anchor 2) from the positions (X_1, Y_1) and ((X_2, Y_2) of the UWB sensors, respectively, and the distance between Anchor 1 and Anchor 2 (d_0) is shown in Figure 10. using the equation.

2.3.2.2 Localization from CCTV

For localization tracking use CCTV to find coordinates. From the image in the area of interest and the location, it can be seen that the image is distorted due to the lens inside the camera, which is corrected by placing the image on a chessboard and detecting it. The image is passed through a mathematical model with a regression coefficient to create a straight line and the robot's position can be determined using an algorithm to detect and locate objects from images [25].

2.4 Multi-Sensor Data Fusion Algorithm

2.4.1 Parallel Fusion Algorithm

Fusion sensor architectures are divided into three types: series: operation based on one output as the next, parallel: Simultaneous operation and combination: series and parallel fusion, with the architecture divided into two types according to the type of data fusion: the first type is voting, weighting, and priority voting, and the second type is divide and conquer. The dataset is divided into subsets of the same size. The classification is then carried out, followed by the inclusion of decision data based on the results of those subsets of the classification [18].

2.4.2 Generalized Reduced Gradient (GRG) Method: A Nonlinear Optimization Algorithm

The Generalized Reduced Gradient (GRG) method is an effective algorithm used to solve non-linear optimization problems, extending the capabilities of the Reduced Gradient method by effectively addressing the constraints of inequality. The process starts by dividing the variable into a base variable (y) and a non-base variable (x) and then reducing the dimension of the problem by displaying y as a function of x, resulting in the target function $F(x) = f(y(x), x)$. This method turns the problem into a minimized $F(x)$ within the scope of $l \le x \le u$ [26, 27].

$$\Delta F(x) = \frac{\partial f}{\partial x} - \frac{\partial f}{\partial y}\left(\frac{\partial g}{\partial y}\right)^{-1}\frac{\partial g}{\partial x} \quad (8)$$

where:

$\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$ *are the gradients of the objective function.*

$\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}$ *are the Jacobian matrices of the constraints.*

GRG is a precise and versatile algorithm for the computation of nonlinear equations. This algorithm is commonly used in software such as Microsoft Excel Solver [28, 29], leveraging techniques such as numerical differentiation and lower-upper factorization to optimize [26, 27] calculations. The GRG stands out for its complex equation management, making it ideal for using [30]. This algorithm requires heavy computation, especially for large matrices, and that converges into a locally appropriate value instead of a generic optimal value [29]. Starting the parameters correctly is considered important for achieving optimal results. Despite these challenges, GRGs remain a powerful tool for non-linear computations due to their robustness and flexibility in reducing selected error statistics [30].

In this study, the GRG algorithm was applied offline to find the most appropriate weighting coefficients (a, b, c, d, and e) for the sensor fusion model. This process is not a real-time adjustment, but rather, the pre-recorded data from each path were processed using the Solver tool in Microsoft Excel. The GRG Nonlinear method was used to adjust the weighting coefficients until the lowest Root Mean Square Error (RMSE) was obtained when comparing the path obtained from the fusion to the ground truth path. This pre-calculated optimal weighting coefficient set was then used in the sensor fusion model for the experiment.

3. Experimental Design

The experimental design in this study focuses on the use of low-cost sensors to localize AGVs and partition the experiments into two main parts: static experiments and dynamic experiments to verify the accuracy of the sensors and the performance of the data fusion techniques. For the static experiment, the test is performed in a predetermined localization within a 4x8 square meter test area, testing position readings from external sensors such as UWB and CCTV, and comparing the different techniques used to calculate the position. The dynamic experiment focuses on the localization test while The AGV moves in a straight line under specified conditions, collecting position data from UWB and CCTV, as well as Pulse Width Modulation (PWM) from encoders and direction values from IMUs, which can be used to analyse and improve the accuracy of the AGV's positioning system. Both of these experiments aim to evaluate the capabilities of various sensors and improve the accuracy of AGV positioning so that they can be used as a practical guideline.

3.1 In static experiments

In experiments, four conditions are determined: (1,1), (3,1), (1,7), and (3,7), as shown in Figure 8, within a 4x8 square meter test area. The AGV moves in four directions at angles of 0, 90, 180, and 270 degrees relative to the x-axis, repeating each movement three times. The installation points of UWB and CCTV anchors are shown in Figure8. The study involves comparing Time-of-Flight signal readings with RSSI reading methods, evaluating the use of the 3P method (trilateration) against the 2P method (triangular), analyzing the position of anchors used in calculations, and comparing UWB and CCTV for localization accuracy.

3.2 In dynamic experiments

In experiments, linear motion is analysed by collecting position data from UWB in the specified conditions, obtaining location data from CCTV, which are processed using a vision-based localization algorithm [25], gathering Pulse Width Modulation (PWM) data from encoders, and recording vehicle direction data from the IMU at average speeds of 0.27 m/s and 0.4 m/s. The study involves comparing movement data obtained solely from the encoder with data incorporating direction readings from the IMU, utilizing average Weighted Parallel Fusion, applying Weighted Parallel Fusion customized in Microsoft Excel for each moving path, and implementing Weighted Parallel Fusion from combined path customizations in Microsoft Excel.

3.3 Timestamp and Communication Management

To ensure data integrity for the data fusion algorithm, the data collection process has specific sampling rates: 10 Hz for UWB IMU wheel encoders and 5 Hz for CCTV. Since data from each sensor is collected asynchronously, the temporal alignment is performed using event-based synchronization in the post-processing step. We set the 'AGV start of movement' as the reference marker to synchronize all datasets. For sensor data communication, the MQTT protocol is used, which has a basic QoS configuration that emphasizes fast data transmission. However, this method may occasionally cause data loss under unstable Wi-Fi network conditions.

4. Results and Discussion

This experiment was divided into two types, namely a Static Experiment, and a Dynamic Experiment, to study the accuracy of the location of each sensor, which consisted of UWB, CCTV, IMU, and Encoder. After that, the position values obtained from each sensor were used to fuse the data (Parallel Fusion) to find the best results. For UWB positioning, the ESP32 DW1000 UWB Module BU01 is used, with anchors placed at all four corners, as shown in Figure 5. The tag is mounted at the center of the AGV. The system reads the signal data as the distance

between the anchors and the tag, then calculates the position and transmits it to MQTT. The obtained position is retrieved and recorded into the AGV through MQTT. For CCTV-based positioning, TP-Link Tapo C200 cameras are used, positioned at four locations. These cameras provide Full HD 1080p resolution and connect via the Real-Time Streaming Protocol (RTSP) to a computer, where the position is calculated. The computed position is then transmitted to MQTT and recorded into the AGV through MQTT. In the static experiment, 4 points are set: (1,1), (3,1), (1,7), and (3,7) by analysing UWB from the recorded distance values to select the measurement method, which has 2 types: RSSI and Tof. When the appropriate measurement method is obtained, the appropriate calculation method is selected, divided into 2 types: trilateration and triangular, to select the appropriate method and use the method used in the experiment together with CCTV to find the error value that occurs. The dynamic experiment focuses on testing AGV positioning while it moves in a straight line under predefined conditions. Position data is collected from both UWB and CCTV. An encoder attached to the motor is used, which counts the pulse width modulation (PWM) signals from the wheel encoders on both front wheels to calculate the position and record it directly into the AGV. Similarly, an IMU (WitMotion WT901C) is connected to the AGV and works with the encoder to compute the position, which is then recorded directly into the AGV. The UWB, encoder, and IMU sensors record data approximately every 0.1 seconds, while the CCTV records the most recently obtained values. From the position values read from each type of sensor, the weight Parallel Fusion method is used by using the weight value from the average of equal weights, from the weight value of the GRG method of each route, and the weight value of the GRG method of both routes combined. The results can be displayed according to the topic.

4.1 static experiments Results

Static experiment using 4 positions with the car facing in 4 directions. The experiment was repeated 3 times, and the experimental results can be presented as follows.

4.1.1 Comparing the ToF signal value with the RSSI value using 2 calculation methods: trilateration distance measurement and triangular distance measurement. It was found that the highest error value was at the x-axis RSSI value with a fault value of 9.2825 m (The trilateration is computed using anchor 4,1,2 at position (1,1) and angle 0 degree in the 2 experiment) meters and the y-axis error value was 7.6216 m (The triangular is computed using anchor 1,2 at position (3,1) and angle 90 degree in the 3 experiment). It was found that the RSSI value was higher than the ToF reading from the x-axis with a maximum error of 1.4828 m (The triangular is computed using anchor 3,4 at position (1,7) and angle 270 degree in the 2 and 3 experiment), and the y-axis had the highest error value of 1.8054 m (The triangular is computed using anchor 4,1 at position (3,7) and angle 180 degree in the 2 and 3 experiment). It was found that the ToF reading method should be selected based on the results of the experiment as shown in Table 1.

Table 1: Summary of error results comparing ToF and RSSI with two calculation methods.

Comparison	Case/Method	Key Finding (Error Value)	Conclusion
RSSI Maximum Error	Trilateration (Anchor 4,1,2; pos (1,1); angle 0°; Exp 2)	9.2825 m (x-axis)	RSSI has very high error.
RSSI Maximum Error	Triangular (Anchor 1,2; pos (3,1); angle 90°; Exp 3)	7.6216 m (y-axis)	RSSI error is unstable in y-axis.
ToF vs. RSSI (Difference)	Triangular (Anchor 3,4; pos (1,7); angle 270°; Exp 2-3)	ToF lower by 1.4828 m (x-axis)	ToF more reliable than RSSI.
ToF vs. RSSI (Difference)	Triangular (Anchor 4,1; pos (3,7); angle 180°; Exp 2-3)	ToF lower by 1.8054 m (y-axis)	ToF more reliable than RSSI.

4.1.2 In comparison of the use of the 3P method (trilateration) and the 2P method (triangular), the maximum error of the x-axis using the triangular method is 1.4824 m (The triangular is computed using anchor 3,4 at position (1,7) and angle 270 degree in the 2 and 3 experiment), the maximum error value of the y-axis when using the triangular method is -1.8054 m (The triangular is computed using anchor 4,1 at position (3,7) and angle 180 degree in the 2 and 3 experiment), the maximum error value of the x-axis when using the trilateration method is 1.4464 m (The

trilateration is computed using anchor 1,2,3 at position (1,7) and angle 90 degree in the 2 and 3 experiment), and the maximum error value of the y-axis when using the triangular method is 1.0464 m (The trilateration is computed using anchor 4,1,2 at position (1,1) and angle 180 degree in the 2 and 3 experiment) as shown in Table 2.

4.1.3 When comparing the positions of the anchors used in the calculations. When at position X=1 Y=7, which is close to position 2,3,4 and has the least error value. Position X=3 Y=1, which is close to position 1,2,3, found that the calculation from anchor points 4,1,2 had the lowest error value, but position 1,2,3 had a lower error value as shown in Table 3. It was found that 2 out of 4, the nearest calculation position has the least error value. Therefore, the nearest Anchor placement position was selected for calculation.

Table 2: The table shows the static error test.

Type	Reading	Method	x (m)	y (m)
UWB	RSSI	Trilateration	9.2825	—
	RSSI	Triangular	—	7.6216
	ToF	—	1.4828	1.8054
	ToF	Trilateration	1.4464	1.0464
CCTV	—	—	0.087	0.214

Table 3: The error results using ToF trilateration calculation.

