การวางแผนการจัดวางปลายนิ้วสำหรับการจับวัตถุที่ไม่รู้จักโดยใช้แนวทางตาในมือแบบสแลม

นายณัชนนท์ วงษ์วิไล

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2556

FINGERTIP PLACEMENT PLANNING FOR UNKNOWN OBJECT GRASPING USING SLAM-BASED EYE-IN-HAND APPROACH

Mr. Natchanon Wongwilai

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Computer Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2013

หัวข้อวิทยานิพนธ์	การวางแผนการจัดวางปลายนิ้วสำหรับการจับวัตถุที่ไม่รู้จักโดยใช้แนวทางตาในมือแบบสแลม
โดย	นายณัชนนท์ วงษ์วิไล
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.นัทที นิภานันท์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร.บัณฑิต เอื้ออาภรณ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.อรรถวิทย์ สุดแสง)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร.นัทที นิภานันท์)

.....กรรมการ

(อาจารย์ ดร.ณัฐพงศ์ ชินธเนศ)

.....กรรมการภายนอกมหาวิทยาลัย

(ดร.ธนะธร พ่อค้า)

ณัชนนท์ วงษ์วิไล : การวางแผนการจัดวางปลายนิ้วสำหรับการจับวัตถุที่ไม่รู้จักโดยใช้แนวทางตาในมือแบบสแลม. (FINGERTIP PLACEMENT PLANNING FOR UNKNOWN OBJECT GRASPING USING SLAM-BASED EYE-IN-HAND APPROACH) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ผศ. ดร.นัทที นิภานันท์, 67 หน้า.

ความล้มเหลวในการจับวัตถุที่ไม่รู้จักของหุ่นยนต์ มีสาเหตุหนึ่งมาจากการระบุตำแหน่งระหว่างหุ่นยนต์และวัตถุที่ผิดพลาด ทำให้ไม่สามารถจัดวางปลายนิ้วลงบนจุดจับที่ต้องการได้ เพื่อแก้ไขปัญหานี้งานวิจัยนี้ได้นำเสนอวิธีการวางแผนการจัดวางปลายนิ้วที่แม่นยำ โดยใช้แนวทางตาในมือที่มีกล้อง DepthSense 325 เป็นอุปกรณ์รับรู้และผ่านการปรับแก้พารามิเตอร์ด้วยวิธีที่นำเสนอแล้ว ซึ่งจากแนวคิดของการระบุตำแหน่งพร้อมกับการสร้างแผนที่ (SLAM) เราจะนำข้อมูลจากอุปกรณ์รับรู้ในขณะที่หุ่นยนต์การเคลื่อนที่ มาใช้ปรับปรุงตำแหน่งของหุ่นยนต์และปรับปรุงแบบจำลองวัตถุไปพร้อมกัน โดยมีจุดมุ่งหมายคือเพิ่มโอกาสความสำเร็จในการจับวัตถุที่ไม่รู้จัก ด้วยการจัดวางปลายนิ้วที่แม่นยำขึ้นนี้ ในตอนท้ายของงานวิจัยได้แสดงผลการทดลองเปรียบเทียบกับวิธีการอื่นเพื่อแสดงถึงประสิทธิผลของวิธีการที่นำเสนอ รวมไปถึงการทดสอบการจับวัตถุจริงในภารกิจหยิบและวางที่แสดงให้เห็นว่าวิธีการที่นำเสนอสามารถใช้ในการจับวัตถุที่ไม่รู้จักได้เป็นอย่างดี

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2556

5470185721 : MAJOR COMPUTER ENGINEERING
KEYWORDS: FINGERTIP PLACEMENT / UNKNOWN OBJECT / GRASPING / EYE-IN-HAND

NATCHANON WONGWILAI: FINGERTIP PLACEMENT PLANNING FOR UNKNOWN OBJECT GRASPING USING SLAM-BASED EYE-IN-HAND APPROACH. ADVISOR: ASST. PROF. NATTEE NIPARNAN, Ph.D., 67 pp.

The cause of failure in unknown object grasping is that unable to put fingertips on desired grasping points because of the localization error between a robot and an object. To solve this problem, we present an accurate fingertip placement planning method using DepthSense 325 as eye-in-hand sensor, calibrated by a novel approach. By using a SLAM concept, while the robot is moving, new information should be obtained from sensor that is used to further accurate the current position of the robot and the model of the object simultaneously in the hope that it would give better grasping result. The experiment results show the comparison of proposed method with the others that confirm effectiveness of our approach. Additionally, we also provide an example of real object grasping in pick-and-place task.

Department: Computer Engineering Student's Signature .....
Field of Study: Computer Engineering Advisor's Signature .....
Academic Year: 2013

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากการสนับสนุนและส่งเสริมเป็นอย่างดี จาก ผู้ช่วยศาสตราจารย์ ดร.นัทที นิภานันท์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งเป็นที่ปรึกษาทั้งในส่วนของแนวทางในการทำงานวิจัยชิ้นนี้ รวมถึงข้อแนะนำต่างๆ ของงานวิจัยและเรื่องอื่นๆ นอกเหนือจากงานวิจัย ข้าพเจ้าจึงขอขอบคุณอาจารย์เป็นอย่างสูงมา ณ ที่นี้

ขอขอบคุณประธานกรรมการสอบวิทยานิพนธ์ ผู้ช่วยศาสตราจารย์ ดร.อรรถวิทย์ สุดแสง ตลอดจนกรรมการสอบวิทยานิพนธ์ อาจารย์ ดร.ณัฐพงศ์ ชินธเนศ และ ดร.ธนะธร พ่อค้า ที่ได้กรุณาสละเวลาตรวจสอบและให้คำแนะนำเกี่ยวกับวิทยานิพนธ์ฉบับนี้จนเสร็จสมบูรณ์

และที่ขาดมิได้คือ ขอขอบคุณเหล่าสมาชิกของห้องปฏิบัติการวิจัยระบบอัจฉริยะ ISL2 ทุกคน ที่ช่วยให้คำแนะนำต่างๆ ทั้งในเรื่องของงานวิจัยและเรื่องอื่นๆ รวมไปถึงแนวทางในการแก้ปัญหาและอุปสรรคต่างๆ

สุดท้ายนี้ขอขอบคุณบิดามารดาผู้ให้กำเนิด รวมไปถึงญาติพี่น้องทุกคนที่ให้กำลังใจและสนับสนุนผู้วิจัยตลอดมา และขอขอบคุณอีกหลายท่านที่ไม่สามารถเอ่ยนามได้ทั้งหมด ณ ที่นี้ด้วยใจจริง

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญรูป

หน้า

บทที่ 1
บทนำ

1.1. ที่มาและความสำคัญของปัญหา

การใช้งานหุ่นยนต์ทั้งในด้านอุตสาหกรรม งานบริการ และการใช้งานในบ้าน จำเป็นที่ต้องมีความสามารถ หลายๆอย่างที่จะช่วยให้สามารถทำงานได้บรรลุวัตถุประสงค์ หนึ่งในความสามารถที่สำคัญคือ ความสามารถในการหยิบจับสิ่งของ เมื่อศึกษารายละเอียดเกี่ยวกับการจับ ในแง่ของการใช้งานจริง สภาพแวดล้อมของการทำงาน มักจะเป็นสภาพแวดล้อมที่ไม่รู้จัก กล่าวคือ หุ่นยนต์ไม่รู้จักตำแหน่ง รูปร่าง และคุณสมบัติของวัตถุที่จะจับ ถือเป็นปัญหาที่สำคัญที่รู้จักกันในชื่อ “ปัญหาการจับวัตถุที่ไม่รู้จัก” ซึ่งการพัฒนาให้หุ่นยนต์สามารถกระทำการจับวัตถุที่ไม่รู้จักได้ ยังคงเป็นปัญหาที่ท้าทาย และมีหลากหลายงานวิจัยที่พยายามพัฒนางานในส่วนนี้ให้ดียิ่งขึ้น

ในปัญหาการจับวัตถุของหุ่นยนต์ ได้มีการศึกษาและพัฒนาในแง่มุมที่หลากหลาย เริ่มตั้งแต่การออกแบบแขนและมือหุ่นยนต์ที่เหมาะสมกับการจับ การวางแผนการเคลื่อนที่ของมือหุ่นยนต์เพื่อเข้าไปจับวัตถุโดยสามารถหลบหลีกสิ่งกีดขวางได้ การคำนวณท่าทางการจับที่สามารถจับวัตถุได้อย่างมีเสถียรภาพ และองค์ความรู้ในอื่นๆ ที่เสริมความสามารถของการจับให้ดีและใช้งานได้ง่ายขึ้น ซึ่งการคำนวณหาท่าทางการจับวัตถุที่เหมาะสม เป็นหนึ่งในแนวทางที่สำคัญ ที่จำเป็นต้องพิจารณาในการจับวัตถุของหุ่นยนต์ โดยจะอาศัยข้อมูลจากอุปกรณ์รับรู้ที่หลากหลาย อาทิเช่น กล้อง (Camera), อุปกรณ์วัดระยะทางด้วยเลเซอร์ (Laser range finder) และอุปกรณ์วัดแรง (Force and contact sensor) เป็นต้น เพื่อนำข้อมูลเหล่านั้นมาใช้ ระบุตำแหน่ง สร้างแบบจำลอง หรือหาคุณสมบัติต่างๆ ของวัตถุ แล้วจึงวิเคราะห์ท่าทางและจุดจับบนพื้นผิวของวัตถุที่เหมาะสม โดยจะคำนึงถึงความสมดุลของแรงที่กระทำต่อวัตถุ เพื่อให้สามารถจับวัตถุได้โดยวัตถุไม่หลุดออกจากมือหุ่นยนต์ ซึ่งมีหลากหลายงานวิจัยที่ได้พัฒนางานในส่วนนี้ และสามารถพิสูจน์ได้ว่าท่าทางการจับที่หามาได้นั้นจะสามารถจับให้วัตถุอยู่ในมือได้อย่างสมดุล แต่ผลการจับวัตถุที่แสดงให้เห็นจริง ยังคงมีผลการจับวัตถุที่ล้มเหลวเป็นส่วนมาก

สาเหตุที่ส่งผลให้การจับวัตถุในงานวิจัยต่างๆ ล้มเหลว เกิดจากหุ่นยนต์ไม่สามารถนำปลายนิ้วไปวางลงบนจุดจับที่ต้องการได้ เนื่องจากสภาพแวดล้อมในโลกจริงนั้น จะมีความไม่แน่นอน (Uncertainty) ที่ส่งผลให้เกิดความผิดพลาดทั้ง การวัดค่าข้อมูลจากอุปกรณ์รับรู้ การควบคุมหุ่นยนต์ รวมไปถึงการประมวลผลต่างๆ ตัวอย่างเช่น ภาพที่ได้จากกล้องมีจุดสีรบกวนเกิดขึ้นแบบสุ่มทำให้การคำนวณแบบจำลองวัตถุผิดไปจากความจริง และการเคลื่อนที่ของหุ่นยนต์มีระยะเกินจากคำสั่งที่สั่งจริงทำให้การระบุตำแหน่งของปลายนิ้วผิดพลาด เป็นต้น ซึ่งผลจากความผิดพลาดเหล่านี้ ส่งผลให้การคำนวณหาจุดจับบนวัตถุ ไม่สามารถทำได้อย่างถูกต้อง และการนำนิ้วไปวางบนจุดจับเหล่านั้น ไม่สามารถนำไปวางให้ตรงจุดได้ โดยอาจกล่าวได้ว่า “แม้สามารถคำนวณหาท่าทางและจุดจับของวัตถุได้ดีเพียงใด หากไม่สามารถวางนิ้วลงบนตำแหน่งที่หามาได้นั้น ผลการจับวัตถุก็ยังคงมีโอกาสที่จะล้มเหลวสูง” ซึ่งหลายงานวิจัยจะไม่พิจารณาความผิดพลาดที่เกิดขึ้น เนื่องจากการจัดการกับความ

ผิดพลาดเหล่านี้ ต้องอาศัยการคำนวณที่ยากและซับซ้อน อีกทั้งความผิดพลาดเหล่านี้ เป็นสิ่งที่ไม่สามารถคาดเดาได้ ว่าจะเกิดขึ้นแบบใด ณ ขณะใด จึงเป็นปัญหาที่ยากต่อการแก้ไขอย่างยิ่ง

ในการจับวัตถุเชิงปฏิบัตินั้น จะมีลำดับขั้นตอนแบบพื้นฐานที่เป็นที่ยอมรับดังนี้ เริ่มต้นด้วยระบบรับข้อมูลของวัตถุจากอุปกรณ์รับรู้ จากนั้นระเบียบวิธีการในการคำนวณจุดจับจะถูกนำมาใช้เพื่อคำนวณหาจุดจับที่เหมาะสมจากข้อมูลวัตถุที่หามาได้ สุดท้ายมือจับของระบบจะถูกกำหนดทิศทางการเคลื่อนที่ด้วยระเบียบวิธีในการนำทาง ที่จะเคลื่อนที่ไปยังตำแหน่งเป้าหมายและทำจัดวางปลายนิ้วให้สัมผัสวัตถุที่จุดจับที่ต้องการ จากลำดับขั้นตอนที่กล่าวมานั้นจะสังเกตได้ว่า การรับข้อมูลของระบบนั้นจะกระทำเพียงครั้งเดียวตอนเริ่มต้นการทำงาน ซึ่งในความเป็นจริงแล้วยังมีข้อมูลที่เป็นประโยชน์อีกมากมายที่สามารถนำมาใช้เพื่อเพิ่มประสิทธิภาพในการทำงานของระบบ อาทิเช่น ข้อมูลในขณะที่แขนหุ่นยนต์กำลังเคลื่อนที่ เป็นต้น

ด้วยเหตุผลดังกล่าว งานวิจัยนี้จึงมีแนวคิดหลักคือ นำอุปกรณ์รับรู้มาใช้แบบแนวทางตาในมือ กล่าวคือจะติดตั้งอุปกรณ์รับรู้บริเวณมือจับของหุ่นยนต์ และแบ่งการเคลื่อนที่ของแขนหุ่นยนต์ออกเป็นทีละก้าว เพื่อให้สามารถรับรู้ข้อมูลในขณะที่แขนหุ่นยนต์กำลังเคลื่อนที่ได้ โดยมีจุดมุ่งเน้นเพื่อพัฒนาและวางแผนวิธีการนำปลายนิ้ว ไปวางยังจุดจับที่กำหนดบนวัตถุอย่างแม่นยำ โดยการใช้ประโยชน์จากข้อมูลที่ได้รับในการจัดการกับความผิดพลาดที่เกิดขึ้น ทั้งความผิดพลาดในการสร้างแบบจำลองของวัตถุ ซึ่งส่งผลโดยตรงต่อการหาจุดจับที่เหมาะสม และความผิดพลาดในการระบุตำแหน่งที่จะส่งผลต่อความแม่นยำในการจับวัตถุ เพื่อเพิ่มประสิทธิภาพในการจับวัตถุที่ไม่รู้จักให้ดียิ่งขึ้น

1.2. งานวิจัยที่เกี่ยวข้อง

งานวิจัยการจับวัตถุในปัจจุบันได้มีการศึกษาและพัฒนาในแง่มุมที่หลากหลาย อาทิเช่น การคำนวณหาวิธีการจับวัตถุ [1-5] การสร้างแบบจำลองวัตถุ [6-10] การระบุตำแหน่งวัตถุ [11-14] การวางแผนการเคลื่อนที่ [15, 16] และการกำหนดกลยุทธ์ต่างๆ [17, 18] เพื่อให้สามารถจับวัตถุได้ดีขึ้น เป็นต้น ซึ่งสิ่งจำเป็นที่งานวิจัยเหล่านี้ต้องมีคือ ข้อมูลของวัตถุและข้อมูลของสิ่งแวดล้อม เพื่อใช้สำหรับอธิบายลักษณะของวัตถุหรือสิ่งแวดล้อมในการจับวัตถุที่แตกต่างกันออกไป การรับรู้ข้อมูลและการนำข้อมูลมาใช้อย่างเหมาะสมจึงเป็นพื้นฐานสำคัญสำหรับการพัฒนางานวิจัยในด้านการจับวัตถุ

สำหรับงานวิจัยในเรื่องการจับวัตถุที่ไม่รู้จัก การรับรู้ข้อมูลจึงเป็นสิ่งจำเป็นในหาข้อมูลของวัตถุสำหรับการสร้างแบบจำลอง ระบุตำแหน่ง หรือจำแนกชนิดของวัตถุ และข้อมูลของสิ่งแวดล้อมสำหรับการวางแผนการเคลื่อนที่เพื่อหลบหลีกสิ่งกีดขวางในการเข้าไปจับวัตถุ โดยงานวิจัยการจับวัตถุที่ไม่รู้จักในปัจจุบัน มีการเลือกใช้อุปกรณ์รับรู้ที่หลากหลาย กล้องวิดีโอ [19-21] เป็นหนึ่งในอุปกรณ์รับรู้ที่นิยมใช้ในงานวิจัย เนื่องจากเป็นอุปกรณ์ที่ใช้งานง่าย ราคาถูก ให้ข้อมูลปริมาณมาก และความเร็วในการส่งข้อมูลสูง แต่กล้องวิดีโอจะให้ข้อมูลภาพที่เป็นสองมิติเท่านั้น ไม่สามารถอธิบายถึงข้อมูลใน 3 มิติ นั่นคือไม่สามารถบอกความลึกระหว่างวัตถุถึงกล้องได้ ทำให้ในหลายงานวิจัยจะมีการประยุกต์ใช้อุปกรณ์รับรู้อื่น ที่แตกต่างกันออกไปเพื่อนำมาหาข้อมูลใน 3 มิติของวัตถุ อาทิเช่น กล้องส

เตอริโอ (Stereo camera) [22-25] อุปกรณ์วัดระยะด้วยเลเซอร์ (Laser range finder) [26-29] หรือแม้กระทั่งอุปกรณ์รับรู้การสัมผัส (Tactile sensor) [10, 30] ที่ใช้ร่วมกับ encoder ของแขนหุ่นยนต์เพื่อระบุตำแหน่งใน 3 มิติของพื้นผิวที่สัมผัส

กล้องไทม์ออฟไฟลท์ (Time-of-flight camera) ซึ่งเป็นกล้องที่ใช้คำนวณระยะทางโดยอาศัยการวัดเวลาเดินทางของแสงที่ปล่อยออกมา เป็นอีกอุปกรณ์หนึ่งที่นิยมใช้ในการหาข้อมูลใน 3 มิติของวัตถุ [31, 32] เนื่องด้วยให้ข้อมูลความลึกได้ในความถี่ที่สูง มีขนาดเล็กและน้ำหนักเบาทำให้สามารถเลือกติดตั้งได้หลากหลายที่โดยไม่กีดขวางการทำงานของระบบ อีกทั้งยังใช้พลังงานในการทำงานน้อย ด้วยเหตุผลดังกล่าวกล้องไทม์ออฟไฟลท์จึงถูกเลือกใช้เป็นอุปกรณ์รับรู้ภายในงานวิจัยนี้

จากที่ได้กล่าวในข้างต้นว่า ข้อมูลของวัตถุและสิ่งแวดล้อมนั้นเป็นสิ่งสำคัญในงานวิจัยการจับวัตถุที่ไม่รู้จัก ถ้าเราเลือกติดตั้งอุปกรณ์รับรู้ให้อยู่นิ่ง ข้อมูลที่ได้จะเป็นข้อมูลเพียงมุมมองเดียว แตกต่างจากแนวทางตาในมือ (Eye-in-hand approach) ที่จะนำอุปกรณ์รับรู้มาติดตั้งไว้กับแขนหุ่นยนต์ สามารถปรับเปลี่ยนมุมมองของอุปกรณ์รับรู้ ไปสำรวจมุมมองอื่นของวัตถุได้ ซึ่งมีหลายงานวิจัยใช้แนวทางนี้ในการพัฒนางานวิจัย เช่น ในงานวิจัย [33] จะสร้างแบบจำลองของวัตถุจากการตัดกันของภาพเงา (Silhouette) ด้วยการเคลื่อนที่แขนหุ่นยนต์รอบวัตถุ จากนั้นจะคำนวณจุดจับแล้วทำการจับวัตถุทันทีแบบอัตโนมัติ ซึ่งคล้ายกับงานวิจัย [21, 34] ที่จะสร้างแบบจำลองของวัตถุจากการตัดกันของภาพเงาเช่นกัน แต่งานวิจัยนี้จะลดเวลาในการทำงานโดยจะเคลื่อนที่แขนหุ่นยนต์ไปพร้อมกับการคำนวณหาแบบจำลองวัตถุ และงานวิจัย [20] ได้นำเสนอวิธีการจับวัตถุที่ไม่รู้จัก มีข้อดีคือ ง่าย เร็ว และทนทานต่อความผิดพลาด โดยในงานนี้จะใช้ข้อมูลภาพขอบของวัตถุ นำมาหาเส้นโค้งที่สามารถล้อมรอบภาพวัตถุที่ได้ และเสนอว่าจุดที่เป็น concave บนเส้นโค้งนั้นน่าจะเป็นจุดจับที่ดี จากนั้นก็ทำการเลื่อนกล้องไปมุมต่างๆ เพื่อหาภาพที่ให้ค่าความโค้งมากที่สุด นั่นคือมุมมองที่จะสามารถหาจุดจับที่ดีได้นั่นเอง

ปริมาณข้อมูลที่มากขึ้น ความผิดพลาดที่เกิดจากความไม่แน่นอนของสิ่งแวดล้อมก็จะมากขึ้นหากไม่มีการจัดการที่ดีก็อาจส่งผลให้มีโอกาสเกิดความล้มเหลวในการจับวัตถุสูง ในหลายปีที่ผ่านมาจึงได้มีแนวคิด การจัดการภายใต้ความไม่แน่นอน (Manipulation under uncertainty) [35] ที่เป็นแนวทางการวิจัยของงานทางด้าน Manipulation ที่อยู่บนพื้นฐานของความไม่แน่นอนที่เกิดขึ้นในสภาพแวดล้อม และพยายามจัดการหรือลดปริมาณความผิดพลาดที่เกิดขึ้น เพื่อเพิ่มความถูกต้องและแม่นยำในการทำงานของหุ่นยนต์ ซึ่งเป็นแนวทางที่มีความสำคัญอย่างยิ่ง ในการพัฒนางานวิจัยการจับวัตถุที่ไม่รู้จัก ภายใต้สภาพแวดล้อมไม่อาจคาดเดาได้ โดยในปัจจุบันมีการพัฒนาภายใต้สภาพแวดล้อมที่หลากหลาย และมีลักษณะการใช้งานที่แตกต่างกันออกไป เช่น การประกอบอุปกรณ์ในโรงงาน [36], การหยิบจับอุปกรณ์ในครัว [37], การทำความสะอาดพื้นโต๊ะ [38] และการยกและวางวัตถุ [39, 40] เป็นต้น

มีหลายแนวทางที่จัดการกับสิ่งแวดล้อมที่ไม่แน่นอน ซึ่งบางวิธีจะขึ้นอยู่กับข้อจำกัดทางแบบจำลองของวัตถุ เช่น บริเวณสัมผัสอิสระ (Independent contact region) [41] ที่จะคำนวณพื้นที่ที่จะวางนิ้วมือ แทนการใช้จุด ซึ่งจะช่วยให้สามารถมีความผิดพลาดในการวางนิ้วได้เล็กน้อย แต่อย่างไรก็ตามวิธีนี้จำเป็นต้องรู้แบบจำลองของวัตถุมาก่อน เป็นต้น ในบางวิธีจะใช้การควบคุมหุ่นยนต์

เพื่อจัดการกับความไม่แน่นอนที่เกิดขึ้น อาทิเช่น Visual Servoing [16, 20, 42] จะใช้ error ที่หาได้จากภาพ มาควบคุมการเคลื่อนที่ของหุ่นยนต์ เพื่อลด error ให้ลู่เข้าสู่ค่าศูนย์ แต่วิธีการนี้จะไม่รับรองผลว่าจะสามารถทำให้ลู่เข้าได้ เป็นต้น หรือในงานวิจัยที่ใช้การรับรู้ข้อมูลแบบก่อนการสัมผัส (Pre-touch sensing) เพื่อกำหนดกลยุทธ์ให้เหมาะสมกับสถานการณ์ ณ ขณะนั้นๆ ตัวอย่างเช่น งานวิจัย [17] ได้นำเสนอวิธีการปรับตำแหน่งของมือก่อนการจับวัตถุ โดยการใช้อุปกรณ์วัดการสัมผัส (Tactile sensor) ซึ่งมีงานวิจัยอื่นๆ ได้ใช้แนวคิดในลักษณะเดียวกัน แต่มีการเปลี่ยนอุปกรณ์รับรู้เป็น Infra-red (IR) [43] อุปกรณ์วัดลำแสง (Light beam sensor) [44] อุปกรณ์วัดแรงและการสัมผัส (Force and tactile sensor) [4] และอุปกรณ์วัดระยะด้วยปรากฏการณ์เสียงจากเปลือยหอย (Seashell effect sensor) [45] ซึ่งวิธีเหล่านี้จำเป็นต้องนำอุปกรณ์รับรู้ เข้าไปใกล้กับวัตถุหรือสัมผัสวัตถุก่อนจึงจะสามารถวัดค่าข้อมูลได้ ซึ่งอาจทำให้วัตถุเลื่อนตำแหน่งหรือเกิดความเสียหายจากการชนของมือหุ่นยนต์ได้

ความผิดพลาดที่เป็นสาเหตุของการจับวัตถุล้มเหลวสามารถแบ่งได้เป็น 2 อย่างคือ ความผิดพลาดในการระบุตำแหน่งของวัตถุเทียบกับแขนหุ่นยนต์ และความผิดพลาดในการสร้างแบบจำลองวัตถุ สำหรับแนวทางตาในมือนั้นการสร้างแบบจำลองวัตถุจะอาศัยตำแหน่งของมือหุ่นยนต์ในการระบุว่าข้อมูลวัตถุที่หาได้อยู่ตำแหน่งใดใน 3 มิติเทียบกับฐานหุ่นยนต์ และใช้วิธีการเดียวกันนี้สำหรับข้อมูลถัดไปที่ได้จากการเปลี่ยนมุมมอง ซึ่งถ้าตำแหน่งของมือหุ่นยนต์นั้นผิดพลาดแบบจำลองวัตถุที่ได้ก็จะผิดพลาดเช่นกัน และในการระบุตำแหน่งมือหุ่นยนต์นั้นก็สามารถใช้ข้อมูลจากอุปกรณ์รับรู้ในปัจจุบันหาความสัมพันธ์เทียบกับข้อมูลแบบจำลองวัตถุที่หาได้ในอดีต เพื่อปรับปรุงตำแหน่งปัจจุบันของมือหุ่นยนต์ได้ จากความสัมพันธ์ระหว่างการระบุตำแหน่งและการสร้างแบบจำลองวัตถุนี้ จะสังเกตได้ว่าเราสามารถใช้การระบุตำแหน่งช่วยปรับปรุงแบบจำลองวัตถุ และสามารถใช้แบบจำลองวัตถุปรับปรุงการระบุตำแหน่งไปพร้อมกันได้ ซึ่งแนวคิดนี้คือแนวคิดของปัญหาการระบุตำแหน่งพร้อมกับการสร้างแผนที่ (Simultaneous localization and mapping: SLAM) [46, 47] เป็นกระบวนการที่หุ่นยนต์สามารถสร้างแผนที่ของสภาพแวดล้อมในขณะเคลื่อนที่และระบุตำแหน่งของตนในแผนที่นั้นไปพร้อมกัน โดยสามารถมองได้ว่าแผนที่คือข้อมูลของแบบจำลองวัตถุนั่นเอง

จากงานวิจัยที่ได้กล่าวมาข้างต้น วิทยานิพนธ์ฉบับนี้จะเลือกใช้แนวทางตาในมือ เนื่องด้วยข้อดีจากการรับรู้ข้อมูลจากหลากหลายมุมมอง โดยใช้กล้องไทม์ออฟไฟลท์ DepthSense 325 [48] เป็นอุปกรณ์รับรู้ของระบบ ที่มีข้อดีคือให้ทั้งข้อมูลความลึกและข้อมูลสี มีขนาดเล็กติดตั้งบนแขนหุ่นยนต์ได้โดยไม่กีดขวางการทำงาน และราคาไม่แพง และจากแนวคิดของ SLAM วิทยานิพนธ์นี้จะอาศัยข้อมูลจากการเคลื่อนที่ของแขนหุ่นยนต์ ในการระบุตำแหน่งหุ่นยนต์พร้อมกับสร้างแบบจำลองวัตถุ โดยมีเป้าหมายคือเพื่อเพิ่มความถูกต้องแม่นยำในการนำปลายนิ้วของมือหุ่นยนต์ ไปวางลงบนพื้นผิวของวัตถุที่ไม่รู้จักในจุดที่กำหนดไว้ รวมไปถึงใช้ประโยชน์จากข้อมูลการระบุตำแหน่งและข้อมูลแบบจำลองวัตถุในการจัดการกับความผิดพลาดที่เกิดขึ้นภายในระบบด้วย

1.3. การนำเสนอและลำดับเนื้อหาวิทยานิพนธ์

การนำเสนอเนื้อหาวิทยานิพนธ์แบ่งออกเป็น 3 ส่วนหลักได้แก่ ทฤษฎีที่เกี่ยวข้อง ขั้นตอนการแก้ปัญหาของวิทยานิพนธ์ และการทดสอบการทำงานและสรุปผล

