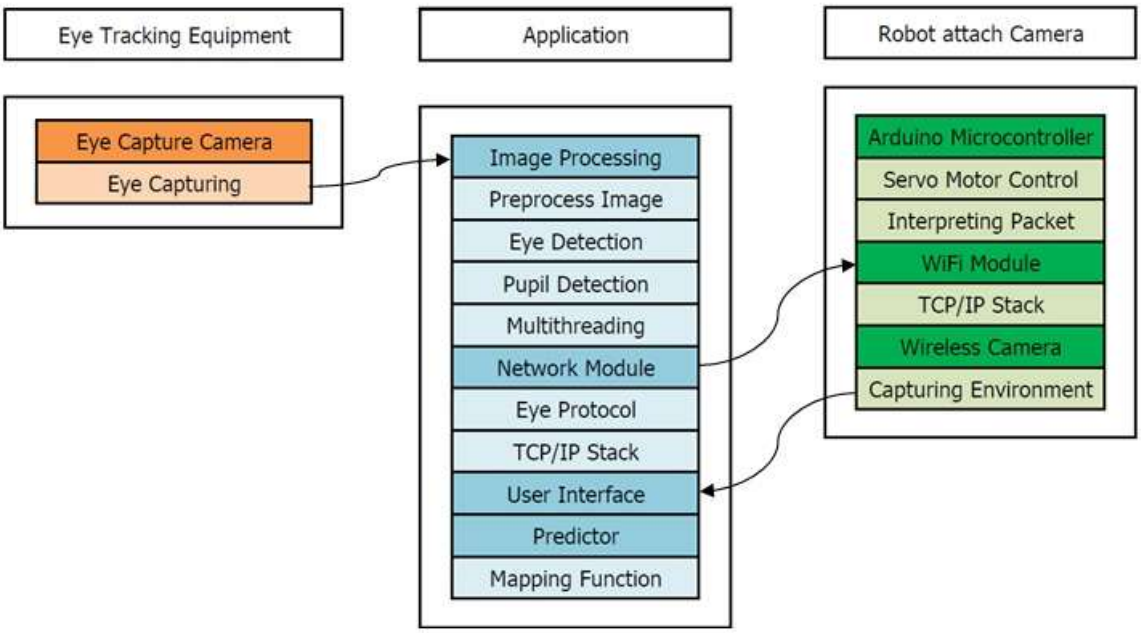
บทที่ 3
การออกแบบและพัฒนา

การออกแบบและพัฒนาระบบการควบคุมกล้องจับภาพบนตัวหุ่นยนต์ด้วยการหมุนของดวงตาแบ่งเป็น 3 ส่วนหลัก ดังแสดงในภาพที่ 3.1



ภาพที่ 3.1 Block diagram ของระบบ

3.1 อุปกรณ์ติดตั้งกล้องจับภาพดวงตา

3.1.1 ออกแบบอุปกรณ์ติดตั้ง

การออกแบบอุปกรณ์ติดตั้งจะออกแบบลักษณะคล้ายหมวกให้สวมใส่ได้ง่ายโดยมีกล้องติดอยู่ข้างหน้าจับการเคลื่อนไหวของดวงตาข้างซ้าย โดยกล้องที่ติดกับหมวกเป็นกล้อง Microsoft LifeCam ซึ่งจะส่งภาพของดวงตาไปยัง Application ด้วย USB 2.0 โดยการออกแบบส่วนนี้ทำขึ้นเพื่อป้องกันปัญหาที่อาจจะเกิดขึ้นกับการจับภาพดวงตา ซึ่งทำให้กล้องสามารถจับภาพดวงตาได้อย่างแม่นยำตลอดเวลาแม้จะเกิดการเคลื่อนไหวของศีรษะก็ตาม ภาพที่ 3.2 แสดงการออกแบบอุปกรณ์ติดตั้งกล้องจับภาพดวงตา และภาพที่ 3.3 แสดงอุปกรณ์ติดตั้งกล้องจับภาพดวงตาที่ได้ออกแบบ

ภาพที่ 3.2 การออกแบบอุปกรณ์ติดตั้งกล้องจับภาพดวงตา

ภาพที่ 3.3 อุปกรณ์ติดตั้งกล้องจับภาพดวงตาที่ได้ออกแบบ

3.1.2 กล้องจับภาพดวงตา

กล้องจับภาพดวงตาจะเลือกใช้กล้อง Microsoft LifeCam Studio ซึ่งสามารถถ่ายภาพได้ความละเอียดสูงสุดที่ 1920 x 1080 pixel ซึ่ง Microsoft LifeCam Studio มีรายละเอียดดังนี้

- 1080p HD Sensor ซึ่งให้คุณภาพของภาพและความคมชัดสูง
- 720p HD video chat

- ใช้วัสดุแก้วคุณภาพสูง lens จึงให้คุณภาพของภาพที่ดี
- TrueColor Technology ช่วยในการควบคุมความสว่างและสีสันโดยอัตโนมัติ-รองรับการใช้งาน Skype ในระดับ HD
- Wideband microphone สำหรับการอัดเสียงอย่างมีคุณภาพและให้เสียงที่เป็นธรรมชาติ
- หมุนได้ 360 องศาทั้ง 2 ทิศทางเพื่อมุมมองรอบด้าน
- Auto focus ตั้งแต่ระยะ 4 นิ้วจนถึงระยะอนันต์
- Wide-angle lens สามารถจับภาพรอบทิศทางมากขึ้น

โดยการที่ผู้พัฒนาเลือกใช้กล้องจับภาพชนิดนี้ เพราะเป็นกล้องที่มีความละเอียดของภาพสูงซึ่งจำเป็นอย่างยิ่ง เนื่องจากการเคลื่อนไหวของตาดำนั้นมองเห็นได้กว้างประมาณ ๑๐๐ องศา ทางด้านข้าง ๖๐ องศา ทางด้านบนและด้านใกล้จมูก และ ๗๕ องศา ทางด้านล่าง ในขณะที่การเคลื่อนที่ของตาดำในกล้องที่มีความละเอียดระดับ 1920 x 1080 pixel นั้นจะสามารถเคลื่อนไปได้ประมาณ 450 pixel ทางด้านข้าง 250 pixel ทางด้านบนและด้านใกล้จมูก และ 300 pixel ทางด้านล่าง ซึ่งเมื่อนำการเปลี่ยนแปลงของ pixel มาใช้ในการขับเคลื่อนแขนหุ่นยนต์จะสามารถเคลื่อนแขนหุ่นยนต์ได้อย่างละเอียดและและมีความล่ายของแขนที่น้อย นอกจากนี้ กล้อง Microsoft Lifecam Studio ยังมีระยะโฟกัสที่น้อยที่สุดที่ห่างพอดีกับระยะห่างระหว่างกล้องกับดวงตาที่เหมาะสมอีกด้วย

3.2 Application

Application ทำงานบนสภาพแวดล้อมของระบบปฏิบัติการ Window 7 โดย Application จะทำหน้าที่หลัก 2 หน้าที่คือประมวลผลภาพจากกล้องจับภาพดวงตาเพื่อตรวจหาดวงตาและทำนายทิศทางของดวงตาดำ อีกหนึ่งหน้าที่คือการรับส่งข้อมูลด้วยระบบ Network บน TCP เพื่อส่งข้อมูลไปยังตัวหุ่นยนต์และรับข้อมูลภาพและคำสั่งจากตัวหุ่นยนต์ได้ ซึ่งการพัฒนา Application จะประกอบด้วยการเลือก Software Development Tool และแนวคิดของการประมวลผลภาพดังนี้

3.2.1 Software Development Tool and Development Language

การพัฒนา Application จะพัฒนาบนสภาพแวดล้อมของระบบปฏิบัติการ Window 7 ซึ่งประกอบด้วย Software Tool เครื่องมือที่ใช้ในการพัฒนาระบบขึ้นดังนี้

1) Microsoft Visual Studio 2008 เป็น Software Tool หลักที่ใช้ในการพัฒนา Window Application Form ซึ่งทาง Microsoft ผู้ออกแบบ Software Tool นี้เพื่อให้พัฒนา Application เพื่อทำงานบนระบบปฏิบัติการของ Window

Microsoft Visual Studio 2008 สามารถสร้าง Window Form ได้อย่างง่ายดาย ด้วย User Graphic Interface และสามารถเขียนโปรแกรมเพื่อควบคุมการทำงาน และ Event ที่เข้าใจง่าย นอกจากนี้มี Text Editor ช่วยทำให้ง่ายต่อการพัฒนาและลดความผิดพลาดในการเขียนได้ Microsoft Visual Studio 2008 มี User Interface ที่ง่ายต่อการ Debug โปรแกรมด้วยการประมวลผลคำสั่งแบบ Break Point ได้และสามารถดูค่าตัวแปรและ Address ของตัวแปรได้ง่าย ดังนั้นทางกลุ่มผู้พัฒนาจึงเลือก Software Tool นี้เพื่อใช้ในการพัฒนา Application

2) ภาษา C# เป็นภาษาเชิง Object Oriented Programming มี Syntax คล้ายกับภาษา Visual Basic สามารถเข้าใจได้ง่าย มีความปลอดภัยในการเข้าถึงตัวแปรด้วย Scope ของ Object สามารถพัฒนา Application ได้โดยใช้เวลาไม่นาน ซึ่งทางกลุ่มจะเลือกใช้ภาษา C# ในการพัฒนา Application บน Microsoft Visual Studio 2008

3) EmguCV 2.3 Library คือ Opensource Library เกี่ยวกับการประมวลผลภาพของคอมพิวเตอร์ซึ่งมีการเปิดเผย SourceCode ผู้พัฒนาสามารถแก้ไขหรือปรับเปลี่ยน Algorithm ให้เหมาะสมตามที่ต้องการได้ EmguCV จะใช้พัฒนาร่วมกับภาษา C# ซึ่งประกอบด้วย Function Call ที่สำเร็จรูปมากมายเกี่ยวกับการประมวลผลภาพ ซึ่งใน Library นี้มีการรวมการประมวลผลภาพเพื่อตรวจหาอวัยวะในส่วนใบหน้าด้วย Haar Algorithm ซึ่งมีความแม่นยำอยู่ในระดับที่ดี นอกจากนี้ยังสามารถหาแหล่งอ้างอิงของข้อมูลได้ง่ายเนื่องจาก EmguCV 2.3 Library ค่อนข้างได้รับความนิยมในการเขียนโปรแกรมเพื่อพัฒนา Application ประมวลผลภาพ

4) MATLab เป็น Software Tool เพื่อใช้ในการประมวลผล Matrix ที่ได้รับความนิยมในทางวิศวกรรม ในการประมวลผลภาพ MATLab ได้รับความนิยมเนื่องจากภาพคือ Matrix ของสีในแต่ละ Pixel ดังนั้น MATLab สามารถใช้ในการประมวลผลภาพด้วยการทำ Operation บน Matrix ซึ่ง MATLab ใช้ภาษา MATLab C ในการพัฒนา MATLab สามารถพัฒนาได้ด้วยระยะเวลาไม่นาน และมี Function ที่เกี่ยวกับการจัดการ Matrix ของภาพค่อนข้างหลากหลาย เช่น การแยก Channel ของภาพ, การตรวจหาขอบของรูปภาพ, และการแปลงภาพเป็น Greyscale เป็นต้น เนื่องจาก MATLab สามารถประมวลผลข้อมูลใน Matrix ในแต่ละ Offset ได้ทำให้สามารถประมวลผลด้วย Algorithm ที่สามารถออกแบบได้เอง ซึ่งทางกลุ่มพัฒนาได้นำ MATLab มาเพื่อเขียน Function ในหารตรวจหาใบหน้าและดวงตา จากนั้น Export Function ในการประมวลผล

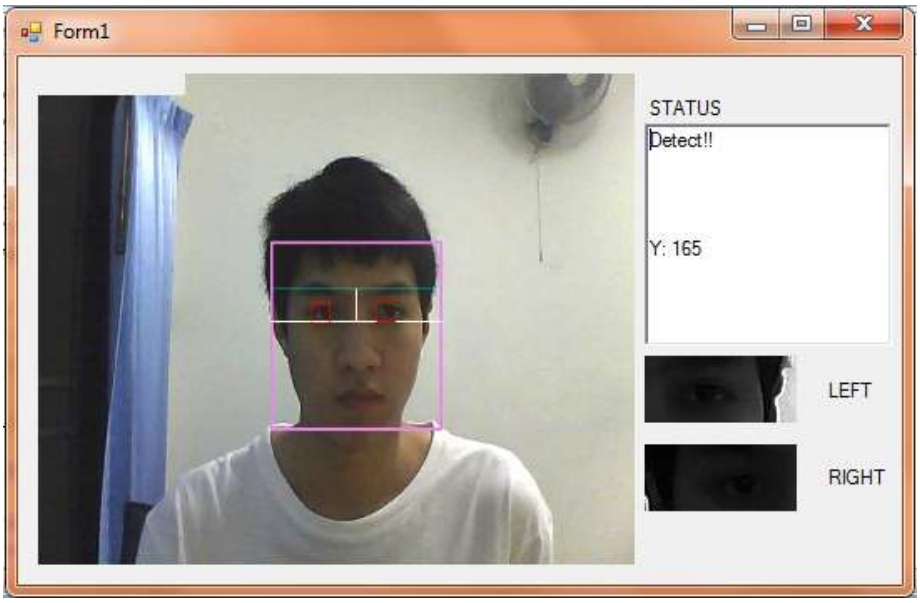
ภาพที่ได้เขียนไว้มาเป็นนามสกุล dll ได้ซึ่งสามารถนำไป Interface กับภาษา C# ได้