Position		RMSE (Anchor Combination)			
x	y	1,2,3	2,3,4	3,4,1	4,1,2
1	1	0.75	0.92	1.23	1.12
3	7	0.77	0.68	0.55	0.58
1	7	0.96	0.74	0.33	0.56
3	1	0.51	0.76	0.96	0.77

4.1.4 The comparison between UWB and CCTV can be shown in Figure 11. when comparing the real position with the CCTV position obtained using the method of vision-based localization algorithm, and the UWB position obtained from CCTV is more accurate. Less dispersed. The positioning accuracy analysis CCTV shows that at position (1,1) with an orientation of 270 degrees, the maximum X error was 0.087 m, with an RMSE of 0.057061 m. At position (1,7) with an orientation of 180 degrees, the maximum Y error was 0.214 m, with an RMSE of 0.097567 m. The overall total RMSE for the system was 0.113028 m.

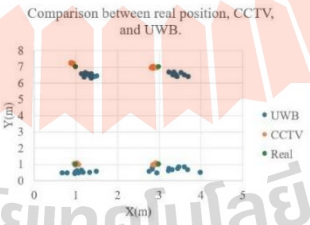


Figure 11. Comparison between real position, CCTV, and UWB.

The results of the error values that occurred in the static experiment can be summarized as shown in Table 2.

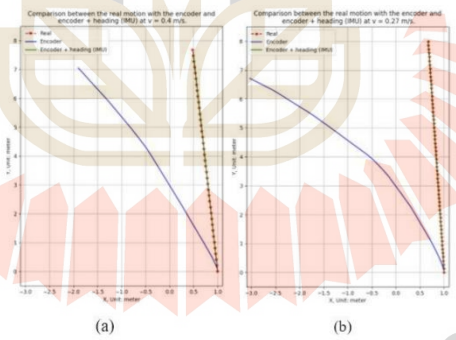
4.2 Dynamic experiment Results

Dynamic experiment by testing 2 speeds at average speeds of 0.4 and 0.27 m/s, which is given the starting point at x = 1, y = 0, moving in a straight line with the wheel radius of 0.052 m, and the distance between the wheels of 0.379 m, and the experimental results can be presented as follows.

4.2.1 Once the starting position is set, collect data that reads the Pulse Width Modulation (PWM) from the direction encoder from the speed difference of the two wheels (Encoder). Compare the direction obtained from the IMU (Encoder + heading (IMU)) using the moving speed from the average of the two wheels. The first path of the movement moved from the position x = 1, y = 0 to x = 0.48, y = 7.68 m with the locomotive starting at an angle of 93.87°, the second path moved from the position x = 1, y = 0 to x = 0.67, y = 8 m with the starting head at an angle of 92.4°. It can be done as shown in Figure 12(a), 12(b).

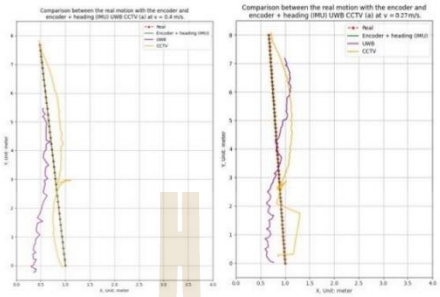
When implementing each type of sensor, it can be displayed as in Figure 13(a)., 13(b). Selected to calculate the position from encoder + heading (IMU) at a speed of 0.4 m/s with a root mean square error (RMSE) of 0.0035 m (x-axis) 0.0731 m(y-axis) and a total RMSE value of 0.0732 m with a maximum error of 0.0101 m(x-axis at(0.48,7.68)) 0.048 m(y-axis at(0.496421,7.437474)), speed 0.27 m/s with a RMSE of 0.0021 m (x-axis) 0.0678 m(y-axis) and a total RMSE value of 0.0678 m with a maximum error of 0.0034 m (x-axis at (0.7169,6.861)) 0.1086 m(y-axis at (0.7046,7.1593)), calculate the position from UWB at a speed of 0.4 m/s with a RMSE of 0.3317 m (x-axis) 1.1858 m (y-axis) and a total RMSE value of 1.2313 m with a maximum error of 0.6562 m(x-axis at(0.956211,0.646737)) 2.2 m(y-axis at(0.48,7.68)), speed 0.27 m/s with a RMSE of 0.2591 m (x-axis) 0.5156 m(y-axis) and a total RMSE value of 0.5770 m with a maximum error of 0.4053 m (x-axis at (0.704678,7.159322)) 0.82 m (y-axis at (0.67,7.8), calculate the position from CCTV at a speed

of 0.4 m/s with a RMSE of 0.1769 m (x-axis) 0.3951 m (y-axis) and a total RMSE value of 0.4329 m with a maximum error of 0.3449 m(x-axis at(0.770105,3.395368) 0.7537 m(y-axis at(0.660632,5.012211)), speed 0.27 m/s with a RMSE of 0.2414 m (x-axis) 0.2628 m (y-axis) and a total RMSE value of 0.3568 m with a maximum error of 0.3725 m(x-axis at (0.932881,1.627119)) 0.4795 m(y-axis at (0.827729,4.176271)). It can be seen that the movement in a straight line is calculated from encoder + heading (IMU). The lowest error rate is followed by CCTV and UWB respectively. When considering the speed, it can be seen that the higher the speed of UWB, the higher the error rate. The integration of the encoder and IMU produced the most accurate localization due to the synergy between dead reckoning and drift correction. While encoders provide incremental movement data, the IMU stabilizes heading estimations, minimizing angular deviation over time. At higher speeds, UWB readings suffered from increased positioning errors. This is primarily attributed to multipath propagation effects and signal delays that occur during rapid movement, affecting both RSSI and ToF estimations. CCTV localization demonstrated high stability under consistent lighting conditions. As the vision algorithm depends on contrast, environments with fixed illumination enhance detection performance and reduce positioning fluctuations.



(a) (b)

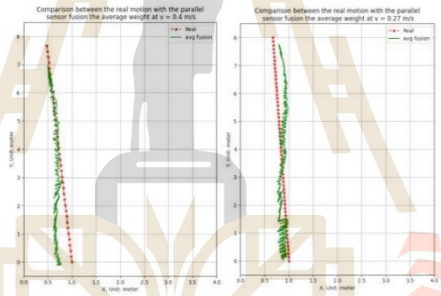
Figure 12. Comparison the real motion with the encoder and encoder + heading (IMU) (a) at v = 0.4 m/s. (b) at v = 0.27 m/s.



(a) (b)

Figure 13. Comparison real motion with encoder and encoder + heading (IMU) UWB CCTV at (a) v = 0.4 m/s. (b) v = 0.27 m/s.

4.2.2 Using a Fusion sensor in parallel, as shown in Figure 14(a)., 14(b). At a speed of 0.4 m/s using the average weight value, the RMSE of 0.1470 m(x-axis) 0.5241 m(y-axis), and a total RMSE value of 0.5444 m with a maximum error of 0.3248 m(x-axis) 1.124 m(y-axis). At a speed of 0.27 m/s using the average weight value, the RMSE of 0.1681 m(x-axis) 0.2732 m(y-axis), and a total RMSE value of 0.3208 m with a maximum error of 0.2317 m(x-axis) 0.3840 m(y-axis).



(a) (b)

Figure 14. Comparison real motion with parallel sensor fusion using average weight at (a) v = 0.4 m/s. (b) v = 0.27 m/s.

4.2.3 Use Weighted Parallel Fusion from the optimization in the Excel program for each movement path. When testing the optimal parameter of the sensor weight multiplier, there are two conditions. The first condition is when the sensor values are transmitted completely, allowing them to be used as described in Equation 9.

$$(x,y) = \frac{a \cdot (Encoder + IMU\ heading)(x,y)}{a + b + c} + \frac{b \cdot UWB(x,y) + c \cdot CCTV(x,y)}{a + b + c} \quad (9)$$

The second condition is when the sensor values are not completely transmitted, and there are no values from the CCTV sensor. In this case, use Equation 10. instead.

$$(x,y) = \frac{d \cdot (Encoder + IMU\ heading)(x,y)}{d + e} + \frac{e \cdot UWB(x,y)}{d + e} \quad (10)$$

The optimal parameter values can be found from the Excel program by selecting the GRG Nonlinear method, which is shown in Table 4, showing weight values used for optimal at a speed of 0.4 m/s and having the RMSE of 0.009 m(x-axis) 0.0616 m(y-axis), and a total RMSE value of 0.0623 m with a maximum error of 0.0211 m(x-axis) 0.1164 m(y-axis) shown as Figure 15(a). the optimal parameter values shown Table 5, weight values used for optimal at a speed of 0.27 m/s and having the RMSE of 0.1681 m(x-axis) 0.2732 m(y-axis), and a total RMSE value of 0.3208 m with a maximum error of 0.2317 m(x-axis) 0.3840 m(y-axis) shown as Figure 15(b).

Table 4: The weight values used for optimal at a speed of 0.4 m/s.

a	b	c	d	e
0.0617	0.00027	0.0038	1.7673	0.0580

4.2.4 Use Weighted Parallel Fusion from the optimization in the Excel program obtained by combining all paths. From Figure 16(a)., 16(b). and Table 6, weight values used for optimal at all speeds have an RMSE value of 0.4 m/s which is 0.0154 m(x-axis) 0.0647 m(y-axis), and a total RMSE 0.0591 m(y-axis), and a total RMSE value of 0.0634 m with a maximum error of 0.035 m(x-axis) 0.0894

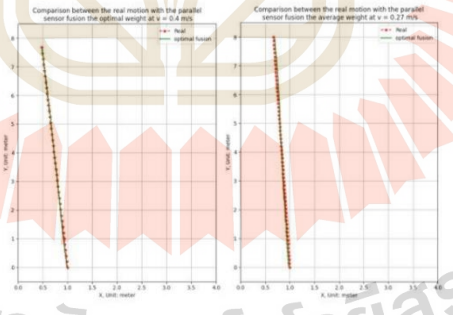
m(y-axis) shown in the Table 7 is the results of the dynamic experiment can be Summarized. value of 0.0665 m with a maximum error of 0.027 m(x-axis) 0.153 m(y-axis), and an SSE value of 0.27 m/s which is 0.0228 m(x-axis)

Table 5: The weight values used for optimal at a speed of 0.27 m/s.

a	b	c	d	e
2.4332	0.1679	0.1057	2.1349	0.2070

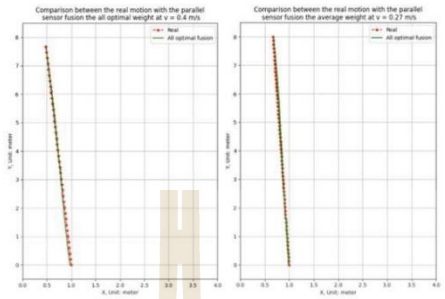
Table 6: The weight values used for all optimal at all speed.

a	b	c	d	e
0.1176	0.0037	0.0106	1.9884	0.1019



(a) (b)

Figure 15. Comparison real motion with parallel sensor fusion using the optimal weight at (a) v = 0.4 m/s. (b) v = 0.27 m/s.



(a) (b)

Figure 16. Comparison real motion with parallel sensor fusion using the optimal weight at (a) v = 0.4 m/s. (b) v = 0.27 m/s.

Table 7: The dynamic error test results.