1) *ทฤษฎีที่เกี่ยวข้อง* จะอธิบายทฤษฎีและหลักการที่เกี่ยวข้องกับการพัฒนาวิธีการวางแผนการจัดวางปลายนิ้วสำหรับการจับวัตถุที่ไม่รู้จัก โดยจะอธิบายไว้ในบทที่ 2 ที่จะกล่าวถึงแนวคิดและหลักการของการระบุตำแหน่งพร้อมกับสร้างแผนที่ และวิธีการประมาณค่าเหมาะสมที่สุด รวมไปถึงทฤษฎีที่ใช้สำหรับการวิเคราะห์และจัดการกับข้อมูลทั้งในสองมิติและสามมิติ เช่น ICP, Optical flow และ RANSAC เป็นต้น
2) *ขั้นตอนการแก้ปัญหาของวิทยานิพนธ์* จะแสดงขั้นตอนของการปรับแก้พารามิเตอร์ทั้งภายในและภายนอกของอุปกรณ์รับรู้ ที่เป็นขั้นตอนที่สำคัญก่อนนำอุปกรณ์รับรู้ไปใช้งานจริง ซึ่งอธิบายไว้ในบทที่ 3 รวมไปถึงจะแสดงผลการทดสอบการปรับแก้พารามิเตอร์ไว้ในตอนท้ายของบทนี้ สำหรับบทที่ 4 จะอธิบายถึงขั้นตอนในการแก้ปัญหาการวางแผนการจัดวางปลายนิ้วโดยใช้แนวคิดใหม่ที่ได้นำเสนอ รวมไปถึงอธิบายการทำงานของวิธีการอื่นที่จะนำมาใช้เปรียบเทียบผลลัพธ์กับวิธีที่นำเสนอ
3) *การทดสอบการทำงานและสรุปผล* ในบทที่ 5 จะแสดงผลการทดลองของวิธีการที่นำเสนอเปรียบเทียบกับวิธีการอื่นที่ได้อธิบายไว้ในบทที่ 4 ซึ่งจะเปรียบเทียบความผิดพลาดที่เกิดขึ้นได้แก่ ความผิดพลาดในการระบุตำแหน่งของหุ่นยนต์เทียบกับวัตถุ ความผิดพลาดของการสร้างแบบจำลองวัตถุ และความผิดพลาดในการจัดวางปลายนิ้ว รวมไปถึงแสดงผลการทดสอบการจับวัตถุจริงในภารกิจหยิบและวางวัตถุ และในบทที่ 6 จะเป็นการสรุปผลการวิจัยและเสนอแนะแนวทางสำหรับการทำวิจัยขั้นถัดไปในอนาคต

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1. การระบุตำแหน่งพร้อมกับการสร้างแผนที่ (Simultaneous localization and mapping: SLAM)

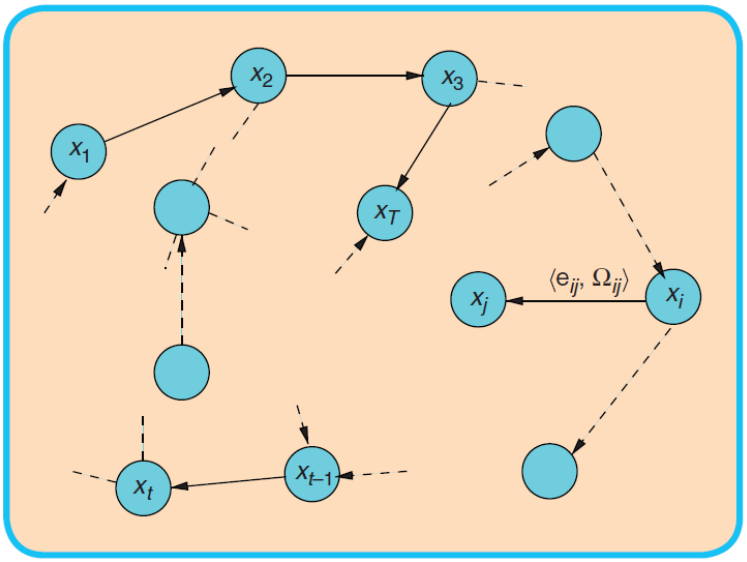
วิธีการระบุตำแหน่งพร้อมกับการสร้างแผนที่ (SLAM) [46, 47] เป็นกระบวนการแก้ปัญหาการเคลื่อนที่ของหุ่นยนต์ในสภาพแวดล้อมที่ไม่รู้จัก ซึ่งในขณะเริ่มต้นการทำงาน หุ่นยนต์จะไม่มีข้อมูลของสภาพแวดล้อม จึงจำเป็นต้องทำการสร้างแผนที่ของสภาพแวดล้อมขึ้นจากข้อมูลที่ได้จากอุปกรณ์รับรู้ จากนั้นจึงระบุตำแหน่งของหุ่นยนต์จากแผนที่นี้ เมื่อหุ่นยนต์เคลื่อนที่ตำแหน่งของหุ่นยนต์จะถูกทำนายขึ้น เพื่อนำไปใช้สร้างแผนที่จากข้อมูลของสภาพแวดล้อมใหม่ที่ได้รับ ซึ่งข้อมูลสภาพแวดล้อมใหม่นี้จะมีความสัมพันธ์กับแผนที่ในอดีตที่สร้างขึ้น ที่สามารถนำไปใช้ปรับปรุงตำแหน่งของหุ่นยนต์ได้ต่อไป การระบุตำแหน่งและการสร้างแผนที่จึงมีความเกี่ยวพันกันมาก ที่ต้องอาศัยการทำงานไปพร้อมกันเพื่อใช้ประโยชน์จากข้อมูลซึ่งกันและกัน โดยในสภาพแวดล้อมที่มีความไม่แน่นอนนั้น อาจกล่าวได้ว่าความไม่แน่นอนในการระบุตำแหน่งขึ้นกับความไม่แน่นอนของแผนที่ และความไม่แน่นอนในการสร้างแผนที่ขึ้นกับความไม่แน่นอนของตำแหน่งหุ่นยนต์ โดยแนวทางการแก้ปัญหา SLAM บนความไม่แน่นอนเหล่านี้ จะอาศัยวิธีการในเชิงความน่าจะเป็น เพื่อทำให้ความไม่แน่นอนเหล่านี้มีค่าน้อยที่สุด

ในงานวิจัยทางด้านการจับวัตถุที่ไม่รู้จักนั้น สามารถนำแนวคิดของ SLAM นี้ไปประยุกต์ใช้ได้ โดยการใช้แนวทางตาในมือที่ติดตั้งอุปกรณ์รับรู้ไว้ที่มือหุ่นยนต์ โดยเปรียบการระบุตำแหน่งหุ่นยนต์ในสภาพแวดล้อมที่ไม่รู้จักคือ การระบุตำแหน่งของมือหุ่นยนต์เทียบกับวัตถุที่ไม่รู้จัก และสำหรับแผนที่ที่สร้างขึ้นเปรียบได้กับแบบจำลองของวัตถุนั่นเอง ซึ่งมีเป้าหมายคือลดความผิดพลาดในการระบุตำแหน่งและความผิดพลาดของแบบจำลองวัตถุ เพื่อเพิ่มโอกาสความสำเร็จในการจับวัตถุ

2.1.1. การระบุตำแหน่งพร้อมกับการสร้างแผนที่โดยอาศัยกราฟ (Graph-based SLAM)

สำหรับแนวทางในการแก้ปัญหา SLAM นั้นมีหลากหลายแนวทาง การระบุตำแหน่งพร้อมกับการสร้างแผนที่โดยอาศัยกราฟ (Graph-based SLAM) [49] เป็นแนวทางหนึ่งที่กำหนดให้ข้อมูลต่างๆ ในระบบอยู่ในรูปแบบของกราฟ โดยมีปม (Node) เป็นตำแหน่งของหุ่นยนต์ และข้อจำกัดเชิงพื้นที่ (Spatial constraint) ของสองปมแทนด้วยเส้นเชื่อม (Edge) ที่หาได้จากการวัดค่า (Measurement) จากรูปที่ 2.1 แสดงให้เห็นถึงลักษณะของกราฟในการแสดงการทำงานของ SLAM โดยเส้นเชื่อมที่เป็นเส้นทึบแสดงถึงการวัดเสมือน (Virtual measurement) ที่บอกความสัมพันธ์เชิงตำแหน่งพร้อมด้วยการกระจายความน่านำเป็น (Probability distribution) ของปมที่อยู่ปลายลูกศรเทียบกับปมที่อยู่อีกด้านของเส้นเชื่อม ที่ทำให้การวัดจริง (Real measurement) ที่ได้จากแต่ละปมสอดคล้องกัน ตัวอย่างเช่น มีข้อมูล point cloud ของสภาพแวดล้อมจากสองมุมมองของกล้องที่แตกต่างกัน เราจะใช้ข้อมูล point cloud ทั้งสองชุดนี้นำมาซ้อนทับให้เข้ากันมากที่สุด เพื่อหาการ

แปลง (Transformation) ระหว่างสองมุมมองนี้ ซึ่งการแปลงที่ได้นี้คือการวัดเสมือนนั่นเอง สำหรับเส้นประคือเส้นเชื่อมที่แสดงความสัมพันธ์เชิงตำแหน่งที่ได้จากการวัดค่าหลายครั้งของชิ้นส่วนเดียวกันในสภาพแวดล้อม



รูปที่ 2.1 กราฟที่เป็นตัวแทนวิธีการทำงานของ SLAM

กำหนดให้ $\boldsymbol{x} = (x_1, \ldots, x_T)^T$ คือเวกเตอร์ของพารามิเตอร์ในระบบ โดย x_i คือตำแหน่งของปมที่ i และให้ z_{ij} และ Ω_{ij} คือค่าเฉลี่ยและเมทริกซ์ข้อมูล (Information matrix) ของการวัดเสมือนระหว่างปมที่ i และ j ตามลำดับ โดยการวัดเสมือนนี้คือการแปลงที่ทำให้ข้อมูลที่วัดได้จาก i ซ้อนทับกับข้อมูลที่วัดได้จาก j มากที่สุด สำหรับการทำนายค่าของการวัดเสมือนกำหนดให้เป็นฟังก์ชัน $\hat{z}_{ij}(x_i, x_j)$ ที่รับค่าปม x_i และ x_j ซึ่งโดยปกติแล้วฟังก์ชันนี้จะให้ค่าเป็นการแปลงแบบสัมพัทธ์ระหว่างปมทั้งสอง สำหรับความน่าจะเป็นแบบลอการึทึม (Log-likelihood) l_{ij} ของการวัด z_{ij} คือ

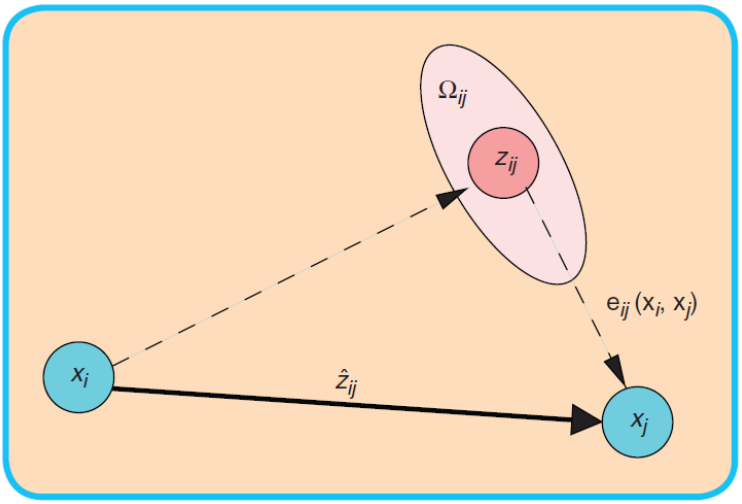
$$l_{ij} \propto [z_{ij} - \hat{z}_{ij}(x_i, x_j)]^T \Omega_{ij} [z_{ij} - \hat{z}_{ij}(x_i, x_j)]$$

โดยกำหนดให้ $e(x_i, x_j, z_{ij})$ คือฟังก์ชันความแตกต่างระหว่างการวัดที่ทำนายได้ $\hat{z}_{ij}$ กับการวัดจากข้อมูลที่ได้จากหุ่นยนต์ z_{ij} และเพื่อความสะดวกในการเขียนสมการจะขอนำดัชนีของการวัดมากำหนดเป็นดัชนีของฟังก์ชันนี้แทนได้เป็นสมการดังนี้

$$e_{ij}(x_i, x_j) = z_{ij} - \hat{z}_{ij}(x_i, x_j)$$

รูปที่ 2.2 แสดงการนิยามเส้นเชื่อมของกราฟ โดยวงกลมสีแดงคือการวัดเสมือน z_{ij} ที่วัดค่าตำแหน่งของปม x_j เทียบกับปม x_i กล่าวคือ z_{ij} จะเปรียบเสมือนตำแหน่งของปม x_j เทียบกับปม x_i ที่ทำการวัดค่ามาได้ โดยมีวงรีสีชมพูบอกถึงความมั่นใจในตำแหน่งที่วัดมาได้นี้ สำหรับวงกลมฟ้าจะแทนตำแหน่งปัจจุบันของแต่ละปม โดยตำแหน่งของปม x_j ในปัจจุบันจะหาค่าตำแหน่งมาจากการ

คาดเดาการแปลงเทียบกับ x_i ด้วย $\hat{z}_{ij}$ โดยในอุดมคติแล้ว z_{ij} ควรจะมีตำแหน่งเดียวกับปม x_j ซึ่งในการนิยามเส้นเชื่อมในกราฟนั้นจะกำหนดโดยฟังก์ชันความผิดพลาด $e_{ij}(x_i, x_j)$ และ Ω_{ij} เท่านั้น (รูปที่ 2.1) ที่สามารถบอกได้ว่าปม x_j มีความผิดพลาดในการวัดและมีความน่าเชื่อถือมากน้อยเพียงใด ซึ่งเป็นข้อมูลที่ใช้ในการหาค่าเหมาะสมที่สุดสำหรับการปรับปรุงตำแหน่งของปมในกราฟ



รูปที่ 2.2 การนิยามเส้นเชื่อมของกราฟ

กำหนดให้ C คือเซตของคู่ดัชนีของปมที่มีการวัดค่า z_{ij} และกำหนดให้ $F(x)$ คือฟังก์ชันผลรวมของความน่าจะเป็นแบบลอการึทึมเชิงลบ (Negative log-likelihood) ของทุกการวัดมีค่าดังสมการนี้

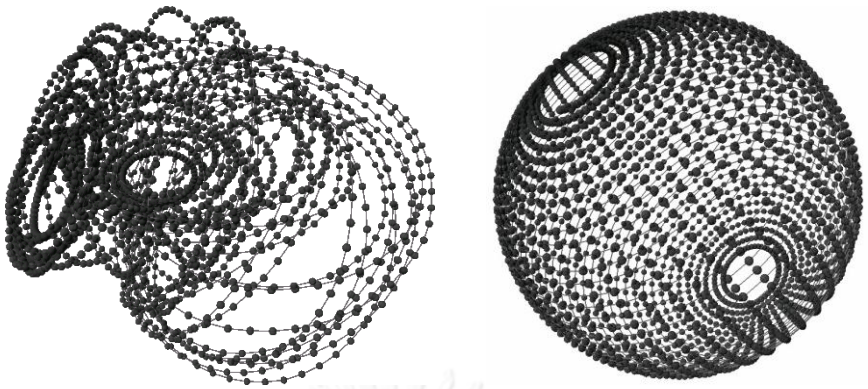
$$F(x) = \sum_{\langle i,j \rangle \in C} e_{ij}^T \Omega_{ij} e_{ij}$$

ซึ่งในการปรับปรุงตำแหน่งปมในกราฟนั้นจะกระทำโดย ทำให้ผลรวมความน่าจะเป็นในระบบมีค่ามากที่สุด นั่นคือการหาเวกเตอร์พารามิเตอร์ของระบบ x^* ที่ทำให้ฟังชัน $F(x)$ มีค่าน้อยที่สุดดังสมการ

$$x^* = \underset{x}{\operatorname{argmin}} F(x)$$

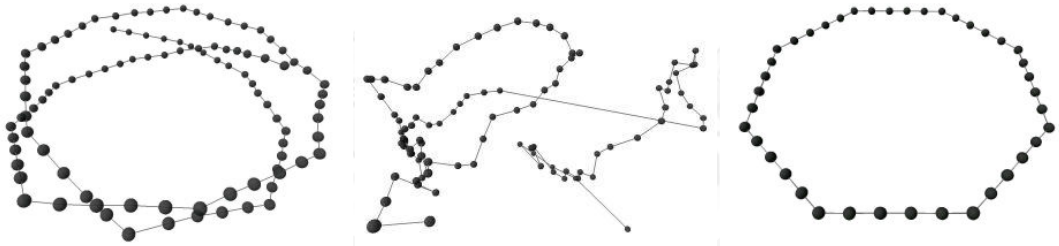
2.1.2. การหาค่าเหมาะสมที่สุดของกราฟ (Graph optimization)

สำหรับการหาค่าเหมาะสมที่สุดของกราฟที่ได้กล่าวมาในข้างต้นนั้น สามารถคำนวณได้ด้วยวิธีการพื้นฐานในการหาค่าเหมาะสมที่สุด อาทิเช่น Nonlinear least-squares, Gauss-Newton และ Levenberg- Marquardt เป็นต้น ซึ่งผลลัพธ์ความถูกต้องของกราฟที่ได้จะขึ้นอยู่กับความซับซ้อนของกราฟ และเวลาในการคำนวณจะแปรผันตามจำนวนเส้นเชื่อมที่มีในกราฟ



รูปที่ 2.3 ตัวอย่างผลลัพธ์การหาค่าเหมาะสมที่สุดด้วย TORO

Tree-based network optimizer (TORO) [50] เป็นกระบวนการแก้ปัญหาการหาค่าเหมาะสมที่สุดของกราฟความน่านะเป็น ซึ่งมีวัตถุประสงค์ในการใช้ในงานวิจัย SLAM ที่มีการใช้กราฟ โดยวิธีการนี้จะใช้แนวทางในการหาค่าเหมาะสมที่สุดด้วย gradient descent จุดเด่นของวิธีการนี้คือใช้ระยะเวลาในการประมวลผลน้อย โดยการแปลงปมของกราฟให้อยู่ในรูปแบบของต้นไม้ ที่มีการปรับแต่งค่าเพื่อให้มีความถูกต้องมากขึ้นด้วย ตัวอย่างเช่น จากรูป 2.3 เป็นข้อมูลที่มีจำนวนเส้นเชื่อม 8,600 เส้นทำการหาค่าเหมาะสมที่สุดด้วย TORO จำนวน 100 รอบใช้เวลาเพียง 21 วินาที อีกทั้งยังมีการแก้ไขปัญหาการกระจายความผิดพลาดในการหมุนที่เกิดขึ้นในกราฟ 3 มิติด้วย ดังรูปที่ 2.4



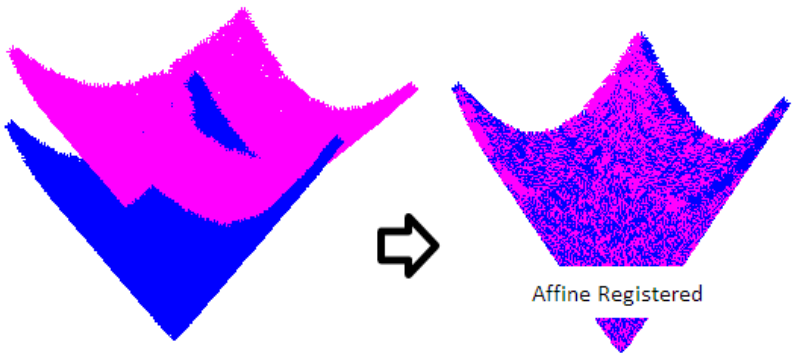
รูปที่ 2.4 ตัวอย่างปัญหาการกระจายความผิดพลาดใน 3 มิติ ซ้าย) ข้อมูลต้นนำเข้า กลาง) ผลลัพธ์จากวิธีการอื่น ขวา) ผลลัพธ์จาก TORO

สำหรับข้อมูลนำเข้าของวิธีการนี้จะเป็นเวกเตอร์ของปมทั้งหมดในกราฟกำหนดให้เป็น $\boldsymbol{x} = (x_1, \dots, x_T)^T$ โดยที่ x_i คือตำแหน่งของปมที่ i และเวกเตอร์ของเส้นเชื่อมซึ่งประกอบด้วย z_{ij} และ Ω_{ij} ทั้งหมดที่มีในกราฟ โดย z_{ij} คือการแปลงที่วัดค่าได้ระหว่างปมที่ j เทียบกับปมที่ i และ Ω_{ij} คือเมทริกซ์ข้อมูลของ z_{ij} ซึ่ง x_i และ z_{ij} จะอยู่ในรูปของเวกเตอร์ $(x, y, z, \alpha, \beta, \gamma)^T$ โดยที่ x, y, z คือตำแหน่งใน 3 มิติและ α, β, γ คือการหมุนในแกน roll pitch และ yaw ตามลำดับ และ Ω_{ij} จะอยู่ในรูปของเมทริกซ์ขนาด 6 x 6 ตามขนาดของเวกเตอร์ปม x_i สุดท้ายผลลัพธ์ที่ได้จาก TORO คือเวกเตอร์ของปมใหม่ที่ผ่านการหาค่าเหมาะสมที่สุดแล้วดังรูปที่ 2.3 ขวา

2.2. การรวมข้อมูลให้เข้ากันในสามมิติ (3D registration)

2.2.1. Iterative Closet Point แบบจุดถึงจุด (Point to point ICP)

Iterative Closet Point (ICP) คือวิธีการที่ทำให้ผลต่างของตำแหน่งจุดระหว่างสอง point cloud มีค่าน้อยที่สุดดังรูปที่ 2.5 [51] ซึ่งมักนำมาใช้ทั้งใน 2 มิติและ 3 มิติ เพื่อสร้างพื้นผิวของสภาพแวดล้อมจากมุมมองของอุปกรณ์รับรู้ที่แตกต่างกัน วางแผนการเคลื่อนที่ที่เหมาะสม จนไปถึงใช้ในการระบุตำแหน่งหุ่นยนต์



รูปที่ 2.5 ตัวอย่างผลลัพธ์จากวิธีการ ICP

โดยจะใช้ข้อมูลนำเข้าเป็น point cloud จำนวนสองชุดคือ point cloud ต้นฉบับกำหนดให้เป็น $P_s = \{p_s^1, \dots, p_s^M\}$ และ point cloud เป้าหมายที่ต้องการแปลงตำแหน่งของจุดใน point cloud ต้นฉบับเพื่อให้ซ้อนทับเข้ากับpoint cloud เป้าหมายนี้ กำหนดเป็น $P_t = \{p_t^1, \dots, p_t^N\}$ สำหรับ ICP แบบจุดถึงจุด (Point to point ICP) จะเป็นวิธีการมาตรฐานที่จะทำให้ผลรวมระยะห่างของจุดที่ใกล้ที่สุดระหว่าง P_t และ P_s มีค่าน้อยที่สุด กล่าวคือจะทำการหาเมทริกซ์การแปลง T^* ที่ทำให้ฟังก์ชันความผิดพลาดดังต่อไปนี้น้อยที่สุด

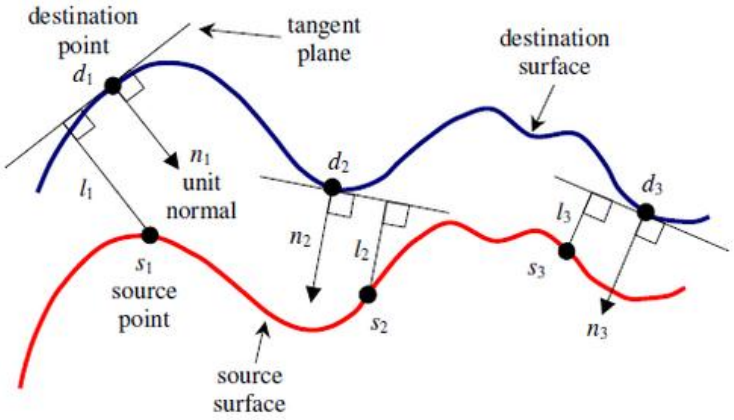
$$e(T) = \sum_{i=1}^{M} \min_{p_t^j \in P_t} \left| T(p_s^i) - p_t^j \right|^2$$

และเมทริกซ์การแปลง T^* จะมีค่าดังสมการต่อไปนี้

$$T^* = \underset{T}{\operatorname{argmin}}\, e(T)$$

ซึ่งจะต้องทำการวนซ้ำสองชั้น กล่าวคือวนซ้ำชั้นแรกเพื่อหาจุด p_t^j ที่ใกล้กับจุด p_s^i มากที่สุด และวนซ้ำชั้นที่สองเพื่อหา T^* ซึ่งวิธีการนี้จะลู่เข้าสู่ค่าต่ำสุดเฉพาะที่ (Local minimum) ดังนั้นจึงต้องเลือกใช้ค่าเริ่มต้นของ T ที่เหมาะสม

2.2.2. Iterative Closet Point แบบจุดถึงระนาบ (Point to plane ICP)



รูปที่ 2.6 ความผิดพลาดแบบจุดถึงระนาบ

เนื่องด้วย ICP แบบจุดต่อจุดนั้นจะมีปัญหาการไถลของพื้นผิวออกด้านข้าง และมีการลู่เข้าสู่ค่าต่ำสุดช้า ซึ่งปัญหานี้จะสามารถแก้ไขได้ด้วยการเปลี่ยนฟังก์ชันความผิดพลาดเป็นแบบจุดต่อระนาบ กล่าวคือจะคำนวณระยะห่างจากจุดใน P_s ถึงระนาบสัมผัส (Tangent plane) ของจุดที่สอดคล้องกัน (Correspondence) ใน P_t ดังรูปที่ 2.6 [52] ซึ่งจะทำให้ฟังก์ชันความผิดพลาดเปลี่ยนเป็น

$$e(T) = \sum_{i=1}^{M} \left[\left(T\left(p_s^i\right) - p_t^{corr(i)} \right) \cdot \boldsymbol{n}_t^{corr(i)} \right]^2$$

โดยที่ $corr(i)$ คือดัชนีของจุดใน P_t ที่มีความสอดคล้องกับจุดดัชนีที่ i ใน P_s และ $\boldsymbol{n}_t^{corr(i)}$ คือเวกเตอร์ปกติของระนาบสัมผัส ณ จุด $p_t^{corr(i)}$ จากนั้นจึงทำการหาค่า T^* ที่ทำให้ฟังก์ชันความผิดพลาดนี้มีค่าน้อยที่สุดต่อไป

สำหรับการหาระนาบสัมผัสของ point cloud นั่นคือการหาเวกเตอร์ปกติของ point cloud นั้นจะใช้วิธีการวิเคราะห์องค์ประกอบหลัก (Principal component analysis: PCA) โดยใช้การวิเคราะห์เวกเตอร์ลักษณะเฉพาะ (Eigen vector) และค่าลักษณะเฉพาะ (Eigen value) ของเมทริกซ์ความแปรปรวนร่วมเกี่ยว (Covariance matrix) กำหนดให้เป็น C ที่สร้างจากจุดเพื่อนบ้านที่ใกล้ที่สุด (Nearest neighbors) ของจุดที่ต้องการหาเวกเตอร์ปกติ โดยที่ C จะคำนวณได้จากสมการต่อไปนี้

$$C = \frac{1}{k} \sum_{j=1}^{k} \left(p_i^j - \bar{p}\right) \cdot \left(p_i^j - \bar{p}\right)^T$$

โดย p_i^j และ k คือจุดเพื่อนบ้านและจำนวนของจุดเพื่อนบ้านของจุด p_i ตามลำดับและ $\bar{p}$ คือค่าเฉลี่ยตำแหน่งใน 3 มิติของจุดเพื่อนบ้านเหล่านี้ จากนั้นจึงหาค่าลักษณะเฉพาะและเวกเตอร์

ลักษณะเฉพาะของเมทริกซ์ C ด้วยวิธีการแยกเมทริกซ์ (Matrix decomposition) จากสมการดังต่อไปนี้

$$C \cdot \vec{v}_j = \lambda_j \cdot \vec{v}_j, \qquad j \in \{0,1,2\}$$

โดย λ_j คือค่าลักษณะเฉพาะลำดับที่ j ของ C และ $\vec{v}_j$ คือเวกเตอร์ลักษณะเฉพาะลำดับที่ j ซึ่งเวกเตอร์ปกติของจุด p_i จะคือ $\vec{v}_j$ ที่มีค่า λ_j น้อยที่สุด ซึ่งจะหาเวกเตอร์ปกตินี้สำหรับทุกจุด p_i ที่เป็นสมาชิกของ point cloud ที่ต้องการ

2.3. การประมาณการเคลื่อนที่จากการมองเห็น (Visual odometry)

ข้อมูลที่ได้จากวิธีที่การประมาณการเคลื่อนที่จากการมองเห็น (Visual odometry) จะเปรียบได้กับข้อมูลที่ได้จากอุปกรณ์วัดระยะทาง (Odometry) ในหุ่นยนต์ แต่จะไม่มีข้อเสียในด้านของปัญหาความแม่นยำอันเนื่องมาจากการลื่นไถลของล้อหุ่นยนต์สำหรับการเคลื่อนที่ในแนวราบ โดยการประมาณการเคลื่อนที่จากการมองเห็นจะเป็นกระบวนการของการระบุตำแหน่งและทิศทางของหุ่นยนต์โดยอาศัยการวิเคราะห์ข้อมูลจากชุดภาพสี ในมุมมองที่แตกต่างกันของกล้อง ซึ่งข้อมูลภาพนี้ให้รายละเอียดที่มากกว่าที่ช่วยให้การประมาณตำแหน่งของหุ่นยนต์มีความแม่นยำที่มากขึ้น สำหรับขั้นตอนวิธีของ visual odometry สามารถอธิบายเป็นลำดับขั้นตอนโดยสังเขปได้ดังนี้

1) รับข้อมูลภาพจากอุปกรณ์รับรู้ ไม่ว่าจะเป็น กล้องวิดีโอ กล้องสเตอริโอ หรือกล้องวิดีโอแบบออมนิ (Omnidirectional camera) จากนั้นทำการปรับแก้ภาพด้วยพารามิเตอร์ภายในของกล้องที่ทราบค่าอยู่แล้ว
2) ตรวจหาความสอดคล้องของ features ในภาพปัจจุบันและภาพก่อนหน้าด้วยวิธี optical flow และทำการตรวจสอบหาความสอดคล้องที่มีความผิดพลาดสูงซึ่งถือว่าเป็นข้อมูลที่ผิดปกติ (Outliers) ที่ต้องทำการลบออก
3) ประมาณการเคลื่อนที่ของกล้องจาก optical flow ที่หามาได้นี้ ด้วยการหาการแปลงใน 3 มิติระหว่างตำแหน่งกล้องในปัจจุบันเทียบกับตำแหน่งกล้องก่อนหน้า ที่ทำให้ฟังก์ชันความผิดพลาดของการฉายภาพ feature จากภาพในปัจจุบันไปยังภาพก่อนหน้ามีค่าน้อยที่สุดด้วยวิธีการหาค่าเหมาะสมที่สุด (Optimization) หรือวิธีการสุ่มตัวอย่าง (Random sampling)
4) ทำการเพิ่มจุด features ที่ใช้ในการหาความสอดคล้องในขั้นตอนการทำ optical flow เป็นระยะ เพื่อให้จุดครอบคลุมทั่วทั้งภาพทำให้สามารถประมาณการเคลื่อนที่ได้อย่างแม่นยำ