3.2.2 Graphic User Interface

การออกแบบหน้าจอของ Graphic User Interface ของ Application จะแบ่งเป็น 2 ส่วนหลักดังนี้

3.2.2.1 การออกแบบ User Interface เพื่อใช้ในการพัฒนา

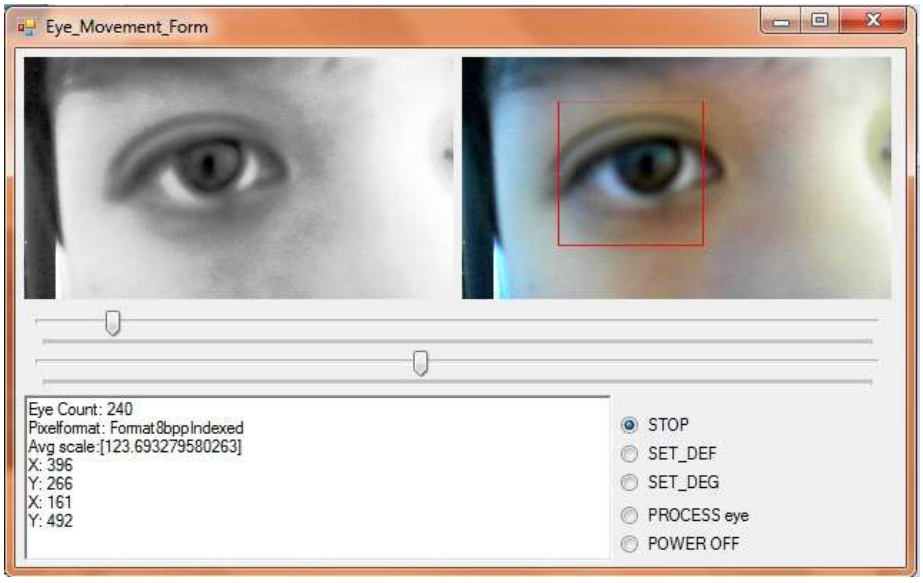
การออกแบบ User Interface เพื่อใช้ในการพัฒนาหรือทดสอบการประมวลผลภาพจะต้องออกแบให้ง่ายต่อการดูผลลัพธ์ของการประมวลผลภาพและง่ายต่อการหาข้อผิดพลาดการทำงานของ Application ซึ่งหน้าจอ Application จะประกอบด้วยการแสดงผลภาพหลาย Picture Box และประกอบด้วย Textbox เพื่อใช้ในการ Input ค่าและแสดงผล Output ค่าแบบ Text ในการตรวจหาข้อผิดพลาด ซึ่งการออกแบบหน้าจอ User Interface ในการพัฒนานั้นจะเปลี่ยนแปลงไปตามความเหมาะสมของการพัฒนา เช่น ในช่วงแรกของการพัฒนาจะเน้นในการประมวลผลภาพจึงออกแบบหน้าจอให้แสดงผลภาพด้วยขนาดที่เหมาะสม ในภายหลังเมื่อสามารถประมวลผลภาพได้ตามต้องการแล้วได้ออกแบบให้มีการแสดงข้อความเพื่อดูผลลัพธ์ตำแหน่งที่ตรวจจับได้เพื่อตรวจสอบความถูกต้อง เป็นต้น ภาพที่ 3.4 แสดง Graphic User Interface สำหรับการพัฒนาเพื่อแสดงผลภาพการตรวจจับใบหน้า



ภาพที่ 3.4 Graphic User Interface สำหรับการพัฒนาเพื่อแสดงผลภาพการตรวจจับใบหน้า

โดยภาพที่ 3.4 แสดงถึง Graphic User Interface ที่ออกแบบเพื่อในใช้การตรวจหาใบหน้าด้วยกล้อง Web Camera บน Notebook Computer เท่านั้น รายละเอียดของ Graphic User Interface แสดงดังภาพที่ 3.5 มีดังนี้

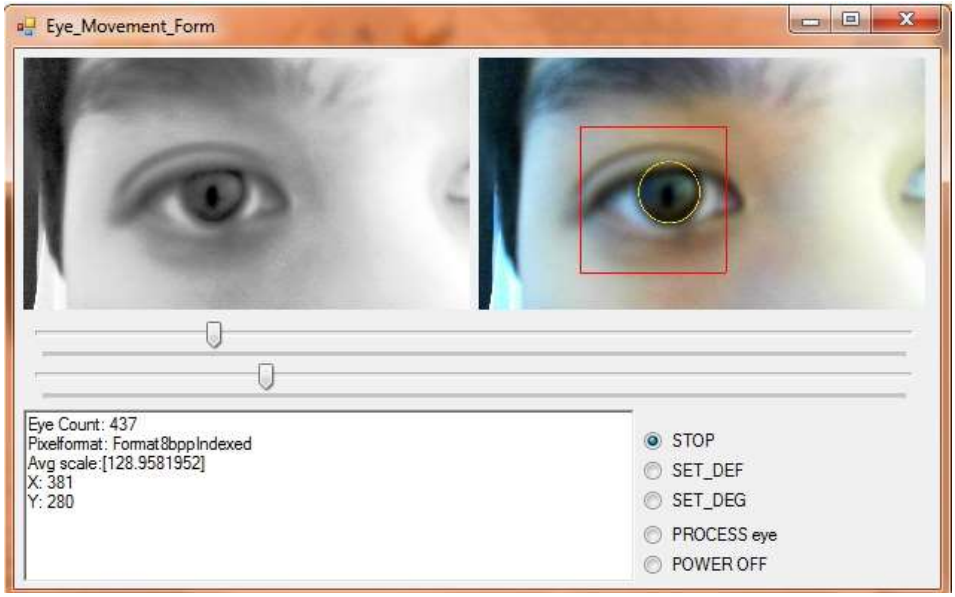
1) Pre processing image คือภาพ raw image จากกล้องจับดวงตาที่ถูกนำมาประมวลผลก่อนในครั้งแรกเพื่อนำไปประมวลผลภาพตรวจจับใบหน้าต่อไป
2) Post processing image คือภาพที่นำมาประมวลผลในการตรวจหาใบหน้าโดยการวาดกรอบสี่เหลี่ยมให้เห็นว่าตรวจจับใบหน้าที่ตำแหน่งใดและมีขนาดของสี่เหลี่ยมเป็นเท่าใดและตำแหน่งที่คาดว่าจะตรวจจับดวงตาคือตำแหน่งใบของใบหน้า
3) Left eye image และ Right eye image จะแสดงภาพดวงตาซ้ายและดวงตาขวาที่ Crop มาจากภาพ Pre processing image เพื่อแสดงให้เห็นว่าการทำ Pre processing image ส่งผลให้เห็นดวงตาดำชัดเท่าใด
4) Information Textbox จะแสดงข้อมูลที่ใช้ในการพัฒนาในการประมวลผลภาพ เช่น พิกัดของใบหน้าและความกว้างของกรอบใบหน้า, พิกัดของดวงตาซ้ายและขวา, และจำนวนภาพที่ถูก process ติดต่อกัน เป็นต้น



ภาพที่ 3.5 Graphic User Interface สำหรับการพัฒนาเพื่อแสดงผลภาพการตรวจจับดวงตา

Graphic User Interface นี้ออกแบบเพื่อในใช้การตรวจหาดวงตาด้วยกล้อง TP Web Camera M490 ที่มีความละเอียด 640 x 480 รายละเอียดของ Graphic User Interface ดังภาพที่ 3.5 มีดังนี้

1) Pre processing image คือภาพ raw image จากกล้องจับดวงตาที่ถูกนำมาประมวลผลก่อนในครั้งแรกเพื่อนำไปประมวลผลภาพตรวจจับใบหน้าต่อไป
2) Post processing image คือภาพที่นำมาประมวลผลในการตรวจหาดวงตาโดยการวาดกรอบสี่เหลี่ยมให้เห็นว่าตรวจจับดวงตาที่ตำแหน่งใดและมีขนาดของสี่เหลี่ยมเป็นเท่าใด
3) Information Textbox จะแสดงข้อมูลที่ใช้ในการพัฒนาในการประมวลผลภาพ เช่น พิกัดของดวงตา, ขนาดความกว้างและความยาวของกรอบดวงตา, จำนวนภาพที่ประมวลผลได้ในหนึ่งวินาที, และจำนวนภาพที่ถูก process ติดต่อกัน เป็นต้น

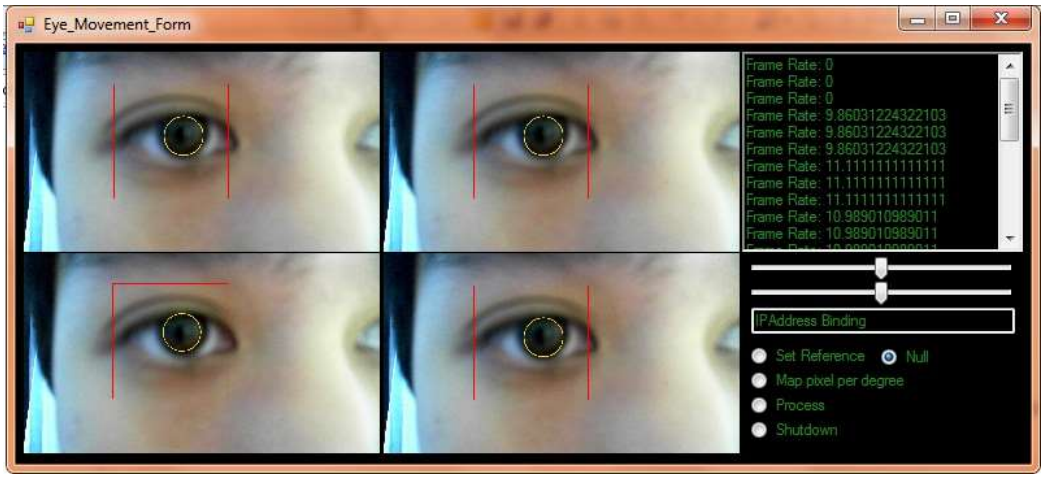


ภาพที่ 3.6 Graphic User Interface สำหรับการพัฒนาเพื่อแสดงผลภาพการตรวจจับรูม่านตาหรือดวงตาดำ

Graphic User Interface นี้ออกแบบเพื่อในใช้การตรวจหาดวงตาด้วยกล้อง Microsoft LifeCam ที่มีความละเอียด 1920*1080 ดังภาพที่ 3.6

รายละเอียดของ Graphic User Interface มีดังนี้

1) Pre processing image คือภาพ raw image จากกล้องจับดวงตาที่ถูกนำมาประมวลผลก่อนในครั้งแรกเพื่อนำไปประมวลผลภาพตรวจจับใบหน้าต่อไป
2) Post processing image คือภาพที่นำมาประมวลผลในการตรวจหาดวงตาโดยการวาดกรอบสี่เหลี่ยมให้เห็นว่าตรวจจับดวงตาที่ตำแหน่งใดมีขนาดของสี่เหลี่ยมเป็นเท่าใด และแสดงการตรวจหารูม่านตาด้วยการแสดงวงกลมภายในกรอบของดวงตาที่ตรวจพบ
3) Information Textbox จะแสดงข้อมูลที่ใช้ในการพัฒนาในการประมวลผลภาพ เช่น พิกัดของดวงตา, ขนาดความกว้างและความยาวของกรอบดวงตา, จำนวนภาพที่ประมวลผลได้ในหนึ่งวินาที, และจำนวนภาพที่ถูก process ติดต่อกัน เป็นต้น



ภาพที่ 3.7 Graphic User Interface สำหรับการพัฒนาเพื่อแสดงผลภาพด้วย Multithread Processing

Graphic User Interface นี้ออกแบบเพื่อในใช้การตรวจหาดวงตาด้วยกล้อง Microsoft LifeCam ที่มีความละเอียด 1920*1080 และทำการประมวลผลด้วย 4 Threads เพื่อประมวลผลตรวจหาดวงตาดำและรูม่านตาดังภาพที่ 3.7