Type	Speed (m/s)	Max Error (m)	RMSE	RMSE(x, y)	x	y
encoder	0.4	0.0805	0.1226	0.0526	0.0707	0.0881
Encoder + IMU		0.0101	0.0480	0.0035	0.0731	0.0732
UWB		0.6562	2.2000	0.3317	1.1858	1.2313
CCTV		0.3449	0.7537	0.1769	0.3951	0.4329
AVG Fusion		0.3248	1.1240	0.1470	0.5241	0.5444
opt Fusion		0.0211	0.1164	0.0090	0.0616	0.0623
All opt Fusion		0.0270	0.1530	0.0154	0.0647	0.0665
encoder	0.27	0.0979	0.1070	0.0637	0.0642	0.0904
Encoder + IMU		0.0034	0.1086	0.0021	0.0678	0.0678
UWB		0.4053	0.8200	0.2591	0.5156	0.5770
CCTV		0.3725	0.4794	0.2414	0.2628	0.3568
AVG Fusion		0.2317	0.3840	0.1681	0.2732	0.3208
opt Fusion		0.0328	0.0690	0.0199	0.0360	0.0411
All opt Fusion		0.0350	0.0894	0.0228	0.0591	0.0634

5. Conclusion

This research developed and validated a low-cost, multi-sensor fusion system for indoor AGV localization, presenting a viable alternative to expensive LiDAR-based solutions. By integrating data from encoders, IMUs, UWB, and CCTV through a parallel fusion architecture optimized with a Generalized Reduced Gradient (GRG) algorithm, we demonstrated a significant enhancement in positioning accuracy. Our findings reveal several key insights. In static tests, the Trilateration method proved more robust than the Triangular method for UWB

positioning, reducing maximum errors to 1.4464 m (x-axis) and 1.0464 m (y-axis). For dynamic localization, the synergy between the encoder and IMU was paramount, yielding the lowest RMSE of 0.0732 m at 0.4 m/s and 0.0678 m at 0.27 m/s. This highlights the effectiveness of combining dead-reckoning data with drift correction from the IMU. Conversely, UWB performance degraded at higher velocities, likely due to multipath propagation and signal delays, while CCTV provided stable, albeit less accurate, localization The most significant contribution of this work is the application of the GRG-optimized weighted fusion, which reduced the final RMSE to as low as 0.0623 m and 0.0411 m at 0.4 m/s and 0.27 m/s, respectively. This demonstrates that a carefully weighted combination of low-cost sensors can achieve the high level of accuracy required for many industrial applications, directly addressing the cost barrier of widespread AGV adoption. While these results are promising, we acknowledge certain limitations. The experiments were conducted in a controlled environment with predefined, straight-line paths. In addition, the GRG weight adjustment is performed offline. The current system uses constant weight. Furthermore, its practical application in industrial environments presents significant challenges, such as noise from machinery and uncontrolled environments. Future research should explore more complex paths and dynamic environments. Future work will focus on four primary areas. First, we plan to implement a real-time adaptive weighting mechanism, possibly using machine-learning algorithms like neural networks, to dynamically adjust sensor weights without manual optimization. Second, integrating Kalman filtering could further mitigate sensor noise and improve state estimation. Third, integrating the system with an IoT platform allows for real-time tracking of multiple AGVs, route planning, and data analysis for process optimization within the framework of a smart factory. Finally, deploying and testing the system in a live warehouse environment will be crucial to validate its practical applicability and robustness.

Acknowledgments: "This research is gratefully acknowledged for the support from Suranaree University of Technology and Department of Mechatronics and Robotics Engineering, School of Engineering and Innovation, Rajamangala University of Technology Tawan-ok. Thailand."

Funding: "This work was supported by Suranaree University of Technology."

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] I. Kubasakova, J. Kubanova, D. Benco, and D. Kadlecová, "Implementation of Automated Guided Vehicles for the Automation of Selected Processes and Elimination of Collisions between Handling Equipment and Humans in the Warehouse," *Sensors*, vol. 24, no. 3, 2024.

[2] A. Thakur and P. Rajalakshmi, "LiDAR-Based Optimized Normal Distribution Transform Localization on 3-D Map for Autonomous Navigation," *IEEE Open Journal of Instrumentation and Measurement*, vol. 3, pp. 1-11, 2024.

[3] K. W. Park and S. Y. Park, "Visual LIDAR Odometry Using Tree Trunk Detection and LIDAR Localization," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLVIII-1/W2-2023, pp. 627-632, 2023.

[4] N. Maitlo, N. Noonari, K. Arshid, N. Ahmed, and S. Duraisamy, "AINS: Affordable Indoor Navigation Solution via Line Color Identification Using Mono-Camera for Autonomous Vehicles," in *2024 IEEE 9th International Conference for Convergence in Technology (I2CT)*, 2024, pp. 1-7.

[5] S. Jeong, M. Ko, and J. Kim, "LiDAR Localization by Removing Moveable Objects," *Electronics*, vol. 12, no. 22, 2023.

[6] S. Odngam, P. Intacharoen, N. Tanman, and C. Sumpavakup, "The Design of Distance-Warning and Brake Pressure Control Systems Incorporating LiDAR Technology for Use in Autonomous Vehicles," *World Electric Vehicle Journal*, vol. 15, no. 12, 2024.

[7] F. Sauerbeck, D. Kulmer, M. Pielmeier, M. Leitenstern, C. Weiß, and J. Betz, "Multi-LiDAR Localization and Mapping Pipeline for Urban Autonomous Driving," in *2023 IEEE SENSORS*, 2023, pp. 1-4.

[8] M. Kascha, K. Xin, X. Zou, A. Sturm, R. Henze, L. Heister, and S. Oezbek, "Monocular Camera Localization for Automated Driving," in *2023 29th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, 2023, pp. 1-6.

[9] T. Kim, S. Lim, G. Shin, G. Sim, and D. Yun, "An Open-Source Low-Cost Mobile Robot System With an RGB-D Camera and Efficient Real-Time Navigation Algorithm," *IEEE Access*, vol. 10, pp. 127871-127881, 2022.

[10] L. Triyono, R. Gernowo, and P. Prayitno, "Optimizing indoor navigation systems through ensemble deep learning techniques for ArUco marker detection," *International Journal of Intelligent Engineering & Systems*, vol. 17, no. 6, 2024.

[11] X. Liu, G. Wang, and K. Chen, "High-precision vision localization system for autonomous guided vehicles in dusty industrial environments," *NAVIGATION: Journal of the Institute of Navigation*, vol. 69, no. 1, 2022.

[12] S. Y. Alaba, "GPS-IMU Sensor Fusion for Reliable Autonomous Vehicle Position Estimation," *ArXiv*, vol. abs/2405.08119, 2024.
[13] A. Charroud, K. El Moutaouakil, V. Palade, and A. Yahyaouy, "Enhanced autoencoder-based LiDAR localization in self-driving vehicles," *Applied Soft Computing*, vol. 152, p. 111225, 2024.
[14] V. Brunacci, A. Dionigi, A. De Angelis, and G. Costante, "Infrastructure-less UWB-based Active Relative Localization," in 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2024, pp. 14079-14086.
[15] K. Wongvichayakul, C. Mitsantisuk, and K. Prompol, "Development of high-precision ultra-wideband (UWB) path following using Kalman filter for automatic guide vehicles," in *Proceedings of IECON 2023 - 49th Annual Conference of the IEEE Industrial Electronics Society*, 2023, pp. 1-6.
[16] M. Kim, Y. Kwon, S. Lee, and S.-e. Yoon, "CCTV-informed human-aware robot navigation in crowded indoor environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5767-5774, 2024.
[17] C. Lundquist, *Sensor Fusion for Automotive Applications*, Ph.D. dissertation, Linköping University, Linköping, Sweden, 2011.
[18] Y. Chandola, J. Virmani, H. S. Bhadauria, and P. Kumar, "Chapter 4 - End-to-end pre-trained CNN-based computer-aided classification system design for chest radiographs," in *Deep Learning for Chest Radiographs, Primers in Biomedical Imaging Devices and Systems*, Academic Press, 2021, pp. 117-140.
[19] R. A. Pashchapur, Y. Chen, and D. Ignatyev, "Low–cost multi-object positioning system with optical sensor fusion," in *AIAA SCITECH 2023 Forum*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, Jan. 2023.
[20] W. Zhu and S. Guo, "Indoor Positioning of AGVs Based on Multi-Sensor Data Fusion Such as LiDAR," *International Journal of Sensors and Sensor Networks*, vol. 12, no. 1, pp. 13-22, 2024.
[21] S. Bouzoualegh, E.-H. Guechi, and R. Kelaiaia, "Model Predictive Control of a Differential-Drive Mobile Robot," *Acta Universitatis Sapientiae Electrical and Mechanical Engineering*, vol. 10, pp. 20-41, Dec. 2018.
[22] R. Dhaouadi and A. A. Hatab, "Dynamic Modelling of Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: A Unified Framework," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 02, 2013.
[23] E. Alcalá, L. Sellart, V. Puig, J. Quevedo, J. Saludes, D. Vázquez, and A. López, "Comparison of two non-linear model-based control strategies for autonomous vehicles," in *Proceedings of the 24th Mediterranean Conference on Control and Automation (MED)*, 2016, pp. 846-851.
[24] F. Che, Q. Z. Ahmed, P. I. Lazaridis, P. Sureephong, and T. Alade, "Indoor Positioning System (IPS) using Ultra-Wide Bandwidth (UWB)-for Industrial Internet of Things (IIoT)," *Sensors*, vol. 23, no. 12, article no. 5710, 2023.
[25] Z. Chen, X. Li, L. Wang, Y. Shi, Z. Sun, and W. Sun, "An Object Detection and Localization Method Based on Improved YOLOv5 for the Teleoperated Robot," *Applied Sciences*, vol. 12, no. 22, pp. 11441, 2022.
[26] L. S. Lasdon, R. L. Fox, and M. W. Ratner, "Nonlinear optimization using the generalized reduced gradient method," *Recherche Opérationnelle*, vol. 8, no. 3, pp. 73-103, 1974.
[27] L. S. Lasdon, A. D. Waren, A. Jain, and M. Ratner, "Design and testing of a generalized reduced gradient code for nonlinear programming," *ACM Transactions on Mathematical Software*, vol. 4, no. 1, pp. 34-50, Mar. 1978.
[28] J. S. Arora, "Chapter 13 - More on Numerical Methods for Constrained Optimum Design," in *Introduction to Optimum Design (Third Edition)*, Academic Press, 2012, pp. 533-573.
[29] D. B. Nugroho, L. P. Panjaitan, D. Kurniawati, Z. Kholil, B. Susanto, and L. R. Sasongko, "GRG non-linear and ARWM method for estimating the GARCH-M, GJR, and log-GARCH models," *Jurnal Teoridan Aplikasi Matematika*, vol. 6, no. 2, pp. 448-460, 2022.
[30] S. J. Terregrossa and U. S¸ener, "Employing a generalized reduced gradient algorithm method to form combinations of steel price forecasts generated separately by ARIMA-TF and ANN models," *Cogent Economics & Finance*, vol. 11, no. 1, 2023.