โดยในวิทยานิพนธ์นี้จะเลือกใช้ Lucas–Kanade optical flow [53] สำหรับตรวจหาความสอดคล้องของ features ระหว่างภาพสองภาพ สำหรับประมาณการเคลื่อนที่ของกล้องนั้นจะเลือกใช้ทั้งแบบหาค่าเหมาะสมที่สุดโดยจะใช้วิธี Levenberg-Marquardt และแบบสุ่มตัวอย่างโดยใช้วิธี Random sample consensus (RANSAC) โดยจะอธิบายหลักการทำงานของทั้งสามกระบวนการดังนี้

2.3.1. Lucas–Kanade optical flow

ในระหว่างที่กล้องหรือวัตถุในภาพเคลื่อนที่ Optical flow จะเป็นวิธีการหาความสัมพันธ์การเคลื่อนที่ของวัตถุหรือ feature ระหว่างสองภาพที่ต่อเนื่องกัน โดยผลลัพธ์ที่ได้จะเป็นเวกเตอร์บอกการเคลื่อนที่ของจุดในภาพก่อนหน้ามายังภาพปัจจุบัน ดังรูปที่ รูปที่ 2.7

รูปที่ 2.7 ผลลัพธ์การทำงานของ optical flow ซ้าย) ภาพก่อนหน้า ขวา) ภาพปัจจุบันที่แสดงเวกเตอร์ผลลัพธ์ของ optical flow

สำหรับ Lucas–Kanade optical flow จะมีสมมติฐานอยู่ 2 ประการคือ ความเข้ม (Intensity) ของพิกเซลใดๆ ในภาพจะไม่เปลี่ยนแปลงในภาพถัดไป กล่าวคือสภาพแสงระหว่างสองภาพที่ติดกันจะไม่มีการเปลี่ยนแปลง และพิกเซลเพื่อนบ้านของพิกเซลที่กำลังคำนวณหา optical flow นั้นจะถือว่ามีการเคลื่อนที่ที่เหมือนกัน เมื่อพิจารณาพิกเซล (x, y) ที่เวลา t จะมีความเข้มเป็น $I(x, y, t)$ และพิกเซลเดียวกันนี้ในภาพถัดไปภายหลังการเคลื่อนที่ด้วยเวลา Δt ไปเป็นระยะทาง $(\Delta x, \Delta y)$ จะมีความเข้มเท่าเดิมดังสมการ

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

จากนั้นจึงประมาณค่าฝั่งขวาของสมการด้วยอนุกรม Taylor (Taylor series) ได้เป็น

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \varepsilon$$

$$I(x + \Delta x, y + \Delta y, t + \Delta t) \approx I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t$$

จากสมการทั้งหมดนี้จะได้ว่า

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$

หารด้วย Δt ทั้งสมการได้เป็นสมการ

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$$

โดยที่ V_x และ V_y คือความเร็วของพิกเซลในแกน x และ y ตามลำดับ สำหรับ $\frac{\partial I}{\partial x}$ และ $\frac{\partial I}{\partial y}$ คือ gradient ของภาพและ $\frac{\partial I}{\partial t}$ คือ gradient ของเวลา โดยเขียนใหม่ให้เป็น I_x, I_y และ I_t ตามลำดับ จึงเขียนสมการใหม่ได้เป็น

$$I_x V_x + I_y V_y = -I_t$$

ซึ่งต้องทำการแก้สมการหา V_x และ V_y ซึ่งคือ optical flow ของพิกเซล $I(x, y, t)$ สำหรับวิธีการแก้สมการนั้นเราจะใช้วิธีของ Lucas–Kanade

กำหนดให้ p คือจุดที่กำลังหา optical flow เมื่อทำการสร้างหน้าต่างโดยมี p เป็นจุดศูนย์กลาง จุดทั้งหมดที่อยู่ในหน้าต่างนี้กำหนดให้เป็นจุด $q_1, \dots, q_n$ จากสมมติฐานที่กล่าวว่าพิกเซลเพื่อนบ้านของพิกเซลที่กำลังคำนวณหา optical flow นั้นจะถือว่ามีการเคลื่อนที่ที่เหมือนกัน กล่าวคือ V_x และ V_y ของจุด $q_1, \dots, q_n$ จะมีค่าเหมือนกันทั้งหมด เขียนเป็นสมการได้ดังนี้

$$I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1)$$
$$\vdots$$
$$I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)$$

ซึ่งสมการเหล่านี้สามารถเขียนให้อยู่ในรูปของ $Av = b$ ได้โดยที่

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix}, v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \text{ และ } b = \begin{bmatrix} -I_t(q_1) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

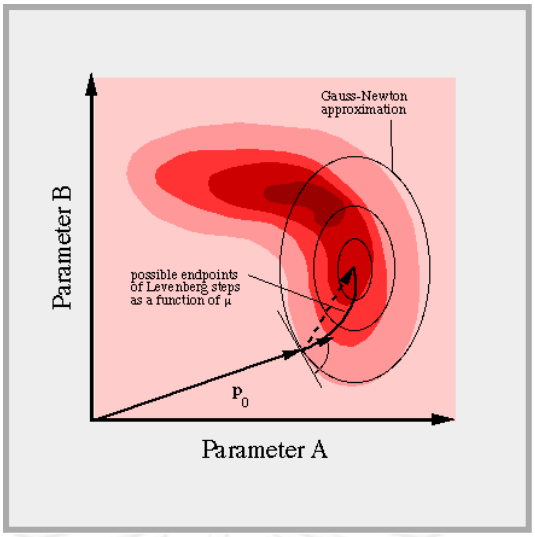
โดยวิธีการของ Lucas–Kanade นี้จะแก้สมการนี้ด้วยหลักการของวิธี least square กล่าวคือ $A^T Av = A^T b$ หรือนั่นคือ $v = (A^T A)^{-1} A^T b$ ซึ่งสุดท้ายแล้วจะได้สมการดังต่อไปนี้

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_y(q_i) I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix}$$

จากนั้นแก้สมการเพื่อหา V_x และ V_y ซึ่งคือ optical flow ของจุด p ที่ต้องการ

2.3.2. Levenberg-Marquardt method

วิธี Levenberg-Marquardt เป็นกระบวนการที่ใช้ในการการหาจุดต่ำที่สุดสำหรับปัญหา non-linear least squares โดยมีแนวคิดผสมผสานกันระหว่างวิธี Gauss-Newton และวิธี gradient descent ดังรูปที่ 2.8 [54] ซึ่งจะมีความทนทานมากกว่าวิธี Gauss-Newton ที่สามารถหาคำตอบได้แม้จุดเริ่มต้นจะอยู่ไกลจากคำตอบมาก แต่อย่างไรก็ตามวิธี Levenberg-Marquardt ยังคงให้ผลลัพธ์เป็นจุดต่ำสุดแบบสัมพัทธ์ การเลือกจุดเริ่มต้นที่เหมาะสมจึงเป็นสิ่งสำคัญ



รูปที่ 2.8 แนวคิดของวิธี Levenberg-Marquardt

วิธี Levenberg-Marquardt จะถูกนำไปใช้ในการแก้ไขปัญหา least squares โดยข้อมูลนำเข้าจะให้เป็นเซตของข้อมูลมาจำนวน m คู่กำหนดให้คือ (x_i, y_i) มีจุดมุ่งหมายเพื่อหาค่าพารามิเตอร์ β ที่ทำให้ฟังก์ชันต่อไปนี้มีค่าน้อยที่สุด

$$S(\beta) = \sum_{i=1}^{m} [y_i - f(x_i, \beta)]^2$$

โดยที่ $f(x_i, \beta)$ คือฟังก์ชันแบบจำลองที่เราใช้ในการแสดงความสัมพันธ์ระหว่าง x_i และ y_i สำหรับวิธีการนั้นจะใช้ลักษณะของวิธีทำซ้ำ (Iterative) โดยเราจะทำการเปลี่ยนค่าของ β ในขั้นตอนถัดไปเป็น $\beta + \delta$ โดยในการหาค่า δ จะเริ่มโดยการประมาณค่าฟังก์ชัน $f(x_i, \beta + \delta)$ ดังนี้

$$f(x_i, \beta + \delta) \approx f(x_i, \beta) + J_i \delta$$

โดยที่ $J_i = \frac{\partial f(x_i, \beta)}{\partial \beta}$ และเมื่อนำสมการนี้รวมกับสมการของ $S(\beta + \delta)$ จะได้เป็น

$$S(\beta + \delta) \approx \sum_{i=1}^{m} [y_i - f(x_i, \beta) - J_i \delta]^2$$

เขียนให้อยู่ในรูปของเวกเตอร์ได้เป็น

$$S(\beta + \delta) \approx \|\boldsymbol{y} - \boldsymbol{f}(\beta) - \boldsymbol{J}\delta\|^2$$

โดยที่ $\boldsymbol{J}$ คือเมทริกซ์ Jacobian ที่มีแถวที่ i คือ J_i สำหรับ $\boldsymbol{f}$ และ $\boldsymbol{y}$ คือเวกเตอร์ที่มีส่วนประกอบที่ i คือ $f(x_i, \beta)$ และ y_i ตามลำดับ ซึ่งสำหรับที่จุดต่ำสุดของ $S(\beta)$ จะมี gradient ของ S เทียบกับ δ มีค่าเท่ากับศูนย์ จากสมการข้างต้นทำการหาอนุพันธ์เทียบกับ δ ซึ่ง $\frac{\partial S(\beta+\delta)}{\partial \delta}$ จะมีค่าเท่ากับศูนย์และทำการปรับแก้สมการใหม่ได้เป็น

$$(\boldsymbol{J}^T\boldsymbol{J})\delta = \boldsymbol{J}^T[\boldsymbol{y} - \boldsymbol{f}(\beta)]$$

จากสมการนี้จะทำให้เราหาค่า δ ได้ ซึ่งจะใช้ค่านี้ในการเพิ่มค่าพารามิเตอร์ β เพื่อทำให้ฟังก์ชัน S ลู่เข้าสู่จุดต่ำสุด โดยจะทำซ้ำจนกระทั่งมีค่า δ น้อยกว่าค่าที่กำหนด และวิธี Levenberg-Marquardt สามารถใช้ตัวประกอบการหน่วงมาใช้ร่วมได้ดังนี้

$$(\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{I})\delta = \boldsymbol{J}^T[\boldsymbol{y} - \boldsymbol{f}(\beta)]$$

โดยที่ λ คือตัวประกอบการหน่วง และ $\boldsymbol{I}$ คือเมทริกซ์เอกลักษณ์ สำหรับ λ จะมีการเปลี่ยนค่า**ใน**ทุกรอบการทำงาน โดย λ จะมีค่าน้อยเมื่อฟังก์ชัน S มีค่าลดลงอย่างรวดเร็ว ซึ่งทำให้มีลักษณะการทำงานใกล้เคียงกับวิธี Gauss-Newton และค่า λ จะมีค่ามากเมื่อฟังก์ชัน S มีค่าลดลงช้ากว่าค่าที่กำหนดไว้ จะทำให้มีลักษณะการทำงานใกล้เคียงกับวิธี gradient descent

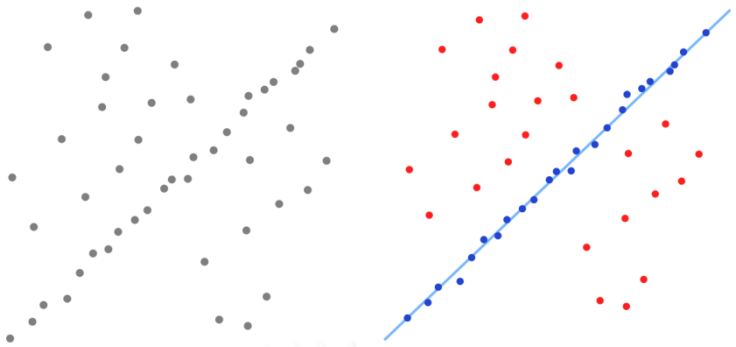
สำหรับวิทยานิพนธ์เล่มนี้จะใช้วิธี Levenberg-Marquardt ในการประมาณการเคลื่อนที่ของกล้องจากการหาความสัมพันธ์ของข้อมูลจุด feature ระหว่างสองภาพ โดยกำหนดให้จุด feature ในภาพแรกแทนด้วยเซต $U = \{u_1, \dots, u_k\}$ และจุดในภาพที่สองซึ่งคือจุดที่ได้จากการหา optical flow ของจุดในเซต U กำหนดให้เป็นเซต $U' = \{u'_1, \dots, u'_k\}$ สำหรับทุกจุด U ในภาพแรกนั้นถูกกำหนดให้ทราบตำแหน่งของจุดเหล่านี้ในสามมิติอยู่แล้ว กำหนดด้วยเซต $P = \{p_1, \dots, p_k\}$ โดย p_i เป็นจุดในสามมิติเทียบกับกรอบอ้างอิงของกล้องของภาพแรก เราจะทำการหาเมทริกซ์การแปลงระหว่างตำแหน่งกล้องของภาพแรกเทียบกับตำแหน่งกล้องของภาพที่สอง กำหนดให้เป็น T^* ที่ทำให้ฟังก์ชันต่อไปนี้มีค่าต่ำที่สุด

$$S(T^*) = \sum_{i=1}^{k} |u'_i - proj((T^*) \cdot p_i)|^2$$

โดยที่ $proj(p_i, T^*)$ คือฟังก์ชันการฉายภาพ และฟังก์ชัน S คือผลรวมกำลังสองของความผิดพลาดในการฉายภาพ

2.3.3. Random sample consensus (RANSAC)

Random sample consensus หรือ RANSAC เป็นกระบวนการการทำซ้ำสำหรับประมาณค่าพารามิเตอร์ของแบบจำลองทางคณิตศาสตร์ จากข้อมูลที่เก็บค่ามาได้ที่มีข้อมูลที่ผิดพลาด (Outlier) รวมอยู่ด้วย โดยมีแนวคิดหลักคือประมาณค่าพารามิเตอร์จากชุดข้อมูลที่ได้จากการสุ่ม โดยจะทำซ้ำจนกว่าพารามิเตอร์ที่ได้จะสามารถเข้ากันได้กับข้อมูลโดยส่วนใหญ่ จึงทำให้วิธีการนี้ทนทานต่อข้อมูลที่ปะปนไปด้วยข้อมูลที่ผิดปกติ ตัวอย่างเช่น รูปที่ 2.9 ซ้าย คือชุดข้อมูลที่ต้องการหาพารามิเตอร์ของเส้นตรงที่เหมาะสมกับข้อมูลนี้ที่สุด และรูปที่ 2.9 ขวา คือผลลัพธ์ของเส้นตรงที่เหมาะสมกับข้อมูลนี้ ซึ่งจะสังเกตเห็นว่าข้อมูลที่ผิดปกติ (สีแดง) จะไม่ส่งผลต่อผลลัพธ์เส้นตรงที่ได้



รูปที่ 2.9 ตัวอย่างผลลัพธ์ของวิธี RANSAC ซ้าย) ข้อมูลนำเข้า ขวา) ผลลัพธ์เส้นตรงที่เหมาะสมกับข้อมูลชุดนี้มากที่สุด

โดยวิธี RANSAC จะมีข้อกำหนดต่างๆ ดังนี้

1) พารามิเตอร์ที่ต้องการหาจะต้องสามารถประมาณได้ด้วยข้อมูลจำนวน N ข้อมูล
2) มีข้อมูลทั้งหมดจำนวน M ข้อมูล
3) p_g คือความน่าจะเป็นที่ข้อมูลที่ถูกเลือกเป็นส่วนหนึ่งของแบบจำลองที่ดี
4) p_{fail} คือความน่าจะเป็นที่กระบวนการนี้จะหยุดการทำงานโดยไม่สามารถหาแบบจำลองที่ดีได้เลย

โดยที่ p_g และ p_{fail} คือค่าความน่าจะเป็นที่จะกำหนดเงื่อนไขในการทำงานของวิธีการนี้ ซึ่งต้องมีการกำหนดค่าก่อนเริ่มต้นการทำงาน สำหรับขั้นตอนการทำงานของวิธี RANSAC สามารถสรุปได้ดังนี้

1) เลือกข้อมูลแบบสุ่มจำนวน N ข้อมูล
2) หาค่าพารามิเตอร์ที่ต้องการจากข้อมูลที่สุ่มมาจำนวน N ข้อมูล โดยกำหนดให้พารามิเตอร์นี้แทนด้วยเวกเตอร์ $\boldsymbol{x}$
3) กำหนดให้ K คือจำนวนข้อมูลที่มีความผิดพลาดเมื่อเทียบกับพารามิเตอร์ $\boldsymbol{x}$ เกินค่า t ที่กำหนดไว้ จากข้อมูลทั้งหมด (จำนวน M ข้อมูล)
4) ถ้า K มีจำนวนมากพอ นั่นคือ $K \geq K_{accept}$ จะสิ้นสุดการทำงานและคืนค่าผลลัพธ์เป็น $\boldsymbol{x}$
5) ทำซ้ำข้อ 1 ถึง 4 จำนวน L ครั้ง
6) ถือว่าวิธีการนี้ล้มเหลวถ้ามาถึงขั้นตอนนี้

สำหรับค่า t และ K_{accept} จะกำหนดโดยผู้ใช้ โดยที่ t จะแสดงถึงความผิดพลาดสูงสุดที่ยอมรับให้เข้ากับพารามิเตอร์ของแบบจำลองได้ สำหรับ K_{accept} นิยมกำหนดเป็นร้อยละเทียบกับ M เพื่อแสดงถึงร้อยละของจำนวนข้อมูลที่ไม่ผิดปกติที่คาดเดาว่ามีอยู่จากข้อมูลทั้งหมด และ L สามารถ

คำนวณค่าได้จากความน่าจะเป็น p_g และ p_{fail} โดยที่ p_{fail} คือความน่าจะเป็นที่ทำซ้ำทั้งหมด L ครั้งแล้วล้มเหลว ซึ่งมีค่าดังนี้

$$p_{fail} = \left(1 - \left(p_g\right)^N\right)^L$$

จากสมการข้างต้นจะสามารถคำนวณค่า L ได้เป็น

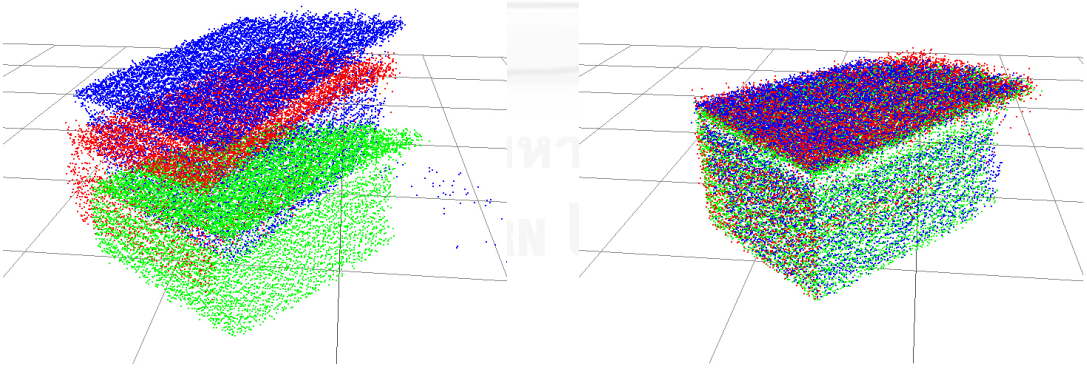
$$L = \frac{\log\left(p_{fail}\right)}{\log\left(1 - \left(p_g\right)^N\right)}$$

ในการประมาณการเคลื่อนที่ของกล้องจากการหาความสัมพันธ์ของข้อมูลจุด feature ระหว่างสองภาพ จะนำวิธี RANSAC มาใช้เนื่องด้วยข้อมูลจุดเหล่านี้มีโอกาสที่จะมีข้อมูลที่ผิดปกติรวมอยู่ด้วย การใช้วิธี RANSAC จึงสามารถช่วยให้ผลลัพธ์ที่ได้มีความทนทานต่อข้อมูลที่ผิดปกติเหล่านี้ ซึ่งจะใช้ร่วมกับวิธี Levenberg-Marquardt ที่กล่าวไว้ในบทที่ 2.3.2 โดยจะใช้ขั้นตอนการทำงานของ RANSAC เป็นหลัก ซึ่งในขั้นตอนการหาค่าพารามิเตอร์นั้นจะเป็นขั้นตอนการประมาณค่าเมทริกซ์การแปลง T^* ด้วยวิธี Levenberg-Marquardt และมีฟังก์ชันความผิดพลาดของจุด u_i' เทียบกับพารามิเตอร์ T^* ที่หาได้เป็น

$$E(T^*) = |u_i' - proj((T^*) \cdot p_i)|^2$$

บทที่ 3
การปรับแก้พารามิเตอร์ของอุปกรณ์รับรู้และแขนหุ่นยนต์

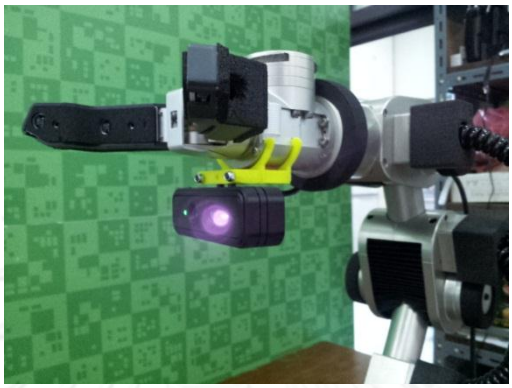
สำหรับการนำอุปกรณ์รับรู้มาติดตั้งที่ปลายแขนหุ่นยนต์ หรือที่เรียกว่าแนวทางตาในมือ (Eye-in-hand approach) เมื่อต้องการหาตำแหน่งของข้อมูลที่ได้จากอุปกรณ์รับรู้ เทียบกับกรอบอ้างอิงของแขนหุ่นยนต์ในปริภูมิสามมิตินั้น จำเป็นที่จะต้องรู้ความสัมพันธ์ของตำแหน่งระหว่างอุปกรณ์รับรู้กับแขนหุ่นยนต์ หรือที่รู้จักกันในชื่อพารามิเตอร์ภายนอก (Extrinsic parameter) อีกทั้งอุปกรณ์รับรู้ในบางชนิดจะมีพารามิเตอร์ภายใน (Intrinsic parameter) ที่ต้องจำเป็นต้องรู้ เพื่อความถูกต้องในการหาความสัมพันธ์เชิงตำแหน่งของข้อมูลเทียบกับแขนหุ่นยนต์ ตัวอย่างเช่น ความยาวโฟกัสของกล้อง (Focal length) และความบิดงอของภาพ (Distortion) เป็นต้น ซึ่งในโลกจริงนั้นเราไม่สามารถรู้พารามิเตอร์เหล่านี้ได้โดยตรง กล่าวคือเราจำเป็นต้องหาวิธีการประมาณค่าและปรับแก้พารามิเตอร์เหล่านี้ โดยวิธีการที่ดีจะส่งผลโดยตรงต่อความถูกต้องของตำแหน่งข้อมูล ดังรูปที่ 3.1 (ซ้าย) แสดงให้เห็นถึงผลลัพธ์การสแกนวัตถุที่ได้จากกล้องความลึก (Depth camera) แสดงในรูปแบบของ point cloud ในปริภูมิ 3 มิติ ที่ทำการวัดค่าพารามิเตอร์ด้วยมือ โดยแต่ละสีคือการเก็บข้อมูลแต่ละครั้ง ในตำแหน่งของกล้องที่แตกต่างกัน ซึ่งผลลัพธ์ที่แสดงให้เห็นถึงความไม่ถูกต้องของตำแหน่งของข้อมูล ที่มีการบิดงอและเหลื่อมซ้อนทับกันอย่างไม่เป็นระเบียบ เปรียบเทียบกับรูปที่ 3.1 (ขวา) คือผลลัพธ์การสแกนวัตถุ ภายหลังการปรับแก้พารามิเตอร์ ที่แสดงให้เห็นถึงความถูกต้องของตำแหน่งข้อมูลที่มากกว่า



รูปที่ 3.1 ผลลัพธ์การสแกนวัตถุที่ได้จากกล้องความลึก ซ้าย) ก่อนการปรับแก้พารามิเตอร์ ขวา) หลังการปรับแก้พารามิเตอร์

ในด้านการจับวัตถุของแขนหุ่นยนต์ ความผิดพลาดของข้อมูลเหล่านี้ จะส่งผลโดยตรงในการระบุตำแหน่งจุดจับ กล่าวคือเมื่อข้อมูลของวัตถุที่ได้รับจากอุปกรณ์รับรู้ผิดพลาด ตำแหน่งจุดจับที่คำนวณได้ก็จะผิดพลาด ไม่สามารถนำนิ้วหุ่นยนต์ไปวางยังตำแหน่งที่ถูกต้องได้ การจับวัตถุจึงล้มเหลว

ในบทนี้จึงจะนำเสนอวิธีการปรับแก้พารามิเตอร์ทั้งภายในและภายนอกของอุปกรณ์รับรู้ซึ่งคือ DepthSense 325 ที่ติดตั้งที่ปลายแขนหุ่นยนต์ Katana 6M180 (รูปที่ 3.2) เพื่อลดความผิดพลาดของข้อมูลเหล่านี้ โดยจะอธิบายกระบวนการในแต่ละขั้นตอนของการปรับแก้พารามิเตอร์ รวมไปถึงผลการทดสอบการปรับแก้พารามิเตอร์ ที่เปรียบเทียบให้เห็นถึงความถูกต้องของตำแหน่งข้อมูล ก่อนและหลังการปรับแก้พารามิเตอร์ด้วยวิธีการที่เรานำเสนอ



รูปที่ 3.2 การติดตั้งกล้อง DepthSense 325 บนปลายแขนหุ่นยนต์ Katana 6M180

3.1. อุปกรณ์ในระบบ

3.1.1. SoftKinetic DepthSense 325

DepthSense รุ่น DS325 [48] (รูปที่ 3.3) ของบริษัท SoftKinetic เป็นอุปกรณ์รับรู้ที่ประกอบไปด้วยกล้องสี (RGB camera) และกล้องความลึกแบบไทม์ออฟไฟลท์ (Time-of-flight camera) ซึ่งมีขนาดเล็กเพียง 10.5 x 3 x 2.3 ลูกบาศก์เซนติเมตร สามารถวัดความลึกได้ใกล้สุด 15 เซนติเมตร และมาพร้อมกับกล้องสีที่มีความละเอียดสูง 1280 x 720 พิกเซล (High-definition) จึงเหมาะแก่การนำมาใช้ในงานที่ใช้แนวทางตาในมือ กล่าวคือสามารถนำ DepthSense 325 มาติดตั้งที่ปลายแขนหุ่นยนต์ได้โดยไม่รบกวนหรือกีดขวางการทำงานของแขนหุ่นยนต์ สามารถรับข้อมูลความลึกได้ในขณะที่แขนอยู่ใกล้วัตถุมาก และได้ข้อมูลสีที่มีความละเอียดสูง ซึ่งข้อมูลทั้งหมดนี้ถูกส่งมาให้แบบทันที (Real-time) โดยคุณสมบัติอื่นๆที่น่าสนใจ แสดงให้เห็นดังตารางที่ 3.1

รูปที่ 3.3 อุปกรณ์ DepthSense 325

ตารางที่ 3.1 คุณสมบัติของกล้อง DepthSense 325

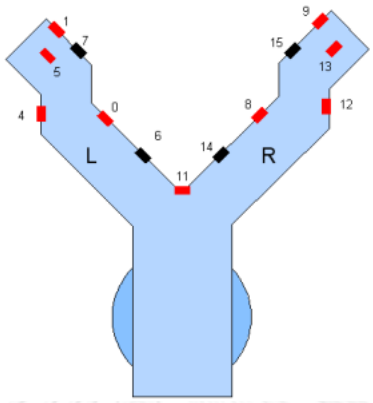
คุณสมบัติ		DepthSense 325
ทั่วไป	ขนาด (กว้าง x สูง x ลึก)	10.5 x 3 x 2.3 ลูกบาศก์เซนติเมตร
	กำลังไฟฟ้า	น้อยกว่า 2.2 วัตต์
	อื่นๆ	การเชื่อมต่อ USB, ไมโครโฟนคู่ และมาตรวัดความเร่งแบบ 3 แกน
กล้องวัดความลึก	ความละเอียด	320 x 240 พิกเซล
	ความกว้างของมุมมอง (แนวนอน x แนวตั้ง x แนวลึก)	74 x 58 x 87 องศา
	ความถี่การรับภาพ	25 – 30 ภาพต่อวินาที
กล้องสี	ความละเอียด	1280 x 720 พิกเซล
	ความกว้างของมุมมอง (แนวนอน x แนวตั้ง x แนวลึก)	63.2 x 49.3 x 75.2 องศา

3.1.2. Neuronics Katana 6M180

รูปที่ 3.4 ซ้าย) แขนหุ่นยนต์ Katana 6M180 ขวา) ปริภูมิการทำงานของแขนหุ่นยนต์ Katana

Katana [55] เป็นแขนหุ่นยนต์ที่ผลิตโดยบริษัท Neuronics จากประเทศสวิตเซอร์แลนด์ ซึ่งได้มีการพัฒนาและผลิตออกมาหลากหลายรุ่น โดยในงานวิจัยนี้จะใช้แขนหุ่นยนต์รุ่น Katana 6M180 (รูปที่ 3.4 ซ้าย) ซึ่งเป็นแขนหุ่นยนต์ที่มีน้ำหนักเบา 4.1 กิโลกรัม สามารถเคลื่อนที่ได้ 6 องศาอิสระ (Degrees of freedom) ขับเคลื่อนโดยมอเตอร์กระแสตรงจำนวน 6 ตัว ที่มีความผิดพลาดในเชิงตำแหน่งน้อยกว่า 0.1 มิลลิเมตร รัศมีการทำงาน 60 เซนติเมตร (รูปที่ 3.4 ขวา) มาพร้อมกับมือจับ