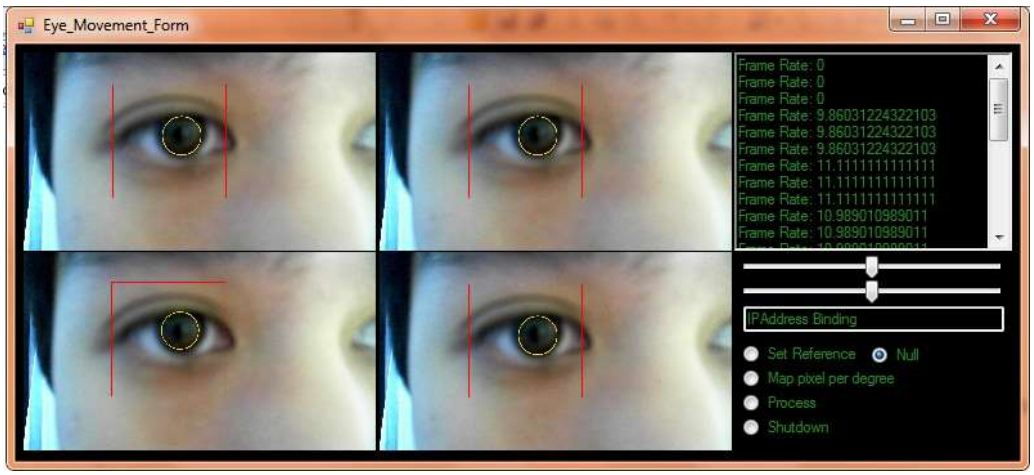
รายละเอียดของ Graphic User Interface มีดังนี้

1) Post processing image จำนวน 4 Picture Box คือภาพที่นำมาประมวลผลในการตรวจหาดวงตาโดยการวาดกรอบสี่เหลี่ยมให้เห็นว่าตรวจจับดวงตาที่ตำแหน่งใดมีขนาดของสี่เหลี่ยมเป็นเท่าใด และแสดงการตรวจหารูม่านตาด้วยการแสดงวงกลมภายในกรอบของ

ดวงตาที่ตรวจพบ การแสดงผลจะเป็น 4 Picture Box ซึ่งแสดงผลให้เห็นในการประมวลผลภาพของแต่ละ Thread

2) Track bar จะแสดงผลให้เห็นตำแหน่งของรูม่านตาที่ตรวจจับได้ในแกน X และแกน Y เพื่อให้ง่ายต่อการดูผลลัพธ์มากกว่าการใช้ Textbox

3) Information Textbox จะแสดงข้อมูลที่ใช้ในการพัฒนาในการประมวลผลภาพด้วย 4 Thread เนื่องจาก Thread มีการทำงานด้วยการ Schedule ของระบบปฏิบัติการดังนั้นการ Debug ด้วย Break Point จึงทำได้ยากแต่ใช้การ Debug ด้วยการแสดงผลเป็น Text ออกทาง Textbox แทนซึ่งแสดงผล เช่น พิกัดของดวงตา, ขนาดความกว้างและความยาวของกรอบดวงตา, จำนวนภาพที่ประมวลผลได้ในหนึ่งวินาที, จำนวนภาพที่ถูก process ติดต่อกัน, ลำดับการเข้าทำงานของ Thread, และลำดับการประมวลผลเสร็จของ Thread เป็นต้น



ภาพที่ 3.8 Graphic User Interface สำหรับการพัฒนาเพื่อหาค่าที่เหมาะสมในการตรวจจับรูม่านตา

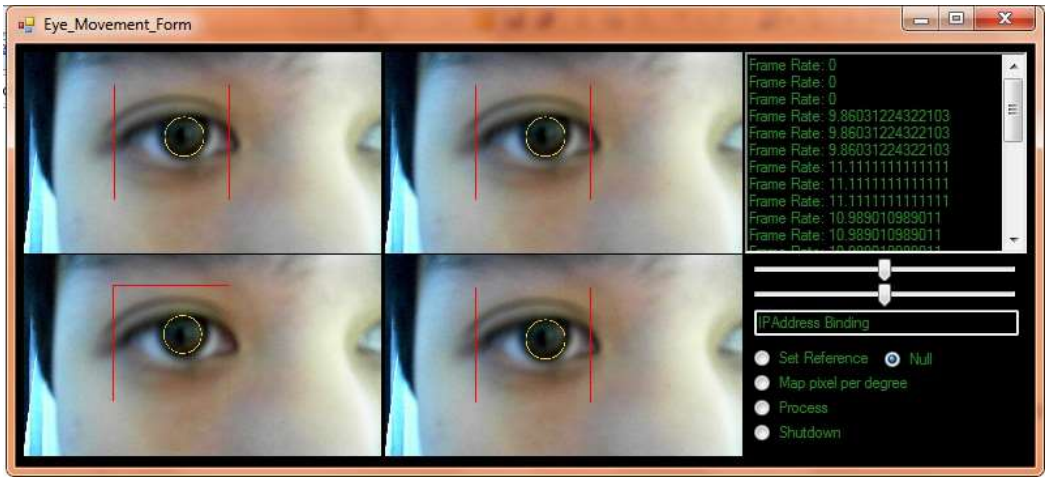
Graphic User Interface นี้ออกแบบเพื่อในใช้การตรวจหาดวงตาด้วยกล้อง Microsoft LifeCam ที่มีความละเอียด 1920*1080 และทำการกำหนดค่าที่เหมาะสมในการใช้ตรวจหาตำแหน่งของรูม่านตาด้วยการกำหนดค่าตัวแปรต่างๆ ดังภาพที่ 3.8

รายละเอียดของ Graphic User Interface มีดังนี้

1) Post processing image จำนวน 4 Picture Box คือภาพที่นำมาประมวลผลในการตรวจหาดวงตาโดยการวาดกรอบสี่เหลี่ยมให้เห็น

ว่าตรวจจับดวงตาที่ตำแหน่งใดมีขนาดของสี่เหลี่ยมเป็นเท่าใด และแสดงการตรวจหารูม่านตาด้วยการแสดงวงกลมภายในกรอบของดวงตาที่ตรวจพบ การแสดงผลจะเป็น 4 Picture Box ซึ่งแสดงผลให้เห็นในการประมวลผลภาพของแต่ละ Thread

2) Track bar จะแสดงผลให้เห็นตำแหน่งของรูม่านตาที่ตรวจจับได้ในแกน X และแกน Y เพื่อให้ง่ายต่อการดูผลลัพธ์มากกว่าการใช้ Textbox
3) Input Textbox ใช้เพื่อการรับค่าเพื่อนำไปใช้ในการกำหนดค่า Parameter ในการตรวจหาตำแหน่งของรูม่านตาให้เหมาะสมโดยจะประกอบด้วย 6 Input Textbox
4) Information Textbox จะแสดงข้อมูลที่ใช้ในการพัฒนาในการประมวลผลภาพด้วย 4 Thread เนื่องจาก Thread มีการทำงานด้วยการ Schedule ของระบบปฏิบัติการดังนั้นการ Debug ด้วย Break Point จึงทำได้ยากแต่ใช้การ Debug ด้วยการแสดงผลเป็น Text ออกทาง Textbox แทนซึ่งแสดงผล เช่น พิกัดของดวงตา, ขนาดความกว้างและความยาวของกรอบดวงตา, จำนวนภาพที่ประมวลผลได้ในหนึ่งวินาที, และจำนวนภาพที่ถูก process ติดต่อกัน, เป็นต้น



ภาพที่ 3.9 Graphic User Interface สำหรับการพัฒนาเพื่อทดสอบการส่งข้อมูลผ่านระบบ Network ด้วย Protocol ที่คิดขึ้นมา

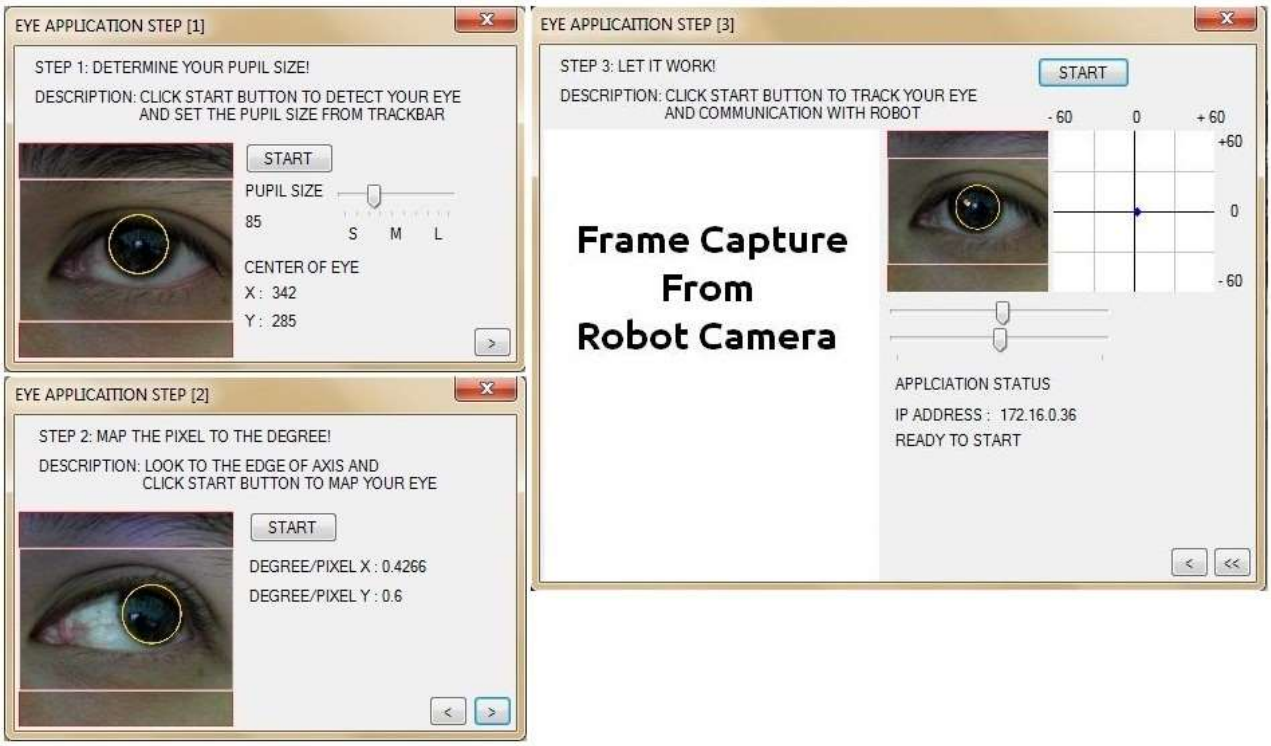
Graphic User Interface นี้ออกแบบเพื่อในใช้การทดสอบการส่งข้อมูลด้วย Protocol ที่ผู้จัดทำคิดขึ้นมา ดังภาพที่ 3.9

รายละเอียดของ Graphic User Interface มีดังนี้

1) Post processing image จำนวน 4 Picture Box คือภาพที่นำมาประมวลผลในการตรวจหาดวงตาโดยการวาดกรอบสี่เหลี่ยมให้เห็นว่าตรวจจับดวงตาที่ตำแหน่งใดมีขนาดของสี่เหลี่ยมเป็นเท่าใด และแสดงการตรวจหารูม่านตาด้วยการแสดงวงกลมภายในกรอบของดวงตาที่ตรวจพบ การแสดงผลจะเป็น 4 Picture Box ซึ่งแสดงผลให้เห็นในการประมวลผลภาพของแต่ละ Thread
2) Track bar จะแสดงผลให้เห็นตำแหน่งของรูม่านตาที่ตรวจจับได้ในแกน X และแกน Y เพื่อให้ง่ายต่อการดูผลลัพธ์
3) Input Textbox ใช้เพื่อการรับค่าเพื่อนำไปใช้ในการกำหนดค่า Parameter ในการตรวจหาตำแหน่งของรูม่านตาให้เหมาะสม
4) Information Textbox จะแสดงข้อมูลที่ใช้ในการส่งและรับข้อมูลผ่านระบบ Network ที่จะติดต่อกับ Wi -Fi Module บนตัวหุ่นยนต์

3.2.2.2 การออกแบบ User Interface เพื่อการใช้งานของ User

การออกแบบ Graphic User Interface เพื่อนำไปใช้งานกับ User จะออกแบบให้มีลักษณะการใช้งานที่ง่ายที่สุดและเน้นความสวยงามของ Application ดังภาพที่ 3.10



ภาพที่ 3.10 User Interface สำหรับใช้การใช้งานของ User

การออกแบ User Interface จะแบ่งเป็น 3 หน้าต่าง ได้แก่ หน้าต่างการตั้งค่าจุดกึ่งกลางของดวงตา, หน้าต่างการตั้งค่าการแปลง Pixel เป็นองศา, และหน้าต่างการประมวลผลดวงตา