ภาคผนวก ข

ชุดคำสั่งในการประมวลผล

คำสั่ง python สำหรับพัฒนา EKF

```
# -*- coding: utf-8 -*-
"""
AGV Fusion (NO UWB) — Compare:
  (A) EKF Normal (measurement noise R จาก sigma-based config/residual)
  (B) EKF Blend per-axis alpha (CAM vs DR)

Features:
- ประเมิน σ_process (one-step prediction error) และ σ_measure (residual vs GT หลัง bias-corr)
- P0 สร้างจาก σ² ที่ประเมินจริง โดยเลือกโหมด:
    * "process"      : ใช้ σ_process² ทั้ง matrix
    * "measurement"  : ใช้ σ_measure² ทั้ง matrix (x,y จาก CAM/DR, θ จาก IMU)
    * "hybrid"       : เลือกแหล่งที่มารายแกน (x,y,θ) และเลือก CAM/DR รายแกนได้
- floors กัน P0 เล็กเกินไป
- พิมพ์/บันทึก σ, σ², P0, Q, R_* เป็นไฟล์ และเก็บ P_hist (N×3×3)
- วิเคราะห์ σ/σ² ของเซนเซอร์แบบ RAW vs Corrected (bias-corr) เทียบกับ GT แล้วบันทึก CSV+JSON

หมายเหตุคอลัมน์ข้อมูลที่ใช้ (ตามไฟล์ที่มีจริง): Time/Time (s), PWM1, PWM2, IMU, CAMx, CAMy, Rx, Ry
"""

import numpy as np
import pandas as pd
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
from pathlib import Path
import json
```

```
# ==============================
# ===== Plot RC & Styles  ======
# ==============================
plt.rcParams.update({
    "font.size": 11,
    "axes.labelsize": 12,
    "axes.titlesize": 14,
    "legend.fontsize": 11,
    "figure.dpi": 100,
})
plt.rcParams['grid.alpha'] = 1.0
plt.rcParams['legend.framealpha'] = 1.0
plt.rcParams['patch.antialiased'] = True

POINT_SIZE = 30
LINE_WIDTH = 1.8

# สไตล์เป็นมิตรกับขาวดำ: ให้ GT เป็นเส้นทึบสีดำ, ที่เหลือแยกด้วย linestyle/marker
ST_GT      = dict(color="black", linestyle="-", marker="*", linewidth=LINE_WIDTH,
markersize=5)
ST_CAM     = dict(linestyle="--", marker="o")
ST_DR      = dict(linestyle=":",  marker="D")
ST_NORM    = dict(linestyle="-.", marker="s")  # EKF Normal
ST_BLEND   = dict(linestyle="-",  marker="^")  # EKF Blend
ST_RAW     = dict(linestyle="--", marker="P")  # EKF Normal (RAW)

def _make_opaque_for_eps(Figliola & Beasley):
    import matplotlib
    for coll in fig.findobj(match=matplotlib.collections.Collection): coll.set_alpha(1.0)
    for line in fig.findobj(match=matplotlib.lines.Line2D):           line.set_alpha(1.0)
    for patch in fig.findobj(match=matplotlib.patches.Patch):         patch.set_alpha(1.0)
```

```
    for leg in fig.legends:
        if leg.get_frame() is not None: leg.get_frame().set_alpha(1.0)
    for ax in fig.get_axes(): ax.grid(alpha=1.0)

def save_fig_all(path_no_ext: str, fig=None, save_eps=True):
    if fig is None: fig = plt.gcf()
    fig.tight_layout()
    fig.savefig(f"{path_no_ext}.png", dpi=600, bbox_inches="tight")
    fig.savefig(f"{path_no_ext}.pdf", dpi=600, bbox_inches="tight")
    if save_eps:
        _make_opaque_for_eps(Figliola & Beasley)
        fig.savefig(f"{path_no_ext}.eps", dpi=600, bbox_inches="tight")
    plt.close(Figliola & Beasley)

def _finite_mask(*arrs):
    m = np.ones_like(np.asarray(arrs[0], dtype=float), dtype=bool)
    for a in arrs:
        m &= np.isfinite(np.asarray(a, dtype=float))
    return m

# ================================
# ===== User Flags & Files =====
# ================================
RUN_NORMAL = True
RUN_BLEND  = True

USE_CAM = True              # ใช้ CAMx/CAMy
USE_DR =  True              # สร้าง DR จาก encoder+IMU (x_dr,y_dr)
USE_IMU_HEADING_MEAS = True  # ใช้ IMU เป็น measurement ของ θ
USE_DR_AS_MEAS = True        # ใช้ DR[x,y] เป็น measurement เพิ่มเติม
IMU_IN_DEG = False           # ถ้า IMU ในไฟล์เป็นองศา ให้ True (จะแปลงเป็น rad)
```

```
FILES = [
    ("S1_701_01.xlsx",      0.1, 167),
    ("S1_701_02_01.xlsx",  0.2,  83),
]

# ===== DR parameters =====
EDGES_PER_PULSE = 1
WHEEL_RADIUS = 0.052    # m (0.104/2)
PPR = 6840
DR_X0, DR_Y0, DR_TH0 = 0.0, 7.0, 0.0
PWM2V_SCALE = 1.155

# ===== Process noise fallback =====
Q_SIGMA_X  = 0.03
Q_SIGMA_Y  = 0.03
Q_SIGMA_TH = 0.02

# ===== Blend weights (per-axis) =====
ALPHA_X = 0.7   # CAM weight for x (DR = 1-ALPHA_X)
ALPHA_Y = 0.3   # CAM weight for y (DR = 1-ALPHA_Y)

# ================================
# ===== Sigma Config ===========
# ================================
USE_AUTO_SIGMA = True  # True = คำนวณ σ จากข้อมูลจริง, False = manual

# MANUAL (ใช้เมื่อ USE_AUTO_SIGMA=False)
SIG_INIT_X   = 0.5
SIG_INIT_Y   = 0.5
SIG_INIT_TH  = np.deg2rad(15.0)
```

```
SIG_PROC_X   = 0.03
SIG_PROC_Y   = 0.03
SIG_PROC_TH  = 0.02

SIG_MEAS_CAM_X = 0.08
SIG_MEAS_CAM_Y = 0.08
SIG_MEAS_DR_X  = 0.10
SIG_MEAS_DR_Y  = 0.10
SIG_MEAS_IMU_TH= np.deg2rad(5.0)

# ==============================
# ===== P0 from σ² (HYBRID) ===
# ==============================
P0_MODE = "hybrid"       # "process" | "measurement" | "hybrid"
P0_AXIS_SOURCE = {
    "x":  "dr",
    "y":  "dr",
    "th": "measurement_auto",
}
P0_FLOORS = (1e-6, 1e-6, np.deg2rad(0.5)**2)  # (x,y in m^2, th in rad^2)

# ==============================
# ===== Utilities =============
# ==============================
def angle_wrap(a): return (a + np.pi) % (2*np.pi) - np.pi

def compute_dt_series(df, fallback=0.1):
    col = 'Time' if 'Time' in df.columns else ('Time (s)' if 'Time (s)' in df.columns else
None)
    if col is None: return np.full(len(df), fallback), float(fallback)
    dt = df[col].diff().fillna(fallback).astype(float).values
```

```
    return dt, float(np.nanmedian(dt))

def compute_real_heading_from_gt(df):
    N = len(df); th = np.full(N, np.nan, dtype=float)
    if not {"Rx","Ry"}.issubset(df.columns): return th
    rx = df["Rx"].to_numpy(dtype=float); ry = df["Ry"].to_numpy(dtype=float)
    for i in range(1, N):
        if np.isfinite(rx[i]) and np.isfinite(ry[i]) and np.isfinite(rx[i-1]) and np.isfinite(ry[i-1]):
            dx = rx[i]-rx[i-1]; dy = ry[i]-ry[i-1]
            if dx!=0.0 or dy!=0.0: th[i] = np.arctan2(dy, dx)
    return th

def rmse(a, b):
    a = np.asarray(a, dtype=float); b = np.asarray(b, dtype=float)
    m = np.isfinite(a) & np.isfinite(b)
    if m.sum()==0: return np.nan
    return float(np.sqrt(np.mean((a[m]-b[m])**2)))

def rmse_xy(xp, yp, xg, yg):
    xp = np.asarray(xp, dtype=float); yp = np.asarray(yp, dtype=float)
    xg = np.asarray(xg, dtype=float); yg = np.asarray(yg, dtype=float)
    m = np.isfinite(xp) & np.isfinite(yp) & np.isfinite(xg) & np.isfinite(yg)
    if m.sum() == 0: return np.nan
    err2 = (xp[m]-xg[m])**2 + (yp[m]-yg[m])**2
    return float(np.sqrt(np.mean(err2)))

def get_time_vector(df):
    if "Time" in df.columns: return df["Time"].to_numpy(dtype=float)
    if "Time (s)" in df.columns: return df["Time (s)"].to_numpy(dtype=float)
    return np.arange(len(df), dtype=float)
```

```
def _finite_var(a):
    a = np.asarray(a, dtype=float); a = a[np.isfinite(a)]
    return float(np.var(a, ddof=1)) if a.size > 1 else np.nan

def safe_var(df, col, ref):
    if (col in df.columns) and (ref in df.columns):
        v = _finite_var(df[col] - df[ref])
        if np.isfinite(v) and v > 0: return v
    return None

def title_for_dt(dtv: float) -> str:
    # ข้อกำหนดพิเศษ: 0.1 "ไม่มี s", 0.2 "มี s"
    if abs(dtv - 0.1) < 1e-6:
        return "Trajectory comparison of ground truth (GT), EKF Normal, and EKF Blend
models at dt = 0.1 s."
    if abs(dtv - 0.2) < 1e-6:
        return "Trajectory comparison of ground truth (GT), EKF Normal, and EKF Blend
models at dt = 0.2 s."
    return f"Trajectory Comparison: Normal vs. Blended EKF at dt={dtv:.3g} s"

# ================================
# ===== DR (Encoder+IMU) ======
# ================================
def pwm_to_velocity(d_pwm1, d_pwm2, r, ppr, edges_per_pulse, dt,
scale=PWM2V_SCALE):
    w_l = (d_pwm1 * 2*np.pi * r) / (ppr * edges_per_pulse * dt)
    w_r = (d_pwm2 * 2*np.pi * r) / (ppr * edges_per_pulse * dt)
    return scale * 0.5 * (w_r + w_l)

def compute_dr_from_encoder_imu(
    df, dt_series,
```

```
    wheel_radius=WHEEL_RADIUS, PPR=PPR, edges_per_pulse=EDGES_PER_PULSE,
    x0=DR_X0, y0=DR_Y0, th0=DR_TH0, scale=PWM2V_SCALE
):
    N = len(df)
    V = np.zeros(N, dtype=float); x_dr = np.zeros(N, dtype=float)
    y_dr = np.zeros(N, dtype=float); th_dr = np.zeros(N, dtype=float)
    x_dr[0], y_dr[0], th_dr[0] = float(x0), float(y0), float(th0)
    for i in range(1, N):
        dt = max(float(dt_series[i]), 1e-6)
        d1 = float(df.loc[i,"PWM1"] - df.loc[i-1,"PWM1"])
        d2 = float(df.loc[i,"PWM2"] - df.loc[i-1,"PWM2"])
        imu = float(df.loc[i,"IMU"])  # rad
        V[i] = pwm_to_velocity(d1, d2, wheel_radius, PPR, edges_per_pulse, dt,
scale=scale)
        x_dr[i]= x_dr[i-1] + V[i]*dt*np.cos(imu)
        y_dr[i]= y_dr[i-1] + V[i]*dt*np.sin(imu)
        th_dr[i]= imu
    return V, x_dr, y_dr, th_dr


# ================================
# ===== Bias (optional) ========
# ================================
def estimate_biases(df):
    out = {}
    if USE_CAM and {"CAMx","Rx"}.issubset(df.columns):
        out.setdefault("CAM", {})["x"] = float((df["CAMx"] - df["Rx"]).replace([np.inf,-np.inf],
np.nan).dropna().mean())
    if USE_CAM and {"CAMy","Ry"}.issubset(df.columns):
        out.setdefault("CAM", {})["y"] = float((df["CAMy"] - df["Ry"]).replace([np.inf,-np.inf],
np.nan).dropna().mean())
    if USE_DR and {"x_dr","Rx"}.issubset(df.columns):
```

```
        out.setdefault("DR", {})["x"] = float((df["x_dr"] - df["Rx"]).replace([np.inf,-np.inf],
np.nan).dropna().mean())
    if USE_DR and {"y_dr","Ry"}.issubset(df.columns):
        out.setdefault("DR", {})["y"] = float((df["y_dr"] - df["Ry"]).replace([np.inf,-np.inf],
np.nan).dropna().mean())
    if "IMU" in df.columns and {"Rx","Ry"}.issubset(df.columns):
        real_h = compute_real_heading_from_gt(df)
        diff = angle_wrap(df["IMU"].to_numpy(dtype=float) - real_h)
        diff = diff[np.isfinite(diff)]
        b = float(np.arctan2(np.sin(diff).mean(), np.cos(diff).mean())) if diff.size>0 else 0.0
        out["IMU"] = {"theta": b}
    return out

def correct_bias(df, biases, suffix="_cb"):
    out = df.copy()
    if USE_CAM and "CAMx" in out.columns and "CAM" in biases and "x" in
biases["CAM"]:
        out["CAMx"+suffix] = out["CAMx"] - biases["CAM"]["x"]
    if USE_CAM and "CAMy" in out.columns and "CAM" in biases and "y" in
biases["CAM"]:
        out["CAMy"+suffix] = out["CAMy"] - biases["CAM"]["y"]
    if USE_DR and "x_dr" in out.columns and "DR" in biases and "x" in biases["DR"]:
        out["x_dr"+suffix] = out["x_dr"] - biases["DR"]["x"]
    if USE_DR and "y_dr" in out.columns and "DR" in biases and "y" in biases["DR"]:
        out["y_dr"+suffix] = out["y_dr"] - biases["DR"]["y"]
    if "IMU" in out.columns and "IMU" in biases and "theta" in biases["IMU"]:
        out["IMU"+suffix] = angle_wrap(out["IMU"].astype(float) -
float(biases["IMU"]["theta"]))
    return out

# ==============================
```

```
# ===== P0 helpers (HYBRID) ===
# ==============================
def build_P0_from_sigma(sig_init):
    sx, sy, sth = sig_init
    return np.diag([sx**2, sy**2, sth**2])

def build_P0_from_variance(vars_):
    vx, vy, vth = vars_
    return np.diag([vx, vy, vth])

def _sigma_measure_for_axis(axis, meas_sig, prefer):
    defaults = {'x': 0.5, 'y': 0.5}  # meters
    key_cam = f"sig_cam_{axis}"
    key_dr  = f"sig_dr_{axis}"
    has_cam = USE_CAM and (key_cam in meas_sig) and
np.isfinite(meas_sig[key_cam])
    has_dr  = USE_DR_AS_MEAS and (key_dr in meas_sig) and
np.isfinite(meas_sig[key_dr])

    if prefer == "cam":
        if has_cam: return float(meas_sig[key_cam])
        if has_dr:  return float(meas_sig[key_dr])
        return defaults[axis]
    if prefer == "dr":
        if has_dr:  return float(meas_sig[key_dr])
        if has_cam: return float(meas_sig[key_cam])
        return defaults[axis]
    # measurement_auto
    if has_cam: return float(meas_sig[key_cam])
    if has_dr:  return float(meas_sig[key_dr])
    return defaults[axis]
```

```
def _pick_sigma_for_axis(sig_proc, meas_sig, axis, mode_for_axis):
    default_sigma = {'x': 0.5, 'y': 0.5, 'th': np.deg2rad(15.0)}
    if axis in ('x','y'):
        if mode_for_axis == "process":
            return float(sig_proc[0] if axis=='x' else sig_proc[1]) if (sig_proc is not None
and all(np.isfinite(sig_proc))) else default_sigma[axis]
        elif mode_for_axis in ("measurement_auto", "cam", "dr"):
            return _sigma_measure_for_axis(axis, meas_sig, prefer=mode_for_axis)
        return _sigma_measure_for_axis(axis, meas_sig, prefer="measurement_auto")
    # th
    if mode_for_axis == "process":
        return float(sig_proc[2]) if (sig_proc is not None and all(np.isfinite(sig_proc))) else