(Gripper) ติดตั้งในทิศทาง 180 องศาเทียบปลายแขนหุ่นยนต์ มีอุปกรณ์รับรู้ระยะแบบอินฟราเรด (Infra-red sensor) และอุปกรณ์รับรู้แรง (Force sensor) ดังรูปที่ 3.5 โดยมีคุณสมบัติอื่นๆแสดงดังตารางที่ 3.2



รูปที่ 3.5 อุปกรณ์รับรู้บนมือจับ สีแดง) อุปกรณ์รับรู้อินฟราเรด สีดำ) อุปกรณ์รับรู้แรง

ตารางที่ 3.2 คุณสมบัติของแขนหุ่นยนต์ Katana 6M180

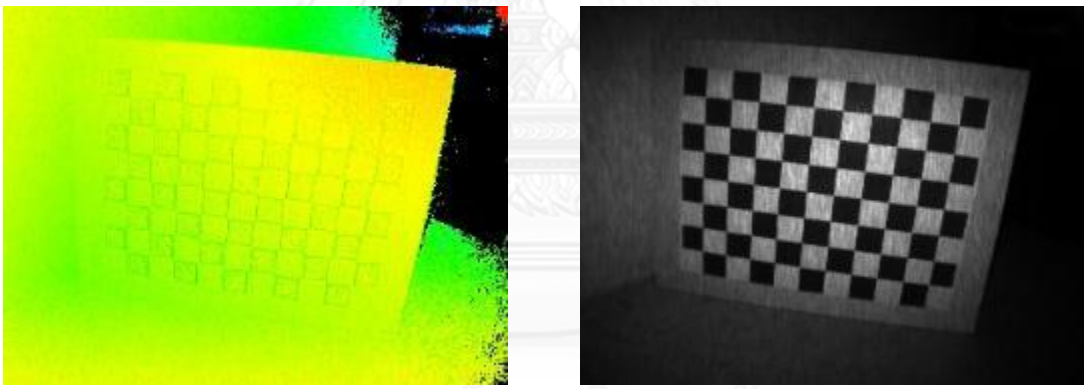
คุณสมบัติ	Katana 6M180
มอเตอร์	มอเตอร์กระแสตรงพร้อมด้วยมาตรวัดตำแหน่งแบบดิจิตอล
ความแม่นยำ	ผิดพลาดไม่เกิน 0.1 มิลลิเมตร
ความสูงฐาน	188 มิลลิเมตร
องศาอิสระ	5 หรือ 6 (ตามจำนวนมอเตอร์)
รัศมีการทำงาน	60 เซนติเมตร
วัสดุ	Anodized Aluminum
น้ำหนัก	4.1 กิโลกรัม
น้ำหนักบรรทุก	500 กรัม
พลังงาน	12 โวลต์ 3.5 แอมแปร์
อัตราเร็ว	เฉลี่ย 90 องศาต่อวินาที

3.2. การปรับแก้พารามิเตอร์ของกล้อง DepthSense 325

เริ่มต้นโดยกำหนดให้แบบจำลองของกล้องในงานวิจัยนี้เป็นแบบกล้องรูเข็ม (Pinhole camera model) ทั้งหมด ให้ $\{C\}$ แทนกรอบอ้างอิงของกล้อง DepthSense โดยจุดกำเนิดอยู่ที่กล้องความลึก มีแกน z พุ่งออกจากกล้อง หรือนั่นคือกรอบอ้างอิงของกล้องความลึกนั่นเอง สำหรับกรอบอ้างอิงของกล้องสีแทนด้วย $\{B\}$

3.2.1. การปรับแก้พารามิเตอร์ภายใน

สำหรับกล้องความลึก หรือกล้องไทม์ออฟไฟลท์ของกล้อง DepthSense จะให้ข้อมูลเป็นภาพความลึก (Depth image) (รูปที่ 3.6 ซ้าย) นอกจากนั้นกล้องความลึกนี้ยังให้ข้อมูลภาพระดับความเชื่อมั่น (Confidence level image) (รูปที่ 3.6 ขวา) โดยจะสังเกตได้ว่าภาพนี้สามารถเห็นลวดลายของกระดานหมากรุกได้อย่างชัดเจน เนื่องมาจากสีดำและสีขาวของกระดานหมากรุกนั้นสะท้อนแสงอินฟราเรดจากกล้อง DepthSense ได้แตกต่างกัน ทำให้ภาพระดับความเชื่อมั่นที่ได้แสดงผลลัพธ์ของความแตกต่างนี้อย่างเด่นชัด

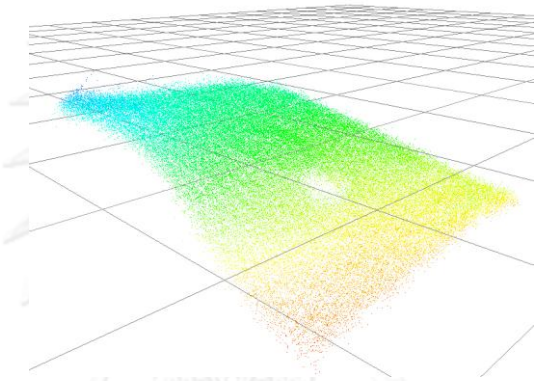


รูปที่ 3.6 ข้อมูลจากกล้องความลึกของ DepthSense 325 ซ้าย) ภาพความลึก ขวา) ภาพระดับความเชื่อมั่น

จากข้อสังเกตเหล่านี้ เราจึงนำภาพระดับความเชื่อมั่นนี้มาใช้ประโยชน์ ในการปรับแก้พารามิเตอร์ภายในของกล้องความลึก โดยการใช้กระดานหมากรุกที่รู้ขนาดและระยะห่างของแต่ละช่องแล้ว มาทำการปรับแก้พารามิเตอร์ภายในด้วยวิธีมาตรฐานโดยใช้ OpenCV [56] ซึ่งจะทำการปรับแก้ทั้ง ความยาวโฟกัส จุดกึ่งกลางภาพ และความบิดงอของภาพ สำหรับกล้องสีนั้นจะใช้วิธีการเดียวกัน แต่จะใช้ข้อมูลภาพสีที่ได้จากกล้องสีแทน

3.2.2. การปรับแก้ค่าความลึก

จากรูปที่ 3.7 คือข้อมูล point cloud จากกล้องความลึกในปริภูมิ 3 มิติของพื้นเรียบที่ได้จากค่าเฉลี่ยของภาพความลึกจำนวน 10 ภาพ โดยสีของแต่ละจุดแสดงถึงความสูงที่แตกต่างกัน ซึ่งแสดงให้เห็นถึงความผิดปกติของข้อมูลที่อ่านค่าได้จากกล้อง ลักษณะที่ผิดปกตินี้ยังคงมีลักษณะเดิมแม้จะเปลี่ยนตำแหน่งหรือทิศทางของกล้อง จึงทำให้เชื่อได้ว่าเราต้องปรับแก้ค่าความลึกในแต่ละพิกเซล โดยเป้าหมายคือหาฟังก์ชันความสัมพันธ์ระหว่าง ระยะทางจากวัตถุถึงกล้องที่อ่านค่าได้กับระยะทางจากวัตถุถึงกล้องที่แท้จริง เพื่อนำฟังก์ชันนี้มาใช้ปรับแก้ค่าความลึกของแต่ละพิกเซลให้มีความถูกต้องมากขึ้น

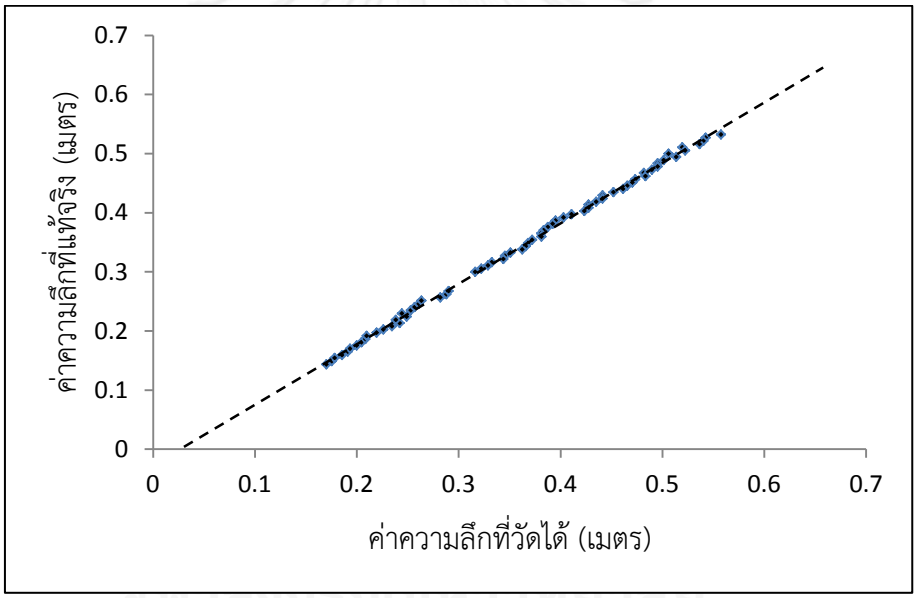


รูปที่ 3.7 ข้อมูล point cloud จากกล้องความลึกในปริภูมิ 3 มิติของพื้นเรียบ

ซึ่งเราสมมติว่าฟังก์ชันนี้มีลักษณะเป็นเส้นตรง เริ่มต้นโดยนำกระดานหมากรุกไปวางไว้บนพื้นผิวที่เรียบดังรูป 3.8 จากนั้นจึงเก็บข้อมูลภาพความลึกและภาพระดับความเชื่อมั่น จากลวดลายของกระดานหมากรุกจะทำให้สามารถหา $\{C\}$ เทียบกับกระดานหมากรุกได้ด้วย OpenCV [56] สำหรับแต่ละพิกเซล p_i ของภาพความลึก โดยการใช้พารามิเตอร์ภายในที่หาได้ในขั้นตอนก่อนหน้าจะได้ $\boldsymbol{p_c}$ คือตำแหน่งของพิกเซล p_i บนระนาบของภาพ (Image plane) เทียบกับ $\{C\}$ จากนั้นทำการลากเส้นตรงจากจุดกำเนิดของ $\{C\}$ ผ่านจุด $\boldsymbol{p_c}$ เส้นตรงนี้จะตัดกับระนาบของกระดานหมากรุก ซึ่งเรารู้ตำแหน่งของ $\{C\}$ เทียบกับกระดานหมากรุก ทำให้สามารถคำนวณระยะทางจากจุดกำเนิดของ $\{C\}$ ถึงจุดที่ตัดกับระนาบของกระดานหมากรุกได้ โดยระยะทางนี้คือระยะทางที่แท้จริงของพิกเซล p_i โดยนำมาจับคู่กับค่าที่อ่านได้จากกล้องความลึกของพิกเซลเดียวกันนี้เป็นหนึ่งคู่ข้อมูล

ทำการย้ายตำแหน่งของกล้องไปยัง 10 ตำแหน่งที่แตกต่างกัน โดยแต่ละตำแหน่งจะทำการเก็บข้อมูลจำนวน 10 ครั้ง ซึ่งรวมแล้วจะได้ข้อมูลระหว่าง ระยะที่แท้จริงกับระยะที่อ่านค่าได้จำนวน 100 คู่ข้อมูลต่อหนึ่งพิกเซล p_i โดยนำข้อมูลเหล่านี้มาสร้างกราฟและหาความสัมพันธ์เชิงเส้นโดยใช้ linear least square จะได้ความสัมพันธ์ดังรูปที่ 3.9 โดยจะทำขั้นตอนข้างต้นเหล่านี้ซ้ำกับทุกพิกเซลของภาพความลึก

รูปที่ 3.8 การติดตั้งอุปกรณ์สำหรับการปรับแก้ค่าความลึก



รูปที่ 3.9 กราฟแสดงความสัมพันธ์ระหว่างค่าความลึกที่วัดได้กับค่าความลึกที่แท้จริง

กำหนดให้ $D_i(x) = a_i x + b_i$ เป็นฟังก์ชันเส้นตรงที่แสดงความสัมพันธ์ระหว่างค่าความลึกที่วัดได้กับค่าความลึกที่แท้จริงของพิกเซลที่ i ซึ่งจากการสังเกตนั้นจะพบว่าในพิกเซลส่วนใหญ่นั้นจะมีค่า a_i ใกล้เคียง 1 และมีจุดตัดแกนที่ไม่ผ่านจุดกำเนิด นั่นคือมีค่า b_i ที่ไม่ใกล้เคียง 0 แสดงให้เห็นโดยค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐานของ a_i และ b_i ในตารางที่ 3.3 ซึ่งเป็นเครื่องพิสูจน์ได้ว่าความผิดพลาดของกล้องความลึกนี้ สามารถแก้ไขได้ด้วยการบวกค่า offset เพิ่มไปในแต่ละพิกเซล โดยมีงานวิจัย [32, 57] ที่ใช้แนวทางการปรับแก้ค่าความลึกแบบนี้เช่นเดียวกัน

ตารางที่ 3.3 ค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐานของสัมประสิทธิ์ของฟังก์ชันปรับแก้ค่าความลึก

สัมประสิทธิ์	ค่าเฉลี่ย	ค่าเบี่ยงเบนมาตรฐาน	ค่าต่ำสุด	ค่าสูงสุด
a_i	1.045	0.022	0.920	1.145
b_i	-0.004	0.016	-0.031	0.028

3.2.3. การปรับแก้พารามิเตอร์ภายนอก

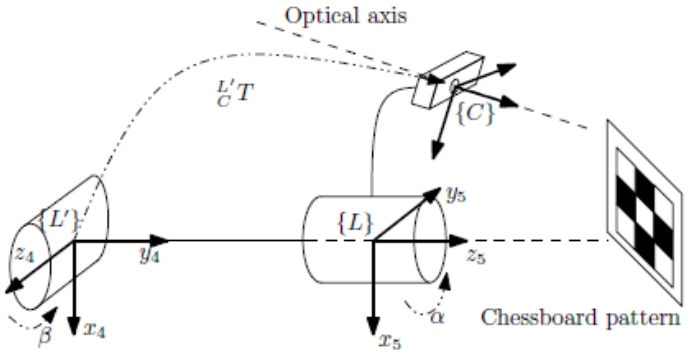
ในส่วนนี้จะทำการประมาณค่าพารามิเตอร์ภายนอกระหว่างกล้องสีกับกล้องความลึก โดยนำกระดานหมากรุกมาวางไว้ด้านหน้ากล้อง DepthSense โดยให้ทั้งกล้องสีและกล้องความลึกมองเห็นกระดานหมากรุกเต็มแผ่น กำหนดให้ $\{P\}$ แทนกรอบอ้างอิงของกระดานหมากรุก จากนั้นจึงหาเมทริกซ์การแปลง (Transformation matrix) ของ $\{B\}$ เทียบกับ $\{P\}$ คือ ${}^{P}_{B}T$ และเมทริกซ์การแปลงของ $\{C\}$ เทียบกับ $\{P\}$ คือ ${}^{P}_{C}T$ ด้วยวิธีมาตรฐานจากการใช้ลวดลายของกระดานหมากรุก ซึ่งเป้าหมายคือการหาพารามิเตอร์ภายนอกของกล้องสีเทียบกับกล้องความลึก หรือนั่นคือการหา ${}^{C}_{B}T$ เมทริกซ์การแปลงของ $\{B\}$ เทียบกับ $\{C\}$ ซึ่งสามารถหาได้จาก ${}^{C}_{B}T = ({}^{P}_{C}T)^{-1} \times {}^{P}_{B}T$ โดยจะทำซ้ำไปเป็นจำนวน 10 ครั้งในตำแหน่งของกล้องที่แตกต่างกัน กำหนดให้ ${}^{C}_{B}T'$ คือเมทริกซ์การแปลงของ $\{B\}$ เทียบกับ $\{C\}$ ที่เราต้องการหาจากการประมาณค่าเฉลี่ยของเมทริกซ์ ${}^{C}_{B}T$ ทั้งหมด จากนั้นจึงนำเมทริกซ์การแปลงทั้งหมดที่ได้มาหาค่าเฉลี่ยการหมุน (Rotation) ด้วยการแปลงเมทริกซ์การหมุนให้อยู่ในรูปของ quaternion ทำการหาค่าเฉลี่ยของการหมุนด้วยวิธีในงานวิจัย [58] แล้วจึงแปลงผลลัพธ์ที่ได้เป็นเมทริกซ์การหมุน $\bar{R}$ สำหรับการประมาณค่าการเลื่อน (Translation) นั้นจะหาโดยการเฉลี่ยค่าจุดกำเนิดของเมทริกซ์การแปลง ${}^{C}_{B}T$ ทั้งหมดที่หามาได้ ได้เป็นจุด $\bar{t}$ ซึ่งสุดท้ายแล้วจะได้ ${}^{C}_{B}T' = \begin{bmatrix} \bar{R} & \bar{t} \\ 0 & 1 \end{bmatrix}$ ที่เป็นค่าประมาณของพารามิเตอร์ภายนอกระหว่างกล้องสีกับกล้องความลึกที่เราต้องการ

3.3. การปรับแก้พารามิเตอร์ภายนอกระหว่างกล้อง DepthSense และแขนหุ่นยนต์ Katana

กำหนดให้ $\{L\}$ และ $\{L'\}$ คือกรอบอ้างอิงของข้อต่อสุดท้ายและข้อต่อรองสุดท้ายของแขนหุ่นยนต์ Katana 6M180 ตามลำดับ โดยกรอบอ้างอิงของแขนหุ่นยนต์กำหนดให้เป็น $\{R\}$ โดยมีจุดกำเนิดอยู่ที่ฐานของแขนหุ่นยนต์ เป้าหมายของขั้นตอนนี้คือหาเมทริกซ์การแปลง ${}^{L}_{C}T$ ซึ่งแทนกรอบอ้างอิงของกล้องเทียบกับข้อต่อสุดท้ายของแขนหุ่นยนต์ ให้ข้อต่อแบบพับ (Revolute joint) ทั้ง 5 ของ Katana 6M180 ถูกกำหนดด้วยตัวเลข 1 ถึง 5 นับจากฐานจนถึงข้อต่อสุดท้าย โดยในขั้นตอนนี้เราจะคำนวณหาเมทริกซ์การแปลง ${}^{L'}_{C}T$ แทนการหา ${}^{L}_{C}T$ โดยตรง ซึ่ง ${}^{L}_{C}T$ นั้นสามารถคำนวณได้จาก ${}^{L'}_{C}T$ ด้วยพารามิเตอร์ Denavit–Hartenberg (DH) ที่รู้ค่าอยู่แล้ว

โดยการคำนวณหา ${}^{L'}_{C}T$ สามารถสรุปได้ดังต่อไปนี้ ในขั้นเตรียมการจะนำกระดานหมากรุกมาวางไว้ให้ถูกเห็นได้ด้วยกล้องวัดความลึก โดยกระดานหมากรุกจะไม่มีการเปลี่ยนแปลงตำแหน่งตลอดการ

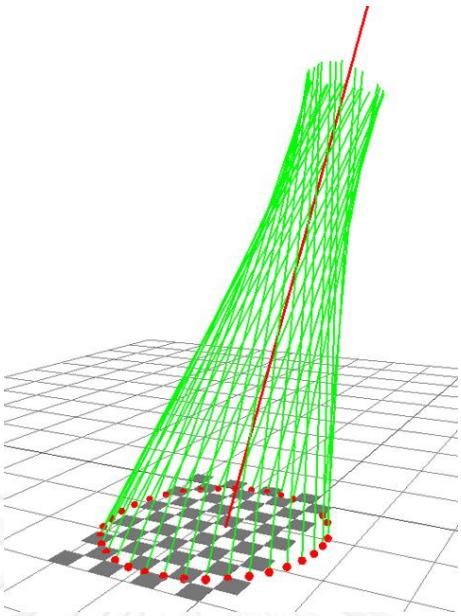
ปรับแก้พารามิเตอร์ กำหนดให้ $\{P\}$ แทนกรอบอ้างอิงของกระดานหมากรุก จากรูปที่ 3.10 แสดงให้เห็นถึงกรอบอ้างอิงของข้อต่อสองข้อสุดท้ายพร้อมด้วยกล้อง โดยมุมของข้อต่อแต่ละข้อถูกสมมติให้มีค่าเท่ากับศูนย์ กำหนดให้ z_5 คือแกน z ของ $\{L\}$ ส่วนแกน y ของข้อต่อที่ 4 นั่นคือ $\{L'\}$ ถูกกำหนดโดย y_4 ซึ่งอยู่แนวเส้นเดียวกับ z_5



รูปที่ 3.10 กรอบอ้างอิงของข้อต่อสองข้อสุดท้ายพร้อมด้วยกล้อง

การคำนวณเริ่มต้นโดยการหาทิศทางของ z_5 เทียบกับ $\{P\}$ จากนั้นจะคำนวณหาจุดเทียบกับ $\{P\}$ ที่ผ่านโดยเส้นตรงที่มี z_5 และ y_4 อยู่ ขั้นตอนที่สามจะคำนวณหาตำแหน่งของ $\{L'\}$ และขั้นตอนสุดท้ายคำนวณหาทิศทางของแกน x ของ $\{L'\}$ นั่นคือ x_4 ซึ่งในขั้นตอนสุดท้ายนี้เรารู้เมทริกซ์การแปลงของ $\{L'\}$ เทียบกับ $\{P\}$ และรู้เมทริกซ์การแปลงของ $\{C\}$ เทียบกับ $\{P\}$ จากการหาพารามิเตอร์ภายนอกด้วยวิธีมาตรฐานจากลวดลายของกระดานหมากรุก จึงทำให้สามารถคำนวณหา $^{L'}_{C}T$ ได้

1) *การคำนวณหาทิศทางของ* z_5 : กำหนดให้ o คือแกนเชิงแสง (Optical axis) ของกล้องที่อยู่บนข้อต่อสุดท้ายของแขนหุ่นยนต์ ซึ่ง o และ z_5 จะไม่ตัดกัน กล่าวคือจะมีส่วนของเส้นตรงที่ตั้งฉากทั้ง o และ z_5 โดยทุกๆมุมองศาที่เปลี่ยนไปของข้อต่อสุดท้าย ความยาวของเส้นนี้และมุมระหว่าง o กับ z_5 จะคงที่ตลอดเนื่องจาก $\{L\}$ นั้นจะถูกหมุนไปพร้อมกับข้อต่อสุดท้ายด้วย กล่าวคือถ้ามีการหมุนข้อต่อสุดท้าย o จะหมุนรอบๆ z_5 เกิดเป็นรูปร่าง hyperboloid ที่มี z_5 เป็นแกนดังรูปที่ 3.11

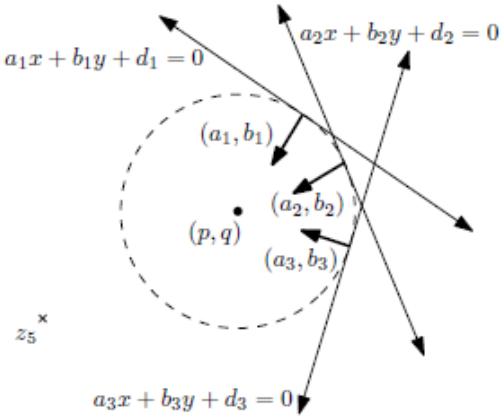


รูปที่ 3.11 ผลลัพธ์การหมุนของข้อต่อสุดท้ายที่แกนเชิงแสง (สีเขียว) หลายแกนก่อตัวเป็นรูปร่าง hyperboloid รอบแกน z_5 (สีแดง)

กำหนดให้ o_i คือแกนเชิงแสงเมื่อมุมของข้อต่อสุดท้ายเป็น α_i ซึ่งสามารถคำนวณหา o_i เทียบกับ $\{P\}$ ด้วยการหาพารามิเตอร์ภายนอกแบบมาตรฐานของกล้องเทียบกับกระดานหมากรุก เนื่องด้วยกระดานหมากรุกอยู่นิ่ง o_i จึงรวมกันเป็นรูปร่าง hyperboloid ในกรอบอ้างอิง $\{P\}$ จากนั้นกำหนดให้ $\hat{\boldsymbol{v}}_i$ เป็นเวกเตอร์หนึ่งหน่วยมีทิศตามแนวเส้นของ o_i พุ่งออกจากกระดานหมากรุก และให้ $\boldsymbol{u} = (u_x, u_y, u_z)^T$ เป็นเวกเตอร์หนึ่งหน่วยมีทิศทางตามแนวเส้น z_5 ซึ่งผลคูณจุด (Dot product) ของ $\boldsymbol{u}/\|\boldsymbol{u}\|$ และ $\hat{\boldsymbol{v}}_i$ จะมีค่าเท่ากับ $\cos(\theta)$ โดยที่ θ คือมุมระหว่าง o_i กับ z_5 ซึ่ง $\hat{\boldsymbol{v}}_i$ จะถูกคำนวณเมื่อหมุนข้อต่อสุดท้ายไปยังแต่ละมุมที่แตกต่างกัน เป็นจำนวน n มุมนั่นคือ $\alpha_i = 2\pi/n * i$ จากนั้นจึงกำหนดระบบเชิงเส้น (Linear system) $A\boldsymbol{u} = b$ โดยที่ $A = (\hat{\boldsymbol{v}}_0, \hat{\boldsymbol{v}}_1, \dots, \hat{\boldsymbol{v}}_i)^T$ และ $b = (1,1, \dots, 1)^T$ ซึ่งระบบเชิงเส้นนี้จะถูกแก้ด้วยวิธี linear least square เพื่อหา $\boldsymbol{u}$ โดยที่ $\boldsymbol{u}$ เป็นเวกเตอร์มีทิศทางตามแนวเส้น z_5 เทียบกับ $\{P\}$ และมีขนาดเท่ากับ $1/\cos(\theta)$

2) *การคำนวณหาจุดบนเส้นตรง* z_5 : จากขั้นตอนที่แล้วทำให้เรารู้ทิศทางของ z_5 แต่ยังไม่รู้ตำแหน่งของมัน ในขั้นตอนนี้จะทำการคำนวณหาตำแหน่งของจุดที่อยู่บน z_5 เริ่มต้นด้วยการกำหนดระนาบ P_{z_5} คือระนาบที่มีทิศทางเดียวกับ z_5 เมื่อเส้น o_i ถูกฉายภาพลงบนระนาบนี้ เส้นที่ได้จะห่อตัวเป็นรูปวงกลม โดยมี z_5 ตัดผ่านจุดศูนย์กลาง ซึ่งกำหนดให้ (p, q) คือจุดศูนย์กลางของวงกลมที่เกิดขึ้นบนระนาบ P_{z_5} ที่จะถูกคำนวณด้วยวิธี least square ต่อไป

พิจารณาวงกลมใน 2 มิติที่มีจุดศูนย์กลางอยู่ที่จุดกำเนิด กำหนดให้ $ax + by + d = 0$ คือสมการของเส้นสัมผัสวงกลม โดยที่ $\boldsymbol{p}$ คือเวกเตอร์ของจุดที่เส้นนั้นสัมผัสวงกลม และให้ $\boldsymbol{d}$ คือเวกเตอร์ (a, b) ซึ่งเป็นที่ชัดเจนว่า $\boldsymbol{p} \cdot (\boldsymbol{d}/\|\boldsymbol{d}\|)$ คือรัศมีของวงกลม



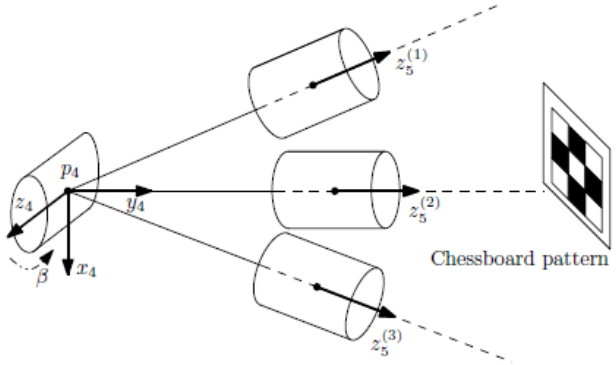
รูปที่ 3.12 วงกลมบนระนาบ P_{z_5}

จากการพิจารณาข้างต้น เราจะทำการกำหนดระบบเชิงเส้นโดยกำหนดให้เวกเตอร์ $\boldsymbol{c} = (p, q, r)$ โดยที่ (p, q) คือจุดศูนย์กลางของวงกลมและ r คือรัศมีของวงกลม เมื่อทำการฉายภาพของ o_i ลงบนระนาบ P_{z_5} จะได้เส้นตรงที่มีสมการเป็น $a_i x + b_i y + d_i = 0$ โดยที่เส้นตรงนี้จะมี (a_i, b_i) เป็นเวกเตอร์ปกติ (Normal vector) ที่มีทิศพุ่งเข้าสู่วงกลม แสดงให้เห็นดังรูปที่ 3.12 จากนั้นกำหนดให้ $b = (d_1, d_2, \dots, d_n)$ และให้ A คือเมทริกซ์ขนาด $n \times 3$ โดยที่มีข้อมูลในแต่ละแถวเป็น $(a_i, b_i, -1)$ จะได้ระบบเชิงเส้นคือ $A\boldsymbol{c} = b$ แก้สมการหา $\boldsymbol{c}$ ด้วยวิธี linear least square ซึ่งจะได้ผลลัพธ์คือ (p, q) ที่สามารถคำนวณเป็นจุดบนเส้น z_5 เทียบกับ $\{P\}$ ได้

3) *การคำนวณตำแหน่งของ* $\{L'\}$: จากขั้นตอนก่อนหน้า ทำให้เรารู้ทิศทางและตำแหน่งของ z_5 เทียบกับ $\{P\}$ ซึ่งเป็นเส้นเดียวกับ y_4 ซึ่งยังไม่เพียงพอที่จะสามารถหาตำแหน่งและแกนของ $\{L'\}$ ได้ เมื่อมุม β ของข้อต่อของกรอบอ้างอิง $\{L'\}$ มีการเปลี่ยนแปลง ทิศทางของ z_5 ก็จะมีการเปลี่ยนตามด้วย ซึ่งเมื่อพิจารณาแกน z_5 ในแต่ละมุม β ที่มีค่าแตกต่างกัน จะพบว่าแกนเหล่านี้จะตัดกันที่จุดจุดหนึ่ง ซึ่งจุดนั้นคือตำแหน่งของ $\{L'\}$ (รูปที่ 3.13) โดยการหมุนของข้อต่อที่ 4 นี้จะเหมือนกับขั้นตอนที่คำนวณทิศทางของ z_5 โดยจะหมุนไปยังมุมที่แตกต่างกันจำนวน m มุมกำหนดโดย β_i ซึ่งการหมุนนี้จะต้องไม่ทำให้กระดานหมากรุกหลุดออกจากกรอบภาพ