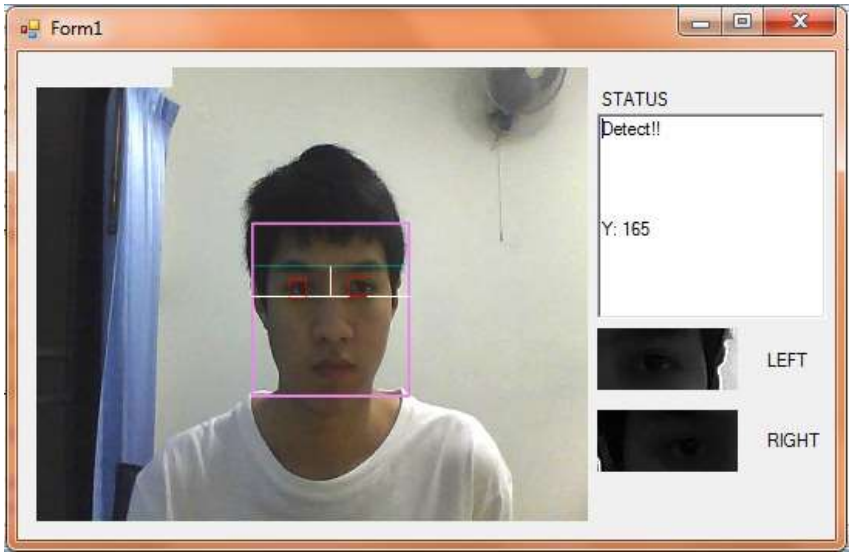
รายละเอียดของ Graphic User Interface แต่ละหน้าต่าง มีดังนี้

1) Eye Application Step [1] ใช้ในการตั้งค่าจุดกึ่งกลางของดวงตา ประกอบด้วยปุ่ม Start เพื่อเริ่มการทำงาน นอกจากนี้สามารถปรับขนาดของรูม่านตาให้เหมาะสมด้วย Trackbar ได้
2) Eye Application Step [2] ใช้ในการตั้งค่าการแปลง Pixel เป็นองศา ประกอบด้วยปุ่ม Start เพื่อเริ่มการทำงาน นอกจากนี้มีปุ่ม <, > ใช้ในการเปลี่ยนหน้าต่างไปยังหน้าต่างก่อนหน้าและหน้าต่างถัดไป
3) Eye Application Step [3] ใช้ในการเริ่มการทำงานของระบบ ประกอบด้วยปุ่ม Start เพื่อเริ่มการทำงาน, Picturebox ด้านซ้ายแสดงผลลัพธ์ภาพที่ถ่ายจากตัวหุ่นยนต์, Picturebox ด้านขวาแสดงถึงภาพดวงตาที่ประมวลผลแล้ว, กราฟแสดงถึงตำแหน่งการมองของดวงตา ประกอบด้วยแกน X, Y ซึ่งจะแสดงด้วยค่าองศา, Trackbar จะแสดงค่าองศาในแนวแกน X, Y คล้ายกับกราฟ, Text Label จะแสดง IP Address ของ Application และแสดงสถานะการทำงานของระบบ

3.2.3 การประมวลผลภาพเพื่อตรวจหาใบหน้า

การประมวลผลภาพเพื่อตรวจหาตำแหน่งของใบหน้าจะใช้ EmguCV Library ซึ่ง EmguCV มี Function สำเร็จรูปในการตรวจหารูปร่างของใบหน้า เรียกว่า Haar Detection ซึ่งเป็น Algorithm ในการหาวัตถุในภาพโดยการสร้างสี่เหลี่ยมขึ้นมาจำนวนมากและใช้ค่าถ่วงน้ำหนักในการตัดสินใจว่าเป็นวัตถุที่ต้องการหรือไม่ Haar Detection ใบ EmguCV สามารถตรวจสอบใบหน้าได้โดยอ้างอิงจาก Face.xml ซึ่งเป็น File ที่เก็บข้อมูลสถิติเพื่อใช้ในการตัดสินใจวัตถุในรูปภาพ

การตรวจสอบใบหน้าทำได้โดยการนำภาพที่ถ่ายจากกล้อง Web Camera ของ Notebook ขนาด 640 x 480 มาทำการ Preprocess ดังภาพที่ 3.11 โดยการแปลงภาพเป็น Grey Scale และทำ Equalizehist เพื่อทำให้ระดับสีเทาของภาพมีความเข้มขึ้น จากนั้นจึงนำภาพที่ทำ Preprocess แล้วมาเป็นภาพ Input ของกระบวนการ Haar Face Detection ซึ่งสามารถเลือกการตรวจหาภาพได้หลายแบบ เช่น Canny Pruning และ Biggest Object ซึ่งจะตรวจหาวัตถุที่ใหญ่ที่สุดผลลัพธ์ที่ได้จะเป็นวัตถุเพียงรูปเดียวเท่านั้น เป็นต้น เมื่อทำ Haar Face Detection แล้วจะได้ตำแหน่งพิกัดมุมซ้ายบนของใบหน้าที่ตรวจสอบพบและขนาดของสี่เหลี่ยมที่ตรวจสอบพบ

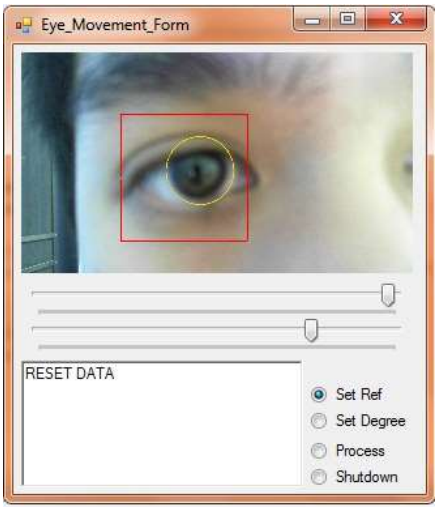


ภาพที่ 3.11 การประมวลผลภาพเพื่อตรวจหาใบหน้า

3.2.4 การประมวลผลภาพเพื่อตรวจหาดวงตา

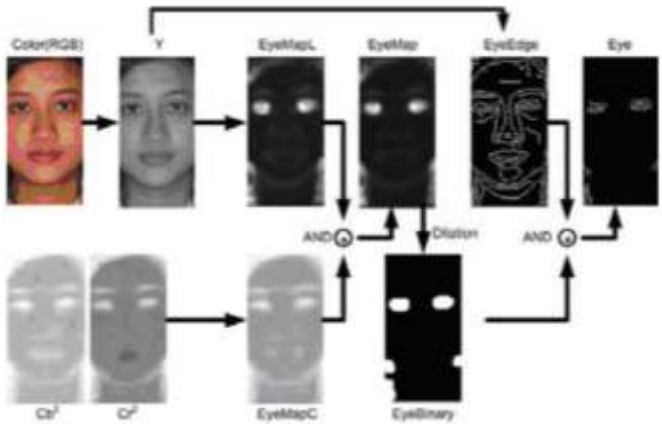
การประมวลผลภาพเพื่อตรวจหาตำแหน่งของดวงตาในรูปภาพ ทางกลุ่มพัฒนาได้ทำการทดลอง 2 วิธีในการตรวจหาตำแหน่งของดวงตาดังนี้

1) กระบวนการตรวจหาดวงตาโดยใช้ EmguCV Library โดยใช้ Haar Eye Detection ซึ่งใน EmguCV มี Function ที่สำเร็จรูปในการหาตำแหน่งของดวงตา โดยการหาตำแหน่งดวงตาเริ่มจากการนำภาพที่ถ่ายจากกล้อง Microsoft LifeCam ที่มีขนาด 1920 x 1080 มาทำการ Preprocess โดยการแปลงภาพเป็น Grey Scale และทำ Equalizehist เพื่อทำให้ระดับสีเทาของภาพมีความเข้มขึ้น จากนั้นจึงนำภาพที่ทำ Preprocess แล้วมาเป็นภาพ Input ของกระบวนการ Haar Eye Detection ซึ่งสามารถเลือกการตรวจหาภาพได้หลายแบบ เช่น Canny Pruning และ Biggest Object ซึ่งจะตรวจหาวัตถุที่ใหญ่ที่สุดผลลัพธ์ที่ได้จะเป็นวัตถุเพียงรูปเดียวเท่านั้น เป็นต้น เมื่อทำ Haar Eye Detection แล้วจะได้ตำแหน่งพิกัดมุมซ้ายบนของดวงตาที่ตรวจสอบพบและขนาดของสี่เหลี่ยมที่ตรวจสอบพบ ภาพที่ 3.12 แสดงการประมวลผลภาพเพื่อตรวจหาดวงตาโดยใช้ EmguCV



ภาพที่ 3.12 การประมวลผลภาพเพื่อตรวจหาดวงตาโดยใช้ EmguCV

2) การตรวจหาดวงตาด้วยการประมวลผลด้วย MATLab ซึ่งการตรวจหาดวงตาจะทำโดย เริ่มจากการนำภาพที่ถ่ายจากกล้อง Microsoft LifeCam ที่มีขนาด 1920 x 1080 มาทำการแปลง Channel ของสีเป็น YCbCr และนำ Channel ของ Y ไปทำกระบวนการ EyeMapL ส่วน Channel Cb, Cr ให้นนำมาทำกระบวนการ EyeMapC และให้นำภาพผลลัพธ์ของ EyeMapL และ EyeMapC มาทำ Operation And กัน ผลลัพธ์ของการทำ And เรียกว่า EyeMap ให้นำภาพผลลัพธ์นี้ไปทำกระบวนการ Dilation จะได้เป็นภาพ Binary ที่มีบริเวณสีขาวเป็นบริเวณดวงตา ให้นำภาพ Binary ที่ได้ไปทำกระบวนการ And กับภาพ Y ที่ได้ทำ Edge Detection ทำให้ได้ภาพเฉพาะส่วนดวงตาตามที่ต้องการ ดังภาพที่ 3.13



ภาพที่ 3.13 การประมวลผลภาพเพื่อตรวจหาดวงตาโดยใช้ Matlab

ซึ่งกระบวนการทั้งหมดทำโดยการเขียน function ใน MATLab และ Export ออกมาเป็น Dll extension เพื่อนำไปเชื่อมโยงกับภาษา C# ได้

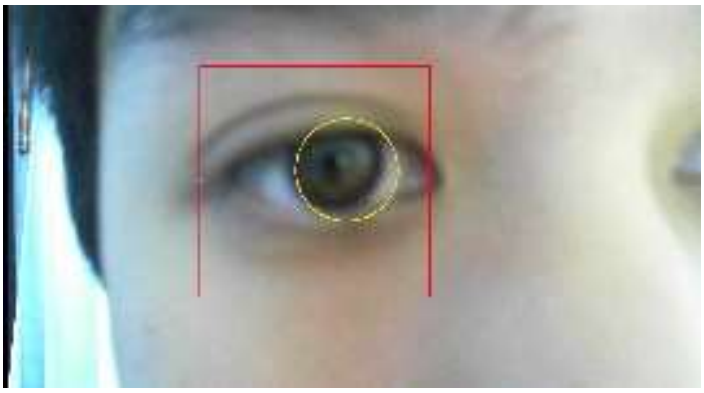
เปรียบเทียบกระบวนการหาตำแหน่งของดวงตาจาก 2 วิธีได้ดังนี้

1) การหาดวงตาโดยใช้ Haar Eye Detection มีความแม่นยำในการตรวจหาตำแหน่งดวงตามากกว่ากระบวนการ EyeMap ซึ่งเขียนใน MATLab
2) ประสิทธิภาพของความเร็วในการประมวลผล EmguCV สามารถทำได้รวดเร็วกว่าการประมวลผลด้วย Function ที่ Export มาจาก MATLab มากจนเห็นได้ชัด

ดังนั้นในการประมวลผลภาพดวงตาทางกลุ่มผู้พัฒนาจึงเลือกใช้ EmguCV Library ด้วย Haar Eye Detection เพื่อนำมาตรวจหาตำแหน่งของดวงตา