default_sigma['th']
    s = float(meas_sig.get("sig_imu_th", default_sigma['th']))
    return s if np.isfinite(s) else default_sigma['th']

def build_P0_hybrid(sig_proc, meas_sig, floors=P0_FLOORS):
    sx  = _pick_sigma_for_axis(sig_proc, meas_sig, 'x',
P0_AXIS_SOURCE.get('x','measurement_auto'))
    sy  = _pick_sigma_for_axis(sig_proc, meas_sig, 'y',
P0_AXIS_SOURCE.get('y','measurement_auto'))
    sth = _pick_sigma_for_axis(sig_proc, meas_sig,
'th',P0_AXIS_SOURCE.get('th','process'))
    vx, vy, vth = max(sx**2,floors[0]), max(sy**2,floors[1]), max(sth**2,floors[2])
    return build_P0_from_variance((vx,vy,vth))

# ==============================
# ===== Sigma Estimation =======
# ==============================
def estimate_sigma_meas_from_residual(df):
```

```
    biases = estimate_biases(df)
    dfc    = correct_bias(df, biases, suffix="_cb")
    out = {}
    def _std(fin): fin = fin[np.isfinite(fin)]; return float(np.std(fin, ddof=1)) if fin.size>1
else np.nan
    if {"CAMx_cb","Rx"}.issubset(dfc.columns): out["sig_cam_x"] = _std((dfc["CAMx_cb"] -
dfc["Rx"]).to_numpy(float))
    if {"CAMy_cb","Ry"}.issubset(dfc.columns): out["sig_cam_y"] = _std((dfc["CAMy_cb"] -
dfc["Ry"]).to_numpy(float))
    if {"x_dr_cb","Rx"}.issubset(dfc.columns): out["sig_dr_x"]  = _std((dfc["x_dr_cb"] -
dfc["Rx"]).to_numpy(float))
    if {"y_dr_cb","Ry"}.issubset(dfc.columns): out["sig_dr_y"]  = _std((dfc["y_dr_cb"] -
dfc["Ry"]).to_numpy(float))
    if "IMU_cb" in dfc.columns and {"Rx","Ry"}.issubset(dfc.columns):
        th_real = compute_real_heading_from_gt(dfc)
        dth = angle_wrap(dfc["IMU_cb"].to_numpy(float) - th_real)
        out["sig_imu_th"] = _std(dth)
    # fallbacks
    for k in ("sig_cam_x","sig_cam_y","sig_dr_x","sig_dr_y"):
        if k not in out or not np.isfinite(out[k]) or out[k] <= 0: out[k] = 0.1
    if "sig_imu_th" not in out or not np.isfinite(out["sig_imu_th"]) or out["sig_imu_th"]
<= 0:
        out["sig_imu_th"] = np.deg2rad(5.0)
    return out


# ================================
# ===== Build R, Q ============
# ================================
def build_Q_from_sigma(sig_proc):
    sx, sy, sth = sig_proc
    return np.diag([sx**2, sy**2, sth**2])
```

```
def build_Rs_from_sigma(meas_sigmas, use_dr_as_meas=True):
    R_cam = R_dr = R_imu = None
    if ("sig_cam_x" in meas_sigmas) and ("sig_cam_y" in meas_sigmas):
        R_cam = np.diag([meas_sigmas["sig_cam_x"]**2, meas_sigmas["sig_cam_y"]**2])
    if use_dr_as_meas and ("sig_dr_x" in meas_sigmas) and ("sig_dr_y" in meas_sigmas):
        R_dr  = np.diag([meas_sigmas["sig_dr_x"]**2,  meas_sigmas["sig_dr_y"]**2])
    if "sig_imu_th" in meas_sigmas:
        R_imu = np.array([[meas_sigmas["sig_imu_th"]**2]])
    return R_cam, R_dr, R_imu


# ==============================
# ===== RAW vs Corrected sigma =
# ==============================
def _finite(a): a=np.asarray(a, dtype=float); return a[np.isfinite(a)]
def _mean(a): a=_finite(a); return float(np.mean(a)) if a.size>0 else np.nan
def _stdv(a): a=_finite(a); return float(np.std(a, ddof=1)) if a.size>1 else np.nan
def _rms(a):  a=_finite(a); return float(np.sqrt(np.mean(a**2))) if a.size>0 else np.nan
def _mae(a):  a=_finite(a); return float(np.mean(np.abs(a))) if a.size>0 else np.nan
def _circ_mean(rad):
    a=_finite(rad);
    return float(np.arctan2(np.sin(a).mean(), np.cos(a).mean())) if a.size>0 else np.nan
def _residual(raw, ref, is_angle=False):
    if raw is None or ref is None: return None
    r = np.asarray(raw, dtype=float) - np.asarray(ref, dtype=float)
    return angle_wrap(r) if is_angle else r


def analyze_sigma_raw_vs_corrected(df):
    out = {'biases': {}, 'stats': {}}
    biases = estimate_biases(df)
    out['biases'] = biases
    dfc = correct_bias(df, biases, suffix="_cb")
```

```
    th_real = compute_real_heading_from_gt(df)

    def _pack(res, is_angle=False):
        if res is None:
            return
{'bias':np.nan,'sigma':np.nan,'var':np.nan,'rmse':np.nan,'mae':np.nan,'available':False}
        if is_angle:
            bias = _circ_mean(res); sig=_stdv(res); rmse=_rms(res); mae=_mae(res)
        else:
            bias = _mean(res); sig=_stdv(res); rmse=_rms(res); mae=_mae(res)
        return {'bias':bias,'sigma':sig,'var':(sig**2 if np.isfinite(sig) else np.nan),
                'rmse':rmse,'mae':mae,'available':True}

    def _add(name, raw_series, cb_series, ref_series=None, is_angle=False):
        res_raw = _residual(raw_series, ref_series, is_angle=is_angle)
        res_cb  = _residual(cb_series,  ref_series, is_angle=is_angle)
        out['stats'][name] = {'raw': _pack(res_raw, is_angle), 'cb': _pack(res_cb, is_angle)}

    if {"Rx","Ry"}.issubset(df.columns):
        _add("cam_x", df.get("CAMx"), dfc.get("CAMx_cb"), ref_series=df["Rx"])
        _add("cam_y", df.get("CAMy"), dfc.get("CAMy_cb"), ref_series=df["Ry"])
        _add("dr_x",  df.get("x_dr"), dfc.get("x_dr_cb"),  ref_series=df["Rx"])
        _add("dr_y",  df.get("y_dr"), dfc.get("y_dr_cb"),  ref_series=df["Ry"])
        _add("imu_th", df.get("IMU"), dfc.get("IMU_cb"), ref_series=th_real,
is_angle=True)
    return out

def print_and_save_sigma_raw_vs_corrected(stem, analysis):
    print(f"\n--- [{stem}] Sensor residual vs GT: RAW (no bias) vs CB ---")
    if analysis.get('biases'): print("Bias used:", analysis['biases'])
    header = "{:>8} | {:>5} | {:>10} | {:>10} | {:>10} | {:>10} | {:>10}"
```

```
    rowfmt = "{:>8} | {:>5} | {:>10.6f} | {:>10.6f} | {:>10.6f} | {:>10.6f} | {:>10.6f}"
    print(header.format("Sensor","Mode","bias","σ","σ²","RMSE","MAE"))
    rows=[]
    def emit(name,label):
        e=analysis['stats'].get(name)
        if not e: return
        for m in ("raw","cb"):
            r=e[m]
            if r["available"]:
                print(rowfmt.format(label, m.upper(), r["bias"], r["sigma"], r["var"], r["rmse"],
r["mae"]))
                rows.append([label,m.upper(), r["bias"], r["sigma"], r["var"], r["rmse"],
r["mae"]])
    emit("cam_x","CAMx"); emit("cam_y","CAMy"); emit("dr_x","DRx"); emit("dr_y","DRy");
emit("imu_th","IMUθ")
    print("--- end ---\n")
    with open(f"{stem}_sigma_raw_vs_cb.json","w",encoding="utf-8") as f:
        json.dump(analysis, f, ensure_ascii=False, indent=2)
    if rows:
        pd.DataFrame(rows,
columns=["Sensor","Mode","Bias","Sigma","Variance","RMSE","MAE"]).to_csv(f"{stem}_sigm
a_raw_vs_cb.csv", index=False)


# ================================
# ===== EKF: Normal ============
# ================================
def run_ekf(df, use_corrected=True, P0=None, Q=None, R_cam=None, R_dr=None,
R_imu=None):
    camx = "CAMx_cb" if use_corrected else "CAMx"
    camy = "CAMy_cb" if use_corrected else "CAMy"
    drx  = "x_dr_cb" if use_corrected else "x_dr"
```

```
    dry  = "y_dr_cb" if use_corrected else "y_dr"
    imu  = "IMU_cb"  if use_corrected else "IMU"

    t = get_time_vector(df)
    dt_series = np.diff(t, prepend=t[0]); dt_series[0] = dt_series[1] if len(dt_series)>1
else 0.1
    dt_series = np.clip(dt_series, 1e-6, None)

    N = len(df)
    x_est = np.zeros((N,3))
    P_hist = np.zeros((N,3,3))
    P = P0.copy() if P0 is not None else np.diag([1.0, 1.0, (15*np.pi/180.0)**2])
    P_hist[0] = P
    Q = Q.copy() if Q is not None else np.diag([Q_SIGMA_X**2, Q_SIGMA_Y**2,
Q_SIGMA_TH**2])

    if USE_DR and drx in df.columns and dry in df.columns and np.isfinite(df.loc[0,drx])
and np.isfinite(df.loc[0,dry]):
        x_est[0] = [df.loc[0,drx], df.loc[0,dry], float(df.loc[0,imu]) if imu in df.columns
else 0.0]
    elif USE_CAM and camx in df.columns and camy in df.columns and
np.isfinite(df.loc[0,camx]) and np.isfinite(df.loc[0,camy]):
        x_est[0] = [df.loc[0,camx], df.loc[0,camy], float(df.loc[0,imu]) if imu in
df.columns else 0.0]
    else:
        x_est[0] = [DR_X0, DR_Y0, float(df.loc[0,imu]) if imu in df.columns else 0.0]

    H_pos = np.array([[1,0,0],[0,1,0]], dtype=float)
    H_th  = np.array([[0,0,1]], dtype=float)

    has_gt = {"Rx","Ry"}.issubset(df.columns)
```

```
    refx   = "Rx" if has_gt else (camx if camx in df.columns else drx)
    refy   = "Ry" if has_gt else (camy if camy in df.columns else dry)

    if R_cam is None and USE_CAM and camx in df.columns and camy in df.columns:
        vx = safe_var(df, camx, refx) or _finite_var(df[camx])
        vy = safe_var(df, camy, refy) or _finite_var(df[camy])
        R_cam = np.diag([max(vx,1e-4), max(vy,1e-4)])

    if R_dr is None and USE_DR_AS_MEAS and drx in df.columns and dry in
df.columns:
        vx = safe_var(df, drx, refx) or _finite_var(df[drx])
        vy = safe_var(df, dry, refy) or _finite_var(df[dry])
        R_dr = np.diag([max(vx,1e-4), max(vy,1e-4)])

    if R_imu is None and USE_IMU_HEADING_MEAS and imu in df.columns:
        if has_gt:
            real_h = compute_real_heading_from_gt(df)
            vth = _finite_var(angle_wrap(df[imu].to_numpy(float) - real_h))
        else:
            vth = _finite_var(df[imu])
        if not (np.isfinite(vth) and vth>0): vth = (5*np.pi/180.0)**2
        R_imu = np.array([[vth]], dtype=float)

    x_pred_hist = np.zeros((N,3))

    for i in range(1, N):
        dt = float(dt_series[i])
        v  = float(df["V_enc"].iloc[i]) if ("V_enc" in df.columns and
np.isfinite(df["V_enc"].iloc[i])) else 0.0
        th_prev = float(x_est[i-1,2])
```

```
        th_imu_now  = float(df[imu].iloc[i])   if imu in df.columns and
np.isfinite(df[imu].iloc[i])   else th_prev
        th_imu_prev = float(df[imu].iloc[i-1]) if imu in df.columns and
np.isfinite(df[imu].iloc[i-1]) else th_prev
        omega = angle_wrap(th_imu_now - th_imu_prev) / dt

        # Prediction
        x_pred = np.empty(3)
        x_pred[0] = x_est[i-1,0] + v*dt*np.cos(th_prev)
        x_pred[1] = x_est[i-1,1] + v*dt*np.sin(th_prev)
        x_pred[2] = angle_wrap(th_prev + omega*dt)

        F = np.array([[1,0,-v*dt*np.sin(th_prev)],
                      [0,1, v*dt*np.cos(th_prev)],
                      [0,0, 1]], dtype=float)

        P = F @ P @ F.T + Q
        x_pred_hist[i] = x_pred

        # CAM
        if USE_CAM and (camx in df.columns) and (camy in df.columns) and (R_cam is
not None):
            if np.isfinite(df.loc[i,camx]) and np.isfinite(df.loc[i,camy]):
                z = np.array([df.loc[i,camx], df.loc[i,camy]], dtype=float)
                y = z - (H_pos @ x_pred)
                S = H_pos @ P @ H_pos.T + R_cam
                try:
                    K = P @ H_pos.T @ np.linalg.inv(S)
                    x_pred = x_pred + K @ y
                    x_pred[2] = angle_wrap(x_pred[2])
                    P = (np.eye(3) - K @ H_pos) @ P
```

```
            except np.linalg.LinAlgError:
                pass

        # DR as measurement
        if USE_DR_AS_MEAS and (drx in df.columns) and (dry in df.columns) and (R_dr is
not None):
            if np.isfinite(df.loc[i,drx]) and np.isfinite(df.loc[i,dry]):
                z = np.array([df.loc[i,drx], df.loc[i,dry]], dtype=float)
                y = z - (H_pos @ x_pred)
                S = H_pos @ P @ H_pos.T + R_dr
                try:
                    K = P @ H_pos.T @ np.linalg.inv(S)
                    x_pred = x_pred + K @ y
                    x_pred[2] = angle_wrap(x_pred[2])
                    P = (np.eye(3) - K @ H_pos) @ P
                except np.linalg.LinAlgError:
                    pass

        # IMU heading
        if USE_IMU_HEADING_MEAS and (imu in df.columns) and (R_imu is not None)
and np.isfinite(df.loc[i,imu]):
            z_th = float(df.loc[i,imu])
            y_th = np.array([angle_wrap(z_th - x_pred[2])])
            S_th = np.array([[P[2,2]]]) + R_imu
            try:
                K_th = (P[:,2:3] @ np.linalg.inv(S_th))
                x_pred = x_pred + (K_th @ y_th).reshape(-1)
                x_pred[2] = angle_wrap(x_pred[2])
                P = (np.eye(3) - K_th @ np.array([[0,0,1]], dtype=float)) @ P
            except np.linalg.LinAlgError:
                pass
```

```
        x_est[i] = x_pred
        P_hist[i] = P

    return x_est, x_pred_hist, P_hist

# ==============================
# ===== EKF: Blend ============
# ==============================
def run_ekf_blend(df, use_corrected=True, P0=None, Q=None, R_cam=None,
R_dr=None, R_imu=None, alpha_pos=None):
    camx = "CAMx_cb" if use_corrected else "CAMx"
    camy = "CAMy_cb" if use_corrected else "CAMy"
    drx  = "x_dr_cb" if use_corrected else "x_dr"