กำหนดให้ $z_5^{(i)}$ คือแกน z_5 เมื่อมุมข้อต่อของ $\{L'\}$ มีค่าเป็น β_i ทิศทางของ $z_5^{(i)}$ ถูกแทนด้วยเวกเตอร์ $\hat{\boldsymbol{z}}_5^{(i)}$ และให้ $c^{(i)}$ คือจุดบนเส้น $z_5^{(i)}$ ที่หาได้จากขั้นตอนที่แล้ว จากนั้นกำหนดให้ $\boldsymbol{p}_4$ คือตำแหน่งของ $\{L'\}$ โดยในเชิงอุดมคติ $\boldsymbol{p}_4$ จะเกิดจากการตัดกันของ $z_5^{(i)}$

ระยะทางสั้นที่สุดระหว่าง $\boldsymbol{p}_4$ และ $z_5^{(i)}$ ถูกกำหนดด้วย $D_i = (\boldsymbol{p}_4 - c^{(i)}) \times \hat{\boldsymbol{z}}_5^{(i)}$ โดยที่ $\hat{\boldsymbol{z}}_5^{(i)}$ เป็นเวกเตอร์หนึ่งหน่วย ซึ่งเราต้องการหา $\boldsymbol{p}_4$ ที่ทำให้ $\sum_{i=1}^{m} D_i^2$ มีค่าน้อยที่สุด ดังนั้นอนุพันธ์ของ D_i^2 เทียบกับ $\boldsymbol{p}_4$ คือ $\frac{d(D_i^2)}{d\boldsymbol{p}_4} = 2\left(\boldsymbol{p}_4 - c^{(i)}\right) - 2\hat{\boldsymbol{z}}_5^{(i)}((\boldsymbol{p}_4 - c^{(i)}) \cdot \hat{\boldsymbol{z}}_5^{(i)})$ ควรมีค่าเป็นศูนย์ โดยผลรวม (Summation) ของสมการนี้สำหรับ $z_5^{(i)}$ ที่แตกต่างกันจำนวน m ตัว จะทำให้ได้ระบบเชิงเส้นที่มี 3 สมการตามขนาดของเวกเตอร์ใน 3 มิติ ซึ่งสุดท้ายจะสามารถแก้ระบบเชิงเส้นนี้เพื่อหา $\boldsymbol{p}_4$ นั่นคือตำแหน่งของ $\{L'\}$ เทียบกับ $\{P\}$



รูปที่ 3.13 การคำนวณตำแหน่งของ $\{L'\}$

4) *การคำนวณทิศทางของ* $\boldsymbol{x_4}$: ทิศทางของ $\boldsymbol{x_4}$ สามารถหาได้โดยง่าย โดยจะลากเส้นตรงผ่าน $\boldsymbol{p_4}$ ไปตามทิศทางของแต่ละ $z_5^{(i)}$ ซึ่งเส้นเหล่านี้จะไปตัดกับระนาบของกระดานหมากรุก เกิดเป็นจุดหลายจุดบนระนาบที่เรียงตัวกันเป็นเส้นตรง โดยกำหนดให้เส้นตรงนี้คือ l_{x_4} และทิศทางของ $\boldsymbol{x_4}$ คือทิศทางของเส้นตรงที่เกิดจากการฉายภาพ l_{x_4} ลงบนระนาบที่มี $\hat{\boldsymbol{z}}_5^{(i)}$ เป็นเวกเตอร์ปกติ

3.4. การทดสอบผลการปรับแก้พารามิเตอร์

เราจะทดสอบผลการปรับแก้พารามิเตอร์ด้วยการสแกนวัตถุ โดยจะใช้วิธีปรับแก้พารามิเตอร์ที่แตกต่างกัน แล้วจึงเปรียบเทียบผลลัพธ์ความถูกต้องของการสแกนวัตถุของวิธีเหล่านี้ ค่ารากที่สองของค่าเฉลี่ยกำลังสองของความผิดพลาด (Root-mean-square error : RMSE) จะถูกนำมาใช้วัดผลโดยใช้ระยะทางสั้นที่สุดระหว่างจุดใน point cloud ที่สแกนได้ไปยังแบบจำลองเรขาคณิตของวัตถุ ซึ่งในการทดสอบนี้จะใช้วัตถุรูปทรงเรียบง่ายจำนวน 3 ชิ้นดังรูปที่ 3.14

รูปที่ 3.14 วัตถุที่นำมาใช้ในการทดสอบผลการปรับแก้พารามิเตอร์

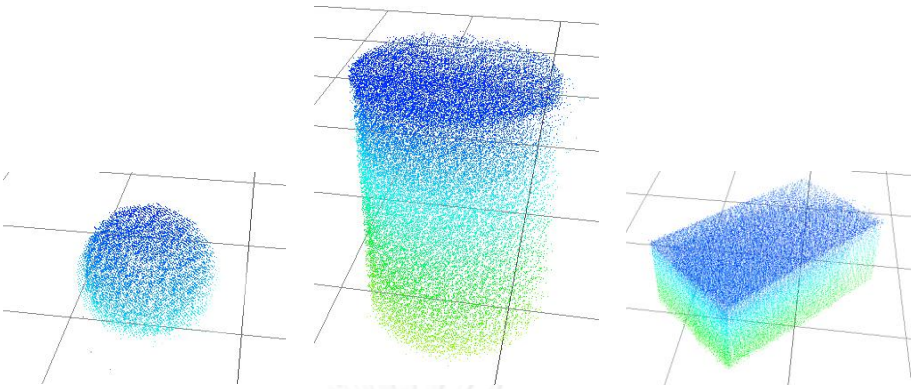
โดยวัตถุแต่ละชิ้นจะถูกนำมาวางบนเสาตั้ง เพื่อให้สามารถแยกวัตถุออกจากระนาบของพื้นได้ง่าย จากนั้นจึงทำการเคลื่อนที่แขนหุ่นยนต์ไปยังตำแหน่งที่แตกต่างกัน 5 ตำแหน่ง โดยแต่ละตำแหน่งจะสแกนวัตถุ 10 ครั้งได้ผลลัพธ์ออกมาเป็น point cloud เทียบกับกรอบอ้างอิงของแขนหุ่นยนต์ โดยกล่องขอบเขต (Bounding box) จะถูกกำหนดอย่างคร่าวๆด้วยมือ จุดใดที่อยู่นอกขอบเขตนี้ซึ่งได้แก่ เสาตั้ง ระนาบพื้น และฉากหลัง จะถูกลบออก ส่วนที่เหลือจะกำหนดให้เป็นวัตถุที่สแกนได้ และนำไปใช้ในการวัดผลต่อไป โดยตัวอย่างผลลัพธ์การสแกนวัตถุแสดงให้เห็นดังรูปที่ 3.1 ในช่วงต้นของบท

สำหรับวัตถุแต่ละชิ้นจะถูกกำหนดแบบจำลองเชิงเรขาคณิต ด้วยการวัดขนาดจากวัตถุจริงด้วยมือ โดยแบบจำลองเหล่านี้จะถูกนำมาปรับการเลื่อนและการหมุนเพื่อให้สวมเข้าพอดีกับวัตถุที่สแกนได้ จากนั้นจะคำนวณหาค่าความผิดพลาดของแต่ละจุด ซึ่งคือระยะทางสั้นที่สุดจากจุดนี้ไปยังแบบจำลองวัตถุ ค่า RMSE จะถูกคำนวณสำหรับวัตถุทุกชิ้น โดยแสดงให้เห็นดังตารางที่ 3.4 ในหน่วยเซนติเมตร โดยหลักที่สองจะเป็นผลลัพธ์จากการใช้วิธีการปรับแก้พารามิเตอร์ทั้งหมดที่ได้กล่าวมาในบทนี้ หลักที่สามจะใช้เฉพาะการปรับแก้พารามิเตอร์ภายนอกระหว่างกล้องกับแขนหุ่นยนต์ และหลักที่สี่ใช้เฉพาะการปรับแก้ค่าความลึก สำหรับหลักสุดท้ายจะเป็นวิธีที่ไม่มีการปรับแก้พารามิเตอร์ใดๆทั้งสิ้น โดยวิธีการที่ไม่มีการหาค่าพารามิเตอร์ภายนอกระหว่างกล้องกับแขนหุ่นยนต์นั้น จะทำการวัดค่า ${}^{L}_{C}T'$ นี้เองด้วยมือ

ตารางที่ 3.4 RMSE (เซนติเมตร)

วัตถุ	**RMSE (เซนติเมตร)**			
	ทั้งหมด	พารามิเตอร์ภายนอก	ค่าความลึก	ไม่มี
ทรงกลม	0.426	0.515	0.692	0.726
ทรงกระบอก	0.714	0.928	1.533	1.998
กล่อง	1.248	2.041	3.670	4.986

ซึ่งจากผลลัพธ์จะเห็นได้ว่าทั้งการปรับแก้ค่าความลึกและการปรับแก้พารามิเตอร์ภายนอก จะส่งผลให้ค่า RMSE มีค่าลดลงอย่างมีนัยสำคัญ และการปรับแก้พารามิเตอร์ที่ใช้วิธีทั้งหมดนั้น ให้ผลลัพธ์ดีที่สุดนั่นคือมีความถูกต้องมากที่สุด ซึ่งจะเห็นการเปลี่ยนแปลงของผลลัพธ์ได้อย่างชัดเจนในกรณีของกล่อง โดยผลลัพธ์ของการสแกนวัตถุแสดงดังรูปที่ 3.15



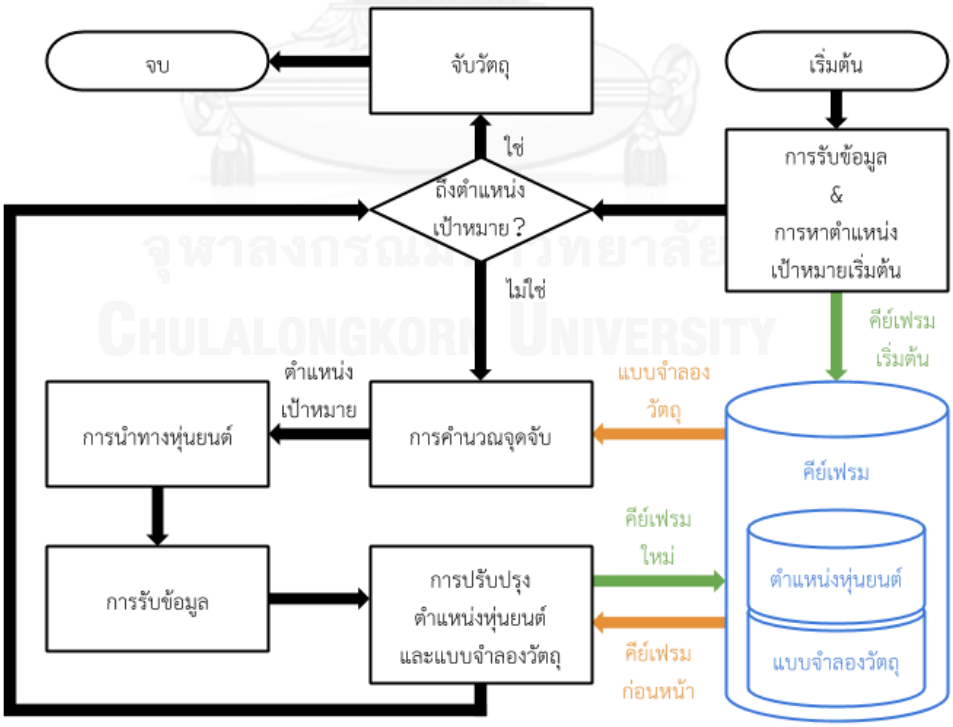
รูปที่ 3.15 ผลลัพธ์การสแกนวัตถุ

บทที่ 4
การวางแผนการจัดวางปลายนิ้ว

เนื้อหาในบทนี้จะกล่าวถึงวิธีการหลักที่จะนำเสนอในวิทยานิพนธ์ฉบับนี้ คือการวางแผนการจัดวางปลายนิ้วสำหรับการจับวัตถุที่ไม่รู้จัก โดยจะอธิบายถึงขั้นตอนของวิธีการ และกลยุทธ์ที่จะใช้ในสถานการณ์ต่างๆ รวมไปถึงอธิบายขั้นตอนวิธีการอื่นๆที่จะนำมาใช้เปรียบเทียบประสิทธิภาพการทำงานของวิธีการที่จะนำเสนอด้วย

4.1. ภาพรวมของระบบ

ลักษณะการทำงานของระบบนี้จะแตกต่างจากกระบวนการจับวัตถุแบบทั่วไป ซึ่งระบบนี้จะรวมขั้นตอนการรับข้อมูล ขั้นตอนการทำทาง และขั้นตอนการคำนวณจุดจับไว้ด้วยกันในระบบเดียว โดยในขณะที่หุ่นยนต์เคลื่อนที่นั้น อุปกรณ์รับรู้ที่ติดอยู่บนแขนหุ่นยนต์จะเก็บข้อมูล เพื่อปรับแก้ตำแหน่งของหุ่นยนต์และปรับแก้แบบจำลองของวัตถุไปพร้อมกับการเคลื่อนที่ โดยจะมีการบันทึกข้อมูลเหล่านี้เก็บเอาไว้ใช้ในขั้นตอนการปรับแก้ครั้งถัดไปเพื่อเพิ่มความถูกต้องของระบบให้มากขึ้น ซึ่งอาจมองได้ว่าระบบนี้คือการนำ SLAM มาประยุกต์ใช้สำหรับงานการจับวัตถุนั่นเอง



รูปที่ 4.1 ผังงานโดยภาพรวมของระบบ

จากรูปที่ 4.1 แสดงให้เห็นถึงภาพรวมของระบบ ซึ่งจะมีการเก็บข้อมูลไว้อยู่ในรูปแบบของคีย์เฟรม (Key frame) โดยในหนึ่งคีย์เฟรมนั้นจะเก็บทั้งข้อมูลหุ่นยนต์และข้อมูลวัตถุเอาไว้ด้วยกัน การทำงานของระบบจะเริ่มต้นโดยรับข้อมูลจากอุปกรณ์รับรู้เก็บรวมกับข้อมูลตำแหน่งของหุ่นยนต์บันทึกไว้เป็นคีย์เฟรมเริ่มต้น หลังจากขั้นตอนเริ่มต้นนี้ระบบจะทำการวนการทำงานซ้ำจนกระทั่งหุ่นยนต์ถึงเป้าหมายที่ต้องการ ซึ่งในแต่ละรอบของการวนซ้ำนั้น ระบบจะทำการหาตำแหน่งเป้าหมายเพื่อให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งนี้ โดยหาได้จากการคำนวณหาจุดจับบนแบบจำลองของวัตถุที่ผ่านการปรับปรุงจากข้อมูลคีย์เฟรมทั้งหมดที่บันทึกไว้แล้ว ในขณะที่หุ่นยนต์เคลื่อนที่ไป ก็จะอ่านข้อมูลจากอุปกรณ์รับรู้มาเพื่อปรับปรุงตำแหน่งปัจจุบันของหุ่นยนต์และแบบจำลองของวัตถุ โดยรวมเอาข้อมูลในคีย์เฟรมก่อนหน้าที่เคยบันทึกไว้มาใช้ร่วมในการปรับปรุงด้วย

สำหรับระเบียบวิธีในการคำนวณหาจุดจับนั้น จะอยู่นอกขอบเขตของวิทยานิพนธ์ฉบับนี้ จึงจะไม่มีการอธิบายในรายละเอียดส่วนนี้ ซึ่งผู้อ่านสามารถเลือกใช้ระเบียบวิธีในการหาจุดจับได้เอง โดยจะต้องได้ผลลัพธ์เป็น ตำแหน่งของจุดจับบนกรอบอ้างอิงของวัตถุ

4.2. รายละเอียดขั้นตอนการวางแผนการจัดวางปลายนิ้ว

4.2.1. การตั้งค่าระบบ

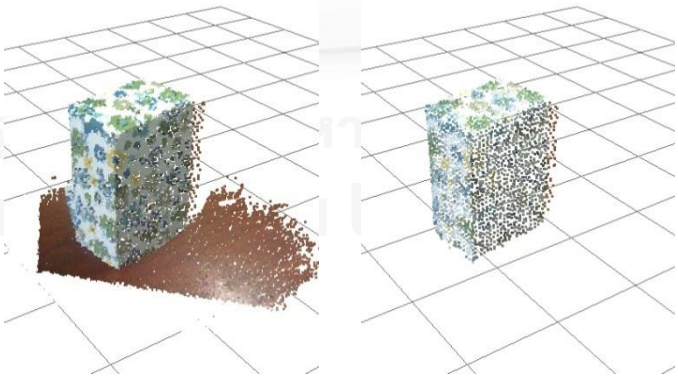
ในระบบนี้จะใช้อุปกรณ์รับรู้คือกล้อง DepthSense 325 ติดตั้งบนปลายแขนหุ่นยนต์ Katana 6M180 ที่ผ่านการปรับแก้พารามิเตอร์ภายใน พารามิเตอร์ภายนอก และค่าความลึกมาแล้ว (บทที่ 3) สำหรับวัตถุที่ใช้นั้นจะมีรูปร่างไม่ซับซ้อนมีลวดลายชัดเจน และวางบนพื้นเรียบไม่มีสิ่งของอื่นขวางหรือบดบังดังรูปที่ 4.2 โดยกำหนดให้ $\{E\}$ แทนกรอบอ้างอิงของ end effector ให้ $\{R\}$ คือกรอบอ้างอิงของฐานหุ่นยนต์ซึ่งกำหนดให้เป็นกรอบอ้างอิงเดียวกับโลก (World frame) และให้ $\{C\}$ คือกรอบอ้างอิงของกล้องโดยจุดกำเนิดอยู่ที่กล้องความลึก มีแกน z พุ่งออกจากกล้อง

รูปที่ 4.2 การตั้งค่าของระบบ

ซึ่งตำแหน่งของหุ่นยนต์ (Robot pose) ในเนื้อหาบทนี้ควรจะกำหนดด้วยเมทริกซ์การแปลงของ $\{E\}$ เทียบกับ $\{R\}$ แต่เนื่องด้วยกล้องนั้นติดอยู่กับปลายแขนหุ่นยนต์ ทำให้เมทริกซ์การแปลงของ $\{C\}$ เทียบกับ $\{E\}$ นั่นคือ ${}^{E}_{C}T$ มีค่าคงที่ เราจึงเลือกใช้เมทริกซ์การแปลงของ $\{C\}$ เทียบกับ $\{R\}$ กำหนดให้เป็นตำแหน่งหุ่นยนต์ (Robot pose) แทน กล่าวคือจะใช้ตำแหน่งของกล้องแทนตำแหน่งของหุ่นยนต์ เพื่อความสะดวกในการคำนวณและการอธิบาย

4.2.2. การรับข้อมูลจากกล้องความลึก

ในขั้นตอนนี้จะทำการอ่านค่าข้อมูลจากกล้องความลึกจำนวน N_{avg} ครั้งได้เป็นภาพความลึกจำนวน N_{avg} ภาพ จากนั้นจึงนำภาพที่ได้มาเฉลี่ยกันเพื่อลดสัญญาณรบกวน (Noise) แล้วจึงทำการแปลงภาพที่ได้ให้อยู่ในรูปแบบของ point cloud เทียบกับกรอบอ้างอิง $\{C\}$ โดยเรารู้ตำแหน่งฉากหลังของพื้นที่ทำงาน (Workspace) อยู่แล้วจึงสามารถลบฉากหลังนี้ออกจาก point cloud ได้โดยง่าย สำหรับระนาบของพื้นที่วัตถุวางอยู่นั้น จะทำการลบออกโดยการใช้ RANSAC เพื่อประมาณตำแหน่งและทิศทางของระนาบนั้น จากนั้นจึงลบจุดที่อยู่ในแนวของระนาบนั้นออกจาก point cloud ในขั้นตอนถัดไปจะทำการลบข้อมูลที่ผิดปกติ (Outlier) ด้วยวิธีทางสถิติโดยจะลบจุดที่มีค่าเบี่ยงเบนมาตรฐานของตำแหน่งมากกว่า 2 เท่าของค่าเบี่ยงเบนมาตรฐานรวม จากนั้นจึงทำการสุ่มตัวอย่าง (Sampling) โดยใช้กริดแบบ voxel (Voxel grid) เพื่อลดปริมาณข้อมูลที่ใช้ประมวลผล ตัวอย่างของ point cloud ที่ได้จากขั้นตอนนี้แสดงให้เห็นดังรูปที่ 4.3



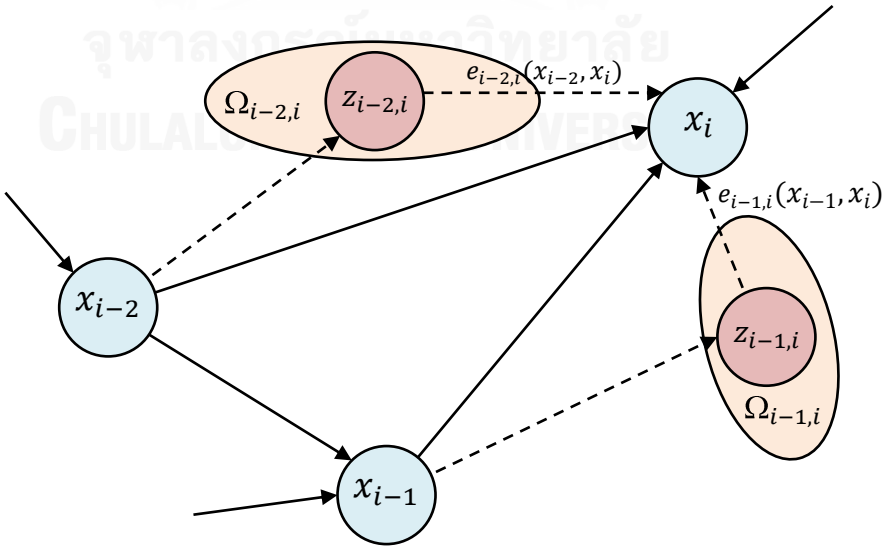
รูปที่ 4.3 ตัวอย่างผลลัพธ์จากขั้นตอนการรับข้อมูลจากกล้องความลึก ซ้าย) point cloud ก่อนการกรองข้อมูล ขวา) point cloud หลังการกรองข้อมูล

โดยกำหนดให้ $P_i = \{p_i^1, p_i^2, \dots, p_i^{M_i}\}$ คือเซตของตำแหน่งของจุดใน point cloud ที่ได้จากขั้นตอนนี้ เทียบกับ $\{C\}$ โดย i หมายถึงคีย์เฟรมลำดับที่ i และ M_i คือจำนวนจุดของ point cloud ในคีย์เฟรมลำดับที่ i

4.2.3. การปรับปรุงตำแหน่งหุ่นยนต์

ขั้นตอนนี้เป็นขั้นตอนหลักที่สำคัญของระบบนี้ โดยจะมีการนำข้อมูลของวัตถุที่หามาจากบทที่ 4.2.2 พร้อมด้วยข้อมูลในอดีตที่ได้มีการบันทึกไว้มาใช้เพื่อปรับแก้ตำแหน่งของหุ่นยนต์ให้มีความถูกต้องมากขึ้น ซึ่งข้อมูลคีย์เฟรมที่ถูกบันทึกเอาไว้นั้นจะประกอบไปด้วยเซตจำนวนสองเซตคือ S_c และ S_{obj} โดยที่ S_c คือเซตที่เก็บตำแหน่งของกล้องเทียบกับกรอบอ้างอิง $\{R\}$ หรือนั่นคือเก็บเมทริกซ์การแปลงของ $\{C\}$ เทียบกับ $\{R\}$ ของคีย์เฟรมทั้งหมดซึ่งกำหนดให้เป็น ${}^R_C T_i$ และเซต S_{obj} จะเก็บข้อมูล point cloud ของวัตถุนั่นคือ P_i ของทุกๆคีย์เฟรม ดังนั้นก่อนที่จะมีการบันทึกคีย์เฟรมลำดับที่ i จะได้ว่า $S_c = \{{}^R_C T_1, \dots, {}^R_C T_{i-1}\}$ และ $S_{obj} = \{P_1, \dots, P_{i-1}\}$ โดยสัญลักษณ์ที่อยู่ทางด้านล่างขวาจะหมายถึงลำดับของคีย์เฟรมที่ข้อมูลนี้ถูกบันทึก สำหรับคีย์เฟรมลำดับที่ 1 นั้นจะเป็นคีย์เฟรมเริ่มต้น (Initial key frame) ที่ถูกบันทึกในขณะที่แขนหุ่นยนต์อยู่ในตำแหน่งเริ่มต้น นั่นคือในขณะเริ่มต้นการทำงานของระบบ

จากแนวคิดของ SLAM แบบอาศัยกราฟ (Graph-based SLAM) เราจะมองตำแหน่งของกล้องเป็นปม (Node) ของกราฟและมองการแปลง (Transformation) ที่วัดค่าได้ระหว่างสองปมใดๆ เป็นเส้นเชื่อม (Edge) ของกราฟ โดยกำหนดให้ $\boldsymbol{x}$ คือเวกเตอร์ของสถานะหุ่นยนต์ (State) หรือนั่นคือเวกเตอร์ของปมตำแหน่งกล้อง และให้ $\boldsymbol{z}$ คือเวกเตอร์ของการวัด (Measurement) นั่นคือเวกเตอร์ของเส้นเชื่อมการแปลงที่วัดค่าได้ระหว่างสองปมใดๆ ซึ่งก่อนการบันทึกคีย์เฟรมที่ i จะได้ว่า $\boldsymbol{x} = (x_1, \dots, x_{i-1})^T$ โดยที่ x_n คือตำแหน่งของกล้องในคีย์เฟรมที่ n โดยมาจากการแปลงเมทริกซ์ ${}^R_C T_n$ ให้อยู่ในรูปของเวกเตอร์ $(x, y, z, roll, pitch, yaw)^T$ และ $\boldsymbol{z} = (z_{1,2}, z_{1,3}, \dots, z_{1,i-1}, z_{2,3}, z_{2,4}, \dots, z_{2,i-1}, \dots, z_{i-2,i-1})^T$ โดย $z_{m,n}$ คือการแปลงของปมในคีย์เฟรมที่ n เทียบกับปมในคีย์เฟรมที่ m ที่วัดค่าได้ซึ่งอยู่ในรูปของเวกเตอร์ $(x, y, z, roll, pitch, yaw)^T$ เช่นกัน โดยมี $\Omega_{m,n}$ เป็นเมทริกซ์ข้อมูลของการวัด $z_{m,n}$ นี้



รูปที่ 4.4 กราฟของตำแหน่งกล้องในคีย์เฟรมที่ i

ในคีย์เฟรมที่ i เราจะทำการเพิ่มปมของตำแหน่งกล้อง x_i เข้าไปในกราฟ โดยจะทำการวัดค่าเทียบกับปม x_1 จนถึงปม x_{i-1} เพื่อหาค่าการวัด $z_{1,i}$ จนถึง $z_{i-1,i}$ ตามลำดับ จากรูปที่ 4.4 แสดงให้เห็นถึงกราฟที่เพิ่มปม x_i เข้าไปแล้ว โดยวงกลมสีฟ้าแสดงถึงปมตำแหน่งกล้อง ณ ปัจจุบัน วงกลมสีแดงคือตำแหน่งของ x_i ที่ได้จากการวัดค่าเทียบกับปมอื่นๆในคีย์เฟรมก่อนหน้า และวงรีสีส้มแสดงถึงเมทริกซ์ข้อมูลที่แสดงถึงความเชื่อมั่นในตำแหน่งของปมที่วัดค่าได้นั้นๆ โดยให้ $e_{m,n}(x_m, x_n)$ คือฟังก์ชั่นความผิดพลาดในการวัดตำแหน่งของปม x_n เทียบกับ x_m ซึ่งวงกลมสีแดงเหล่านี้ในเชิงอุดมคตินั้นควรจะมีตำแหน่งเดียวกัน กล่าวคือเมื่อทำการวัดค่าตำแหน่งของปม x_i เทียบกับปมอื่นๆแล้วควรจะให้ค่าผลลัพธ์ตำแหน่งของกล้องเทียบกับโลกเป็นค่าเดียวกัน กล่าวคือฟังก์ชันความผิดพลาด $e_{m,n}(x_m, x_n)$ ต้องมีค่าเท่ากับศูนย์ ดังนั้นเราจึงต้องทำการปรับแก้ตำแหน่งของกล้องเพื่อให้ค่าความผิดพลาดในกราฟนี้น้อยที่สุด โดยเราจะใช้ TORO [50] ในการหาค่าเหมาะสมที่สุด

สำหรับค่าการวัด $z_{j,i}$ นั่นคือการแปลงระหว่างตำแหน่งกล้องในคีย์เฟรมที่ i เทียบกับตำแหน่งกล้องในคีย์เฟรมที่ j โดยที่ $j = 1, \dots, i-1$ ซึ่งจะหาได้โดยใช้วิธี ICP แบบจุดถึงระนาบ ระหว่างข้อมูล point cloud P_i กับ P_j โดยกำหนดให้ T^* คือเมทริกซ์การแปลงที่เป็นผลลัพธ์จากขั้นตอนการทำ ICP ที่ทำให้ฟังก์ชันความผิดพลาดต่อไปนี้มีค่าน้อยที่สุด กำหนดให้ $corr(k)$ คือดัชนีของจุดใน P_j ที่มีความสอดคล้อง (Correspondence) กับจุดดัชนีที่ k ใน P_i ซึ่งเราจะใช้ระยะทางที่สั้นที่สุดระหว่างสองจุดเป็นความสอดคล้องนี้

$$E_{j,i}(T^*) = \sum_{k=1}^{M_i} \left[\left((T^*)(p_i^k) - p_j^{corr(k)} \right) \cdot \boldsymbol{n}_j^{corr(k)} \right]^2$$