3.2.5 การประมวลผลภาพเพื่อตรวจหารูม่านตา

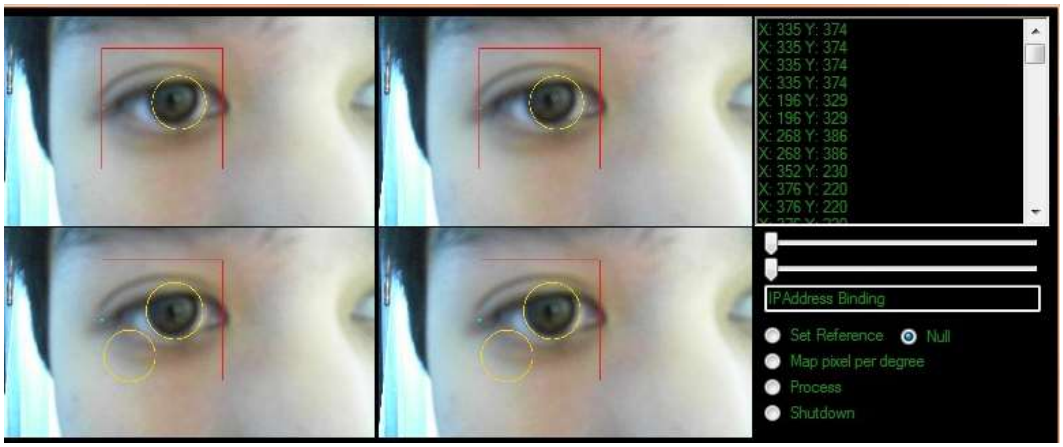
การประมวลผลภาพเพื่อตรวจหาตำแหน่งรูม่านตาทำโดยการนำภาพที่ได้แปลงเป็น Grey Scale และ Equalizehist นำมาทำ ROI เพื่อด้วยตำแหน่งของดวงตาที่ตรวจหาพบจากการบวนการ Eye Detection จากนั้นนำภาพ ROI ไปทำการหาวงกลมในภาพโดยใช้ EmguCV Library ด้วย Circle Detection ในการหาวงกลมจะต้องกำหนดค่า Threshold เพื่อแยกแยะระดับ Grey Scale ของวงกลมที่ต้องการหาและต้องกำหนดขนาดของวงกลมที่ต้องการหาได้แก่ ค่าระดับสี Threshold ของการทำ Canny Edge Detection, ค่าระดับสี Threshold ของการทำ Accumulator, ค่ารัศมีของวงกลมน้อยที่สุดและมากที่สุดที่ต้องการหา, และค่าระยะมากที่สุดและน้อยที่สุดของวงกลมวงถัดไปที่ต้องการหาอ้างอิงจากวงกลมวงที่ตรวจสอบพบ เป็นต้น การหาตำแหน่งของรูม่านตาจะได้พิกัดกึ่งกลางของวงกลมและขนาดของรัศมี โดยภาพที่ 3.14 แสดงการประมวลผลภาพเพื่อตรวจหารูม่านตา



ภาพที่ 3.14 การประมวลผลภาพเพื่อตรวจหารูม่านตา

3.2.6 การหาตำแหน่งอ้างอิงของรูม่านตา

การหาตำแหน่งอ้างอิงของรูม่านตาเพื่อใช้เป็นจุดอ้างอิงกับรูม่านตาเมื่อขยับไปในทิศทางอื่นๆ ซึ่งในการหาตำแหน่งอ้างอิงของรูม่านตาทำได้โดยการให้ผู้ทดสอบมองตรงไปข้างหน้าเพื่อให้รูม่านตาอยู่กึ่งกลางของดวงตานั่นเอง จากนั้นโปรแกรมจำทำการตรวจหาดวงตาและหาพิกัดกึ่งกลางของรูม่านตา โดยทำการประมวลผลภาพ 35 ภาพติดต่อกันและหาค่าเฉลี่ยของพิกัดกึ่งกลางของรูม่านตาแกน X, Y ที่ตรวจสอบพบ จากนั้นจะนำค่าที่ได้ไปทำการอ้างอิงเมื่อมีการเคลื่อนไหวของดวงตา ดังภาพที่ 3.15



ภาพที่ 3.15 การประมวลผลภาพเพื่อตรวจหาตำแหน่งอ้างอิงของรูม่านตา

3.2.7 การแปลง Pixel เป็นองศา

เมื่อทำการหาตำแหน่งอ้างอิงของรูม่านตาแล้วจะต้องทำการหาว่าดวงตาเคลื่อนที่ไปเพื่อมองในทิศทางใด วิธีในการคำนวณว่าดวงตาจะมองในทิศทางใดจะทำการหาตำแหน่งของรูม่านตาปัจจุบันและนำมาเทียบกับตำแหน่งของรูม่านตาอ้างอิงที่ได้ทำไว้ก่อนแล้ว

กระบวนหาทิศทางเริ่มจากการตรวจสอบตำแหน่งพิกัดของรูม่านตานำมาหาระยะห่างที่สัมพันธ์กับตำแหน่งพิกัดอ้างอิงของรูม่านตา โดยการนำพิกัด X ของรูม่านตาปัจจุบันมาลบกับพิกัด X ของรูม่านตาอ้างอิงและทำในการ Y เช่นกัน เมื่อนำมาลบกันแล้วจะได้ทิศทางการเคลื่อนที่ของรูม่านตาดังนี้

1) ค่าระยะห่างของพิกัดแกน X เทียบกับจุดอ้างอิงเป็นค่าติดลบ = ดวงตาเคลื่อนที่ในแนวแกน X ไปทางซ้าย
2) ค่าระยะห่างของพิกัดแกน X เทียบกับจุดอ้างอิงเป็นค่าติดบวก = ดวงตาเคลื่อนที่ในแนวแกน X ไปทางขวา

3) ค่าระยะห่างของพิกัดแกน Y เทียบกับจุดอ้างอิงเป็นค่าติดลบ = ดวงตาเคลื่อนที่ในแนวแกน Y ไปทางด้านบน
4) ค่าระยะห่างของพิกัดแกน Y เทียบกับจุดอ้างอิงเป็นค่าติดบวก = ดวงตาเคลื่อนที่ในแนวแกน Y ไปทางด้านล่าง

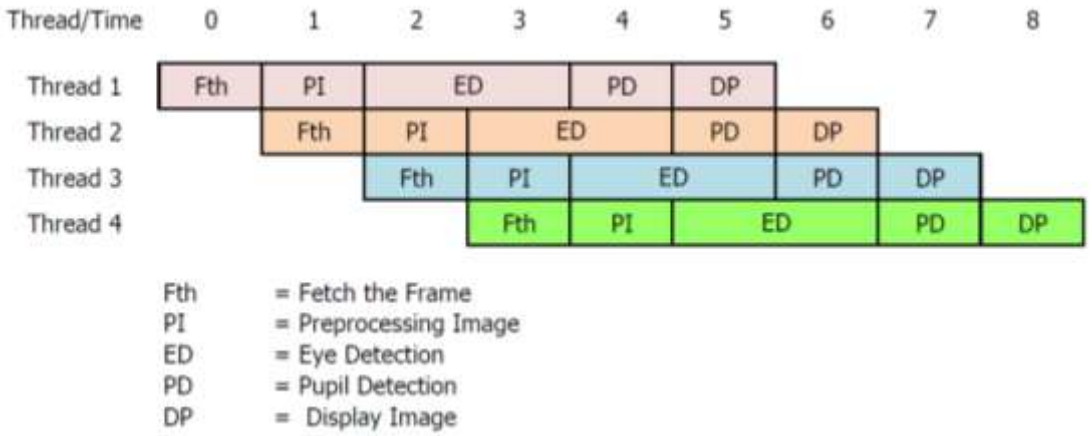
กระบวนการที่กล่าวด้านบนจะช่วยให้หาทิศทางของการเคลื่อนที่ของรูม่านตาได้ แต่ในการ Map จาก Pixel ของภาพเป็นองศาจะต้องทำตามขั้นตอนดังนี้

1) ทดสอบด้วยการกำหนดจุดอ้างอิงด้านขวาบน เช่น การมองตำแหน่งขอบจอขวาบนของ Notebook เป็นต้น และในการทดสอบจะต้องรู้ระยะห่างของผู้ทดสอบกับวัตถุที่มอง
2) ทำการมองจุดขวาบนที่กำหนดไว้แล้วให้โปรแกรมทำการตรวจหาพิกัดของรูม่านตาด้วย 35 ภาพติดต่อกันเพื่อหาค่าเฉลี่ยจากนั้นนำค่าพิกัด X, Y ที่ได้มาหาระยะห่าง Pixel กับจุดพิกัดอ้างอิงของการมองตรงไปข้างหน้า
3) นำระยะห่าง Pixel ที่ได้ของแกน X, Y มาทำการคำนวณมุมจากสามเหลี่ยม Pythagoras ด้วยการใช้ Arccos เพื่อหามุมของสามเหลี่ยมในแนวแกน X และมุมของสามเหลี่ยมในแนวแกน Y
4) เมื่อได้ค่ามุมของแนวแกน X, Y แล้วให้นำมุมที่ได้มาหารด้วยระยะห่างของพิกัดกึ่งกลางและพิกัดขวาบนทำให้ได้ค่า Pixel/องศา

ค่า Pixel/องศา จะใช้ในการทำนายมุมของทิศทางการมองของดวงตาได้แต่เป็นการเทียบความสัมพันธ์แบบ linear ซึ่งอาจจะไม่ตรงหรือมีความผิดพลาดเกิดขึ้นเล็กน้อยแต่สามารถใช้ในการหามุมในการมองได้

3.2.8 การประมวลผลภาพด้วย Multithreading

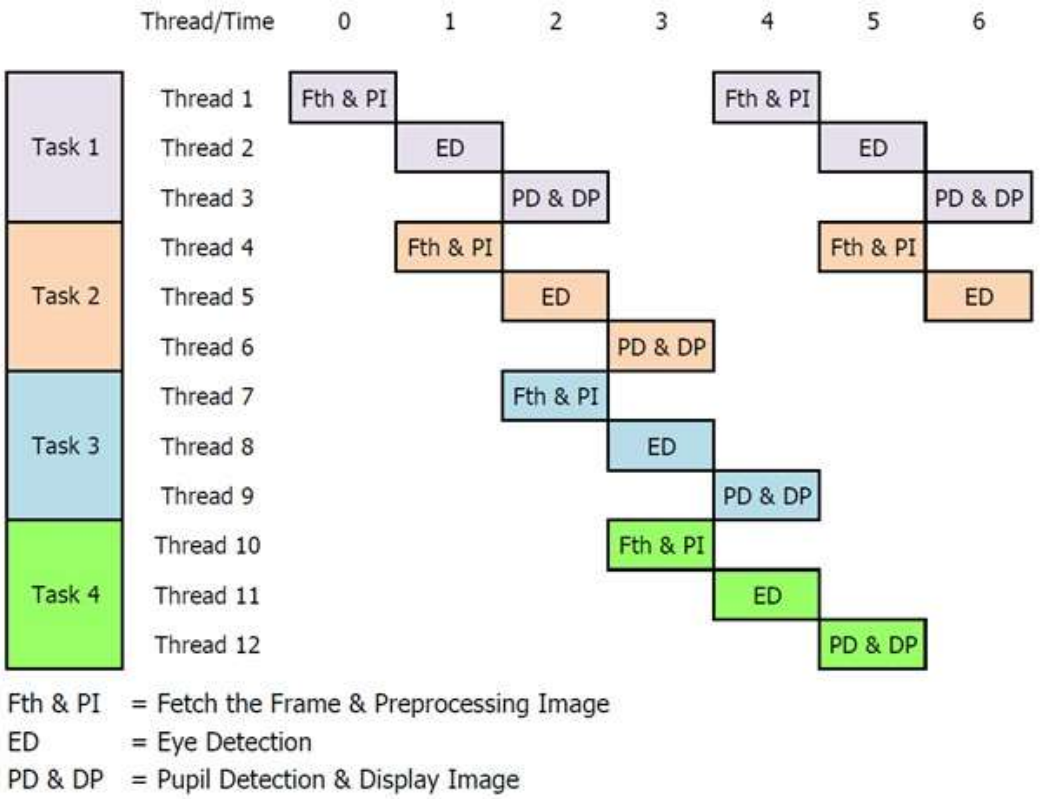
การประมวลผลภาพขนาด 1920 x 1080 ซึ่งเป็นขนาดที่มีความละเอียดสูง และการประมวลผลภาพประกอบด้วยกระบวนการหลายขั้นตอนทำให้การประมวลผลภาพค่อนข้างช้า โดยสามารถประมวลผลได้ประมาณ 5.77 ภาพต่อวินาที ดังนั้นทางกลุ่มผู้พัฒนาจึงนำวิธีการประมวลผลแบบ Parallel มาใช้ในการประมวลผลภาพ ซึ่งแนวความคิดในการประมวลผลนี้คือการใช้ Multithreading ซึ่งแบ่งการประมวลผลภาพเป็นหลาย Thread ทำงานขนานกันคล้ายกับ Pipeline



ภาพที่ 3.16 Pipeline ของการทำงาน Multithreading 4 threads

การแบ่งการประมวลผลภาพแบบ 4 Threads

ภาพที่ 3.16 แสดงการทำงาน Multithreading 4 threads ในการทำงานของ 4 Threads จะมีขั้นตอนการทำงานเหมือนกันทั้ง 4 Threads คือการดึงภาพจากกล้อง, การทำ Preprocessing image, การตรวจหาตำแหน่งของดวงตา, การตรวจหาตำแหน่งของรูม่านตา, และการแสดงผลภาพหลังการประมวลผล ดังนั้นการทำงานของ Thread จะทำงานแบบ Pipeline ซึ่งในบางขั้นตอน Thread จะต้องรอการทำงานกัน ได้แก่ ดึงภาพจากกล้องเพียง 1 ตัวจะต้องรอการเข้าถึงตัวแปรของกล้องและการเข้าถึง Picturebox ใน Graphic User Interface จะต้องรอการเข้าถึง Picturebox เพียง 1 Thread ในเวลาใดๆเท่านั้น