    dry  = "y_dr_cb" if use_corrected else "y_dr"
    imu  = "IMU_cb"  if use_corrected else "IMU"

    t = get_time_vector(df)
    dt_series = np.diff(t, prepend=t[0]); dt_series[0] = dt_series[1] if len(dt_series)>1
else 0.1
    dt_series = np.clip(dt_series, 1e-6, None)

    N = len(df)
    x_est = np.zeros((N,3))
    P_hist = np.zeros((N,3,3))

    if alpha_pos is None:
        ax, ay = float(ALPHA_X), float(ALPHA_Y)
    elif isinstance(alpha_pos, (int, float)):
        ax = ay = float(alpha_pos)
    elif isinstance(alpha_pos, (list, tuple)) and len(alpha_pos) == 2:
        ax, ay = float(alpha_pos[0]), float(alpha_pos[1])
```

```
    elif isinstance(alpha_pos, dict):
        ax = float(alpha_pos.get('x', ALPHA_X)); ay = float(alpha_pos.get('y', ALPHA_Y))
    else:
        ax, ay = float(ALPHA_X), float(ALPHA_Y)
    ax = float(np.clip(ax, 0.0, 1.0)); ay = float(np.clip(ay, 0.0, 1.0))

    P = P0.copy() if P0 is not None else np.diag([1.0, 1.0, (15*np.pi/180.0)**2])
    P_hist[0] = P
    Q = Q.copy() if Q is not None else np.diag([Q_SIGMA_X**2, Q_SIGMA_Y**2,
Q_SIGMA_TH**2])

    if USE_DR and drx in df.columns and dry in df.columns and np.isfinite(df.loc[0,drx])
and np.isfinite(df.loc[0,dry]):
        x_est[0] = [df.loc[0,drx], df.loc[0,dry], float(df.loc[0,imu]) if imu in df.columns
else 0.0]
    elif USE_CAM and camx in df.columns and camy in df.columns and
np.isfinite(df.loc[0,camx]) and np.isfinite(df.loc[0,camy]):
        x_est[0] = [df.loc[0,camx], df.loc[0,camy], float(df.loc[0,imu]) if imu in
df.columns else 0.0]
    else:
        x_est[0] = [DR_X0, DR_Y0, float(df.loc[0,imu]) if imu in df.columns else 0.0]

    H_pos = np.array([[1,0,0],[0,1,0]], dtype=float)
    H_th  = np.array([[0,0,1]], dtype=float)

    has_gt = {"Rx","Ry"}.issubset(df.columns)
    refx   = "Rx" if has_gt else (camx if camx in df.columns else drx)
    refy   = "Ry" if has_gt else (camy if camy in df.columns else dry)

    def _fv(a): a = np.asarray(a, dtype=float); a = a[np.isfinite(a)]; return float(np.var(a,
ddof=1)) if a.size>1 else np.nan
```

```
    def _sv(df_, col, ref):
        if (col in df_.columns) and (ref in df_.columns):
            v = _fv(df_[col] - df_[ref]);
            if np.isfinite(v) and v>0: return v
        return None

    if R_cam is None and USE_CAM and camx in df.columns and camy in df.columns:
        vx = _sv(df, camx, refx) or _fv(df[camx]); vy = _sv(df, camy, refy) or _fv(df[camy])
        R_cam = np.diag([max(vx,1e-4), max(vy,1e-4)])

    if R_dr is None and USE_DR_AS_MEAS and drx in df.columns and dry in
df.columns:
        vx = _sv(df, drx, refx) or _fv(df[drx]); vy = _sv(df, dry, refy) or _fv(df[dry])
        R_dr = np.diag([max(vx,1e-4), max(vy,1e-4)])

    if R_imu is None and USE_IMU_HEADING_MEAS and imu in df.columns:
        if has_gt:
            real_h = compute_real_heading_from_gt(df)
            vth = _fv(angle_wrap(df[imu].to_numpy(float) - real_h))
        else:
            vth = _fv(df[imu])
        if not (np.isfinite(vth) and vth > 0): vth = (5*np.pi/180.0)**2
        R_imu = np.array([[vth]], dtype=float)

    x_pred_hist = np.zeros((N,3))

    for i in range(1, N):
        dt = float(dt_series[i])
        v  = float(df["V_enc"].iloc[i]) if ("V_enc" in df.columns and
np.isfinite(df["V_enc"].iloc[i])) else 0.0
        th_prev = float(x_est[i-1,2])
```

```
        th_imu_now  = float(df[imu].iloc[i])   if imu in df.columns and
np.isfinite(df[imu].iloc[i])   else th_prev
        th_imu_prev = float(df[imu].iloc[i-1]) if imu in df.columns and
np.isfinite(df[imu].iloc[i-1]) else th_prev
        omega = angle_wrap(th_imu_now - th_imu_prev) / dt

        # Prediction
        x_pred = np.empty(3)
        x_pred[0] = x_est[i-1,0] + v*dt*np.cos(th_prev)
        x_pred[1] = x_est[i-1,1] + v*dt*np.sin(th_prev)
        x_pred[2] = angle_wrap(th_prev + omega*dt)

        F = np.array([[1,0,-v*dt*np.sin(th_prev)],
                      [0,1, v*dt*np.cos(th_prev)],
                      [0,0, 1]], dtype=float)

        P = F @ P @ F.T + Q
        x_pred_hist[i] = x_pred

        # Build blended position meas (per-axis α)
        have_cam = USE_CAM and camx in df.columns and camy in df.columns \
              and np.isfinite(df.loc[i,camx]) and np.isfinite(df.loc[i,camy])
        have_dr  = USE_DR  and drx  in df.columns and dry  in df.columns \
              and np.isfinite(df.loc[i,drx])  and np.isfinite(df.loc[i,dry])

        z_pos = None; R_pos = None
        if have_cam and have_dr:
            cam_x, cam_y = float(df.loc[i,camx]), float(df.loc[i,camy])
            dr_x,  dr_y  = float(df.loc[i,drx]),  float(df.loc[i,dry])
            z_x = ax * cam_x + (1.0-ax) * dr_x
            z_y = ay * cam_y + (1.0-ay) * dr_y
```

```
    z_pos = np.array([z_x, z_y], dtype=float)

    Rx = Ry = None
    if R_cam is not None:
        Rx = (ax**2) * R_cam[0,0]; Ry = (ay**2) * R_cam[1,1]
    if R_dr  is not None:
        Rx = (Rx if Rx is not None else 0.0) + ((1.0-ax)**2) * R_dr[0,0]
        Ry = (Ry if Ry is not None else 0.0) + ((1.0-ay)**2) * R_dr[1,1]
    if (Rx is not None) and (Ry is not None):
        R_pos = np.diag([max(Rx,1e-8), max(Ry,1e-8)])
elif have_cam:
    z_pos = np.array([float(df.loc[i,camx]), float(df.loc[i,camy])], dtype=float)
    R_pos = R_cam
elif have_dr:
    z_pos = np.array([float(df.loc[i,drx]), float(df.loc[i,dry])], dtype=float)
    R_pos = R_dr

# Position update
if (z_pos is not None) and (R_pos is not None):
    y  = z_pos - (H_pos @ x_pred)
    S  = H_pos @ P @ H_pos.T + R_pos
    try:
        K = P @ H_pos.T @ np.linalg.inv(S)
        x_pred = x_pred + K @ y
        x_pred[2] = angle_wrap(x_pred[2])
        P = (np.eye(3) - K @ H_pos) @ P
    except np.linalg.LinAlgError:
        pass

# Heading update
```

```
        if USE_IMU_HEADING_MEAS and (imu in df.columns) and np.isfinite(df.loc[i,imu])
and (R_imu is not None):
            z_th = float(df.loc[i,imu])
            y_th = np.array([angle_wrap(z_th - x_pred[2])])
            S_th = H_th @ P @ H_th.T + R_imu
            try:
                K_th = P @ H_th.T @ np.linalg.inv(S_th)
                x_pred = x_pred + (K_th @ y_th).reshape(-1)
                x_pred[2] = angle_wrap(x_pred[2])
                P = (np.eye(3) - K_th @ H_th) @ P
            except np.linalg.LinAlgError:
                pass

        x_est[i] = x_pred
        P_hist[i] = P

    return x_est, x_pred_hist, P_hist

# ================================
# ===== Plots (scatter) ========
# ================================
def plot_trajectory_compare(rx, ry, trajs, labels, title, out_no_ext):
    fig, ax = plt.subplots(figsize=(9,8))
    if rx is not None and ry is not None:
        xgt = np.asarray(rx, dtype=float); ygt = np.asarray(ry, dtype=float)
        mgt = _finite_mask(xgt, ygt)
        if mgt.any(): ax.plot(xgt[mgt], ygt[mgt], label="GT (Rx,Ry)", **ST_GT)

    styles = [ST_NORM, ST_RAW, ST_BLEND]  # ลำดับที่เราจะใช้งานบ่อย
    for (xy, lb, style) in zip(trajs, labels, styles + [ST_CAM]*max(0, len(trajs)-3)):
        x,y = xy
```

```
        if x is None or y is None: continue
        x = np.asarray(x, dtype=float); y=np.asarray(y, dtype=float)
        m = _finite_mask(x,y)
        if m.any(): ax.scatter(x[m], y[m], s=POINT_SIZE, label=lb, **style)

    ax.set_aspect('equal', adjustable='box')
    ax.grid(True, alpha=1.0); ax.set_xlabel("X (m)"); ax.set_ylabel("Y (m)")
    ax.set_title(title); ax.legend()
    save_fig_all(out_no_ext, fig=fig)

def plot_heading_series(t, real_h, th_series, labels, title, out_no_ext):
    t = np.asarray(t, dtype=float)
    fig, ax = plt.subplots(figsize=(11,5))
    if real_h is not None:
        rh = np.asarray(real_h, dtype=float); m = _finite_mask(t, rh)
        if m.any(): ax.plot(t[m], np.degrees(rh[m]), label="Real θ from GT (deg)",
**ST_GT)
    styles = [ST_NORM, ST_RAW, ST_BLEND] + [ST_DR]
    for th, lb, style in zip(th_series, labels, styles):
        if th is None: continue
        th = np.asarray(th, dtype=float); m = _finite_mask(t, th)
        if m.any(): ax.scatter(t[m], np.degrees(th[m]), s=POINT_SIZE, label=lb, **style)
    ax.set_xlabel("Time (s)"); ax.set_ylabel("Heading (deg)")
    ax.set_title(title); ax.grid(True, alpha=1.0); ax.legend()
    save_fig_all(out_no_ext, fig=fig)

# ==============================
# ===== Save matrices bundle ===
# ==============================
def save_matrix_bundle(stem, tag, P0, Qm, R_cam, R_dr, R_imu, sig_proc, meas_sig):
    out_dir = Path(".")
```

```
    bundle = {
        "file": stem,
        "tag": tag,
        "sig_process": {
            "sigma_x": float(sig_proc[0]), "sigma_y": float(sig_proc[1]), "sigma_th":
float(sig_proc[2]),
            "var_x": float(sig_proc[0]**2), "var_y": float(sig_proc[1]**2), "var_th":
float(sig_proc[2]**2),
        },
        "sig_measure": {},
    }
    for k in ("sig_cam_x","sig_cam_y","sig_dr_x","sig_dr_y","sig_imu_th"):
        if k in meas_sig and np.isfinite(meas_sig[k]):
            bundle["sig_measure"][k] = {"sigma": float(meas_sig[k]), "var":
float(meas_sig[k]**2)}

    def _save_txt(name, M):
        if M is None: return
        np.savetxt(out_dir / f"{stem}_{tag}_{name}.txt", M, fmt="%.8f")

    _save_txt("P0", P0); _save_txt("Q", Qm); _save_txt("R_cam", R_cam);
_save_txt("R_dr", R_dr); _save_txt("R_imu", R_imu)

    with open(out_dir / f"{stem}_{tag}_sigmas.json", "w", encoding="utf-8") as f:
        json.dump(bundle, f, ensure_ascii=False, indent=2)

    print(f"\n--- [{stem}] {tag}: SIGMA & MATRICES ---")
    sp = bundle["sig_process"]
    print(f"σ_process:  x={sp['sigma_x']:.6f}  y={sp['sigma_y']:.6f}  th={sp['sigma_th']:.6f}  |
σ²: x={sp['var_x']:.6f}  y={sp['var_y']:.6f}  th={sp['var_th']:.6f}")
    if bundle["sig_measure"]:
```

```
        print("σ_measure:", " | ".join([f"{k}: σ={v['sigma']:.6f}, σ²={v['var']:.6f}" for k,v in
bundle["sig_measure"].items()]))
    def _pretty(name, M):
        print(f"{name}:\n{np.array2string(M, formatter={'float_kind':lambda x: f'{x: .8f}'})}"
if M is not None else f"{name}: None")
    _pretty("P0", P0); _pretty("Q", Qm); _pretty("R_cam", R_cam); _pretty("R_dr", R_dr);
_pretty("R_imu", R_imu)
    print("--- end ---\n")


# ==============================
# ===== Main ==================
# ==============================
if __name__ == "__main__":
    rmse_rows = []

    for fpath, dt_expect, N in FILES:
        df = pd.read_excel(fpath).iloc[:N].reset_index(drop=True)

        if IMU_IN_DEG and "IMU" in df.columns:
            df["IMU"] = np.deg2rad(df["IMU"].astype(float))

        # DR build
        dt_series, _ = compute_dt_series(df, fallback=dt_expect)
        if USE_DR and {"PWM1","PWM2","IMU"}.issubset(df.columns):
            V, x_dr, y_dr, th_dr = compute_dr_from_encoder_imu(
                df, dt_series, wheel_radius=WHEEL_RADIUS, PPR=PPR,
edges_per_pulse=EDGES_PER_PULSE,
                x0=DR_X0, y0=DR_Y0, th0=DR_TH0, scale=PWM2V_SCALE
            )
            df["V_enc"] = V; df["x_dr"] = x_dr; df["y_dr"] = y_dr; df["th_dr"] = th_dr
```

```
        # Bias correction
        biases = estimate_biases(df)
        dfc = correct_bias(df, biases, suffix="_cb")

        # RAW vs Corrected sigma analytics
        raw_cb_report = analyze_sigma_raw_vs_corrected(df)
        print_and_save_sigma_raw_vs_corrected(Path(fpath).stem, raw_cb_report)

        # Build P0, Q, R
        if USE_AUTO_SIGMA:
            meas_sig = estimate_sigma_meas_from_residual(dfc)
            sig_proc = (
                float(meas_sig.get("sig_dr_x", 0.1)),
                float(meas_sig.get("sig_dr_y", 0.1)),
                float(meas_sig.get("sig_imu_th", np.deg2rad(5.0)))
            )
            Qm  = build_Q_from_sigma(sig_proc)
            if P0_MODE == "process":
                P0 = build_P0_from_variance((sig_proc[0]**2, sig_proc[1]**2,
sig_proc[2]**2))
            elif P0_MODE == "measurement":
                sx = meas_sig.get("sig_cam_x", meas_sig.get("sig_dr_x", 0.5))
                sy = meas_sig.get("sig_cam_y", meas_sig.get("sig_dr_y", 0.5))
                sth = meas_sig.get("sig_imu_th", np.deg2rad(15.0))
                P0 = build_P0_from_variance((sx**2, sy**2, sth**2))
            elif P0_MODE == "hybrid":
                P0 = build_P0_hybrid(sig_proc=sig_proc, meas_sig=meas_sig,
floors=P0_FLOORS)
            else:
                P0 = build_P0_from_variance((sig_proc[0]**2, sig_proc[1]**2,
sig_proc[2]**2))
```