โดยที่ $\boldsymbol{n}_j^{corr(k)}$ คือเวกเตอร์ปกติของจุด $p_j^{corr(k)}$ ในเซต P_j โดยอธิบายวิธีการหาเวกเตอร์นี้แล้วในบทที่ 2.2.2 ซึ่งจะได้ว่า $z_{j,i}$ หาค่าได้จากการแปลงเมทริกซ์ T^* ให้อยู่ในรูปของ $(x, y, z, roll, pitch, yaw)^T$ นั่นเอง และเมทริกซ์ข้อมูล $\Omega_{j,i}$ ของค่าการวัดนี้จะมีค่าดังนี้

$$\Omega_{j,i} = \begin{bmatrix} \frac{w_t}{E_{j,i}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{w_t}{E_{j,i}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{w_t}{E_{j,i}} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{w_r}{E_{j,i}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{w_r}{E_{j,i}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{w_r}{E_{j,i}} \end{bmatrix}$$

โดย w_t และ w_r คือค่าถ่วงน้ำหนักมีค่าเท่ากับ 1 และ 0.001 ตามลำดับได้จากการปรับค่าด้วยมือ โดยจะสมมติว่าค่าตำแหน่งและทิศทางของสถานะที่อยู่ในเวกเตอร์ $(x, y, z, roll, pitch, yaw)^T$ แต่ละค่านั้นเป็นอิสระจากกัน (Independence) ซึ่งเมื่อฟังก์ชันความผิดพลาด $E_{j,i}$ มีค่ามากจะทำให้ค่าในเมทริกซ์ข้อมูล $\Omega_{j,i}$ มีค่าน้อยนั่นหมายถึงมีความเชื่อมั่นในตำแหน่งที่วัดค่าได้น้อยนั่นเอง จากนั้นจะทำการเพิ่มข้อมูลเหล่านี้ลงไปในกราฟนั่นคือการเพิ่ม $z_{1,i}, \dots, z_{i-1,i}$ และ x_i ไปยังเวกเตอร์ $\boldsymbol{z}$ และ $\boldsymbol{x}$ ตามลำดับ โดย x_i หาค่ามาจากเมทริกซ์การแปลง ${}^R_C T_i$ ซึ่งหมายถึงตำแหน่งของกล้อง ณ

ขณะปัจจุบัน ที่หาค่าได้จาก encoders ของแขนหุ่นยนต์ แล้วจึงทำการหาค่าเหมาะสมที่สุดของกราฟนี้ด้วย TORO ซึ่งสมาชิกทั้งหมดของเวกเตอร์ $\boldsymbol{x}$ จะถูกปรับแก้ซึ่งกำหนดให้เป็นเวกเตอร์ใหม่ $\boldsymbol{x}'$ และกำหนดให้เซต S_c' คือเซตที่แปลงสมาชิกทั้งหมดของ $\boldsymbol{x}'$ ให้อยู่ในรูปของเมทริกซ์การแปลงเทียบกับ $\{R\}$ ซึ่งในขั้นตอนสุดท้ายเราจะทำการบันทึกคีย์เฟรม โดยการเพิ่ม P_i ลงไปยังเซต S_{obj} และเซตของตำแหน่งกล้องจะมีค่าเป็น $S_c = S_c'$

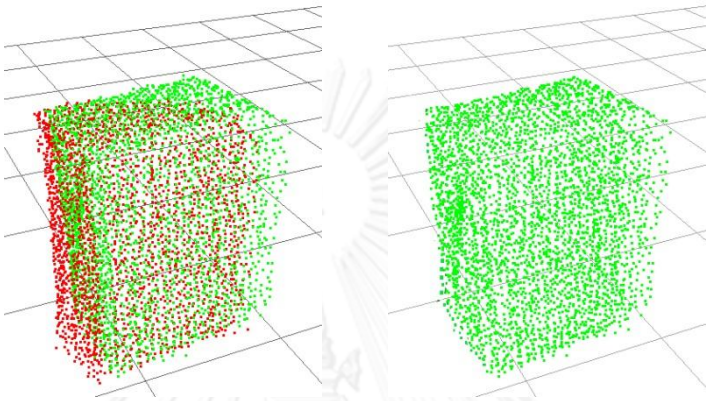
4.2.4. การปรับปรุงตำแหน่งและแบบจำลองวัตถุ

เนื่องจากตำแหน่งของกล้องในทุกคีย์เฟรมนั้นจะมีค่าที่เปลี่ยนแปลงไป ภายหลังจากการเพิ่มคีย์เฟรมใหม่ ดังนั้นข้อมูล point cloud ของวัตถุในเซต S_{obj} จะมีค่าเปลี่ยนไปเมื่อเทียบกับ $\{R\}$ จึงเป็นเหตุผลให้ควรทำการปรับปรุงแบบจำลองวัตถุ โดยการใช้ข้อมูลจากทุกคีย์เฟรมมาคำนวณใหม่ทุกครั้งภายหลังมีการเพิ่มคีย์เฟรม ซึ่งการปรับปรุงแบบจำลองวัตถุนั้นจะเริ่มต้นจากการรวมข้อมูล point cloud ของทุกคีย์เฟรมเทียบกับ $\{R\}$ โดยแปลงให้อยู่ในรูปแบบของต้นไม้ KD เพื่อความรวดเร็วในการคำนวณหาจุดที่อยู่ใกล้ที่สุด จากนั้นจึงหาจุดที่อยู่ใกล้กันมากที่สุดในทุกๆ คีย์เฟรม แทนจุดเหล่านี้ด้วยจุดที่หามาจากค่าเฉลี่ยของตำแหน่งในสามมิติของจุดเหล่านี้ ซึ่งสามารถอธิบายได้ด้วยรหัสเทียมในตารางที่ 4.1 โดยกำหนดให้ N_k คือจำนวนคีย์เฟรมในปัจจุบัน

ตารางที่ 4.1 รหัสเทียมของวิธีการรวมข้อมูล point cloud ของวัตถุ

Input: S_c, S_{obj}
for $n \leftarrow 2$ **to** N_k **do** // ทำซ้ำในคีย์เฟรมที่ 2 จนถึงคีย์เฟรมสุดท้าย
  $P_n \leftarrow transformAllPoints(P_n, {}^{R}_{C}T_n)$ // แปลงทุกจุดในเซตเทียบกับโลก
  $P_{n-1} \leftarrow transformAllPoints(P_{n-1}, {}^{R}_{C}T_{n-1})$
  $tree_{n-1} \leftarrow kdtree(P_{n-1})$ // สร้างต้นไม้ KD-tree จากเซต
  for all $p_n^m \in P_n$ **do** // ทำซ้ำทุกจุดในเซต
    $p_{nearest} \leftarrow findNearestPoint(tree_{n-1}, p_n^m)$ // หาจุดที่ใกล้ที่สุดในต้นไม้
    If $euclideanDist(p_n^m, p_{nearest}) < t_{maxdist}$ **then** // ตรวจสอบระยะห่างใกล้พอที่จะหาค่าเฉลี่ยหรือไม่
      $P_n \leftarrow P_n - \{p_n^m\}$ // ลบจุดออกจากเซต
      $P_{n-1} \leftarrow P_{n-1} - \{p_{nearest}\}$ // ลบจุดออกจากเซต
      $p\_avg \leftarrow (w_n p_{nearest} + p_n^m)/(w_n + 1)$ // เฉลี่ยจุดแบบมีน้ำหนัก
      $P_n \leftarrow P_n \cup \{p\_avg\}$ // เพิ่มจุดที่เฉลี่ยแล้วในเซต
    end if
  end for
  $P_n \leftarrow P_n \cup P_{n-1}$ // รวมจุดที่ไม่มีจุดที่ใกล้พอที่จะหาค่าเฉลี่ย
end for
return P_{N_k} // คืนค่าเซตของจุดที่ผ่านการรวมในทุกคีย์เฟรมแล้ว

โดยกำหนดให้ $t_{maxdist}$ มีค่าเป็น 5 มิลลิเมตร ซึ่งค่านี้หาได้จากการทดลองปรับค่าด้วยมือ และให้ w_n มีค่าเป็น $n-1$ โดยที่ $n = 2, \ldots, N_k$ เพื่อให้ผลการเฉลี่ยของจุดในคีย์เฟรมปัจจุบันไม่ส่งผลกระทบต่อแบบจำลองวัตถุมากจนเกินไป จากรูปที่ 4.5 คือตัวอย่างการรวมข้อมูล point cloud ของวัตถุ



รูปที่ 4.5 ตัวอย่างการรวมข้อมูล point cloud ของวัตถุ ซ้าย) ก่อนการรวม ขวา) หลังการรวม

ภายหลังจากการรวมข้อมูล point cloud ทั้งหมดแล้วจะได้ผลลัพธ์เป็นเซตของจุดเทียบกับ $\{R\}$ กำหนดให้เป็น P_{obj} ซึ่งเราจะใช้ข้อมูลจุด point cloud ในเซตนี้ในการหากรอบอ้างอิงของวัตถุ ซึ่งกำหนดเป็น $\{O\}$ โดยการใช้วิธีการวิเคราะห์องค์ประกอบหลัก (Principal Component Analysis: PCA) จะทำให้เราได้แกน x, y และ z ของวัตถุเทียบกับ $\{R\}$ กำหนดให้เป็นเวกเตอร์หนึ่งหน่วย $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$ และ $\hat{\boldsymbol{z}}$ ตามลำดับ และได้จุดศูนย์กลางมวล (Center of mass) เทียบกับ $\{R\}$ กำหนดโดยจุด p_{cm} ดังนั้นเราจะได้เมทริกซ์การแปลงของ $\{O\}$ เทียบกับ $\{R\}$ ของคีย์เฟรมล่าสุดได้เป็น ${}^{R}_{O}T_{N_k} = \begin{bmatrix} \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{\boldsymbol{z}} & p_{cm} \\ 0 & 0 & 0 & 1 \end{bmatrix}$ จากนั้นจึงหาขนาดของวัตถุโดยใช้กล่องขอบเขต (Bounding box) แบบง่าย กำหนดให้ d_x, d_y และ d_z คือขนาดของวัตถุในแนวแกน x, y และ z ของ $\{O\}$ ตามลำดับ โดยสามารถหาค่าได้ดังนี้

$$d_x = \max_{p_i \in P_{obj}} ((p_i - p_{cm}) \cdot \hat{\boldsymbol{x}}) - \min_{p_j \in P_{obj}} ((p_j - p_{cm}) \cdot \hat{\boldsymbol{x}})$$

$$d_y = \max_{p_i \in P_{obj}} ((p_i - p_{cm}) \cdot \hat{\boldsymbol{y}}) - \min_{p_j \in P_{obj}} ((p_j - p_{cm}) \cdot \hat{\boldsymbol{y}})$$

$$d_z = \max_{p_i \in P_{obj}} ((p_i - p_{cm}) \cdot \hat{\boldsymbol{z}}) - \min_{p_j \in P_{obj}} ((p_j - p_{cm}) \cdot \hat{\boldsymbol{z}})$$

โดยสุดท้ายแล้วข้อมูลที่จะส่งไปยังขั้นตอนการหาจุดจับนั้นคือ P_{obj}, ${}^{R}_{O}T_{N_k}$, d_x, d_y และ d_z ซึ่งผลลัพธ์ที่ต้องการจากขั้นตอนการหาจุดจับนั้นจะเป็นข้อมูลของจุดจับเทียบกับ $\{O\}$ โดยรูปแบบและจำนวนของข้อมูลจุดจับนั้นจะขึ้นอยู่กับกระบวนการที่เลือกใช้และจำนวนนิ้วของหุ่นยนต์ตามลำดับ

4.2.5. การนำทางหุ่นยนต์และเงื่อนไขการเพิ่มข้อมูลคีย์เฟรม

ขั้นตอนนี้จะใช้ข้อมูลจุดจับเทียบกับ $\{O\}$ ที่ได้จากขั้นตอนการหาจุดจับ ซึ่งเรารู้พารามิเตอร์ DH ของมือจับและแขนหุ่นยนต์ ทำให้สามารถคำนวณเมทริกซ์การแปลงของ $\{E\}$ ในขณะที่กำลังจับวัตถุในตำแหน่งจุดจับเหล่านี้เทียบกับ $\{O\}$ ได้ โดยกำหนดให้เป็น ${}^{O}_{E}T_{grasp}$ สำหรับการเคลื่อนที่เพื่อเข้าไปวางนิ้วบนวัตถุนั้นจะทำการเคลื่อนที่แบบเส้นตรง (Linear movement) โดยส่งคำสั่งตำแหน่งเป้าหมาย ${}^{R}_{E}T_{step}$ ซึ่งคือตำแหน่งของ end effector เทียบกับฐานหุ่นยนต์ ผ่านทาง SDK ของแขนหุ่นยนต์ สำหรับเมทริกซ์การแปลงนี้สามารถคำนวณได้จาก ${}^{R}_{E}T_{step} = {}^{R}_{O}T_{N_k} \cdot {}^{O}_{E}T_{grasp}$ ในขณะที่แขนหุ่นยนต์กำลังเคลื่อนที่นั้น จะมีการอ่านค่าตำแหน่งของหุ่นยนต์เพื่อตรวจสอบว่าผ่านเงื่อนไขของการเพิ่มคีย์เฟรมหรือไม่ ถ้าผ่านก็จะมีการบันทึกคีย์เฟรมใหม่เกิดขึ้น ซึ่งจะส่งผลให้ได้เมทริกซ์การแปลง ${}^{R}_{E}T_{step}$ ใหม่ โดยถ้าเมทริกซ์การแปลง ${}^{R}_{E}T_{step}$ มีการเปลี่ยนแปลงค่าจากเดิม การเคลื่อนที่ของแขนหุ่นยนต์ในปัจจุบันจะถูกยกเลิกทันที และจะเคลื่อนที่ไปยังตำแหน่งเป้าหมายใหม่แทน

สำหรับเงื่อนไขในการเพิ่มคีย์เฟรมนั้นจะอธิบายได้ดังนี้ กำหนดให้ d คือระยะทางแบบยุคลิด (Euclidean distance) ระหว่างตำแหน่งหุ่นยนต์ปัจจุบัน ${}^{R}_{E}T$ กับตำแหน่งหุ่นยนต์เป้าหมาย ${}^{R}_{E}T_{step}$ และระยะทางแบบยุคลิด ระหว่างตำแหน่งหุ่นยนต์ของคีย์เฟรมล่าสุด ${}^{R}_{E}T_{N_k}$ กับตำแหน่งหุ่นยนต์เป้าหมาย ${}^{R}_{E}T_{step}$ กำหนดให้เป็น D และให้ N_{kmax} คือจำนวนคีย์เฟรมมากที่สุดที่อนุญาตให้มีได้ ซึ่งจะมีการเพิ่มคีย์เฟรมเมื่อ $d_{min} \leq d < D/2$ และ $N_k \leq N_{kmax}$ โดย d_{min} คือระยะทางน้อยที่สุดที่อนุญาตให้มีการเพิ่มคีย์เฟรมได้

4.2.6. การประมาณตำแหน่งของหุ่นยนต์โดยใช้ข้อมูลจากกล้องสี

เนื่องด้วยข้อจำกัดของอุปกรณ์รับรู้ หรือกล้อง DepthSense นั้นจะสามารถวัดค่าความลึกได้ที่ระยะใกล้ที่สุด 15 เซนติเมตร ซึ่งในกรณีที่แขนหุ่นยนต์เคลื่อนที่เข้าใกล้วัตถุมากนั้น จะทำให้ระบบอยู่ในสถานะที่ไม่มีข้อมูลความลึกสำหรับใช้ปรับปรุงตำแหน่งของหุ่นยนต์ เมื่อเคลื่อนที่แขนหุ่นยนต์จากตำแหน่งสุดท้ายที่มีข้อมูลความลึกไปยังตำแหน่งเป้าหมาย จึงอาจทำให้เกิดความผิดพลาดในการระบุตำแหน่งของหุ่นยนต์ ที่แปรผันตามระยะทางที่เคลื่อนที่ไป และส่งผลให้เกิดการชนกับวัตถุหรือไม่สามารถวางนิ้วบนตำแหน่งจุดจับได้อย่างถูกต้อง โดยในขั้นตอนนี้จะอาศัยข้อมูลจากกล้องสีที่สามารถรับรู้ข้อมูลได้ขณะอยู่ใกล้วัตถุ มาทำการประมาณค่าการเคลื่อนที่ของหุ่นยนต์ เพื่อลดความผิดพลาดในการระบุตำแหน่งของหุ่นยนต์ในขณะที่ระบบไม่สามารถรับรู้ข้อมูลความลึกได้

โดยการอาศัยความสอดคล้องระหว่างข้อมูลในสองมิติกับข้อมูลในสามมิติ (2D-3D correspondence) เพื่อให้สามารถใช้ข้อมูลในสองมิติบอกถึงการเปลี่ยนแปลงข้อมูลในสามมิติได้ กล่าวคือใช้ข้อมูลสองมิติจากภาพสีเพื่อบอกถึงการแปลงของตำแหน่งกล้องได้ เริ่มต้นโดยนำข้อมูลในสามมิติคือ point cloud เทียบกับ $\{R\}$ ของวัตถุ P_{obj} ในคีย์เฟรมสุดท้าย มาทำการเลือกจุดแบบสุ่มด้วยความน่าจะเป็น 1 ใน 5 เพื่อลดปริมาณข้อมูลที่ใช้คำนวณ โดยกำหนดให้จุดที่เลือกมานี้อยู่ในเซต P'_{obj} จากนั้นจึงทำการฉายภาพของจุดเหล่านี้ลงบนระนาบของภาพสี ดังจุดสีเขียวในรูปที่ 4.6 ด้วยสมการดังต่อไปนี้

$$u_i = proj(^B_R T \cdot p_i), \qquad p_i \epsilon P'_{obj}$$

โดยที่ u_i คือจุดในสองมิติบนภาพสีของการฉายภาพจุดในสามมิติ p_i กำหนดให้ U คือเซตที่มีสมาชิกเป็น u_i ที่หามาได้ทั้งหมด ฟังก์ชัน $proj$ คือฟังก์ชันการฉายภาพของกล้องสีที่ทราบพารามิเตอร์ภายในอยู่แล้ว และ $^B_R T$ คือเมทริกซ์การแปลงของฐานหุ่นยนต์เทียบกับกรอบอ้างอิงของกล้องสีที่กำหนดโดย $\{B\}$ ซึ่งมีค่าเท่ากับ $^B_C T \cdot (^R_C T_{N_k})^{-1}$ โดย $^R_C T_{N_k}$ คือเมทริกซ์การแปลงของกล้องในคีย์เฟรมสุดท้าย และ $^B_C T$ คือเมทริกซ์การแปลงของพารามิเตอร์ภายนอกระหว่างกล้องความลึกเทียบกับกล้องสี ซึ่งสามารถหาค่าด้วยวิธีที่กล่าวไว้ในบทที่ 3.2.3

รูปที่ 4.6 การฉายภาพจุดในสามมิติลงบนภาพสีในสองมิติ

ภายหลังการเคลื่อนที่ของหุ่นยนต์ภาพสีจะมีการเปลี่ยนแปลง เราจึงจะติดตามว่าจุดในเซต U บนภาพในอดีต ควรเปลี่ยนแปลงไปอย่างไรในภาพปัจจุบัน โดยจะใช้วิธี pyramids Lucas-Kanade optical flow [53] ซึ่งผลลัพธ์ที่ได้ (รูปที่ 4.7) กำหนดให้เป็นเซต U' จะเป็นเซตของจุดบนภาพปัจจุบันที่รู้ความสอดคล้อง (Correspondence) กับจุดบนภาพในอดีต กล่าวคือจะรู้ความสอดคล้องกับจุดในสามมิติภายในเซต P'_{obj} นั่นเอง โดยเซต U' นั้นจะมีขนาดน้อยกว่าหรือเท่ากับขนาดของเซต U เนื่องด้วยการฉายภาพของจุดในเซต P'_{obj} มาเป็นจุดบนภาพนั้นมีเพียงบางส่วนที่ฉายภาพลงมาบนจุดที่เป็น feature ของภาพพอดีทำให้สามารถตรวจพบและติดตามได้ ส่วนจุดที่ฉายภาพลงมาไม่ตรงกับ feature ใดๆนั้น จะถูกลบออกไปเพราะไม่สามารถตรวจพบและติดตามได้ด้วยวิธี optical flow นี้ ดังจุดสีแดงในรูปที่ 4.7

รูปที่ 4.7 ผลลัพธ์การติดตามจุดบนภาพด้วยวิธี optical flow

จากนั้นจะทำการประมาณค่าตำแหน่งของหุ่นยนต์ โดยกำหนดให้ T^* คือค่าประมาณของเมทริกซ์การแปลงของ $\{R\}$ เทียบกับ $\{B\}$ ในปัจจุบัน และให้ E_{proj} คือฟังก์ชันค่าความผิดพลาดในการฉายภาพ (Projection error) ซึ่งเราต้องการหา T^* ที่ทำให้ค่าความผิดพลาดในการฉายภาพนี้มีค่าน้อยที่สุด โดย E_{proj} มีค่าดังสมการต่อไปนี้

$$E_{proj}(T^*) = \sum_{u_i' \epsilon U'} \|u_i' - proj(T^* \cdot p_i)\|$$

โดยที่ p_i เป็นสมาชิกของเซต P'_{obj} ซึ่งโดยข้อเท็จจริงแล้วจุดในเซต P'_{obj} ที่ฉายภาพลงบนภาพสีนั้นอาจเป็นจุดที่ถูกบังและไม่สามารถมองเห็นจุดนั้นได้จากกล้องสี ทำให้จุดเหล่านี้บนภาพสีจะส่งผลให้ค่าความผิดพลาด E_{proj} มีความผิดพลาดเกินจริง เราจึงจำเป็นต้องกำหนดให้จุดเหล่านี้เป็นข้อมูลที่ผิดปกติ (Outlier) แต่ในความเป็นจริงนั้นการหาจุดเหล่านี้เป็นไปได้ยาก ในขั้นตอนนี้จึงจะเลือกใช้วิธี RANSAC เพื่อหลีกเลี่ยงความผิดพลาดที่เกิดจากข้อมูลที่ผิดปกติเหล่านี้โดยจะใช้ E_{proj} เป็นฟังก์ชันความผิดพลาด และให้ T^* มีค่าคาดเดาเริ่มต้น (Initial guess) เป็นเมทริกซ์การแปลง ${}^B_R T$ ที่หาค่าได้ด้วย encoder ของแขนหุ่นยนต์ในปัจจุบัน ซึ่งสุดท้ายแล้วจะได้ตำแหน่งของหุ่นยนต์ หรือเมทริกซ์การแปลงของ $\{E\}$ เทียบกับ $\{R\}$ มีค่าเท่ากับ $(T^*)^{-1} \cdot {}^B_C T \cdot {}^C_E T$ โดยที่ ${}^B_C T$ และ ${}^C_E T$ ทราบค่าอยู่แล้วจากขั้นตอนการปรับแก้พารามิเตอร์

โดยจะทำขั้นตอนเหล่านี้ซ้ำตั้งแต่ขั้นตอนการหา optical flow ไปพร้อมกับการเคลื่อนที่ของหุ่นยนต์ โดยก่อนการทำซ้ำในครั้งถัดไปจะกำหนดให้ $U = U'$ เพื่อใช้ติดตามจุดในเซต U นี้ในภาพถัดไปด้วย optical flow และกำหนดให้เซต P'_{obj} ไม่มีการเปลี่ยนแปลงค่าตลอดการทำงานในขั้นตอนนี้ ซึ่งจะหยุดการทำงานเมื่อหุ่นยนต์ถึงตำแหน่งเป้าหมาย หรือจำนวนของจุดที่หาได้จาก optical flow มีค่าน้อยกว่า 20 จุด

4.3. สรุปขั้นตอนการวางแผนการจัดวางปลายนิ้ว

ในส่วนนี้จะสรุปขั้นตอนการทำงานของระบบสำหรับวิธีที่นำเสนอ รวมถึงวิธีการอื่นที่จะใช้สำหรับเปรียบเทียบประสิทธิภาพการทำงาน โดยจะแบ่งออกเป็นทั้งหมด 5 วิธีการซึ่งสามารถอธิบายรายละเอียดโดยสังเขปได้ดังนี้ *วิธีการที่ 1* จะเป็นวิธีการทั่วไปที่จะสแกนวัตถุแล้วคำนวณหาจุดจับ จากนั้นจะคำนวณตำแหน่งหุ่นยนต์เทียบกับจุดจับนั้น แล้วจึงเคลื่อนที่เข้าจับวัตถุ โดยไม่มีการใช้ข้อมูลใดๆระหว่างการเคลื่อนที่ ซึ่งเราจะใช้วิธีนี้เป็นเส้นฐาน (Baseline) สำหรับการเปรียบเทียบ โดยผลการทดลองของวิธีการอื่นๆ ควรจะให้ผลลัพธ์ที่ดีกว่าวิธีการนี้ทั้งหมด

วิธีการที่ 2 จะใช้แนวคิดของการเก็บข้อมูลในขณะเคลื่อนที่ โดยวิธีการนี้จะเคลื่อนที่ไปเป็นระยะทางครึ่งหนึ่งของตำแหน่งหุ่นยนต์ปัจจุบันเทียบกับตำแหน่งเป้าหมาย แล้วทำการเก็บข้อมูลของวัตถุมาปรับปรุงแบบจำลองวัตถุให้มีความถูกต้องมากยิ่งขึ้น เพื่อให้สามารถคำนวณจุดจับที่ถูกต้องมากขึ้นจากการปรับปรุงของแบบจำลองวัตถุนี้ โดยไม่มีการปรับปรุงตำแหน่งหุ่นยนต์ และจะทำซ้ำเช่นนี้จนกว่าจะใกล้ตำแหน่งเป้าหมายเพียงพอ

ซึ่งจาก *วิธีการที่ 2* นั้นความถูกต้องของแบบจำลองวัตถุ จะขึ้นอยู่กับความถูกต้องของตำแหน่งหุ่นยนต์ด้วย ดังนั้นสำหรับ *วิธีการที่ 3* และ *วิธีการที่ 4* จะใช้ข้อมูลในอดีตมาปรับแก้ทั้งตำแหน่งหุ่นยนต์และแบบจำลองวัตถุ โดย *วิธีการที่ 4* นั้นจะมีการหาค่าเหมาะสมที่สุด (Optimization) สำหรับการปรับปรุงตำแหน่งของหุ่นยนต์ให้มีความถูกต้องมากที่สุด ในขณะที่ *วิธีการที่ 3* ไม่มีการกระทำในส่วนนี้ และวิธีการสุดท้าย *วิธีการที่ 5* (วิธีการที่นำเสนอ) จะเหมือน *วิธีการที่ 4* โดยจะมีการแก้ไขปัญหาของการใช้อุปกรณ์รับรู้แบบตาในมือ ที่ไม่สามารถรับรู้ข้อมูลความลึกได้ในขณะที่อยู่ใกล้วัตถุ โดยเพิ่มการที่ใช้ข้อมูลจากกล้องสี เพื่อใช้ประมาณค่าตำแหน่งหุ่นยนต์ ในขณะที่ไม่สามารถรับรู้ข้อมูลความลึกได้ ซึ่งสุดท้ายแล้วเราคาดหวังว่า *วิธีการที่ 5* จะให้ผลลัพธ์ในการทดลองดีที่สุด ในขณะที่ *วิธีการที่ 1* จะให้ผลลัพธ์ที่แย่ที่สุด สำหรับรายละเอียดการทำงานแบบเป็นขั้นตอนของแต่ละวิธีการสามารถอธิบายได้ดังตารางที่ 4.2

ตารางที่ 4.2 สรุปรายละเอียดการทำงานแบบเป็นขั้นตอนของแต่ละวิธีการ

ขั้นตอน	วิธีการที่					หมายเหตุ
	1	2	3	4	5	
1. รับข้อมูลจากกล้องความลึกและอ่านข้อมูลตำแหน่งหุ่นยนต์ บันทึกเป็นคีย์เฟรมเริ่มต้น	✓	✓	✓	✓	✓	บทที่ 4.2.2
2. คำนวณหาจุดจับจากแบบจำลองวัตถุจากข้อมูลคีย์เฟรมเริ่มต้น	✓	✓	✓	✓	✓	-
3. คำนวณหาตำแหน่งหุ่นยนต์เป้าหมายจากจุดจับที่คำนวณมาได้	✓	✓	✓	✓	✓	บทที่ 4.2.5
4. เคลื่อนที่แขนหุ่นยนต์ไปเป็นระยะทางครึ่งหนึ่งของระยะทางจากตำแหน่งหุ่นยนต์ปัจจุบันกับตำแหน่งเป้าหมาย	✗	✓	✓	✓	✓	บทที่ 4.2.5
5. รับข้อมูลจากกล้องความลึกและอ่านข้อมูลตำแหน่งปัจจุบันของหุ่นยนต์ บันทึกเพิ่มเป็นคีย์เฟรมใหม่	✗	✓	✓	✓	✓	บทที่ 4.2.2
• มีการปรับปรุงตำแหน่งหุ่นยนต์	✗	✗	✓	✓	✓	บทที่ 4.2.3
• มีการหาค่าเหมาะสมที่สุดของตำแหน่งหุ่นยนต์	✗	✗	✗	✓	✓	บทที่ 4.2.3
• มีการปรับปรุงแบบจำลองวัตถุ	✗	✓	✓	✓	✓	บทที่ 4.2.4
6. คำนวณหาจุดจับจากแบบจำลองวัตถุ ที่ผ่านการปรับปรุงจากข้อมูลคีย์เฟรมล่าสุด	✗	✓	✓	✓	✓	-
7. ทำซ้ำข้อ 3-6 จนถึงระยะใกล้สุดที่กำหนด (d_{min}) หรือมีจำนวนคีย์เฟรมเท่ากับ N_{kmax}	✗	✓	✓	✓	✓	บทที่ 4.2.5
8. รับข้อมูลจากกล้องสี นำมาประมาณค่าตำแหน่งหุ่นยนต์ ไปพร้อมกับการเคลื่อนที่ของหุ่นยนต์เข้าสู่ตำแหน่งเป้าหมาย	✗	✗	✗	✗	✓	บทที่ 4.2.6
9. ทำซ้ำข้อ 8 จนกระทั่งถึงตำแหน่งเป้าหมาย หรือเข้าเงื่อนไขที่ให้หยุดการทำงาน	✗	✗	✗	✗	✓	บทที่ 4.2.6
10. เคลื่อนที่เข้าสู่ตำแหน่งเป้าหมายและทำการวางนิ้วลงบนวัตถุ	✓	✓	✓	✓	✓	บทที่ 4.2.5