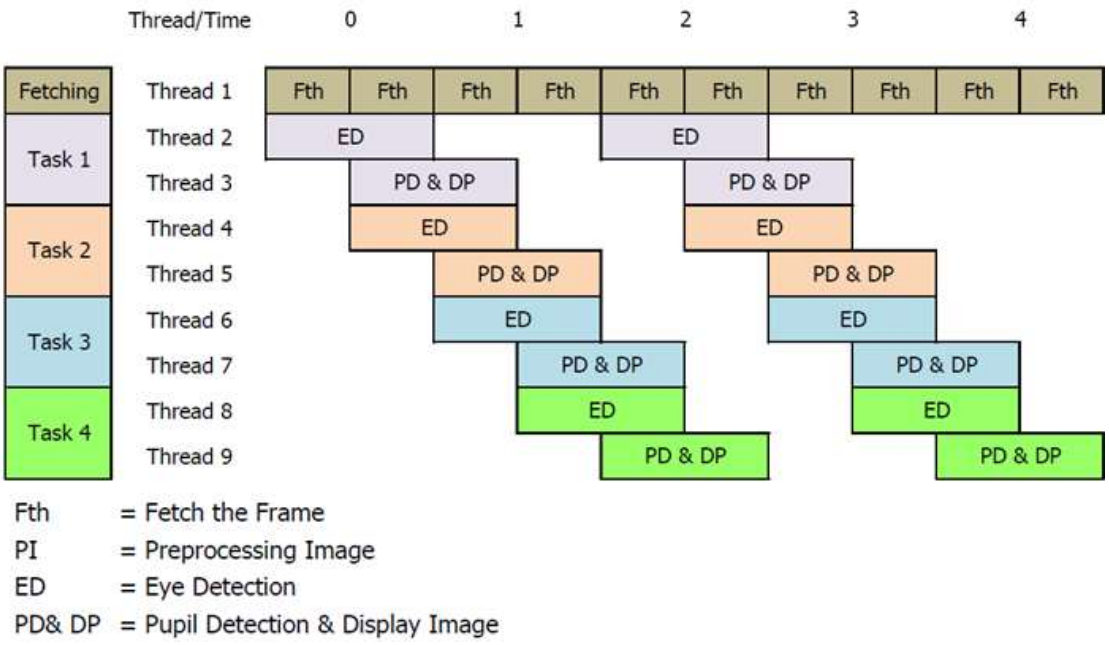
ภาพที่ 3.17 Pipeline ของการทำงาน Multithreading 12 threads

การแบ่งการประมวลผลภาพแบบ 12 Threads

ภาพที่ 3.17 แสดงการทำงาน 12 Threads การแบ่งการทำงานการประมวลผลภาพเป็น 12 Threads แต่ยังคงประมวลผลแบบ 4 Frame พร้อมๆกันโดยการแบ่งให้เกิดงานในการประมวลผล 3 งาน ซึ่งแต่ละงานประกอบด้วย 3 Threads การแบ่งงานในการประมวลผลภาพ 1 Frame ออกเป็นงานย่อย 3 งานคือ

1) การดึงภาพจากกล้องและ Preprocessing image
2) การตรวจหาตำแหน่งของดวงตา
3) การตรวจหาตำแหน่งของรูม่านตาและการแสดงผลลัพธ์บน Picturebox

ดังนั้นการประมวลผล 4 Frame พร้อมๆกันจะให้ 12 Thread ทำงานร่วมกัน ซึ่งข้อจำกัดของการประมวลผลแบบ 12 Thread คือ การรอการเข้าถึงตัวแปรของกล้องเพียง 1 ตัว, การ Thread ก่อนหน้าที่ขึ้นต่อกันทำงานเสร็จ, และการรอการเข้าถึง Picturebox ใน Graphic User Interface



ภาพที่ 3.18 Pipeline ของการทำงาน Multithreading 9 threads

การแบ่งการประมวลผลภาพแบบ 9 Threads

การแบ่งการทำงานการประมวลผลภาพเป็น 9 Threads ดังภาพที่ 3.18 แต่ยังคงประมวลผลแบบ 4 Frame พร้อมๆกัน โดยการแบ่งให้มีเพียง 1 Thread เท่านั้นที่ทำการดึงภาพจากกล้องจับดวงตา และอีก 8 Threads ให้ทำงานโดยเหมือนมีการแบ่งงานเป็น 4 งานซึ่งแต่ละงานมี 2 Thread ช่วยกันประมวลผล โดยมีการแบ่งงานเป็น 2 Thread ดังนี้

1) การตรวจหาตำแหน่งของดวงตา
2) การตรวจหาตำแหน่งของรูม่านตาและการแสดงผลลัพธ์บน Picturebox

ดังนั้นการประมวลผล 4 Frame พร้อมๆกันจะให้ 9 Thread ทำงานร่วมกัน ซึ่งข้อจำกัดของการประมวลผลแบบ 9 Thread คือ การรอการเข้าใช้งานภาพที่ดึงได้จากกล้องจับดวงตาซึ่ง Thread ที่เข้าถึงต้องมีเพียง 1 Thread ดังนั้น Thread ที่เหลือจะต้องรอคอยการเข้าถึง, การรอ Thread ก่อนหน้าที่ขึ้นต่อกันทำงานเสร็จ, และการรอการเข้าถึง Picturebox ใน Graphic User Interface

ประสิทธิภาพของการประมวลผล Multithreading แบบ 4 Thread, 12 Threads, และ 9 Threads

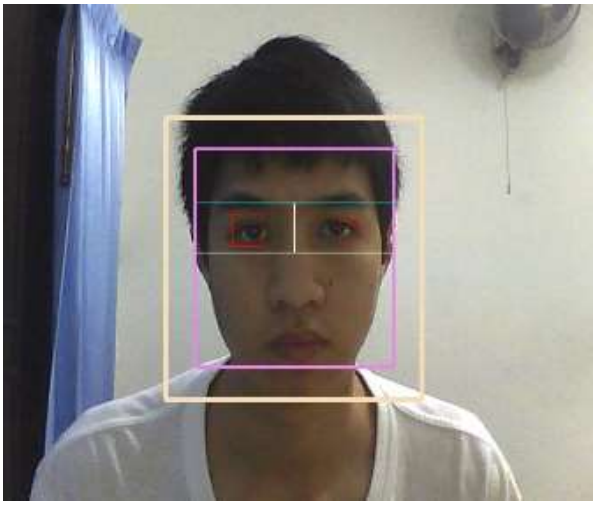
ในการประมวลผลแบบ 4 Threads พบว่าความเร็วในการประมวลผลภาพเพิ่มขึ้นสูงสุด 2.96 เท่า โดยสามารถประมวลผลได้ที่ความเร็ว 13 - 17 ภาพต่อวินาที

ในการประมวลผลแบบ 12 Threads พบว่าความเร็วในการประมวลผลภาพยังคงมีความเร็วเท่ากับการประมวลผลแบบ 4 Threads ซึ่งอาจเกิดจากการที่ทรัพยากรของเครื่องคอมพิวเตอร์ที่ใช้ประมวลผลภาพมีความแตกต่างเมื่อเทียบกับการ Fork Thread

ในการประมวลผลแบบ 9 Threads พบว่าความเร็วในการประมวลผลภาพยังคงมีความเร็วใกล้เคียงกับการประมวลผลแบบ 4 Threads

3.2.9 การ Optimization ในการประมวลผลภาพด้วย Region of Interest

การประมวลผลภาพขนาด 1920 x 1080 ซึ่งมีขนาดใหญ่ การประมวลผลภาพทั้งภาพทำให้การประมวลผลช้า การทำ Optimization จะเป็นการทำให้ขนาดของภาพที่ใช้ในการประมวลผลเฉพาะพื้นที่ที่สนใจเท่านั้น เมื่อขนาดของภาพเล็กลงทำให้สามารถประมวลผลภาพได้เร็วขึ้น ดังภาพที่ 3.19



ภาพที่ 3.19 การทำ Optimization ด้วย Region of interest

การทำ Optimization ทำตามขั้นตอนดังนี้

1) นำภาพมาประมวลผลเพื่อหาตำแหน่งของดวงตา
2) เก็บค่าตำแหน่งและขนาดของสี่เหลี่ยมที่ตรวจดวงตาพบ
3) ในการตรวจสอบดวงตาครั้งถัดไปให้ทำการนำค่าพิกัดและขนาดสี่เหลี่ยมของดวงตาที่ตรวจพบครั้งแรกมาขยายแกน X, Y เพิ่มอีก 10% ซึ่งแสดงในกรอบสีครีม
4) ทำ ROI กับภาพ raw image ในขนาดสี่เหลี่ยมที่ขยายขึ้นจากนั้นนำภาพ ROI ที่ได้มาประมวลผล preprocessing และหาตำแหน่งดวงตาต่อไป
5) หากตรวจหาตำแหน่งของดวงตาในภาพ ROI ไม่พบให้ทำในขั้นตอนที่ 1 ใหม่เพื่อหาขนาดของสี่เหลี่ยมและพิกัดอ้างอิงใหม่
6) หากตรวจหาตำแหน่งของดวงตาในภาพ ROI พบให้ทำการอ้างอิงในการขยายสี่เหลี่ยมจากตำแหน่งที่พบล่าสุด

ในการขยายของขนาดสี่เหลี่ยมขึ้นเพื่อทำนายว่าดวงตาจะยังคงอยู่ภายในกรอบที่ขยายขึ้น ซึ่งหากทำนายถูกจะทำให้ลดเวลาการประมวลผลภาพได้

ข้อจำกัดในการทำการ Optimization คือการใช้ Multithread ในการประมวลผล ซึ่ง Multithread จะประมวลผลโดยไม่เรียงลำดับและระบบปฏิบัติการจะทำการ Scheduling ให้ Thread แต่ละตัวประมวลผลเอง ดังนั้นในการจัดการเพื่อหาพิกัดและขนาดของสี่เหลี่ยมที่ตรวจพบล่าสุดทำได้ยากและการเข้าถึงตัวแปรเพื่อเขียนค่าร่วมกันจะต้องรอการเข้าถึงทีละ Thread

ทางกลุ่มผู้พัฒนาได้ทดลองใช้กระบวนการ Optimization ในการประมวลผลภาพแบบ 1 Thread พบว่าสามารถเพิ่มประสิทธิภาพในการประมวลผลให้ดีขึ้นได้ แต่ในการประมวลผลแบบ Multithread ทางกลุ่มผู้พัฒนาจึงตัดวิธีการ Optimization ออกไป

3.2.10 การติดต่อผ่านทางระบบ Network ด้วย Socket Programming

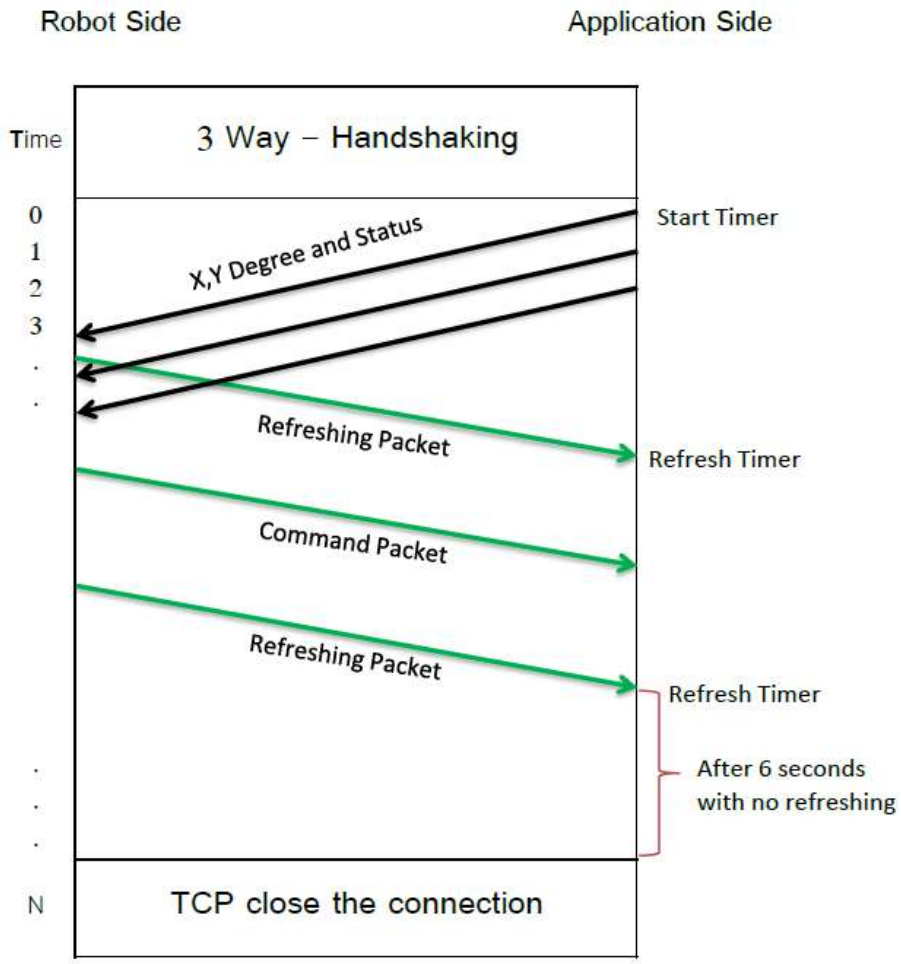
การติดต่อทาง Network ระหว่าง Application และตัวหุ่นยนต์ จะติดต่อด้วย Protocol TCP บน Transport Layer และติดต่อด้วย Protocol ที่ทางกลุ่มผู้พัฒนาคิดขึ้นมาบน Application Layer

