ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

รายงานความก้าวหน้าโครงงานทางวิศวกรรม

(Senior Project Progress Report)

ประจำปีการศึกษา 2553

ชื่อเรื่อง (ภาษาไทย) ระบบหุ่นยนต์หลายตัวสำหรับการควบคุมวัตถุ

ชื่อเรื่อง (ภาษาอังกฤษ) Multi-Robot System for Object Manipulation

ชื่อ-นามสกุล	**รหัสประจำตัวนิสิต**	**ลายมือชื่อนิสิต**
นายรุ่งโรจน์ จินตเมธาสวัสดิ์	5031061821	(……..……..…….……)
(Mr.Rungroj Jintamethasawat)		

อาจารย์ที่ปรึกษาโครงงาน	**ลายมือชื่อ**
ผศ.ดร.อรรถวิทย์ สุดแสง	(……..……..……)
(Asst.Prof.Dr.Attawith Sudsang)	
อ.ดร.นัทที นิภานันท์	(……..……..……)
(Dr.Nattee Niparnan)	

สารบัญ

1. บทนำ **1**

2. รายละเอียดการดำเนินงานที่ผ่านมา **2**

2.1 ศึกษา Console และ Library ที่นำมาใช้สั่งงานหุ่นยนต์ 2

2.2 ศึกษาวิธีการเขียนโปรแกรมบนหุ่นยนต์ 3

2.3 สร้าง Console ไว้สำหรับเขียนโปรแกรมหุ่นยนต์ 5

2.4 ศึกษาวิธีการสื่อสารระหว่างหุ่นยนต์แบบ Peer-to-Peer 6

2.5 ศึกษา PicoC Interpreter 7

2.6 ออกแบบสนามและวัตถุสำหรับหุ่นยนต์ 7

2.7 เขียนโปรแกรมจับภาพวัตถุ ที่ได้รับมาจากกล้องบนตัวหุ่นยนต์ 8

3. ความก้าวหน้าเมื่อเทียบกับกำหนดการที่วางไว้ **9**

4. อุปสรรคและแนวทางการแก้ไข **11**

4.1 อุปสรรคเกี่ยวกับตำแหน่งและการเคลื่อนที่ของหุ่นยนต์ ที่มีผลต่อค่าสีของภาพวัตถุ 11

4.2 ข้อจำกัดของกล้องบนตัวหุ่นยนต์ 12

4.3 ความไม่คุ้นเคยของผู้พัฒนาต่อภาษา PicoC 12

4.4 อุปสรรคในการติดต่อสื่อสารระหว่างหุ่นยนต์ภายในกลุ่ม 13

5. **แผนการดำเนินงานขั้นต่อไป** **14**

5.1 แก้ปัญหาที่เกิดขึ้น ตามแนวทางที่ได้วางแผนไว้ 14

5.2 ออกแบบแขนสำหรับจับวัตถุ 15

5.3 พัฒนาระบบหุ่นยนต์ตัวเดียว เพื่อให้สามารถทำงานได้อย่างถูกต้อง 15

5.4 พัฒนาระบบหุ่นยนต์หลายตัว เพื่อให้สามารถทำงานได้อย่างถูกต้อง 15

5.5 ย้ายส่วนประมวลผลภาพ จากเดิมที่อยู่บนคอมพิวเตอร์ ไปอยู่บนตัวหุ่นยนต์ 15

5.6 ทดสอบระบบหุ่นยนต์หลายตัว เพื่อหาประสิทธิภาพเทียบกับระบบหุ่นยนต์ตัวเดียว 15

6. สรุปผลการดำเนินงาน **16**

7. เอกสารอ้างอิง **17**

1. บทนำ

โครงงานนี้ ได้ออกแบบระบบหุ่นยนต์หลายตัว โดยหุ่นยนต์ภายในกลุ่มจะต้องช่วยกันเคลื่อนย้ายวัตถุ 3 ชิ้น ไปยังเป้าหมายที่กำหนดไว้ให้ วัตถุทั้ง 3 ชิ้น คือ ทรงกระบอกสีน้ำเงิน ทรงสี่เหลี่ยมสีเขียว และปริซึมหน้าตัดรูปสามเหลี่ยมสีแดง

ระบบหุ่นยนต์หลายตัวที่พัฒนาขึ้น จะใช้หุ่นยนต์ยี่ห้อ Surveyor SRV-1 หุ่นยนต์ภายในกลุ่มจะทำงานภายในสนามพื้นสีขาว และจะต้องติดต่อสื่อสารกันภายใต้เครือข่ายไร้สาย (Wireless LAN) เครือข่ายเดียวกัน และระบบหุ่นยนต์หลายตัวที่พัฒนาขึ้นจะประกอบด้วยหุ่นยนต์แบบเดียวกัน (Homogeneous) นอกจากนี้ ระบบหุ่นยนต์หลายตัวที่พัฒนาขึ้น เป็นระบบขนาดเล็ก สามารถรองรับหุ่นยนต์ได้ไม่เกิน 5 ตัว

สำหรับการพัฒนาระบบหุ่นยนต์หลายตัว จะแบ่งเป็น 5 ส่วนใหญ่ๆ คือ

1) การจัดเตรียมวัสดุที่เกี่ยวข้องกับการทำโครงงาน ได้แก่ การเตรียมสนาม วัตถุ และหุ่นยนต์ Surveyor SRV-1
2) การพัฒนาโปรแกรมสำหรับบริหารระบบหุ่นยนต์หลายตัว หรือที่เรียกว่า Console
3) การพัฒนาส่วนประมวลผลภาพที่ได้รับมาจากกล้องบนตัวหุ่นยนต์ เพื่อหาตำแหน่งของวัตถุ
4) การออกแบบวิธีการที่ให้หุ่นยนต์นำวัตถุไปวางไว้ ณ เป้าหมายที่กำหนดไว้
5) การทดสอบประสิทธิภาพของระบบหุ่นยนต์หลายตัว เทียบกับระบบหุ่นยนต์ตัวเดียว