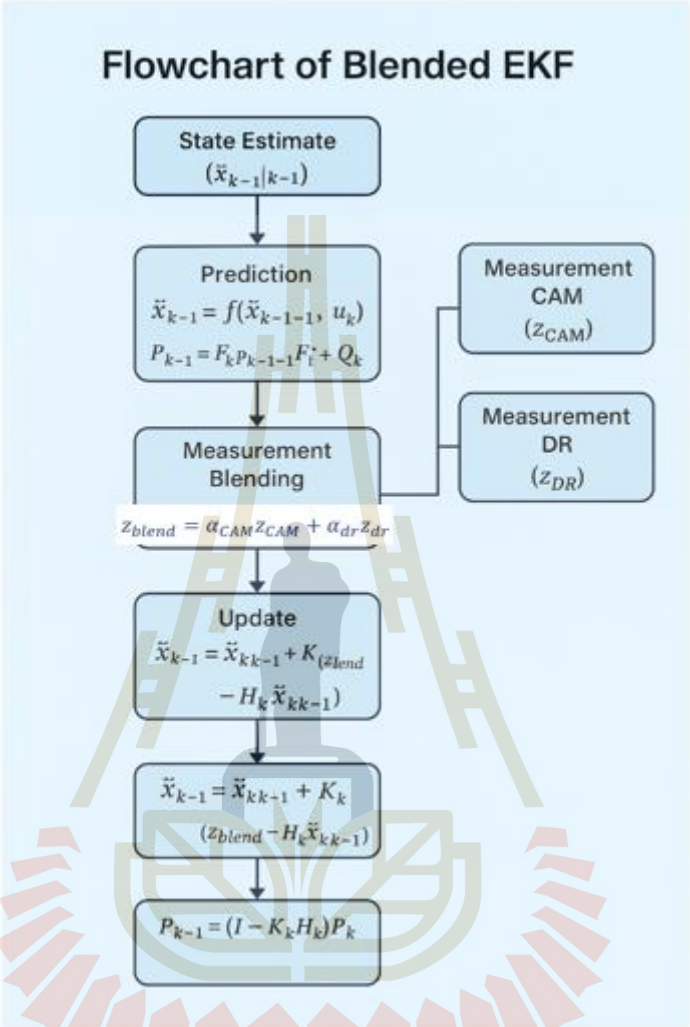
```
            # floors
            P0[0,0] = max(P0[0,0], P0_FLOORS[0])
            P0[1,1] = max(P0[1,1], P0_FLOORS[1])
            P0[2,2] = max(P0[2,2], P0_FLOORS[2])

            R_cam, R_dr, R_imu = build_Rs_from_sigma(meas_sig,
use_dr_as_meas=USE_DR_AS_MEAS)
        else:
            P0 = build_P0_from_sigma((SIG_INIT_X, SIG_INIT_Y, SIG_INIT_TH))
            Qm = build_Q_from_sigma((SIG_PROC_X, SIG_PROC_Y, SIG_PROC_TH))
            meas_sig = {
                "sig_cam_x": SIG_MEAS_CAM_X, "sig_cam_y": SIG_MEAS_CAM_Y,
                "sig_dr_x" : SIG_MEAS_DR_X,  "sig_dr_y" : SIG_MEAS_DR_Y,
                "sig_imu_th": SIG_MEAS_IMU_TH,
            }
            sig_proc = (SIG_PROC_X, SIG_PROC_Y, SIG_PROC_TH)
            R_cam, R_dr, R_imu = build_Rs_from_sigma(meas_sig,
use_dr_as_meas=USE_DR_AS_MEAS)

        stem = Path(fpath).stem
        save_matrix_bundle(stem=stem, tag="normal_setup",
                           P0=P0, Qm=Qm, R_cam=R_cam, R_dr=R_dr, R_imu=R_imu,
                           sig_proc=sig_proc, meas_sig=meas_sig)

        # Run filters
        results = {}
        if RUN_NORMAL:
            # Corrected
            xekf_n, _, P_hist_n = run_ekf(dfc, use_corrected=True, P0=P0, Q=Qm,
R_cam=R_cam, R_dr=R_dr, R_imu=R_imu)
            results["normal"] = xekf_n
```

```
            np.savetxt(f"{stem}_P_last_normal.txt", P_hist_n[-1], fmt="%.8f")
            # RAW
            xekf_n_raw, _, P_hist_n_raw = run_ekf(df, use_corrected=False, P0=P0,
Q=Qm, R_cam=R_cam, R_dr=R_dr, R_imu=R_imu)
            results["normal_raw"] = xekf_n_raw
            np.savetxt(f"{stem}_P_last_normal_raw.txt", P_hist_n_raw[-1], fmt="%.8f")

        if RUN_BLEND:
            xekf_b, _, P_hist_b = run_ekf_blend(dfc, use_corrected=True, P0=P0, Q=Qm,
R_cam=R_cam, R_dr=R_dr, R_imu=R_imu,
                                                alpha_pos=(ALPHA_X, ALPHA_Y))
            results["blend"] = xekf_b
            np.savetxt(f"{stem}_P_last_blend.txt", P_hist_b[-1], fmt="%.8f")

        # Save CSV & Plots
        t  = get_time_vector(df)
        rx = df["Rx"].to_numpy(dtype=float) if "Rx" in df.columns else None
        ry = df["Ry"].to_numpy(dtype=float) if "Ry" in df.columns else None
        real_h = compute_real_heading_from_gt(df)

        for mode, X in results.items():
            pd.DataFrame({"Time": t,
                          f"x_ekf_{mode}": X[:,0],
                          f"y_ekf_{mode}": X[:,1],
                          f"th_ekf_{mode}(rad)": X[:,2]}).to_csv(f"{stem}_ekf_{mode}.csv",
index=False)

        trajs, labels = [], []
        if "normal" in results:
            trajs.append((results["normal"][:,0], results["normal"][:,1])); labels.append("EKF
Normal (Bias Corrected)")
```

```
        if "normal_raw" in results:
            trajs.append((results["normal_raw"][:,0], results["normal_raw"][:,1]));
labels.append("EKF Normal (RAW)")
        if "blend" in results:
            trajs.append((results["blend"][:,0], results["blend"][:,1])); labels.append(f"EKF
Blend (αx={ALPHA_X:.2f}, αy={ALPHA_Y:.2f})")

        plot_trajectory_compare(rx, ry, trajs=trajs, labels=labels,
                                title=title_for_dt(dt_expect),
                                out_no_ext=f"{stem}_traj_compare_normal_vs_blend")

        th_series, h_labels = [], []
        if "normal" in results:      th_series.append(results["normal"][:,2]);
h_labels.append("EKF θ (Normal)")
        if "blend" in results:       th_series.append(results["blend"][:,2]);
h_labels.append(f"EKF θ (Blend αx={ALPHA_X:.2f}, αy={ALPHA_Y:.2f})")
        if "normal_raw" in results: th_series.append(results["normal_raw"][:,2]);
h_labels.append("EKF θ (Normal RAW)")
        if "IMU_cb" in dfc.columns: th_series.insert(0, dfc["IMU_cb"].to_numpy(float));
h_labels.insert(0, "IMU corrected")

        plot_heading_series(t, real_h, th_series=th_series, labels=h_labels,
                            title=f"{stem}: Heading — Normal vs Blend",
                            out_no_ext=f"{stem}_heading_compare_normal_vs_blend")

        # RMSE Summary
        if (rx is not None) and (ry is not None):
            if "normal" in results:
                rmse_rows.append({"File": Path(fpath).name, "Mode": "Normal",
                                  "RMSE_X": rmse(results["normal"][:,0], rx),
                                  "RMSE_Y": rmse(results["normal"][:,1], ry),
```

```
                "RMSE_XY": rmse_xy(results["normal"][:,0],
results["normal"][:,1], rx, ry)})
        if "normal_raw" in results:
            rmse_rows.append({"File": Path(fpath).name, "Mode": "Normal (RAW)",
                "RMSE_X": rmse(results["normal_raw"][:,0], rx),
                "RMSE_Y": rmse(results["normal_raw"][:,1], ry),
                "RMSE_XY": rmse_xy(results["normal_raw"][:,0],
results["normal_raw"][:,1], rx, ry)})
        if "blend" in results:
            rmse_rows.append({"File": Path(fpath).name, "Mode":
f"Blend(αx={ALPHA_X:.2f}, αy={ALPHA_Y:.2f})",
                "RMSE_X": rmse(results["blend"][:,0], rx),
                "RMSE_Y": rmse(results["blend"][:,1], ry),
                "RMSE_XY": rmse_xy(results["blend"][:,0], results["blend"][:,1],
rx, ry)})

    if rmse_rows:
        rd = pd.DataFrame(rmse_rows)
        print("\n=== RMSE Summary (Normal vs Blend) ===")
        print(rd.to_string(index=False))
        rd.to_csv("rmse_report_ekf_normal_vs_blend.csv", index=False)
```