3.2.10.1 TCP socket Programming

การติดต่อผ่าน Network พัฒนาด้วย Socket Programming ด้วยภาษา C# การพัฒนาจะประกอบด้วยการเปิด Listening IP และ Port เพื่อรอรับการติดต่อจากตัวหุ่นยนต์ ซึ่งได้ออกแบบให้สามารถรองรับการเชื่อมต่อได้มากที่สุด 16 การเชื่อมต่อหรือ 16 Socket นั่นเอง

3.2.10.2 Protocol เฉพาะบน Application Layer

ทางกลุ่มผู้พัฒนาได้คิดค้น Protocol ในการรับส่งข้อมูลและคำสั่งระหว่าง Application และตัวหุ่นยนต์ให้สามารถทำงานแบบอัตโนมัติได้ การออกแบบ Protocol ในการรับส่งข้อมูลจะเป็นดังภาพที่ 3.20



ภาพที่ 3.20 Protocol การรับส่งข้อมูลบนชั้น Application layer

การทำงานเริ่มหลังจากการเชื่อมต่อด้วย 3 Way Handshaking ดังนี้

Application จะทำการตั้งค่าเวลาเพื่อใช้ในการยกเลิกเชื่อมต่อเมื่อไม่ได้รับการติดต่อตามเวลาที่กำหนด โดยจะตั้งเวลา 6000 มิลลิวินาที และ Application จะทำการส่งข้อมูลองศาของแกน X, Y ไปยังตัวหุ่นยนต์ ในขณะเดียวกัน Application จะรอรับคำสั่งจากตัวหุ่นยนต์ด้วย Header Format ที่ออกแบบขึ้น เช่น การขอยกเลิกการเชื่อมต่อ, การ Refresh การตั้งเวลาการยกเลิกการเชื่อมต่อ, และการกำหนดความเร็วในการส่งข้อมูลองศาของ Application 2 ระดับคือ ระดับเร็วและช้า เป็นต้น

ตัวหุ่นยนต์เมื่อทำการเชื่อมต่อกับ Application แล้วจะทำการรอรับข้อมูลองศาและเมื่อได้รับข้อมูลองศาแล้วจะทำการส่งข้อมูลเพื่อไป Refresh Session ที่ Application

การออกแบบ Header Format เพื่อใช้ในการติดต่อมีดังนี้

1) การติดต่อจาก Application ไปยังตัวหุ่นยนต์จะใช้รูปแบบคือ

$x$$y$$z$

x คือข้อมูลองศาในแกน X ซึ่งที่มีขนาดความยาว 1 ตัวอักษร

y คือข้อมูลองศาในแกน Y ซึ่งที่มีขนาดความยาว 1 ตัวอักษร

z คือข้อมูลคำสั่งจาก Application ไปยังตัวหุ่นยนต์

z ประกอบด้วย

0 คือการสั่งให้หุ่นยนต์หมุนตามองศาที่กำหนด

1 คือการสั่งให้หุ่นยนต์อยู่ใน Idle process หรือไม่ขยับตามองศาที่ส่งมา ซึ่ง Application ส่งมาเพื่อให้หุ่นยนต์รับรู้ว่ายังเชื่อมต่ออยู่

2) การติดต่อจากตัวหุ่นยนต์ไปยัง Application จะมีรูปแบบการส่งดังนี้

@01@ หมายถึงการขอปิดการเชื่อมต่อ

@02@ หมายถึงการขอ Refresh Session Time เพื่อบอก Application รู้ว่ายังติดต่ออยู่

@03@ หมายถึงการขอเปลี่ยนโหมดการส่งข้อมูลให้ Application ส่งข้อมูลช้าลง

@04@ หมายถึงการขอเปลี่ยนโหมดการส่งข้อมูลให้ Application ส่งข้อมูลเร็วขึ้น

3.2.11 การรับภาพจากกล้องบนตัวหุ่นยนต์

การรับภาพจากกล้องบนตัวหุ่นยนต์จะรับภาพด้วยการส่งข้อมูลแบบไร้สายจาก Wireless Camera ที่ติดอยู่บนตัวหุ่นยนต์ ซึ่ง Wireless Camera จะส่งด้วยคลื่นความถี่ 2.4 GHz ด้วยรูปแบบ AV มาที่อุปกรณ์รับข้อมูลและอุปกรณ์รับข้อมูลจะแปลงเป็นการเชื่อมต่อแบบ USB เข้าที่คอมพิวเตอร์ ดังนั้นในการเขียนโปรแกรมเพื่อดึงภาพจากกล้อง Wireless ยังคงสามารถใช้คำสั่งเดียวกับการดึงภาพจากกล้อง Web Camera แบบมีสายได้

3.3 ตัวหุ่นยนต์ติดกล้องถ่ายสภาพแวดล้อม

3.3.1 Micro Controller

การพัฒนาตัวหุ่นยนต์จะใช้ Microcontroller ตระกูลATmega โดยทางกลุ่มผู้พัฒนาเลือกใช้ Micro Controller ATmega328 ซึ่งทำงานร่วมกับ Arduino Microcontroller Board ซึ่งทางกลุ่มผู้พัฒนามีเหตุผลในการเลือกใช้ดังนี้

1) เนื่องจาก Arduino ใช้ภาษา C ในการพัฒนาและมีรูปแบบของคำสั่งที่ง่ายต่อการเข้าใจและพัฒนา
2) Compiler ที่ใช้ในการแปลภาษาสามารถหาได้ง่ายและมีการปรับปรุงเสมอ
3) Arduino มี Library ในการควบคุม Servo Motor และการติดต่อ Serial Communication ทำให้ง่ายต่อการพัฒนาในภาษาระดับสูง
4) Website ของ Arduino มีข้อมูลเกี่ยวข้องกับการเขียนโปรแกรม, คำสั่ง, และ Library ที่ครบถ้วนและมีคนใช้งาน Arduino Board กันมาก

ทางกลุ่มผู้พัฒนาเลือกใช้ Arduino รุ่น Fino (Arduino Compatible) Board - Duemilanove ATmega328 ดังภาพที่ 3.21

ภาพที่ 3.21 Arduino รุ่น Fino (Arduino Compatible) Board - Duemilanove ATmega328

โดย Fino (Arduino Compatible) Board มีคุณสมบัติดังนี้

1) ATmega328 28pin DIP Microcontroller พร้อมด้วย Duemilanove bootloader
2) ATmega328 ประกอบด้วย 32KB Memory, 1KB EEPROM, 2KB SRAM
3) Quartz crystal 16 MHz
4) Standard ICSP 6pins ซึ่งพัฒนาด้วย AVR ISP
5) เชื่อมต่อกับคอมพิวเตอร์และอัพโหลดโปรแกรมผ่าน USB port
6) ATmega328 Microcontroller ทำงานในระดับความต่างศักย์ 5V
7) Fino (Arduino Compatible) Board ทำงานในระดับความต่าง 7-12V
8) สามารถจ่ายพลังงานให้ Fino (Arduino Compatible) Board ได้โดยการเชื่อมต่อ USB หรือ Power supply ภายนอกซึ่งแหล่งพลังงานจะถูกเลือกโดยอัตโนมัติ
9) Fino (Arduino Compatible) Board มี resettable polyfuse ซึ่งสามารถปกป้อง USB port ของคอมพิวเตอร์จากการช็อตหรือการจ่ายกระแสเกิน
10) Digital Input / Output are 14 pins (PWM 6 channels)
11) Analog Input is 6pins
12) DIP Switch SPST ใช้ในการปิด auto reset และ yellow LED (D13)
13) FT232RL ขับเคลื่อนการเชื่อมต่อสัญญาณกับคอมพิวเตอร์
14) ประกอบด้วย Standard Arduino Pin Header - 8 pins & 6 pins (Female) ซึ่งสามารถทับซ้อนกับอุปกรณ์ภายนอกหรือ Arduino board อื่นๆ

15) ประกอบด้วย 3.1VDC output จาก FT232RLสำหรับการพัฒนาในรูปแบบอื่นๆ

16) ขนาดของ Board 53.1mm (ความกว้าง) * 68.6mm (ความยาว) * 15.0mm (ความสูง)

3.3.2 กล้องบนตัวหุ่นยนต์

กล้องบนตัวหุ่นยนต์จะเลือกใช้กล้อง Wireless Camera ที่ส่งข้อมูลด้วยคลื่นความถี่ 2.4 GHz ในรูปแบบ AV มาที่ Adaptor ตัวรับข้อมูลแบบ 2.4GHz เพื่อแปลงเป็นการเชื่อมต่อแบบ USB เข้ากับคอมพิวเตอร์ได้

การส่งข้อมูลของกล้อง Wireless จะสามารถส่งได้ 4 Channel ของคลื่นความถี่ 2.4 GHz ซึ่งตัวรับจะต้องรับที่ Channel ของการส่งที่ถูกต้อง

Wireless Camera มีแบตเตอรี่ในตัวสามารถส่งข้อมูลได้ติดต่อกันเป็นเวลา 2 ชั่วโมง ซึ่งทางกลุ่มผู้พัฒนาคิดว่าเหมาะสมในการนำมาใช้งานเนื่องจากสามารถส่งข้อมูลได้เป็นเวลานานพอ ดังภาพที่ 3.22

USB Adaptor คืออุปกรณ์ที่ใช้ในการรับข้อมูลแบบไร้สายที่ความถี่ 2.4 GHz จากนั้นจะแปลงรูปแบบข้อมูลให้สามารถติดต่อได้แบบ USB ดังภาพที่ 3.23

ภาพที่ 3.22 กล้อง Wireless camera

ภาพที่ 3.23 USB Adapter

3.3.3 Servo Motor

Servo motor ดังภาพที่ 3.24 ถูกนำมาใช้ในการเคลื่อนไหวของหุ่นยนต์เพื่อทำให้กล้องจับภาพสามารถขยับได้ในลักษณะครึ่งทรงกลม ซึ่งทางกลุ่มผู้พัฒนาได้เลือกใช้ MKS DS1210 Titanium Gear Standard Digital Servo ซึ่งเป็น Servo motor ที่มีความแข็งแรง ทนทาน เหมาะสำหรับนำมาใช้ทำการพัฒนา โดย MKS DS1210 Servo motor นั้นมีคุณสมบัติดังนี้

- Dead band: 0.002ms (Default)
- Control System: +Pulse Width Control
- Working frequence: 120Hz
- (RX) Required Pulse: 3.0~5.0 Volt Peak to Peak Square Wave
- Operating Voltage: 4.8~6.0 V DC Volts
- Operating Temperature Range: -10 to + 60 Degree C
- Operating Speed (4.8V): 0.15 sec/60° degrees at no load
- Operating Speed (6V): 0.12 sec/60° dagrees at no load
- Stall Torque (4.8V): 8.05 kg-cm (111.79 oz/in)
- Stall Torque (6V): 10 kg-cm (138.87 oz/in)
- 360° Modifiable: NO
- Motor Type: DC Motor
- Potentiometer Drive: Direct Drive
- Driver Type: FET
- Bearing Type: Dual Ball Bearings
- Gear Type: metal gear
- Connector Wire Length: 15.0 cm (5.9 in)
- Dimensions: 40X20X40.30 mm (1.5748X0.7874X1.58 in)
- Weight: 56 g (1.97 oz)

ภาพที่ 3.24 MKS DS1210 Titanium Gear Standard Digital Servo

3.3.4 WiFi Module

WiFi Module ใช้ในการติดต่อการส่งข้อมูลผ่านระบบเครือข่ายไร้สายแบบ Infrastructure Mode โดย WiFi Module จะต้องติดตั้งอยู่บนตัวหุ่นยนต์เพื่อให้หุ่นยนต์สามารถส่งรับข้อมูลกับ Application