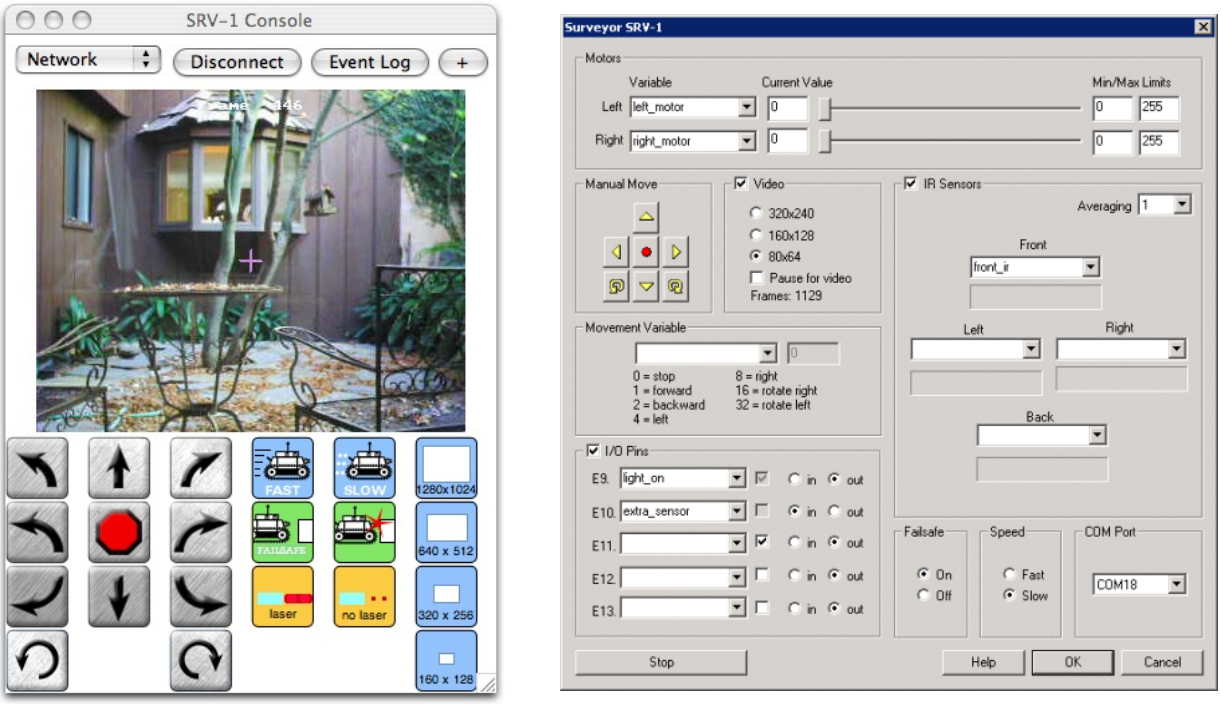
ในส่วนของการจัดเตรียมวัสดุ และการพัฒนาโปรแกรมบริหารระบบหุ่นยนต์หลายตัว ผู้พัฒนาได้ดำเนินการพัฒนาแล้ว สำหรับส่วนประมวลผลภาพ ผู้พัฒนาได้พัฒนาไปแล้วบางส่วน และพบปัญหาในระหว่างการพัฒนาด้วย และสำหรับการออกแบบวิธีการที่ให้หุ่นยนต์นำวัตถุไปวางไว้ ณ เป้าหมาย กับการทดสอบประสิทธิภาพของระบบหุ่นยนต์หลายตัว ผู้พัฒนายังไม่ได้เริ่มดำเนินการแต่อย่างใด

ประโยชน์ที่คาดว่าจะได้รับจากการพัฒนาระบบนี้ คือ ระบบหุ่นยนต์หลายตัว จะต้องทำงานได้เร็วกว่าระบบหุ่นยนต์ตัวเดียว และระบบหุ่นยนต์หลายตัวจะต้องทนต่อความผิดพลาดได้ดีกว่าระบบหุ่นยนต์ตัวเดียว

2. รายละเอียดการดำเนินงานที่ผ่านมา

2.1 ศึกษา Console และ Library ที่นำมาใช้สั่งงานหุ่นยนต์

เนื่องจากหุ่นยนต์ Surveyor SRV-1 เป็นหุ่นยนต์ Open Source ที่ใช้ระบบปฏิบัติการ Linux ดังนั้น ผู้พัฒนาทุกคนจึงมีสิทธิ์พัฒนาโปรแกรมที่ใช้ควบคุมหุ่นยนต์ หรือที่เรียกว่า Console หรือพัฒนา Library เพื่อใช้ควบคุมหุ่นยนต์ได้เอง นอกจากนี้ยังสามารถแก้ไข Firmware ภายในตัวหุ่นยนต์ได้อีกด้วย ตัวอย่าง Console ควบคุมหุ่นยนต์ Surveyor SRV-1 ที่ใช้กันอย่างแพร่หลาย เช่น Java Console [3] ส่วน Library สำหรับหุ่นยนต์ ที่ใช้กันอย่างแพร่หลาย คือ Library ของโปรแกรม RoboRealm [2]



รูปที่ 1 (ซ้าย) Java Console สำหรับควบคุมหุ่นยนต์ Surveyor SRV-1

(ขวา) ตัวอย่างการนำ Library ของ RoboRealm มาใช้สร้างเป็น Console ควบคุมหุ่นยนต์ Surveyor SRV-1

การสั่งงานหุ่นยนต์ Surveyor SRV-1 สามารถทำได้ 2 แบบ คือ

1) โดยการส่งคำสั่งไปในรูปแบบของ Byte Array ผ่านโปรโตคอล TCP หรือ UDP สำหรับรูปแบบของคำสั่ง สามารถดูได้ที่ http://www.surveyor.com/SRV_protocol.html [4]

2) โดยการเขียนโปรแกรมลงบนตัวหุ่นยนต์โดยตรง ซึ่งอาจใช้วิธีการเก็บโปรแกรมไว้ในหน่วยความจำแฟลซ (Flash Memory) ของหุ่นยนต์ หรือใช้วิธีแก้ไข Firmware บนตัวหุ่นยนต์ก็ได้

จากการศึกษาในขั้นตอนนี้ ผู้พัฒนาได้เห็นตัวอย่าง Console เป็นจำนวนมาก ที่พัฒนาขึ้นเพื่อใช้ควบคุมหุ่นยนต์ Surveyor SRV-1 อย่างไรก็ตาม งานของผู้พัฒนาเป็นงานเฉพาะทาง ที่จำเป็นต้องใช้ Console ที่มีรูปแบบการทำงานแตกต่างจาก Console ที่มีอยู่แล้ว กล่าวคือ Console ต้องสามารถสั่งงานหุ่นยนต์ได้หลายๆ ตัวพร้อมกัน อีกทั้งยังมี Library เป็นจำนวนมาก ที่สามารถนำมาใช้สร้าง Console ได้อย่างสะดวก ผู้พัฒนาจึงใช้วิธีสร้าง Console ใช้เอง โดยอ้างอิงจากซอร์สโค้ดของ Console ที่มีอยู่แล้ว