บทที่ 5
การทดลองการวางแผนการจัดวางปลายนิ้วและผลการทดลอง

วิธีการทั้ง 5 วิธีที่ได้กล่าวไว้ในบทที่ 4 จะถูกนำมาทดลองโดยให้หุ่นยนต์พยายามนำนิ้วไว้วางบนพื้นผิววัตถุตามจุดจับที่คำนวณได้ เปรียบเทียบผลลัพธ์ของทั้ง 5 วิธีทั้งในด้านความถูกต้องในการระบุตำแหน่ง ความถูกต้องในการสร้างแบบจำลองวัตถุ และความถูกต้องในการจัดวางปลายนิ้ว โดยในส่วนสุดท้ายจะทำการทดสอบในเชิงคุณภาพของวิธีการที่ได้นำเสนอ ด้วยภารกิจอย่างง่ายคือการหยิบและวางวัตถุ (Pick and place)

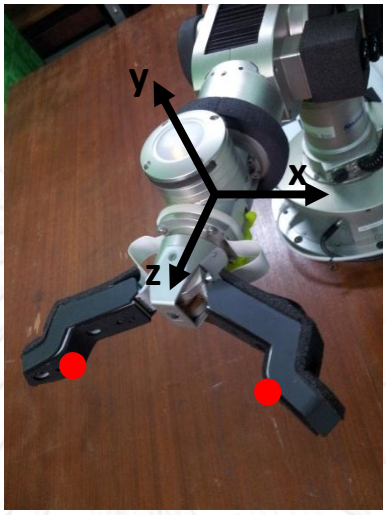
5.1. สภาพแวดล้อมในการทดลอง

5.1.1. อุปกรณ์และพื้นที่การทดลอง

สำหรับการทดลองนี้จะใช้อุปกรณ์รับรู้คือกล้อง DepthSense 325 ติดตั้งบนปลายแขนหุ่นยนต์ Katana 6M180 และพื้นที่ในการทดลองจะเป็นระนาบเรียบไม่มีสิ่งของอื่นขวางหรือบดบังการทำงานของระบบอย่างที่ได้กล่าวไว้แล้วในบทก่อนหน้า สิ่งที่แตกต่างคือในการทดลองนี้จะมีอุปกรณ์รับรู้เพิ่มเติมคือกล้องสีความละเอียดสูง กำหนดให้เป็นกล้องอ้างอิง ซึ่งติดตั้งไว้ด้านข้างของพื้นที่การทำงานดังรูปที่ 5.1 และผ่านการปรับแก้พารามิเตอร์ทั้งภายในและภายนอกเทียบกับ $\{R\}$ แล้ว เพื่อใช้สำหรับหาข้อมูล ground truth ของระบบ โดยกำหนดให้กรอบอ้างอิงของกล้องนี้คือ $\{G\}$

รูปที่ 5.1 การติดตั้งอุปกรณ์และพื้นที่การทดลอง

โดยมือจับหรือ end effector ของหุ่นยนต์ซึ่งมีนิ้วจำนวน 2 นิ้ว มีกรอบอ้างอิงดังรูปที่ 5.2 สำหรับจุดสีแดงในรูปคือบริเวณส่วนโค้งที่จะสัมผัสกับวัตถุในขณะจับวัตถุ โดยกำหนดให้เป็นจุดกึ่งกลางของส่วนโค้งนี้ ซึ่งจากการอ่านค่า encoder ของแขนหุ่นยนต์และพารามิเตอร์ DH ที่ทราบค่าอยู่แล้วจะทำให้เราสามารถคำนวณตำแหน่งของจุดสัมผัสทั้งสองนี้เทียบกับ $\{R\}$ ได้โดยง่าย



รูปที่ 5.2 กรอบอ้างอิงและจุดสัมผัสของมือจับ

เพื่อเป็นการเน้นยำว่าวิธีการทำงานของระบบที่ได้นำเสนอสามารถจัดการกับความไม่แน่นอนที่เกิดขึ้นในสิ่งแวดล้อมได้ เราจะสมมติให้หุ่นยนต์มีความผิดพลาดเกิดขึ้นในส่วนของการควบคุม โดยในขณะที่มีการเพิ่มคีย์เฟรม สัญญาณรับกวนแบบเกาส์เซียน (Gaussian noise) จะถูกใส่เพิ่มให้กับตำแหน่งปัจจุบันของหุ่นยนต์ ประกอบด้วยความผิดพลาดในการเลื่อนและการหมุนที่มีค่าเฉลี่ยเป็น 0 และมีค่าเบี่ยงเบนมาตรฐานเป็น $10 \cdot d_t$ มิลลิเมตรและ $0.5 \cdot d_r$ องศาตามลำดับ โดย d_t และ d_r คือตัวแปรที่กำหนดปริมาณความผิดพลาดที่เกิดขึ้น ซึ่งมีค่าแปรผันตรงตามระยะทางแบบยูคลิดและระยะห่างแบบควอเทอร์เนียนตามลำดับ ของตำแหน่งหุ่นยนต์ปัจจุบันกับตำแหน่งหุ่นยนต์ในคีย์เฟรมก่อนหน้า

5.1.2. วัตถุในการทดลอง

วัตถุที่ใช้นั้นจะเป็นวัตถุแข็งแกร่ง (Rigid body) ที่มีรูปร่างไม่ซับซ้อนและมีลวดลายชัดเจน ซึ่งในการทดลองนี้จะใช้วัตถุ 4 ชนิดคือ กล่อง ทรงกระบอก มุมฉาก และกล่องรูปตัวที ดังรูปที่ 5.2 โดยมีขนาดของด้านที่สั้นที่สุดไม่เกินขนาดความกว้างของมือจับ

รูปที่ 5.3 วัตถุที่ใช้ในการทดลอง

โดยวัตถุทุกชิ้นจะติดกระดานหมากรุกไว้บนระนาบของวัตถุ ที่สามารถมองเห็นได้ชัดเจนโดยกล้องอ้างอิง เพื่อใช้ในการหาข้อมูล ground truth สำหรับการวัดผล

5.2. ขั้นตอนการทดลอง

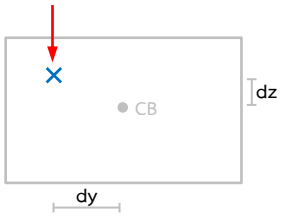
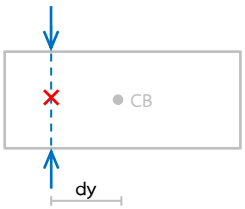
จะทำการทดลองโดยใช้วิธีการทั้ง 5 วิธีที่ได้กล่าวไว้ในบทที่ 4 โดยแต่ละวิธีการนั้นจะทำการทดลองกับวัตถุ 4 ชิ้น แต่ละชิ้นจะทำการทดลองโดยใช้ 2 รูปแบบการจับคือ จับด้านบน (Top-grasp) และจับด้านข้าง (Side-grasp) ซึ่งรูปแบบการจับเหล่านี้จะมีวิธีการคำนวณจุดจับที่แตกต่างกันออกไป โดยจะมีพารามิเตอร์ที่ใช้กำหนดจุดจับของแต่ละรูปแบบดังตารางที่ 5.1 พารามิเตอร์ที่กำหนดไว้ในรูปของสองหลักสุดท้ายของตารางคือระยะห่างของจุดจับในแต่ละแกนเทียบกับจุดศูนย์กลางของกล่องขอบเขต (Center of bounding box) รวมไปถึงมุมของทิศทางการเข้าจับวัตถุเทียบกับแกน z ของวัตถุด้วย โดยการทดลองในรูปแบบการจับด้านบนนั้นจะใช้พารามิเตอร์จุดจับจำนวน 2 แบบโดยทำการทดลองแบบละ 5 ครั้ง ซึ่งใน 5 ครั้งนี้จะกำหนดให้ตำแหน่งเริ่มต้นของหุ่นยนต์และตำแหน่งเริ่มต้นของวัตถุเหมือนกันทั้ง 5 ครั้ง สำหรับการทดลองในรูปแบบการจับข้างจะเหมือนกับการจับด้านบน นั่นคือใช้พารามิเตอร์จุดจับจำนวน 2 แบบแบบละ 5 ครั้ง

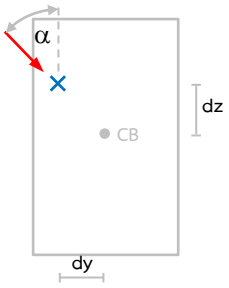
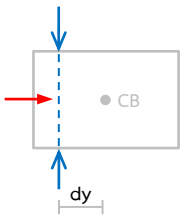
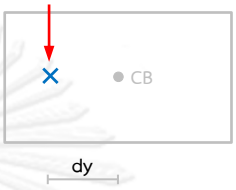
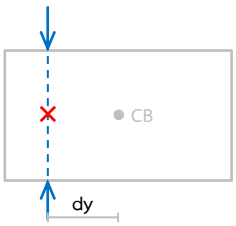
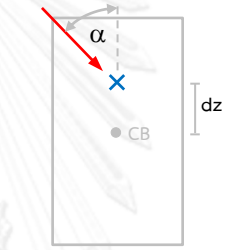
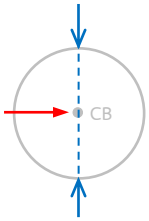
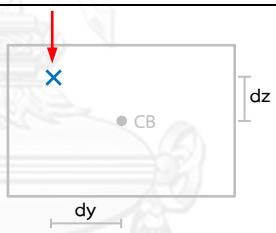
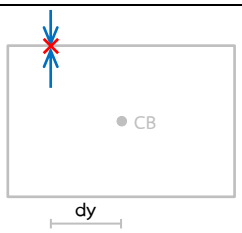
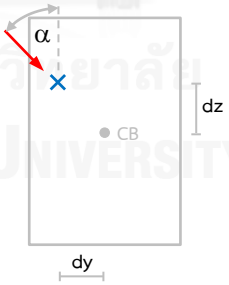
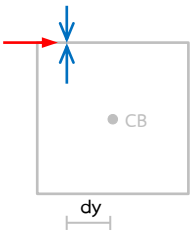
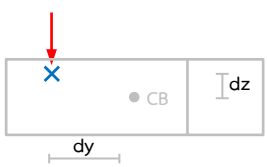
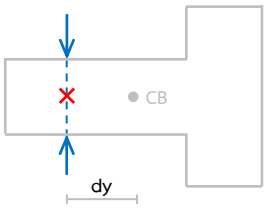
โดยขั้นตอนการทดลองนั้นจะเริ่มโดยนำวัตถุมาวางไว้ยังตำแหน่งเริ่มต้น เคลื่อนที่แขนไปยังตำแหน่งเริ่มต้น จากนั้นทำการเคลื่อนที่แขนหุ่นยนต์เข้าไปวางนิ้วบนวัตถุโดยใช้วิธีการและพารามิเตอร์จุดจับตามที่กำหนด โดยจะบันทึกผลการทดลองเมื่อหุ่นยนต์ถึงตำแหน่งเป้าหมายและทำการวางนิ้วบนวัตถุแล้ว แล้วจึงทำการทดลองในรอบถัดไป ซึ่งผลการวางนิ้วนั้นอาจจะล้มเหลวได้ (รูปที่ 5.4) โดยจะแบ่งความล้มเหลวนี้เป็น 2 แบบคือ เกิดการชน (Collision) ที่เกิดขึ้นในระหว่างการเคลื่อนที่ (ก่อนการบีบนิ้ว) ไม่ว่าจะเกิดการชนกับวัตถุ พื้นโต๊ะ หรือชนกับแขนหุ่นยนต์เอง และเกิดการไถล (Slip) เมื่อทำการบีบนิ้วถ้าวัตถุถูกบีบไถลหลุดออกจากมือจะถือว่าล้มเหลว และจะไม่นำผลการทดลองของการจัดวางปลายนิ้วในครั้งที่เกิดความล้มเหลวนั้นมาประมวลผลรวมกับการทดลองอื่น โดยถ้าเกิดการชนหรือการไถลในขณะบีบนิ้วแล้วยังสามารถวางนิ้วบนวัตถุได้จะไม่ถือว่าเป็นความล้มเหลว

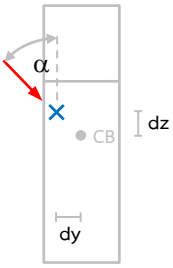
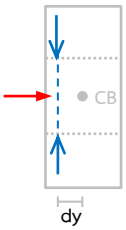
รูปที่ 5.4 ตัวอย่างความล้มเหลวในการวางนิ้ว บน) เกิดการชน ล่าง) เกิดการไถล

การวัดผลการทดลองจะวัดผลโดยเฉลี่ยผลที่ได้จากการทดลองทั้งหมดแยกตามวิธีการและชนิดวัตถุ ซึ่งแบ่งการวัดผลเป็น 3 อย่างคือ ความถูกต้องการระบุตำแหน่งของแขนหุ่นยนต์เทียบกับวัตถุ ความถูกต้องในการสร้างแบบจำลองวัตถุ และความถูกต้องในการวางนิ้ว และสุดท้ายจะเป็นการทดสอบให้เห็นถึงคุณภาพของวิธีการที่นำเสนอ โดยการให้ทำภารกิจหยิบและวางวัตถุ ซึ่งก่อนทำการทดลองจะทำการกำหนดพารามิเตอร์ต่างๆ ดังนี้ $N_{avg} = 5$, $d_{min} = 15$ เซนติเมตร และ $N_{avg} = 6$

ตารางที่ 5.1 พารามิเตอร์ของตำแหน่งและทิศทางการจับวัตถุที่ใช้ในการทดลอง

วัตถุ	**รูปแบบการจับ**	**ลักษณะการวางวัตถุ** 	**ตำแหน่งจุดจับ (สีฟ้า) และทิศทางการจับ (สีแดง)**	
			มุมมองด้านข้าง 	มุมมองด้านบน 
กล่อง	จับด้านบน			

	จับด้านข้าง			
ทรงกระบอก	จับด้านบน			
	จับด้านข้าง			
มุมฉาก	จับด้านบน			
	จับด้านข้าง			
กล่องรูปตัวที	จับด้านบน			

	จับด้านข้าง			

5.3. การวัดผลการทดลองการระบุตำแหน่ง

การวัดผลนี้จะเป็นตัวบ่งชี้ว่าก่อนเข้าทำการจับวัตถุนั้น การระบุตำแหน่งของหุ่นยนต์เทียบกับวัตถุมีความถูกต้องมากน้อยเพียงใด โดยจะใช้ข้อมูลตำแหน่งของหุ่นยนต์เทียบกับวัตถุสุดท้ายที่ประมาณค่าได้จากระบบกำหนดให้เป็น ${}^{O}_{E}T_{last}$ มาเทียบหาความผิดพลาดกับข้อมูล ground truth ที่กำหนดโดย ${}^{O}_{E}T_{gt}$ ซึ่งมีค่าเท่ากับ ${}^{O}_{G}T \cdot {}^{G}_{R}T \cdot {}^{R}_{E}T$ โดยที่ ${}^{O}_{G}T$ คือเมทริกซ์การแปลงระหว่าง $\{G\}$ เทียบกับวัตถุ หาได้โดยกระดานหมากรุกที่ติดอยู่ที่วัตถุ ${}^{G}_{R}T$ ทราบค่าอยู่แล้วจากการปรับแก้พารามิเตอร์ของกล้องอ้างอิง และ ${}^{R}_{E}T$ คือตำแหน่งหุ่นยนต์ปัจจุบันเทียบกับ $\{R\}$ หาค่าได้จาก encoder ของแขนหุ่นยนต์

สำหรับ *วิธีการที่ 1* ถึง *วิธีการที่ 4* จะหา ${}^{O}_{E}T_{last}$ ได้จากข้อมูลคีย์เฟรมสุดท้าย และ *วิธีการที่ 5* จะใช้ข้อมูลสุดท้ายจากการประมาณตำแหน่งหุ่นยนต์โดยใช้กล้องสี (บทที่ 4.2.6) ในการหา ${}^{O}_{E}T_{last}$ โดยความผิดพลาดที่จะวัดผลแบ่งเป็น 2 ค่าคือ ความผิดพลาดของตำแหน่ง (ตารางที่ 5.2) คือระยะทางแบบยูคลิดระหว่าง ${}^{O}_{E}T_{last}$ กับ ${}^{O}_{E}T_{gt}$ และความผิดพลาดของทิศทาง (ตารางที่ 5.3) คือระยะห่างแบบควอเทอร์เนียน (Quaternion distance) ระหว่าง ${}^{O}_{E}T_{last}$ กับ ${}^{O}_{E}T_{gt}$ โดยจะเฉลี่ยผลที่ได้จากการทดลองทั้งหมดแยกตามวิธีการและชนิดวัตถุ ได้ผลการทดลองดังตารางที่ 5.2 และตารางที่ 5.3

ตารางที่ 5.2 ความผิดพลาดของตำแหน่งในการระบุตำแหน่ง

วิธีการ	**ความผิดพลาดของตำแหน่ง (มิลลิเมตร)**			
	กล่อง	ทรงกระบอก	มุมฉาก	กล่องรูปตัวที
วิธีการที่ 1	20.52	24.16	30.35	21.85
วิธีการที่ 2	13.59	14.31	28.56	14.72
วิธีการที่ 3	9.25	9.52	8.98	7.71
วิธีการที่ 4	9.14	9.48	8.91	7.55
วิธีการที่ 5	7.33	8.40	7.35	6.42

ตารางที่ 5.3 ความผิดพลาดของทิศทางในการระบุตำแหน่ง

วิธีการ	ความผิดพลาดของทิศทาง (องศา)			
	กล่อง	ทรงกระบอก	มุมฉาก	กล่องรูปตัวที
วิธีการที่ 1	4.22	5.65	8.82	4.01
วิธีการที่ 2	2.47	3.93	6.93	2.23
วิธีการที่ 3	1.31	1.55	1.04	1.02
วิธีการที่ 4	1.21	1.52	0.95	0.94
วิธีการที่ 5	1.26	1.57	0.72	0.99

จากตารางทั้งสอง *วิธีการที่ 1* ถึง *วิธีการที่ 4* นั้นจะให้ผลการทดลองเป็นไปตามที่คาดไว้ กล่าวคือความผิดพลาดในการระบุตำแหน่งของหุ่นยนต์มีแนวโน้มที่ดีขึ้นตามลำดับ โดยจะสังเกตได้ว่าความผิดพลาดที่ลดลงระหว่าง *วิธีการที่ 3* และ *วิธีการที่ 4* นั้นมีค่าน้อยเมื่อเทียบกับผลต่างระหว่าง *วิธีการที่ 1* กับ *วิธีการที่ 2* หรือผลต่างระหว่าง *วิธีการที่ 2* กับ *วิธีการที่ 3* ซึ่งมีสาเหตุมาจากวิธีการที่ได้นำเสนอนั้นจะทำการเคลื่อนแขนหุ่นยนต์เข้าตรงหาวัตถุโดยไม่มีการสำรวจใดๆ ทำให้ข้อมูลที่ได้มีลักษณะคล้ายเดิม ผลลัพธ์ของการหาค่าเหมาะสมที่สุดจึงถูกจำกัดไว้ด้วยข้อมูลเหล่านี้

สำหรับ *วิธีการที่ 5* ในขั้นตอนการประมาณตำแหน่งของหุ่นยนต์ด้วยกล้องสีนั้น จะประมาณตำแหน่งในสามมิติของหุ่นยนต์ด้วยข้อมูลในสองมิติ ซึ่งข้อมูลเหล่านี้มีข้อมูลที่ผิดปกติปนอยู่มากทั้งข้อมูลจากการฉายภาพที่ผิด และข้อมูลที่ผิดพลาดจากการติดตามตำแหน่งด้วย optical flow ทำให้ตำแหน่งของหุ่นยนต์ที่ประมาณได้นั้นมีโอกาสที่จะผิดพลาดมากขึ้นตามปริมาณของข้อมูลที่ผิดปกติเหล่านี้ จึงอาจเป็นสาเหตุให้ความผิดพลาดของทิศทางในการระบุตำแหน่งในตารางข้างต้น ที่บางชนิดของวัตถุมีแนวโน้มของความผิดพลาดที่เพิ่มขึ้น แต่ผลต่างของความผิดพลาดเหล่านี้ระหว่าง *วิธีการที่ 4* กับ *วิธีการที่ 5* ถือว่ามีค่าไม่สูงเมื่อเทียบกับผลต่างในวิธีการอื่นๆ อีกทั้งผลต่างความผิดพลาดของตำแหน่งระหว่าง *วิธีการที่ 5* กับ *วิธีการที่ 4* เมื่อเทียบกับผลต่างความผิดพลาดของ *วิธีการที่ 3* กับ *วิธีการที่ 4* มีค่าดีขึ้นอย่างเห็นได้ชัด

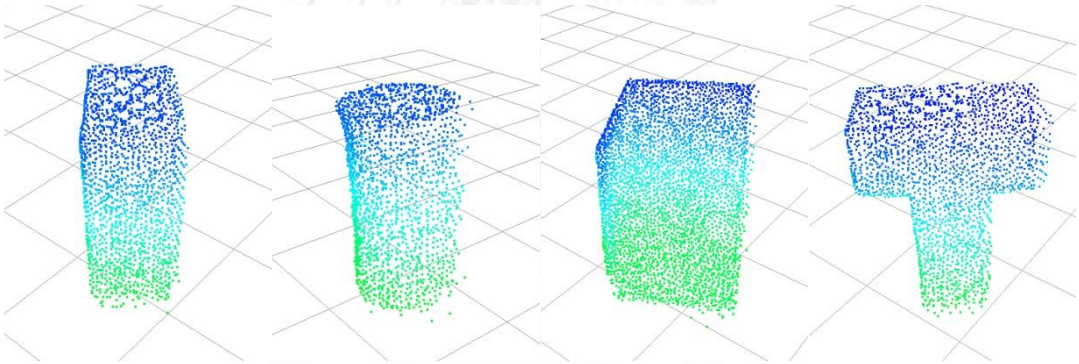
5.4. การวัดผลการทดลองการสร้างแบบจำลองวัตถุ

ในส่วนนี้แบบจำลองของวัตถุ หรือนั่นคือกล่องขอบเขต (Bounding box) จะถูกนำมาใช้เพื่อเปรียบเทียบความถูกต้องของแบบจำลองวัตถุที่สร้างขึ้น ซึ่งจะนำกล่องขอบเขตที่หาได้ในคีย์เฟรมสุดท้ายมาเปรียบเทียบกับกล่องขอบเขตของวัตถุจริงที่มาจากการวัดค่าด้วยมือ โดยนำกล่องขอบเขตทั้งสองมาวางซ้อนทับกันให้ตำแหน่งจุดศูนย์กลางของกล่องและทิศทางเหมือนกัน จากนั้นจึงหาความผิดพลาดของแบบจำลองวัตถุในรูปแบบของร้อยละ ซึ่งมีค่าเท่ากับร้อยละของ ปริมาตรส่วนที่ไม่มีการ

ซ้อนทับกันของกล่องขอบเขตทั้งสอง เทียบกับปริมาตรส่วนที่ซ้อนทับกัน ได้ผลการทดลองดังตารางที่ 5.4 และผลลัพธ์แบบจำลองวัตถุของวิธีการที่นำเสนอ (*วิธีการที่ 5*) แสดงให้เห็นดังรูปที่ 5.5

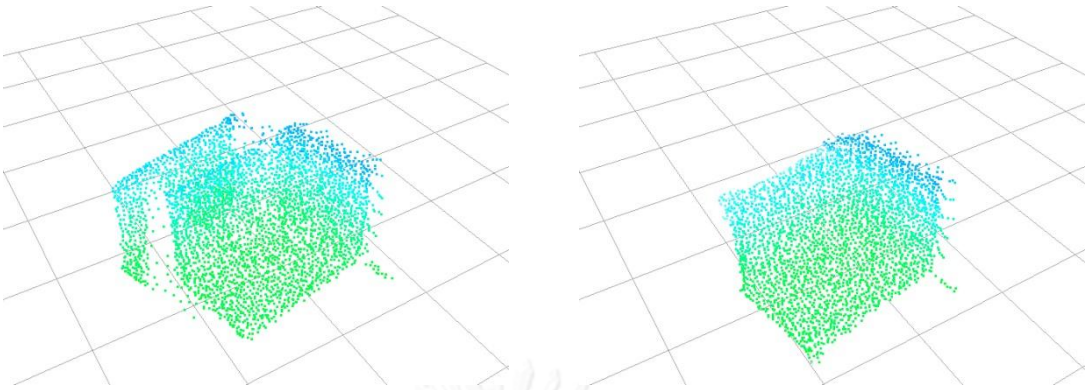
ตารางที่ 5.4 ความผิดพลาดในการสร้างแบบจำลองวัตถุ

วิธีการ	**ความผิดพลาดของแบบจำลองวัตถุ (ร้อยละ)**			
	กล่อง	ทรงกระบอก	มุมฉาก	กล่องรูปตัวที
วิธีการที่ 1	52.72	57.76	62.83	58.53
วิธีการที่ 2	39.73	50.39	60.61	45.75
วิธีการที่ 3	23.79	29.45	21.40	17.53
วิธีการที่ 4 และ *วิธีการที่ 5*	18.47	20.34	12.87	16.91



รูปที่ 5.5 ผลลัพธ์แบบจำลองวัตถุโดยวิธีการที่นำเสนอ

สำหรับ *วิธีการที่ 4* และ *วิธีการที่ 5* นั้นมีขั้นตอนการทำงานในส่วนของการปรับปรุงแบบจำลองวัตถุเหมือนกันทุกประการ ผลการทดลองในตารางนี้จึงรวมผลลัพธ์ของทั้งสองวิธีนี้ไว้ด้วยกัน โดยภาพรวมแล้วความผิดพลาดของแบบจำลองวัตถุมีแนวโน้มลดลงตามวิธีการ ตามที่คาดไว้ และสังเกตได้ว่าผลต่างของความผิดพลาดระหว่าง *วิธีการที่ 2* กับ *วิธีการที่ 3* มีค่าลดลงมากอย่างเห็นได้ชัดเมื่อเทียบกับผลต่างความผิดพลาด ระหว่าง *วิธีการที่ 2* กับ *วิธีการที่ 1* เนื่องจากการรวมข้อมูล point cloud ของแบบจำลองวัตถุโดยไม่ผ่านการปรับปรุงตำแหน่งของกล้องก่อนนั้นจะส่งผลให้แบบจำลองวัตถุมีความถูกต้องน้อยลงตามปริมาณความผิดพลาดของตำแหน่งกล้องดังรูปที่ 5.6 ซ้าย (*วิธีการที่ 2*) โดยเปรียบเทียบกับรูปที่ 5.6 ขวา (*วิธีการที่ 3*) ที่มีการปรับปรุงตำแหน่งของกล้องก่อนที่จะมีการรวมข้อมูลแบบจำลองวัตถุ จึงเป็นสาเหตุให้เกิดผลลัพธ์ความผิดพลาดดังตารางข้างต้น



รูปที่ 5.6 ผลลัพธ์แบบจำลองวัตถุ มุมฉาก ซ้าย) วิธีการที่ 2 ขวา) วิธีการที่ 3

5.5. การวัดผลผลการทดลองการจัดวางปลายนิ้ว

ภายหลังการวางนิ้วบนวัตถุแล้ว จะทำการวัดผลความผิดพลาดโดยคำนวณหาตำแหน่งของจุดสัมผัสของมือจับทั้งสองจุด (รูปที่ 5.2) เทียบกับวัตถุ โดยตำแหน่งวัตถุนั้นอาจเกิดการเปลี่ยนตำแหน่งจากการบีบนิ้วของมือจับ (รูปที่ 5.7) จึงต้องทำการคำนวณตำแหน่งวัตถุใหม่โดยใช้กล้องอ้างอิงตรวจหากระดานหมากรุกบนวัตถุ จากนั้นจึงนำตำแหน่งในสามมิติของจุดสัมผัสทั้งสองจุดมาหาความผิดพลาดเทียบกับตำแหน่งของจุดจับเป้าหมายที่หาจากพารามิเตอร์จุดจับของการทดลองนั้นๆ (ตารางที่ 5.1) ความผิดพลาดที่ได้จะเกิดจากการเฉลี่ยกันของความผิดพลาดของจุดสัมผัสทั้งสอง ตารางที่ 5.5 แสดงถึงความผิดพลาดของตำแหน่งในการวางนิ้วโดยใช้ระยะทางแบบยูคลิดในการวัดผล สำหรับความผิดพลาดในทิศทางนั้นจะหามุมระหว่างแกน z ของมือจับและเวกเตอร์ทิศทางการเข้าจับเป้าหมาย (หาได้จากพารามิเตอร์จุดจับ) ซึ่งผลลัพธ์ความผิดพลาดของทิศทางแสดงดังตารางที่ 5.6 โดยหลักที่เป็นสีเทาของทั้งสองตารางนี้คือ จำนวนครั้งที่ทำการวางนิ้วบนวัตถุสำเร็จ กล่าวคือไม่เกิดความล้มเหลวแบบการชนและการไถล

ตารางที่ 5.5 ความผิดพลาดของตำแหน่งในการจัดวางปลายนิ้ว

วิธีการ	**ความผิดพลาดของตำแหน่ง (มิลลิเมตร)**							
	กล่อง		ทรงกระบอก		มุมฉาก		กล่องรูปตัวที	
วิธีการที่ 1	10	20.12	8	27.03	15	39.36	9	22.28
วิธีการที่ 2	16	13.70	14	19.52	18	29.43	14	16.86
วิธีการที่ 3	20	8.54	19	9.73	20	8.27	20	9.51
วิธีการที่ 4	20	8.04	20	9.13	20	8.07	20	8.79
วิธีการที่ 5	20	6.01	20	7.04	20	5.84	20	6.39

ตารางที่ 5.6 ความผิดพลาดของทิศทางในการจัดวางปลายนิ้ว

วิธีการ	ความผิดพลาดของทิศทาง (องศา)							
	กล่อง		ทรงกระบอก		มุมฉาก		กล่องรูปตัวที	
วิธีการที่ 1	10	5.14	8	5.94	15	4.92	9	3.20
วิธีการที่ 2	16	3.58	14	4.12	18	3.79	14	1.85
วิธีการที่ 3	20	2.40	19	2.49	20	1.65	20	1.06
วิธีการที่ 4	20	2.31	20	2.49	20	1.49	20	1.05
วิธีการที่ 5	20	2.38	20	2.52	20	1.45	20	1.10