Flow Chat กระบวนการทำงานของโปรแกรม Blended EKF



ประวัติผู้เขียน

นายนพรุจ เขียวนาค เกิดเมื่อวันที่ 3 เดือนธันวาคม ปี พ.ศ.2528 เริ่มเข้าศึกษาชั้นประถมศึกษาที่โรงเรียนบ้านเขาพญาปราบ จังหวัดนครราชสีมา และศึกษาต่อระดับมัธยมศึกษาตอนต้นและตอนปลายที่โรงเรียนปักธงชัยประชานิรมิต จังหวัดนครราชสีมา ต่อมาในปี พ.ศ. 2547 ได้เข้าศึกษาระดับปริญญาตรี สาขาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา และสำเร็จการศึกษาในปี พ.ศ. 2550 หลังจากนั้นได้ศึกษาต่อระดับปริญญาโท สาขาวิศวกรรมเครื่องกล มหาวิทยาลัยเทคโนโลยีสุรนารี และสำเร็จการศึกษาในปี พ.ศ. 2554 หลังจากสำเร็จการศึกษา นายนพรุจได้ทำงานในภาคอุตสาหกรรม ต่อมาในระหว่างปี พ.ศ. 2557–2565 ได้ดำรงตำแหน่งอาจารย์ประจำ ที่มหาวิทยาลัยเทคโนโลยีราชมงคลตะวันออก วิทยาเขตบางพระ จังหวัดชลบุรี และได้ลาศึกษาต่อระดับปริญญาในสาขาวิชาวิศวกรรมเครื่องกล หลักสูตรวิศวกรรมเครื่องกลและระบบกระบวนการ มหาวิทยาลัยเทคโนโลยีสุรนารี เมื่อเข้าศึกษาต่อแล้วได้รับทุนการศึกษาของนักศึกษาระดับบัณฑิตศึกษาที่คณาจารย์ได้รับทุนวิจัยจากแหล่งทุนภายนอก โดยอาจารย์ที่ปรึกษา รองศาสตราจารย์ ดร.จิระพล ศรีเสริฐผล เป็นผู้ติดต่อให้รับทุน โดยในระหว่างการศึกษาได้มีผลงานทางวิชาการดังปรากฏในรายละเอียด ภาคผนวก ก.