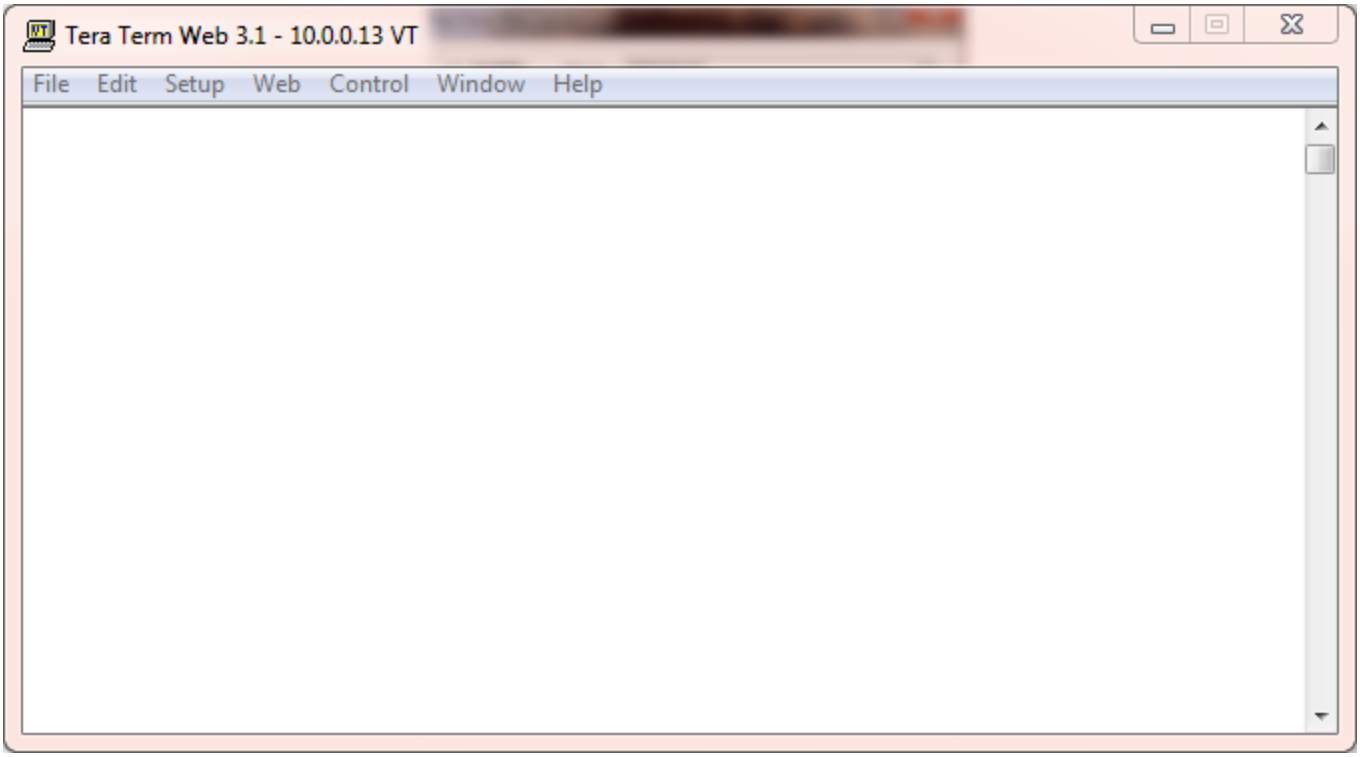
2.2 ศึกษาวิธีการเขียนโปรแกรมบนหุ่นยนต์

การเขียนโปรแกรมลงบนตัวหุ่นยนต์ สามารถทำได้โดยการเก็บโปรแกรมลงในหน่วยความจำแฟลซของหุ่นยนต์ หรือใช้การแก้ไข Firmware แต่จากการปฏิบัติ ผู้พัฒนาพบว่า การเก็บโปรแกรมลงในหน่วยความจำแฟลซสะดวกกว่ามาก และยืดหยุ่นกว่าในกรณีที่ต้องการโหลดโปรแกรมอื่นใส่ในตัวหุ่นยนต์ ผู้พัฒนาจึงเลือกใช้วิธีการเก็บโปรแกรมไว้ในหน่วยความจำแฟลซของหุ่นยนต์ แล้วจึงค่อยสั่งให้โปรแกรมนั้นทำงาน

การส่งโปรแกรมจากคอมพิวเตอร์ไปเก็บลงในหน่วยความจำแฟลซของหุ่นยนต์ สามารถทำได้ 2 วิธีหลัก คือ

1) ใช้โปรโตคอล XMODEM [5] ซึ่งเป็นโปรโตคอลอย่างง่ายที่ใช้ในการส่งไฟล์ และมีความสามารตรวจสอบข้อผิดพลาดที่เกิดจากการส่ง โปรแกรมที่รองรับการส่งไฟล์ผ่านโปรโตคอล XMODEM เช่น โปรแกรม TeraTerm [6]

2) ใช้วิธีการสร้าง Console โดย Console จะเรียกใช้โปรโตคอลของหุ่นยนต์ Surveyor SRV-1 เพื่อส่ง Byte Array ไปบอกหุ่นยนต์ที่จะโหลดโปรแกรมใส่ก่อน จากนั้น Console จึงค่อยส่งโปรแกรมไป



รูปที่ 2 โปรแกรม TeraTerm ที่รองรับโปรโตคอล XMODEM

จากการศึกษาในขั้นตอนนี้ ผู้พัฒนาพบว่าการใช้โปรแกรม TeraTerm ในการส่งไฟล์ มีความยุ่งยากกว่าการสร้าง Console ขึ้นมาใช้เอง ทั้งนี้เนื่องจากการส่งโปรแกรมผ่านโปรแกรม TeraTerm จะต้องนำโปรแกรมเก็บลงไฟล์ก่อน แล้วจึงค่อยส่งไฟล์ไป และหากต้องการให้โปรแกรมบนหุ่นยนต์ทำงาน จะต้องเรียกใช้โปรแกรม TeraTerm อีกครั้งหนึ่ง แต่ถ้าเป็น Console ที่เขียนขึ้นเอง ผู้พัฒนาสามารถออกแบบ Console ให้มีที่ว่างสำหรับใช้แก้ไขโปรแกรม และมีปุ่มสำหรับโหลดโปรแกรมลงบนตัวหุ่นยนต์ รวมทั้งมีปุ่มสั่งให้โปรแกรมบนตัวหุ่นยนต์ทำงานอีกด้วย ผู้พัฒนาจึงใช้วิธีสร้าง Console สำหรับโหลดโปรแกรมขึ้นเอง โดยจะนำการทำงานทั้งหมดไปรวมไว้กับ Console ที่สร้างไว้ในขั้นตอนแรก