รูปที่ 5.7 ตัวอย่างการเลื่อนตำแหน่งของวัตถุในขณะการวางนิ้วของมือจับ

ผลความผิดพลาดในตารางที่ 5.5 และตารางที่ 5.6 มีแนวโน้มตามที่คาดไว้ กล่าวคือมีความผิดพลาดทั้งตำแหน่งและทิศทางลดลงตามวิธีการที่ใช้ รวมถึงอัตราความสำเร็จในการวางนิ้วที่มีค่าเพิ่มขึ้นด้วย สำหรับความผิดพลาดของทิศทางใน *วิธีการที่ 5* นั้นมีแนวโน้มเพิ่มขึ้นในบางชนิดวัตถุซึ่งอาจมีสาเหตุมาจากข้อมูลที่ผิดปกติในประมาณตำแหน่งหุ่นยนต์ด้วยกล้องสี ตามที่ได้อธิบายไว้แล้วในบทที่ 5.3

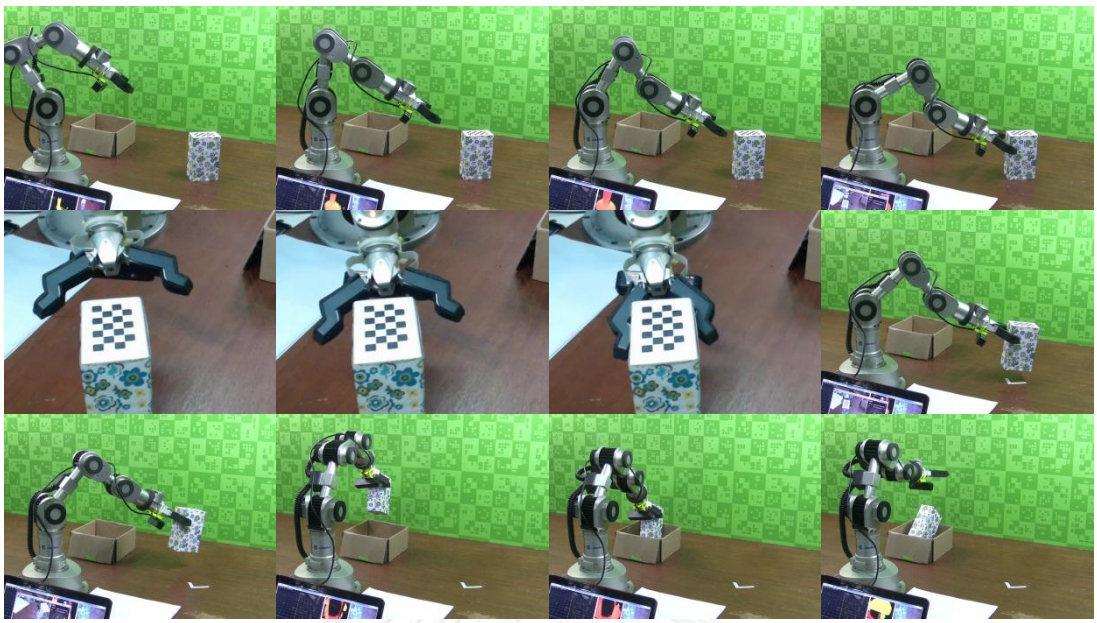
สำหรับวัตถุทรงกระบอกนั้นจะมีผลรวมจำนวนครั้งที่ทำการวางนิ้วสำเร็จน้อยที่สุด เนื่องจากพื้นผิวของทรงกระบอกมีลักษณะเป็นส่วนโค้ง ความผิดพลาดของตำแหน่งจุดจับเพียงเล็กน้อย จะทำให้เวกเตอร์ปกติของจุดสัมผัสเปลี่ยนทิศทาง ส่งผลให้เกิดแรงผลักวัตถุออกจากจุดจับที่ต้องการ ซึ่งความล้มเหลวของการวางนิ้วบนวัตถุทรงกระบอกนี้ จะเป็นความล้มเหลวแบบการไถลเป็นส่วนใหญ่ สำหรับวัตถุที่มีจำนวนครั้งที่ทำการวางนิ้วสำเร็จมากที่สุดคือวัตถุมุมฉาก เนื่องด้วยวัตถุชนิดนี้มี

ลักษณะบริเวณจุดจับเป็นระนาบบาง โอกาสการชนกันระหว่างมือจับและวัตถุจึงมีน้อย อีกทั้งระนาบนี้ยังมีพื้นที่มากทำให้ในการวางนิ้วจึงมีโอกาสมากที่จะวางนิ้วสำเร็จ โดยความล้มเหลวส่วนใหญ่ที่เกิดกับวัตถุมุมฉาก กล่อง และกล่องรูปตัวทีนั้นจะเป็นความล้มเหลวแบบการชน

5.6. การทดสอบการจับวัตถุ

การทดสอบนี้จะให้หุ่นยนต์ทำภารกิจหยิบวัตถุและวางลงในกล่องที่เตรียมไว้ โดยจะทำภารกิจนี้กับวัตถุทั้ง 4 ชนิดโดยใช้วิธีการที่นำเสนอ (*วิธีการที่ 5*) เพื่อทดสอบคุณภาพการใช้งานจริงของระบบและวิธีที่นำเสนอ ผลการทดสอบแสดงดังรูปที่ 5.8 – 5.11 คือลำดับภาพของการทำภารกิจนี้ ซึ่งสามารถหยิบและวางวัตถุทั้ง 4 ชนิดได้สำเร็จอย่างราบรื่น

รูปที่ 5.8 ลำดับภาพการทดสอบหยิบและวางวัตถุกล่อง (จากซ้ายไปขวา บนลงล่าง)



รูปที่ 5.9 ลำดับภาพการทดสอบหยิบและวางวัตถุทรงกระบอก (จากซ้ายไปขวา บนลงล่าง)

รูปที่ 5.10 ลำดับภาพการทดสอบหยิบและวางวัตถุมุมฉาก (จากซ้ายไปขวา บนลงล่าง)

รูปที่ 5.11 ลำดับภาพการทดสอบหยิบและวางวัตถุกล่องรูปตัวที (จากซ้ายไปขวา บนลงล่าง)

บทที่ 6
สรุปการวิจัยและแนวทางการวิจัยในขั้นถัดไป

6.1. สรุปการวิจัย

จากผลการทดลองข้างต้น ได้แสดงผลการทดลองในรูปแบบของความผิดพลาดในการระบุตำแหน่ง ความผิดพลาดของแบบจำลองวัตถุ และความผิดพลาดในการจัดวางปลายนิ้ว ซึ่งในการทดลองได้ใช้วัตถุหลากหลายรูปแบบทั้ง วัตถุรูปร่างเรียบง่าย (กล่อง) วัตถุที่มีพื้นผิวโค้ง (ทรงกระบอก) วัตถุที่มีความบาง (มุมฉาก) และวัตถุที่มีเหลี่ยมมุมมาก (กล่องรูปตัวที) อีกทั้งยังมีการกำหนดวิธีการอื่นๆ เพื่อใช้สำหรับเปรียบเทียบผลลัพธ์กับวิธีที่นำเสนอ สำหรับลักษณะการใช้ข้อมูลของ *วิธีการที่ 1* ถึง *วิธีการที่ 5* นั้นมีการใช้ประโยชน์จากข้อมูลจากน้อยไปมากตามลำดับ ซึ่งผลการทดลองนั้นได้แสดงให้เห็นชัดเจนว่าปริมาณการใช้ประโยชน์จากข้อมูล จะส่งผลต่อความถูกต้องของระบบ

สำหรับวิธีการที่นำเสนอนั้น ความถูกต้องในการจัดวางปลายนิ้วจะขึ้นอยู่กับความผิดพลาดที่เกิดขึ้นในระบบ ตั้งแต่ความถูกต้องของการปรับแก้พารามิเตอร์ภายในและภายนอกของอุปกรณ์รับรู้ ความถูกต้องของข้อมูลที่รับมาจากอุปกรณ์รับรู้ ความผิดพลาดในการระบุตำแหน่ง และความผิดพลาดในการสร้างแบบจำลองวัตถุ ซึ่งจากการผลทดลองที่ได้และการสังเกตของผู้ทำวิจัย สามารถวิเคราะห์และสรุปปัจจัยที่เป็นสาเหตุของความผิดพลาดเหล่านี้ ได้ดังต่อไปนี้

ในด้านของข้อมูลที่รับจากอุปกรณ์รับรู้นั่นคือ DepthSense 325 จากการสังเกตจะพบว่าอุปกรณ์รับรู้ชนิดนี้จะให้ลักษณะของข้อมูลไม่เหมือนกัน ในแต่ละชนิดและสีของพื้นผิวของวัสดุที่แตกต่างกัน จึงจำเป็นที่จะต้องควบคุมชนิดของวัตถุที่ใช้ในงานวิจัยนี้ อีกทั้งข้อมูลที่ได้มีปริมาณของสิ่งรบกวนมาก ทำให้เกิดข้อจำกัดในประสิทธิภาพของการทำงาน ที่ต้องใช้เวลาเพิ่มขึ้นเพื่อเฉลี่ยข้อมูลสำหรับการจัดการกับสิ่งรบกวนเหล่านี้

การปรับปรุงตำแหน่งของหุ่นยนต์ จะอาศัยข้อมูล point cloud จากวัตถุมาหาความสัมพันธ์กับข้อมูลในอดีตด้วยวิธี ICP ซึ่งถ้าวัตถุมีรูปร่างลักษณะที่เรียบง่ายมีเหลี่ยมและมุมน้อย ก็จะส่งผลให้วิธีการนี้มีความผิดพลาดสูง แต่ถ้าใช้วัตถุที่มีความซับซ้อนมากเกินไป อาจเกิดการบังกันของวัตถุ หรือนั่นคือข้อมูลที่ได้ไม่สามารถอธิบายรูปร่างส่วนที่สำคัญของวัตถุได้ครบ ก็จะส่งผลต่อความผิดพลาดในตำแหน่งของหุ่นยนต์เช่นกัน

ลวดลายของวัตถุจะเป็นสิ่งสำคัญในขั้นตอนการประมาณตำแหน่งหุ่นยนต์ด้วยกล้องสี ซึ่งลวดลายที่มี feature น้อยและไม่เด่นชัดจะส่งผลให้การตรวจพบและติดตาม feature เหล่านี้มีความผิดพลาดสูง ก่อให้เกิดเป็นข้อมูลที่ผิดปกติ โดยการเลือกวิธีที่จะนำข้อมูลเหล่านี้ไปใช้จะสามารถช่วยให้ลดความผิดพลาดที่จะเกิดขึ้นได้ อาทิเช่น การใช้วิธี RANSAC ในงานวิจัยนี้ ที่เป็นการหลีกเลี่ยงการนำข้อมูลที่ผิดปกติเหล่านี้ไปใช้

ในงานวิจัยนี้จะกำหนดให้หุ่นยนต์เคลื่อนที่เป็นเส้นตรงไปยังตำแหน่งเป้าหมาย โดยไม่ได้วางแผนการเคลื่อนที่ของแขนหุ่นยนต์เพื่อสำรวจในมุมมองอื่นๆ ที่จะพบข้อมูลใหม่ของวัตถุ ด้วยเหตุนี้

ตำแหน่งเริ่มต้นของหุ่นยนต์และตำแหน่งการวางของวัตถุ จึงมีผลต่อข้อมูลที่จะได้รับอย่างมาก ซึ่งถ้าเรากำหนดตำแหน่งเหล่านี้ไม่ดี ตลอดการเคลื่อนที่ของแขนหุ่นยนต์นั้น จะไม่เห็นข้อมูลที่สำคัญของวัตถุ ส่งผลให้เกิดความผิดพลาดสูง และการวางนิ้วบนวัตถุจะล้มเหลวในที่สุด อีกทั้งในการหาค่าเหมาะสมที่สุดของกราฟตำแหน่งหุ่นยนต์ จะถูกจำกัดขีดความสามารถในการปรับปรุงตำแหน่งหุ่นยนต์จากข้อมูลที่มีอยู่อย่างจำกัดนี้

และประการสุดท้ายคือ ข้อจำกัดของแขนหุ่นยนต์ โดยในรูปแบบการจับวัตถุแบบด้านข้าง แขนหุ่นยนต์ไม่สามารถเปลี่ยนทิศทางการหมุนในแกน yaw เทียบกับวัตถุได้ ทำให้การวางวัตถุต้องวางให้มีทิศทางเดียวกับทิศทางการเข้าจับของแขนหุ่นยนต์นี้ ซึ่งในความเป็นจริงนั้นการกำหนดทิศทางนี้ทำได้เพียงการประมาณ จึงทำให้วัตถุถูกเลื่อนตำแหน่งหรือเปลี่ยนทิศทาง จากการบีบนิ้วของแขนหุ่นยนต์

โดยสรุปแล้ว วิทยานิพนธ์ฉบับนี้ได้นำเสนอแนวคิดใหม่เป็นกรอบงานสำหรับการจับวัตถุที่ไม่รู้จัก โดยจะอาศัยลักษณะการเคลื่อนที่ทีละก้าว เพื่อเก็บข้อมูลของวัตถุในขณะการเคลื่อนที่ มาใช้ในการปรับปรุงตำแหน่งของหุ่นยนต์ไปพร้อมกับแบบจำลองของวัตถุ เพื่อลดความผิดพลาดในการวางปลายนิ้วลงบนจุดจับที่ต้องการ ซึ่งเป็นสาเหตุสำคัญที่ทำให้การจับวัตถุล้มเหลว ซึ่งกรอบงานที่นำเสนอนี้ได้รวมขั้นตอนของการจับวัตถุเอาไว้ทั้งหมด ทั้งการเก็บข้อมูล การระบุตำแหน่งหุ่นยนต์ สร้างแบบจำลองวัตถุ คำนวณจุดจับ และการวางแผนการเคลื่อนที่ โดยในแต่ละขั้นตอนเหล่านี้สามารถนำงานวิจัยอื่นเข้ามาประยุกต์ใช้ได้ อาทิเช่น ขั้นตอนการหาจุดจับสามารถเปลี่ยนไปใช้งานวิจัยในการคำนวณหาจุดจับอื่นๆ [1-5] โดยเปลี่ยนลักษณะของข้อมูลวัตถุไปตามข้อมูลนำเข้าของงานวิจัยนั้นๆ ในการเคลื่อนที่ของหุ่นยนต์นั้นก็สามารถวางแผนกำหนดเส้นทางการเคลื่อนที่เพื่อหลบหลีกสิ่งกีดขวาง [15, 16] และสามารถกำหนดมุมมองถัดไปของการเก็บข้อมูลเพื่อให้ได้ข้อมูลสำคัญของวัตถุด้วยงานวิจัยประเภท Next Best View Planning [27] เป็นต้น

โดยความแม่นยำที่เพิ่มขึ้นในการจัดวางปลายนิ้วบนจุดจับที่ต้องการ ที่ได้จากวิธีการในงานวิจัยนี้เป็นเพียงปัจจัยหนึ่งที่จะช่วยพัฒนากระบวนการจับวัตถุให้ดีขึ้น ซึ่งยังคงมีอีกหลากหลายปัจจัยอาทิเช่น การคำนวณหาจุดจับที่เหมาะสมจากข้อมูลเพียงบางส่วนของวัตถุ และกระบวนการในการสำรวจหาข้อมูลใหม่ๆของวัตถุ เป็นต้น ซึ่งโดยภาพรวมแล้วงานวิจัยนี้จะสามารถนำไปประยุกต์ใช้ร่วมกับงานวิจัยในด้านการจับวัตถุอื่นๆ ที่จะเป็นส่วนที่ช่วยเติมเต็มเพื่อให้กระบวนการจับวัตถุที่ไม่รู้จักนั้นสมบูรณ์มากยิ่งขึ้น

6.2. แนวทางการวิจัยในขั้นถัดไป

สำหรับประเด็นสำคัญที่จะพิจารณาในงานวิจัยขั้นถัดไปนั้นคือ การวางแผนการเคลื่อนที่ของแขนหุ่นยนต์ ซึ่งในงานวิจัยนี้ไม่มีการวางแผนในส่วนนี้ ทำให้การรับรู้ข้อมูลในระบบถูกจำกัดไว้ การวางแผนการเคลื่อนที่ของแขนหุ่นยนต์เพื่อสำรวจในมุมมองที่คาดว่าจะพบข้อมูลที่สำคัญของวัตถุ จึงสามารถช่วยเพิ่มความถูกต้องในการจัดวางปลายนิ้วได้มากขึ้น โดยสามารถมองเป็นปัญหา Next Best

View Planning [27] ที่จะใช้ข้อมูลของวัตถุที่มีอยู่ในมาพิจารณาว่า ควรจะเปลี่ยนมุมมองไปอย่างไร เพื่อให้สามารถเก็บข้อมูลของวัตถุได้ดีที่สุด

ในวิทยานิพนธ์นี้จะมีลักษณะการทำงานของระบบ เคลื่อนที่ไปยังตำแหน่งเป้าหมายที่คำนวณมาได้ เมื่อเข้าถึงตำแหน่งนั้นจะทำการจับวัตถุทันที โดยไม่มีการประเมินคุณภาพของการจับนั้นๆ กล่าวคือ ถ้าเราสามารถประเมินคุณภาพของตำแหน่งจุดจับที่หามาได้นั้น เราจะสามารถปรับเปลี่ยนกลยุทธ์ในการจับวัตถุตามสถานการณ์ได้ อาทิเช่น ถ้าได้ตำแหน่งจุดจับที่มีคุณภาพดีพอแล้ว อาจจะทำการเข้าจับวัตถุเลย หรือลดความถี่ในการเก็บข้อมูลลง เป็นต้น ซึ่งการประเมินคุณภาพนี้จะสามารถบอกได้ถึงความเป็นไปได้ที่จะจับวัตถุได้สำเร็จ ที่เป็นตัวชี้วัดหนึ่งที่สำคัญในการบอกให้ระบบรู้ว่า ณ ขณะนี้คุณภาพของตำแหน่งจุดจับที่ได้เป็นอย่างไร ต้องแก้ไขอย่างไรให้ดีขึ้น และถ้าเข้าจับวัตถุตอนนี้จะมีโอกาสสำเร็จมากน้อยเพียงใด ซึ่งถือว่าเป็นประโยชน์อย่างยิ่งในปัญหาการจับวัตถุที่ไม่รู้จัก จึงควรพิจารณาประเด็นนี้ในการวิจัยขั้นถัดไป

ในส่วนของการปรับปรุงตำแหน่งหุ่นยนต์โดยใช้ข้อมูลความลึกนั้น ยังคงใช้ประโยชน์จากข้อมูลไม่เต็มประสิทธิภาพ กล่าวคือสามารถนำข้อมูลจากภาพสีมาใช้ในขั้นตอนนี้ เพื่อช่วยเพิ่มความถูกต้องในการหาความสัมพันธ์เทียบกับข้อมูลในอดีตได้ อีกทั้งข้อมูลภาพสีนี้ยังสามารถอธิบายรูปร่างหรือคุณลักษณะสำคัญของวัตถุได้ ที่สามารถนำไปใช้ประโยชน์ในการสร้างแบบจำลองวัตถุให้ดียิ่งขึ้น

ในแง่ของการใช้งานจริงนั้น วิธีการในงานวิจัยที่นำเสนอนั้น ยังคงมีข้อจำกัดอยู่หลายประการที่ทำให้การนำมาใช้งานจริงนั้นเป็นไปได้ยาก เช่น ข้อจำกัดในจำนวนและรูปร่างของวัตถุ และประสิทธิภาพเชิงเวลาของการทำงาน เป็นต้น ดังนั้นงานวิจัยในขั้นถัดไปจะพิจารณาประเด็นต่างๆ ที่ได้กล่าวมาในข้างต้น รวมไปถึงพัฒนาให้สามารถใช้งานได้กับวัตถุที่มีอยู่ทั่วไป และคำนึงถึงประสิทธิภาพเชิงเวลาสำหรับการใช้งานจริงด้วย

รายการอ้างอิง

1. Miller, A.T., et al. *Automatic grasp planning using shape primitives*. in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA '03*. 2003.
2. Bohg, J., et al., *Data-Driven Grasp Synthesis—A Survey.* IEEE Transactions on Robotics, 2013. **Early Access Online**.
3. Huebner, K., S. Ruthotto, and D. Kragic. *Minimum volume bounding box decomposition for shape approximation in robot grasping*. in *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008*. 2008.
4. Felip, J. and A. Morales. *Robust sensor-based grasp primitive for a three-finger robot hand*. in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*. 2009.
5. Speth, J., A. Morales, and P.J. Sanz. *Vision-based grasp planning of 3D objects by extending 2D contour based algorithms*. in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*. 2008.
6. Walck, G. and M. Drouin. *Automatic observation for 3D reconstruction of unknown objects using visual servoing*. in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2010.
7. Krainin, M., et al., *Manipulator and object tracking for in-hand 3D object modeling.* The International Journal of Robotics Research, 2011. **30**(11): p. 1311-1327.
8. Chari, V., et al. *Convex bricks: A new primitive for visual hull modeling and reconstruction*. in *2012 IEEE International Conference on Robotics and Automation (ICRA)*. 2012.
9. Feng, Y., Y. Wu, and L. Fan. *On-line Object Reconstruction and Tracking for 3D Interaction*. in *2012 IEEE International Conference on Multimedia and Expo (ICME)*. 2012.
10. Ilonen, J., J. Bohg, and V. Kyrki. *Fusing visual and tactile sensing for 3-D object reconstruction while grasping*. in *2013 IEEE International Conference on Robotics and Automation (ICRA)*. 2013.
11. Yang, Y. and Q.-X. Cao, *Monocular vision based 6D object localization for service robot's intelligent grasping.* Computers & Mathematics with Applications, 2012. **64**(5): p. 1235-1241.
12. Asif, U., M. Bennamoun, and F. Sohel. *Real-time pose estimation of rigid objects using RGB-D imagery*. in *2013 8th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 2013.

13.Chen, C.-H. and H.-P. Huang, *Pose estimation for autonomous grasping with a robotic arm system.* Journal of the Chinese Institute of Engineers, 2013. **36**(5): p. 638-646.
14.Lysenkov, I. and V. Rabaud. *Pose estimation of rigid transparent objects in transparent clutter*. in *2013 IEEE International Conference on Robotics and Automation (ICRA)*. 2013.
15.Vahrenkamp, N., et al. *Integrated Grasp and motion planning*. in *2010 IEEE International Conference on Robotics and Automation (ICRA)*. 2010.
16.Gratal, X., et al., *Visual servoing on unknown objects.* Mechatronics, 2012. **22**(4): p. 423-435.
17.Hsiao, K., et al. *Contact-reactive grasping of objects with partial shape information*. in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2010.
18.Leeper, A., et al., *Using Near-Field Stereo Vision for Robotic Grasping in Cluttered Environments*, in *Experimental Robotics*, O. Khatib, V. Kumar, and G. Sukhatme, Editors. 2014, Springer Berlin Heidelberg. p. 253-267.
19.Saxena, A., J. Driemeyer, and A.Y. Ng, *Robotic Grasping of Novel Objects using Vision.* The International Journal of Robotics Research, 2008. **27**(2): p. 157-173.
20.Calli, B., M. Wisse, and P. Jonker. *Grasping of unknown objects via curvature maximization using active vision*. in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011.
21.Lippiello, V., et al., *Visual Grasp Planning for Unknown Objects Using a Multifingered Robotic Hand.* IEEE/ASME Transactions on Mechatronics, 2013. **18**(3): p. 1050-1059.
22.Nagata, K., et al. *Picking up an indicated object in a complex environment*. in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2010.
23.Kootstra, G., et al., *Enabling grasping of unknown objects through a synergistic use of edge and surface information.* The International Journal of Robotics Research, 2012. **31**(10): p. 1190-1213.
24.Rasolzadeh, B., et al., *An Active Vision System for Detecting, Fixating and Manipulating Objects in the Real World.* The International Journal of Robotics Research, 2010. **29**(2-3): p. 133-154.
25.Popovic, M., et al. *Grasping unknown objects using an Early Cognitive Vision system for general scene understanding*. in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011.
26.Jung, J., et al. *A novel 2.5D pattern for extrinsic calibration of tof and camera fusion system*. in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011.

27.Kriegel, S., et al. *A surface-based Next-Best-View approach for automated 3D model completion of unknown objects*. in *2011 IEEE International Conference on Robotics and Automation (ICRA)*. 2011.

28.Ruhnke, M., et al. *Range sensor based model construction by sparse surface adjustment*. in *2011 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*. 2011.

29.Fanfani, M. and C. Colombo, *LaserGun: A Tool for Hybrid 3D Reconstruction*, in *Computer Vision Systems*, M. Chen, B. Leibe, and B. Neumann, Editors. 2013, Springer Berlin Heidelberg. p. 274-283.

30.Bimbo, J., et al. *Object pose estimation and tracking by fusing visual and tactile information*. in *2012 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. 2012.

31.May, S., et al., *Three-dimensional Mapping with Time-of-flight Cameras.* J. Field Robot., 2009. **26**(11\-12): p. 934-965.

32.Foix, S., G. Alenya, and C. Torras, *Lock-in Time-of-Flight (ToF) Cameras: A Survey.* IEEE Sensors Journal, 2011. **11**(9): p. 1917-1926.

33.Bone, G.M., A. Lambert, and M. Edwards. *Automated modeling and robotic grasping of unknown three-dimensional objects*. in *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008*. 2008.

34.Lippiello, V., F. Ruggiero, and B. Siciliano, *Floating Visual Grasp of Unknown Objects Using an Elastic Reconstruction Surface*, in *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Editors. 2011, Springer Berlin Heidelberg. p. 329-344.

35.*ICRA 2011 Workshop on Manipulation Under Uncertainty | Willow Garage*. 2014/02/03/05:50:00; Available from: http://www.willowgarage.com/muu11.

36.Zhao, G., Y. Jia, and Z. Ou. *Construction of vision-based manipulation system for 3D industrial objects*. in *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 2009.

37.Chang, L.Y. and N. Pollard. *Posture optimization for pre-grasp interaction planning*. in *Proc. of the Workshop on Manipulation under Uncertainty at the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2011.

38.Hess, J., J. Sturm, and W. Burgard. *Learning the State Transition Model to Efficiently Clean Surfaces with Mobile Manipulation Robots*. in *Proc. of the Workshop on Manipulation under Uncertainty at the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2011.

39.Tellex, S., et al. *Interpreting Robotic Mobile Manipulation Commands Expressed in Natural Language*. in *Proc. of the Workshop on Manipulation under Uncertainty at the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2011.

40.Felip, J., J. Bernabé, and A. Morales. *Emptying the box using blind haptic manipulation primitives*. in *Proc. of the Workshop on Manipulation under Uncertainty at the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2011.

41.Roa, M.A. and R. Suarez, *Computation of Independent Contact Regions for Grasping 3-D Objects.* IEEE Transactions on Robotics, 2009. **25**(4): p. 839-850.

42.Vahrenkamp, N., et al. *Visual servoing for humanoid grasping and manipulation tasks*. in *8th IEEE-RAS International Conference on Humanoid Robots, 2008. Humanoids 2008*. 2008.

43.Hsiao, K., et al. *Reactive grasping using optical proximity sensors*. in *IEEE International Conference on Robotics and Automation, 2009. ICRA '09*. 2009.

44.Teichmann, M. and B. Mishra. *Reactive algorithms for grasping using a modified parallel jaw gripper*. in*, 1994 IEEE International Conference on Robotics and Automation, 1994. Proceedings*. 1994.

45.Jiang, L.-T. and J.R. Smith. *A unified framework for grasping and shape acquisition via pretouch sensing*. in *2013 IEEE International Conference on Robotics and Automation (ICRA)*. 2013.

46.Bailey, T. and H. Durrant-Whyte, *Simultaneous localization and mapping (SLAM): part II.* IEEE Robotics Automation Magazine, 2006. **13**(3): p. 108-117.

47.Durrant-Whyte, H. and T. Bailey, *Simultaneous localization and mapping: part I.* IEEE Robotics Automation Magazine, 2006. **13**(2): p. 99-110.

48.*DepthSense Cameras*. 2014/03/13/12:58:12; Available from: http://www.softkinetic.com/en-us/products/depthsensecameras.aspx.

49.Grisetti, G., et al., *A Tutorial on Graph-Based SLAM.* IEEE Intelligent Transportation Systems Magazine, 2010. **2**(4): p. 31-43.

50.Grisetti, G., et al. *Efficient estimation of accurate maximum likelihood maps in 3D*. in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007*. 2007.

51.*Finite Iterative Closest Point - File Exchange - MATLAB Central*. 2014/03/26/00:59:49; Available from: http://www.mathworks.com/matlabcentral/fileexchange/file_infos/24301-finite-iterative-closest-point.

52.Low, K.-l., *Linear least-squares optimization for point-toplane ICP surface registration*. 2004.

53.Bouguet, J.-y., *Pyramidal implementation of the Lucas Kanade feature tracker.* Intel Corporation, Microprocessor Research Labs, 2000.

54.*iTOUGH2 Minimization Algorithms*. 2014/03/27/19:26:50; Available from: http://esd.lbl.gov/itough2/minimization/minalg.html.

55.*AAI Canada, Inc. - Intelligent Robots - Harmonic Arm*. 2014/02/03/06:09:58; Available from: http://www.aai.ca/robots/h_arm.html.

56.*OpenCV | OpenCV*. 2014/02/03/06:05:43; Available from: http://opencv.org/.

57.Cree, M.J., et al., *Analysis of the SoftKinetic DepthSense for Range Imaging*, in *Image Analysis and Recognition*, M. Kamel and A. Campilho, Editors. 2013, Springer Berlin Heidelberg. p. 668-675.

58.Markley, F.L., et al., *Averaging Quaternions.* Journal of Guidance, Control, and Dynamics, 2007. **30**(4): p. 1193-1197.

ภาคผนวก

ประวัติผู้เขียนวิทยานิพนธ์

นายณัชนนท์ วงษ์วิไล เกิดเมื่อวันจันทร์ที่ 5 ธันวาคม พ.ศ.2531 ที่จังหวัดราชบุรี สำเร็จการศึกษาระดับปริญญาตรี หลักสูตรวิศวกรรมศาสตรบัณฑิต (เกียรตินิยม อันดับสอง) สาขาวิศวกรรมคอมพิวเตอร์ จากคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2553 และเข้าศึกษาในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ ณ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2554

ได้รับทุนอัจฉริยะคืนรังจากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2554 ถึง 2555