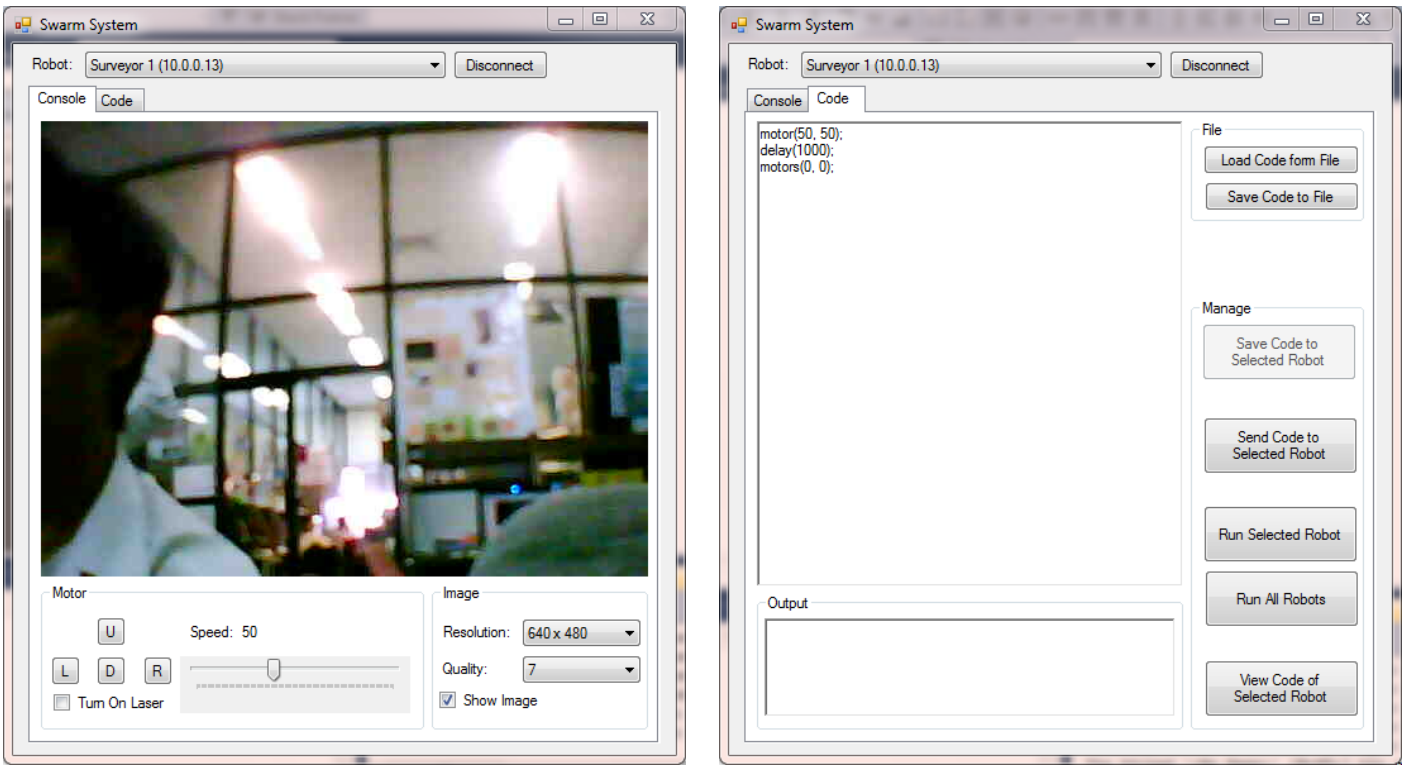
2.3 *สร้าง Console ไว้สำหรับเขียนโปรแกรมหุ่นยนต์*

จากปัญหาที่ว่า Console ที่มีอยู่ในปัจจุบัน ไม่สามารถเชื่อมต่อหุ่นยนต์ได้หลายตัวในเวลาเดียวกัน และการเขียนโปรแกรมลงในหน่วยความจำแฟลซของหุ่นยนต์มีความลำบากพอสมควร ผู้พัฒนาจึงทำการพัฒนา Console ขึ้นใช้งานเอง โดยผู้พัฒนาจะใช้ Library ของ AForge.NET [1] เพื่อพัฒนา Console และ Console ที่พัฒนาขึ้น จะมีความสามารถในการสั่งงานหุ่นยนต์เบื้องต้น เช่น การควบคุม Motor การสั่งงาน Laser การรับภาพจากกล้องบนตัวหุ่นยนต์มาแสดงผลและประมวลผล เป็นต้น และมีความสามารถในการเขียนโปรแกรมลงในหน่วยความจำแฟลซของตัวหุ่นยนต์

การสั่งงานทั่วไป สามารถทำได้โดยการส่งข้อความในรูปแบบที่กำหนดให้ ผ่านโปรโตคอล TCP หรือ UDP เช่น หากต้องการสั่งให้ Motor เดินหน้าเป็นเวลา 0.7 วินาที จะต้องส่ง Byte Array เป็น 0x4D 0x32 0x46 0x14 (หรือเขียนอยู่ในรูปข้อความอักขระได้เป็น “M22F”) หรือหากต้องการสั่งเปิด Laser จะต้องส่ง Byte Array เป็น 0x6C (หรือเขียนอยู่ในรูปข้อความอักขระได้เป็น “l”) เป็นต้น

การเขียนโปรแกรมลงบนหน่วยความจำแฟลซ สามารถทำได้โดยมีขั้นตอนดังนี้

1) ส่งข้อความ “zc” เพื่อทำการล้างข้อมูลเดิมในหน่วยความจำแฟลซออกให้หมด
2) ส่งข้อความ “E” เพื่อเข้าสู่โหมด Editor สำหรับแก้ไขข้อมูลในหน่วยความจำแฟลซ
3) สำหรับแต่ละบรรทัดของโปรแกรม ให้ส่งข้อความ “I” ไปบอกหุ่นยนต์ก่อน เพื่อบอกว่าจะทำการเพิ่มข้อมูลลงในหน่วยความจำแฟลซ
4) ส่งโปรแกรมบรรทัดนั้นเข้าไป
5) ส่งข้อความ 0x1B หรือเป็นข้อความแสดงการกด Esc Key เพื่อออกจากโหมดการเพิ่มข้อมูล
6) ทำขั้นตอนที่ 3-5 เรื่อยๆ จนส่งโปรแกรมครบทุกบรรทัด

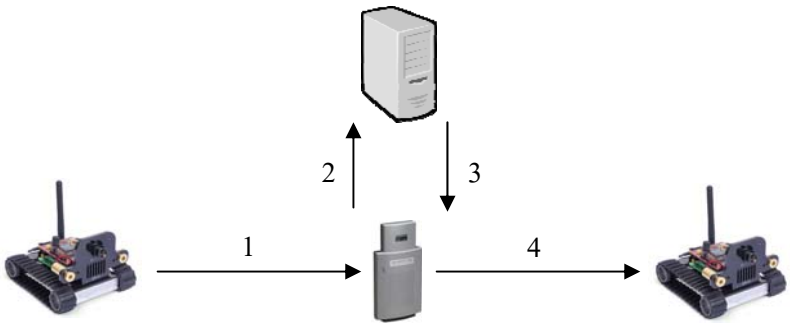


รูปที่ 3 (ซ้าย) Console ที่พัฒนาขึ้น เพื่อใช้สั่งงานพื้นฐานให้กับหุ่นยนต์

(ขวา) Console ที่พัฒนาขึ้น เพื่อใช้สำหรับโหลดโปรแกรมลงบนหน่วยความจำแฟลช

2.4 ศึกษาวิธีการสื่อสารระหว่างหุ่นยนต์แบบ Peer-to-Peer

ในระบบหุ่นยนต์หลายตัว หุ่นยนต์แต่ละตัว มีความจำเป็นต้องติดต่อสื่อสารกับหุ่นยนต์ตัวอื่นๆ แต่เนื่องจากหุ่นยนต์ Surveyor SRV-1 ยังไม่มี Library สำหรับการติดต่อสื่อสารระหว่างหุ่นยนต์กับหุ่นยนต์โดยตรง [11] ดังนั้น ผู้พัฒนาจึงต้องจำลองการสื่อสารระหว่างหุ่นยนต์แบบ Peer-to-Peer ขึ้นมาเอง โดยการทำงานของระบบการสื่อสารแบบ Peer-to-Peer แบบเสมือน จะมีพื้นฐานมาจากระบบการสื่อสารแบบ Infrastructure



รูปที่ 4 วิธีการติดต่อสื่อสารระหว่างหุ่นยนต์ 2 ตัว

2.5 ศึกษา PicoC Interpreter [7], [8]

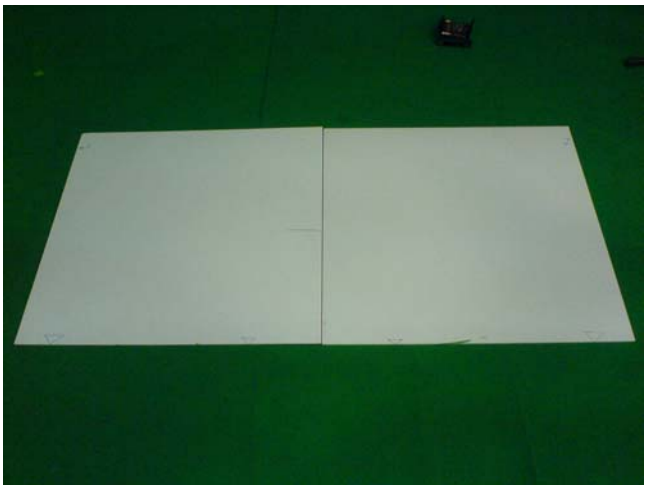
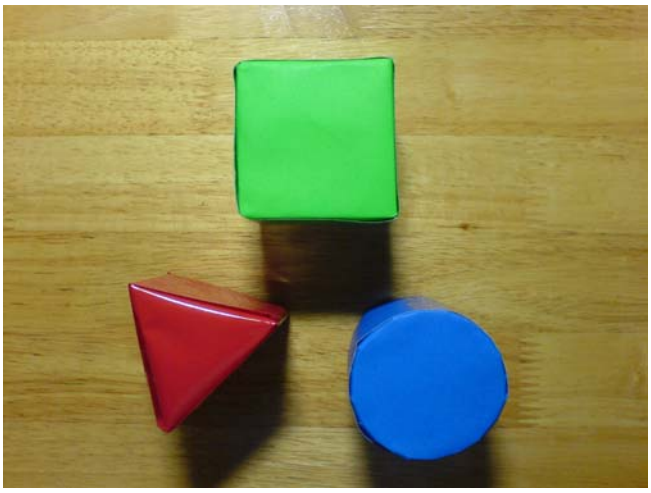
PicoC เป็นภาษาที่มีการทำงานแบบ Interpreter และมีพื้นฐานการทำงานมาจากภาษา C แต่ได้ถูกลดทอนฟังก์ชันการทำงานบางอย่างออกไป เพื่อให้ตัว PicoC มีขนาดเล็ก และเหมาะที่จะนำไปใส่ไว้ในอุปกรณ์ขนาดเล็ก เช่น หุ่นยนต์ เป็นต้น

เช่นเดียวกัน หุ่นยนต์ Surveyor SRV-1 ก็รองรับโปรแกรมที่เขียนในรูปแบบ PicoC จากการศึกษาในขั้นตอนนี้ ผู้พัฒนาพบว่า PicoC ในหุ่นยนต์ Surveyor SRV-1 สามารถรองรับการสั่งงานทั่วไป เช่น การสั่งงาน Motor การสั่งงาน Laser หรือการสั่งงานผ่านอุปกรณ์ตรวจหาระยะทาง เป็นต้น และรองรับการประมวลผลภาพ (Image Processing) [9] เช่น การหาสีของวัตถุ การตรวจหาวัตถุ การหาขอบของวัตถุ เป็นต้น รายละเอียดของการสั่งงานหุ่นยนต์ด้วย PicoC สามารถดูได้ที่ http://www.surveyor.com/C.html

อย่างไรก็ตาม บริษัท Surveyor ยังไม่มีคู่มือการเขียนโปรแกรมรูปแบบ PicoC อย่างละเอียด เพื่อสั่งงานหุ่นยนต์ ผู้พัฒนาจึงตัดสินใจเปลี่ยนวิธีการประมวลผลภาพ จากที่ทำบนหุ่นยนต์ มาทำบนคอมพิวเตอร์ชั่วคราว

2.6 *ออกแบบสนามและวัตถุสำหรับหุ่นยนต์*

ในขั้นตอนนี้ จะนำแผ่นพลาสติกแข็งสีขาว มาวางต่อกันดังรูป จากนั้น จึงทำการสร้างวัตถุ 3 ชิ้น คือ ทรงกระบอกสีน้ำเงิน ทรงสี่เหลี่ยมสีเขียว และปริซึมหน้าตัดรูปสามเหลี่ยมสีแดง

รูปที่ 5 (ซ้าย) สนามพื้นสีขาว

(ขวา) วัตถุ 3 ชิ้น แต่ละชิ้นมีสีและรูปทรงแตกต่างกันไป

2.7 เขียนโปรแกรมจับภาพวัตถุ ที่ได้รับมาจากกล้องบนตัวหุ่นยนต์

ในขั้นตอนนี้ ผู้พัฒนาจะใช้ Library AForge.NET [1] ซึ่งเป็น Library ที่สามารถทำการประมวลผลภาพได้ และเป็น Library เดียวกันกับที่ใช้สั่งงานหุ่นยนต์ Surveyor SRV-1

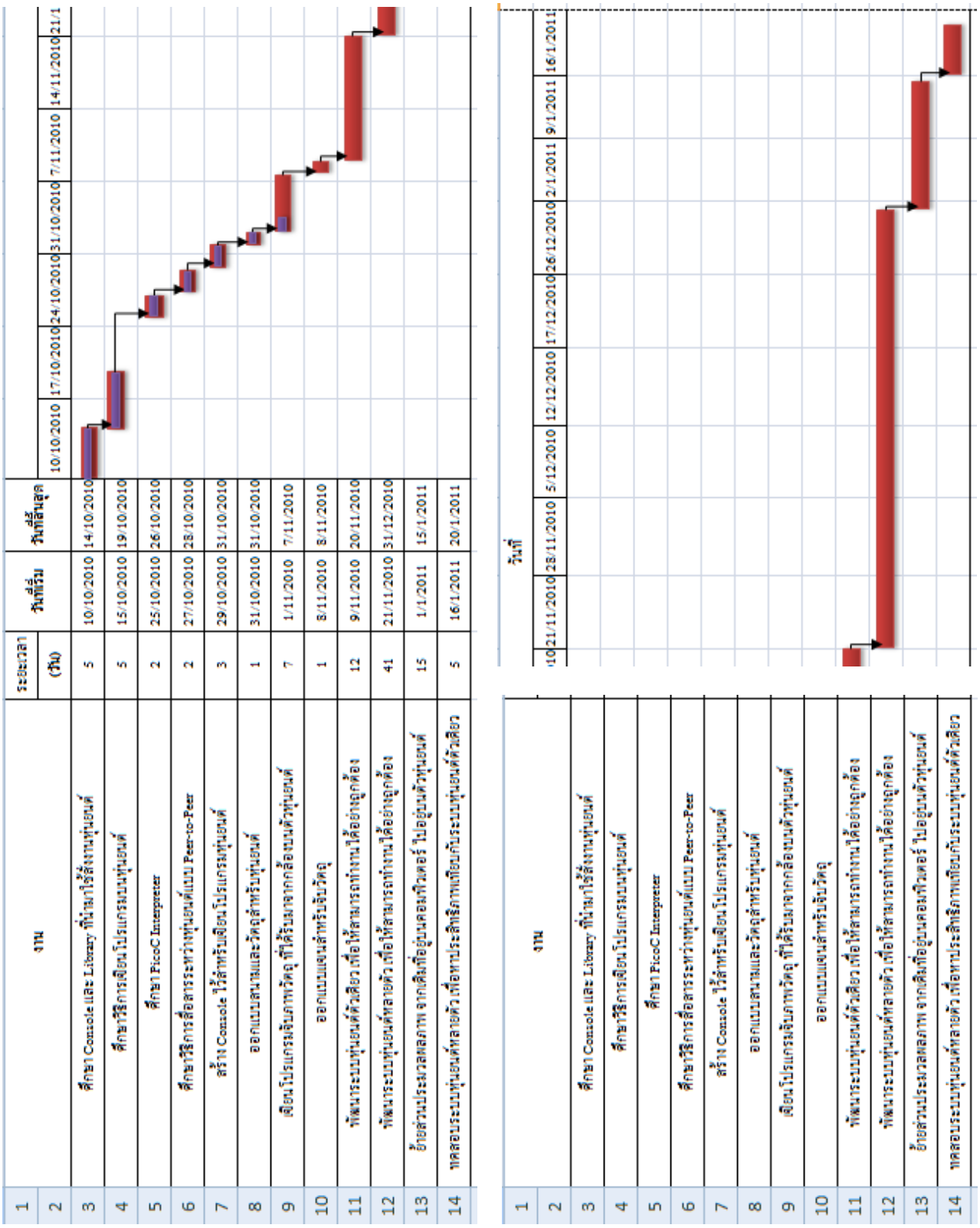
ขั้นตอนนี้เป็นขั้นตอนที่ผู้พัฒนากำลังดำเนินการอยู่ และพบว่ามีปัญหาพอสมควร ซึ่งปัญหาเหล่านี้จะกล่าวต่อไปในหัวข้อ อุปสรรคและแนวทางแก้ไข

3. ความก้าวหน้าเมื่อเทียบกับกำหนดการที่วางไว้

จากตารางเวลาการดำเนินงาน ผู้พัฒนาพบว่า ความก้าวหน้าของโครงงาน ช้ากว่าที่ได้กำหนดไว้ ทั้งนี้เพราะในช่วงเดือนตุลาคม ผู้พัฒนามีความจำเป็นต้องไปทำกิจกรรมอย่างอื่น และผู้พัฒนาไม่มีความคุ้นเคยกับการเขียนโปรแกรมลงบนหุ่นยนต์ Surveyor SRV-1 รวมทั้งยังไม่มีความรู้เกี่ยวกับการพัฒนาระบบในลักษณะเช่นนี้มาก่อน ทำให้เกิดปัญหาตามมามากมาย อย่างไรก็ตาม ความล้าช้าของโครงงาน ก็เป็นแรงกระตุ้นให้ผู้พัฒนาตื่นตัวมากขึ้น

งาน	ระยะเวลา (เดือน)				
	ก.ย. 53	ต.ค. 53	พ.ย. 53	ธ.ค. 53	ม.ค. 54
ศึกษาค้นคว้าข้อมูลที่เกี่ยวข้อง					
พัฒนาระบบตรวจจับวัตถุ					
พัฒนาอัลกอริทึมในการวางแผนการเคลื่อนที่หุ่นยนต์					
พัฒนาหุ่นยนต์หนึ่งตัว เพื่อให้สามารถนำวัตถุไปวางไว้ในตำแหน่งที่ถูกต้องได้					
พัฒนาระบบหุ่นยนต์หลายตัว เพื่อให้สามารถนำวัตถุไปวางไว้ในตำแหน่งที่ถูกต้องได้					
เปรียบเทียบผลการทำงานที่ได้ ระหว่างระบบหุ่นยนต์หลายตัว กับหุ่นยนต์ตัวเดียว					

รูปที่ 6 แผนการดำเนินงานแสดงถึงระยะเวลาที่คาดว่าจะต้องใช้



รูปที่ 7 แผนการดำเนินงานแสดงถึงระยะเวลาที่ใช้จริง

4. อุปสรรคและแนวทางการแก้ไข

4.1 อุปสรรคเกี่ยวกับตำแหน่งและการเคลื่อนที่ของหุ่นยนต์ ที่มีผลต่อค่าสีของภาพวัตถุ

เมื่อหุ่นยนต์อยู่กับที่ เราสามารถทำการหาวัตถุได้อย่างถูกต้อง โดยการบอกค่าสีของวัตถุที่ต้องการ แล้วจึงทำการประมวลผลภาพอย่างง่ายบนคอมพิวเตอร์ แต่เมื่อหุ่นยนต์เปลี่ยนตำแหน่ง ค่าแสงที่เข้าสู่กล้องของหุ่นยนต์ก็ย่อมเปลี่ยนไป ทำให้ค่าสีของวัตถุเปลี่ยนไปด้วย เพราะระบบหุ่นยนต์ที่พัฒนาขึ้น จะทำการเก็บค่าสีของวัตถุเมื่อตอนสั่งให้ระบบเริ่มทำงานเท่านั้น ดังนั้น ในระหว่างการประมวลผล เมื่อพบว่าวัตถุมีค่าสีไม่ตรงกับค่าสีที่เก็บไว้ตั้งแต่ตอนแรก ระบบก็จะประมวลผลผิดว่าไม่พบวัตถุที่ต้องการหา หรือประมวลผลผิดว่าเป็นวัตถุอีกชิ้นหนึ่ง ทั้งๆ ที่หุ่นยนต์อาจจะพบวัตถุชิ้นนั้นแล้วก็ได้ ตัวอย่างของการเปลี่ยนตำแหน่งหุ่นยนต์ ที่มีผลต่อสีของภาพ แสดงไว้ในรูปที่ 6

รูปที่ 8 (ซ้าย) เมื่อหุ่นยนต์อยู่ไกลวัตถุ วัตถุจะมีแสงน้อย

(ขวา) เมื่อหุ่นยนต์อยู่ใกล้วัตถุ วัตถุจะมีแสงมาก และค่าสีของวัตถุจะเปลี่ยนไป

<u>แนวทางการแก้ไข</u>

1) ทำการปรับปรุงค่าสีของวัตถุ เมื่อพบว่าค่าสีที่ตรวจจับได้เริ่มเปลี่ยนแปลงไป

2) ติดตั้งฉากกันแสงให้กับสนามของหุ่นยนต์

4.2 ข้อจำกัดของกล้องบนตัวหุ่นยนต์

กล้องบนตัวหุ่นยนต์ สามารถปรับคุณภาพของภาพได้ 8 ระดับ และปรับความละเอียดได้ 3 ระดับ คือ 160x120 pixel, 320x240 pixel และ 640x480 pixel ถึงแม้ว่าผู้พัฒนาได้ทดลองตั้งค่าความละเอียดของภาพเป็น 320x240 และตั้งค่าคุณภาพของภาพเป็นระดับกลาง แต่อัตราการส่งภาพจากตัวหุ่นยนต์มายังคอมพิวเตอร์ก็ยังน้อยกว่า 10 ภาพต่อวินาที ซึ่งช้ามากเมื่อมีความจำเป็นต้องนำภาพไปประมวลผลแบบ Real-Time

สมมติฐานของผู้พัฒนา คือ ความล่าช้าของการส่งภาพ ไม่ได้ขึ้นกับความเร็วของเครือข่ายไร้สาย แต่ขึ้นกับความเร็วของซีพียูบนตัวหุ่นยนต์ เพราะการส่งข้อมูลผ่านเครือข่ายไร้สาย สามารถส่งได้ในปริมาณที่มากกว่า 1 Megabytes ต่อวินาที ซึ่งเพียงพอต่อการส่งภาพที่ความละเอียด และคุณภาพตามที่ได้กำหนดไว้

<u>แนวทางการแก้ไข</u>

1) ทำให้หุ่นยนต์เคลื่อนที่ช้าลง เพื่อให้ทำประมวลผลภาพแบบ Real-Time ได้ทันท่วงที

2) พยายามย้ายการประมวลผลภาพทั้งหมด ให้ไปอยู่ในตัวหุ่นยนต์ ซึ่งจะต้องอาศัยความรู้เกี่ยวกับ PicoC ด้วย

4.3 ความไม่คุ้นเคยของผู้พัฒนาต่อภาษา PicoC [10]

ถึงแม้ว่าภาษา PicoC จะมีความคล้ายคลึงกับภาษา C แต่ผู้พัฒนาก็ไม่เคยใช้ภาษา PicoC สำหรับทำการประมวลผลภาพบนตัวหุ่นยนต์ Surveyor SRV-1 มาก่อน อีกทั้งคำสั่งในการประมวลผลภาพบนตัวหุ่นยนต์ ที่อยู่ในรูปแบบ PicoC ถูกพัฒนาขึ้นโดยบริษัท Surveyor เพียงผู้เดียว แต่บริษัทก็ยังไม่ได้ออกคู่มือประกอบการใช้งานที่ละเอียด ทำให้ผู้พัฒนาต้องใช้เวลาศึกษานานกว่าปกติ และ ณ ขณะนี้ ผู้พัฒนาก็ยังไม่สามารถเรียกใช้คำสั่ง

ในการประมวลผลภาพบนตัวหุ่นยนต์ได้ ทำให้ผู้พัฒนาจำเป็นต้องนำภาพจากหุ่นยนต์ มาประมวลผลที่คอมพิวเตอร์เป็นการชั่วคราว

<u>แนวทางการแก้ไข</u>

1) เรียนรู้การใช้งาน PicoC โดยเร็วที่สุด เพื่อให้สามารถทำการประมวลผลภาพบนตัวหุ่นยนต์ได้
2) หากผู้พัฒนาไม่สามารถเขียนโปรแกรมเพื่อให้หุ่นยนต์ทำการประมวลผลภาพได้ ผู้พัฒนาจะต้องพัฒนาวิธีการรับ-ส่งภาพจากหุ่นยนต์ไปยังคอมพิวเตอร์ให้รวดเร็วกว่านี้

4.4 อุปสรรคในการติดต่อสื่อสารระหว่างหุ่นยนต์ภายในกลุ่ม [11]

เนื่องจากหุ่นยนต์แต่ละตัวในระบบหุ่นยนต์หลายตัว จำเป็นต้องมีการติดต่อสื่อสารกับหุ่นยนต์ตัวอื่น เพื่อให้สามารถประสานงานกัน และทำงานได้อย่างถูกต้อง แต่หุ่นยนต์ Surveyor SRV-1 ยังไม่รองรับการติดต่อสื่อสารกันระหว่างหุ่นยนต์กับหุ่นยนต์โดยตรง (Peer-to-Peer) ผู้พัฒนาจึงต้องหาวิธีการ เพื่อทำให้หุ่นยนต์ 2 ตัวใดๆ สามารถติดต่อสื่อสารกันได้

<u>แนวทางการแก้ไข</u>

1) จำลองการติดต่อสื่อสารแบบ Peer-to-Peer ระหว่างหุ่นยนต์ 2 ตัว โดยใช้พื้นฐานจากระบบเครือข่ายแบบ Infrastructure ดังนี้ได้แสดงไว้ในหัวข้อที่ 2.4

5. แผนการดำเนินงานขั้นต่อไป

5.1 แก้ปัญหาที่เกิดขึ้น ตามแนวทางที่ได้วางแผนไว้

ปัญหาที่ได้กล่าวไว้ในหัวข้อที่ 4 ผู้พัฒนาจะเร่งแก้ไขโดยพลัน แต่หากผู้พัฒนาพบว่า ปัญหาเหล่านั้นไม่มีผลหรือมีผลน้อยมากต่อการพัฒนาระบบในส่วนอื่น ผู้พัฒนาจะย้อนกลับมาแก้ไขในภายหลัง สำหรับปัญหาเรื่องการเปลี่ยนค่าสีของภาพวัตถุ และปัญหาเรื่องการเขียนโปรแกรมสั่งงานหุ่นยนต์ด้วยภาษา PicoC เป็นปัญหาที่มีผลต่อการทำงานของระบบโดยรวม ดังนั้น ผู้พัฒนาก็จะทำการแก้ไขโดยด่วน แต่ปัญหาเรื่องคุณภาพของภาพกับความเร็วของภาพที่รับเข้ามา เป็นปัญหาที่สามารถแก้ไขได้โดยการปรับความเร็วของการเคลื่อนที่ของหุ่นยนต์ให้ช้าลง ผู้พัฒนาจึงเก็บปัญหานี้ไว้ใน To-Do List และจะทำการแก้ไขในภายหลัง สำหรับปัญหาเรื่องการสื่อสารแบบ Peer-to-Peer ระหว่างหุ่นยนต์ ผู้พัฒนาได้ทำการแก้ไขไปแล้ว

ปัญหาที่มีอยู่	**การจัดการกับปัญหา**
อุปสรรคเกี่ยวกับตำแหน่ง และการเคลื่อนที่ของหุ่นยนต์ ที่มีผลต่อค่าสีของภาพวัตถุ	ต้องรีบแก้ไขโดยพลัน
ข้อจำกัดของกล้องบนตัวหุ่นยนต์	สามารถแก้ไขได้ในภายหลัง
ความไม่คุ้นเคยของผู้พัฒนาต่อภาษา PicoC	ต้องรีบแก้ไขโดยพลัน
อุปสรรคในการติดต่อสื่อสารระหว่างหุ่นยนต์ภายในกลุ่ม	ได้รับการแก้ไขแล้ว

รูปที่ 9 ปัญหาและการจัดการกับปัญหา

5.2 ออกแบบแขนสำหรับจับวัตถุ

ทำการออกแบบแขนจับวัตถุที่จะนำไปติดตั้งบนตัวหุ่นยนต์ ให้มีน้ำหนักเบา ทนทาน และสามารถประคองวัตถุไว้ได้เมื่อหุ่นยนต์เคลื่อนที่

5.3 พัฒนาระบบหุ่นยนต์ตัวเดียว เพื่อให้สามารถทำงานได้อย่างถูกต้อง

ทำการทดลองเพื่อให้หุ่นยนต์เพียงหนึ่งตัว นำวัตถุไปวางไว้ในตำแหน่งปลายทางได้ถูกต้อง

5.4 พัฒนาระบบหุ่นยนต์หลายตัว เพื่อให้สามารถทำงานได้อย่างถูกต้อง

เมื่อหุ่นยนต์หนึ่งตัวทำงานได้ถูกต้องแล้ว ก็จะทำการเพิ่มจำนวนหุ่นยนต์ให้มีตั้งแต่ 2 ตัวขึ้นไป โดยการพัฒนาให้หุ่นยนต์มีการประสานงานกัน เพื่อช่วยกันนำวัตถุไปวางไว้ ๆ ตำแหน่งปลายทางได้ถูกต้อง

5.5 ย้ายส่วนประมวลผลภาพ จากเดิมที่อยู่บนคอมพิวเตอร์ ไปอยู่บนตัวหุ่นยนต์

ทำการศึกษาภาษา PicoC ที่ใช้ในการประมวลผลภาพบนหุ่นยนต์ จากนั้น จึงย้ายส่วนประมวลผลภาพจากเดิมที่อยู่บนคอมพิวเตอร์ ไปอยู่ในตัวหุ่นยนต์แทน โดยจะไม่เปลี่ยนแปลงขั้นตอนการประมวลผลภาพแต่อย่างใด

5.6 ทดสอบระบบหุ่นยนต์หลายตัว เพื่อหาประสิทธิภาพเทียบกับระบบหุ่นยนต์ตัวเดียว

ทำการเปรียบเทียบประสิทธิภาพที่ได้ ระหว่างหุ่นยนต์ตัวเดียว กับระบบหุ่นยนต์หลายตัว และทำการเปรียบเทียบเวลาที่ใช้เคลื่อนย้ายวัตถุไปยังตำแหน่งปลายทาง ระหว่างระบบหุ่นยนต์ตัวเดียว กับระบบหุ่นยนต์หลายตัว ทั้งนี้เพื่อแสดงให้เห็นถึงประโยชน์ของการนำระบบหุ่นยนต์หลายตัวมาใช้งาน

6. สรุปผลการดำเนินงาน

จากการดำเนินงานตลอดเวลา 20 วัน ผู้พัฒนาได้พยายามพัฒนาระบบหุ่นยนต์ตัวเดียวก่อน โดยเริ่มจากการศึกษาข้อมูลที่เกี่ยวข้อง ได้แก่ การเขียนโปรแกรมลงบนหุ่นยนต์ Surveyor SRV-1 การควบคุมหุ่นยนต์ Surveyor SRV-1 เบื้องต้น และการส่งข้อมูลระหว่างหุ่นยนต์ด้วยกันเอง หรือระหว่างหุ่นยนต์กับคอมพิวเตอร์ จากนั้นจึงเตรียมอุปกรณ์ที่เกี่ยวข้องกับการพัฒนาระบบ ไม่ว่าจะเป็นการสร้างวัตถุสำหรับให้หุ่นยนต์เคลื่อนย้าย หรือการสร้างสนามสำหรับระบบหุ่นยนต์หลายตัว และขั้นตอนที่ผู้พัฒนากำลังดำเนินการอยู่ คือ พัฒนาวิธีการประมวลผลภาพที่ได้รับมาจากกล้องบนตัวหุ่นยนต์ เพื่อหาตำแหน่งของวัตถุ

อย่างไรก็ตาม ในระหว่างการดำเนินงาน ผู้พัฒนาได้พบปัญหามากมาย ซึ่งเกิดจากการที่ผู้พัฒนาไม่คุ้นเคยกับฮาร์ดแวร์และซอฟต์แวร์ที่ใช้ในการพัฒนาระบบ และเกิดจากข้อจำกัดของฮาร์ดแวร์และซอฟต์แวร์เอง ทั้งนี้ ผู้พัฒนาได้พยายามแก้ไขปัญหาให้เร็วที่สุด โดยการศึกษาข้อมูลเพิ่มเติม หรือใช้วิธีการอื่นมาทดแทน ปัญหาเหล่านี้ทำให้แผนการดำเนินงานจริงล่าช้ากว่าที่กำหนดไว้

นอกจากนี้ ยังมีขั้นตอนการดำเนินงานอีกหลายขั้นตอน ที่ผู้พัฒนายังไม่ได้เริ่มดำเนินการ ซึ่งผู้พัฒนาจะรีบดำเนินการให้ทันตามแผนการดำเนินงานที่ได้วางไว้ และผู้พัฒนาหวังเป็นอย่างยิ่งว่า ระบบหุ่นยนต์หลายตัวที่พัฒนาขึ้น จะต้องมีประสิทธิภาพดีกว่าระบบหุ่นยนต์ตัวเดียว ทั้งในแง่ของความเร็วในการทำงาน และความทนทานต่อความผิดพลาดที่เกิดขึ้น

7. เอกสารอ้างอิง

[1] AForge.NET. **AForge.NET:: Framework Features – Surveyor Robotics.**

[Online] Available from: http://www.aforgenet.com/framework/features/surveyor_robotics.html [2010].

[2] RoboRealm. **RoboRealm Tutorials.**

[Online] Available from: http://www.roborealm.com/tutorials.php [2010].

[3] Surveyor Corporation. **Surveyor SRV-1 Blackfin Setup.**

[Online] Available from: http://www.surveyor.com/blackfin/SRV_setup_bf.html [2010, Apr 24].

[4] Surveyor Corporation. **Definition of the SRV-1 Control Protocol (Blackfin Version).**

[Online] Available from: http://www.surveyor.com/SRV_protocol.html [2010, Apr 24].

[5] Webopedia. **What's Xmodem?.**

[Online] Available from: http://www.webopedia.com/TERM/X/Xmodem.html [2010].

[6] Wikipedia. **Tera Term.**

[Online] Available from: http://en.wikipedia.org/wiki/Tera_Term [2010, Sep 8].

[7] Surveyor Corporation. **SRV-1 C Interpreter.**

[Online] Available from: http://www.surveyor.com/C.html [2010, May 7].

[8] Google Project Hosting. **Picoc. A very small C interpreter..**

[Online] Available from: http://code.google.com/p/picoc/ [2010].

[9] Surveyor Robotics Forum. **Picoc image processing.**

[Online] Available from: http://www.surveyor.com/cgi-bin/yabb2/YaBB.pl?num=1275549473 [2010, Jun 3].

[10] Surveyor Robotics Forum. **Using picoC.**

[Online] Available from: http://www.surveyor.com/cgi-bin/yabb2/YaBB.pl?num=1239459317 [2009, Apr 11].

[11] Surveyor Robotics Forum. **Communication among several SRV-1 robots?.**

[Online] Available from: http://www.surveyor.com/cgi-bin/yabb2/YaBB.pl?num=1259086356 [2009, Nov 24].