WiFi Module ที่ทางกลุ่มผู้พัฒนาเลือกใช้คือ M03 - LVTTL UART WiFi Module ดังภาพที่ 3.25 ซึ่งจะติดต่อกับ Microcontroller ด้วย UART communication ประกอบด้วยขา Rx, Tx, Vcc,และ Gnd ซึ่งการส่งข้อมูลแบบ Serial จะมีความเร็วช้าแต่เนื่องจากการส่งข้อมูลขนาดเล็กคือคำสั่งจากตัวหุ่นยนต์ไปยัง Application ดังนั้นจึงเลือกใช้ WiFi Module แบบ UART โดยรายละเอียดของ UART WiFi Module มีดังนี้

ภาพที่ 3.25 M03 - LVTTL UART WiFi Module

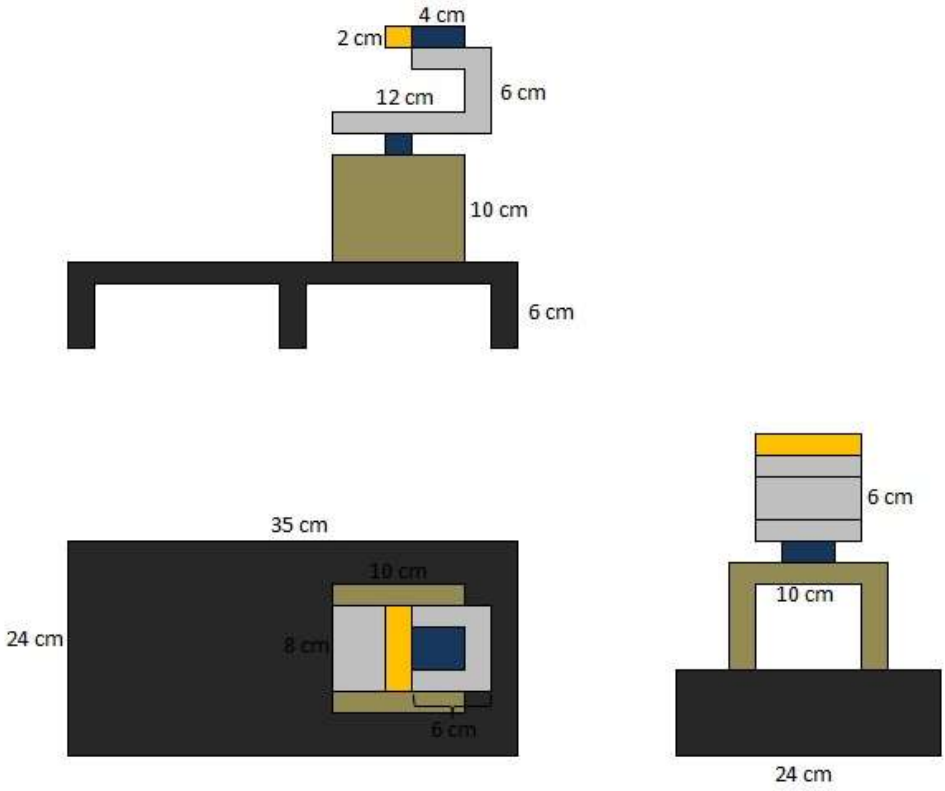
M03 - LVTTL UART to Wi-Fi คือ โมดูลที่ใช้สำหรับแปลงการรับส่งข้อมูลในรูปแบบ UART เป็นการรับส่งข้อมูลในรูปแบบของ Wireless LAN หรือ Wi-Fi (IEEE 802.11b/g) ซึ่งภายในโมดูลมี Software TPC/IP Stack อยู่ทำให้ใช้งานได้ง่าย สะดวก และรวดเร็ว เหมาะสำหรับนำมาใช้กับระบบประมวลผลหรือไมโครคอนโทรลเลอร์ที่มีความเร็วในการประมวลผลต่ำ ใช้

หน่วยความจำน้อย และลดเวลาในการพัฒนาลง โดยโมดูลมีการจัดสรร Software ที่เป็นส่วนของ TCP/IP Stack และการควบคุม Hardware ที่ใช้รับส่งข้อมูลผ่านทาง Wireless ไว้แล้ว ดังนั้นเพียงแค่ตั้งค่าตัวโมดูลให้สามารถเชื่อมต่อกับระบบ Network ผ่านทาง Software ที่ผู้ผลิตได้จัดเตรียมไว้ให้ก็สามารถใช้สื่อสารข้อมูลได้เลย นอกจากนี้ยังสามารถสั่งงานหรือเปลี่ยนแปลงค่าต่างๆ ผ่านทาง AT Command ด้วยการเขียนโปรแกรมจากไมโครคอนโทรลเลอร์ได้อีกด้วย

- interface
  - 2x4 Pin of interface
  - สามารถรับส่งข้อมูลผ่านทาง UART ที่ความเร็ว 1200 – 115200 bps
  - รองรับ Hardware Flow Control RTS/CTS
  - ใช้กระแสไฟฟ้า 3.1V DC
- Wireless
  - รองรับการรับส่งข้อมูลผ่าน Wireless ตามมาตรฐาน IEEE 802.11 b/g
  - ใช้ช่วงความถี่ 2.412 – 2.484 GHz
  - รองรับการใช้งานแบบ Ad hoc และ Infrastructure
  - รองรับมาตรฐานความปลอดภัย WEP64/ WEP128/ TKIP/ CCMP(AES)/ WEP/ WPA-PSK/ WPA2-PSK
  - รองรับ Network Protocol แบบ TCP/ UDP/ ICMP/ DHCP/ DNS/ HTTP

3.3.5 การออกแบบตัวหุ่นยนต์

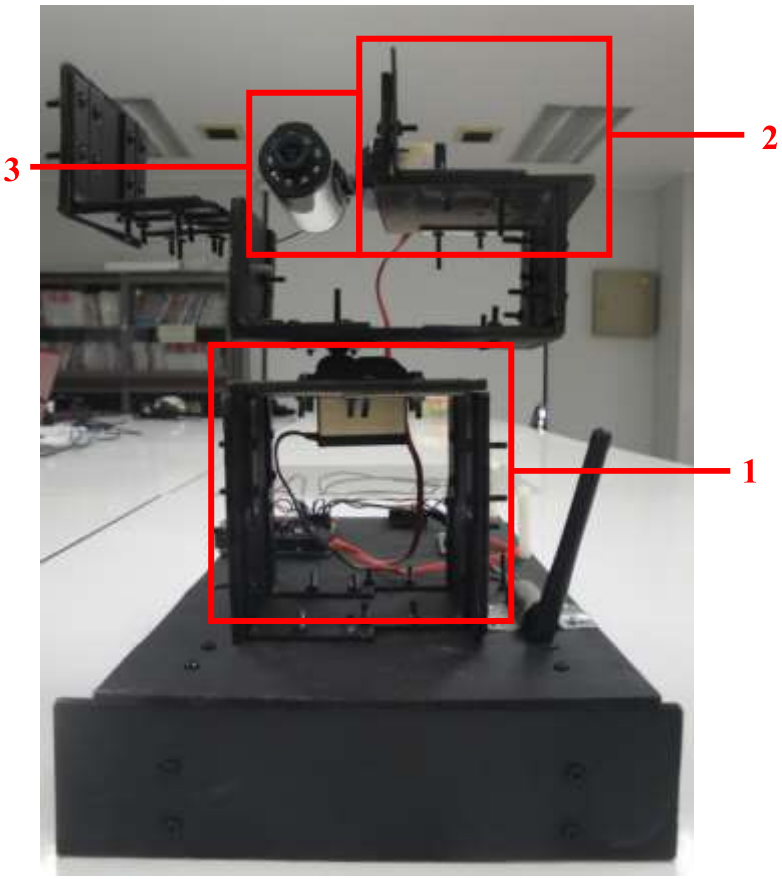
ในการออกแบบตัวหุ่นยนต์ ทางผู้พัฒนาได้ออกแบบให้ตัวหุ่นยนต์มีลักษณะเป็นแขนที่ยกสูงขึ้นจากพื้นโดยมีฐานเป็นแผ่นระนาบขนาดใหญ่ ซึ่งในส่วนของแขนหุ่นยนต์จะใช้พื้นที่ประมาณ ½ ของฐาน โดยพื้นที่ของฐานที่เหลือจะนำมาใช้ในการวาง Micro controller และ WiFi module โดยขนาดของฐานถูกออกแบบมาให้มีขนาดใหญ่เพียงพอในการรองรับน้ำหนักของแขนหุ่นยนต์ และยังสามารถรองรับการสั่นสะเทือนและแรงเหวี่ยงจากการหมุนของแขนหุ่นยนต์เพื่อให้ไม่มีการเคลื่อนที่ส่วนเกินได้ ภาพที่ 3.26 – 3.29 แสดงการออกแบบตัวหุ่นยนต์



ภาพที่ 3.26 โครงสร้างของหุ่นยนต์

ภาพที่ 3.27 โครงสร้างของหุ่นยนต์ที่ออกแบบมุมมอง Top View

ภาพที่ 3.28 โครงสร้างของหุ่นยนต์ที่ออกแบบมุมมอง Side View



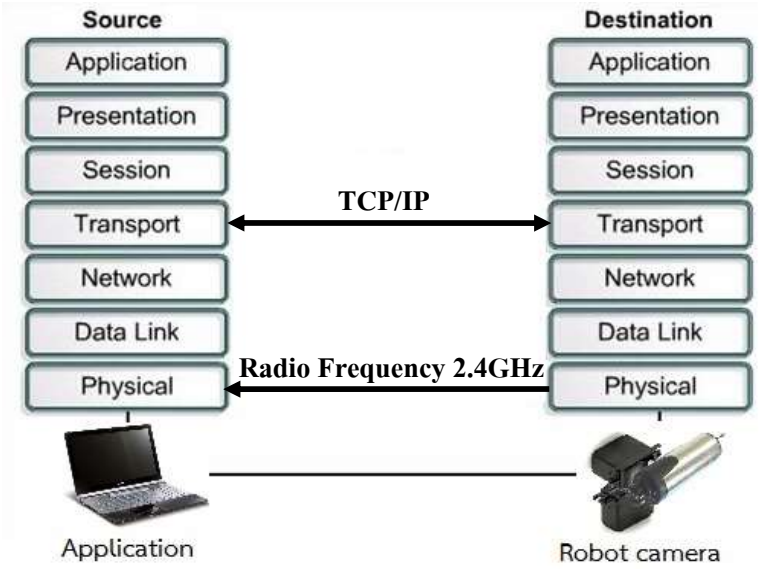
ภาพที่ 3.29 โครงสร้างของหุ่นยนต์ที่ออกแบบมุมมอง Front View

ในส่วนของการออกแบบแขนหุ่นยนต์ ทางผู้พัฒนาได้นำ Servo motor จำนวน 2 ตัว มาใช้ในการหมุนกล้อง โดยมี Servo Motor จำนวน 1 ตัวใช้ในการเคลื่อนที่แนวแกนนอนจะติดตั้งกับฐานด้านล่างซึ่งแสดงในส่วนที่ 1 ดังภาพที่ 3.29 แล Servo motor จำนวน 1 ตัวใช้ในการเคลื่อนที่ในแนวแกนตั้งจะติดตั้งกับฐานด้านบนในส่วนที่ 2 ดังภาพที่ 3.29 ซึ่งทางผู้พัฒนาได้ออกแบบให้แกนการหมุนของมอเตอร์มีตำแหน่งที่ใกล้ชิดกันที่สุด เพื่อให้การหมุนของมอเตอร์มีความคล้ายคลึงกับการเคลื่อนที่ของตามนุษย์มากที่สุด กล้อง Wireless ที่นำมาติดกับตัวหุ่นยนต์จะติดตั้งกับ Servo Motor ตัวบนเพื่อใช้หมุนในแนวแกนตั้งแสดงในส่วนที่ 3 ดังภาพที่ 3.29 และภาพที่ 3.30 แสดงการติดกล้องเข้ากับตัว Servo Motor การออกแบบตัวฐานจะออกแบบให้ยกสูงขึ้นจากพื้นเพื่อลดการสั่นสะเทือนโดยยกสูงจากพื้นประมาณ 6 เซนติเมตร

ภาพที่ 3.30 การออกแบบการหมุนของกล้องจับภาพ

3.4 การติดต่อระหว่าง Application และตัวหุ่นยนต์

การติดต่อสื่อสารเพื่อรับส่งข้อมูลองศาและคำสั่งผ่านระบบ Network จะพัฒนาด้วย WiFi Infrastructure Mode และในการส่งภาพจากตัวหุ่นยนต์จะส่งด้วย Wireless กลับมายัง Application ลักษณะของการสื่อสารข้อมูลจะเป็นดังภาพที่ 3.31



ภาพที่ 3.31 ลักษณะการสื่อสารของข้อมูลในแต่ละส่วน

ในการส่งข้อมูลระหว่าง Application กับตัวหุ่นยนต์จะเป็นการสื่อสารแบบ 2 ทิศทาง แต่การสื่อสารระหว่าง Application กับ อุปกรณ์จับภาพดวงตาจะสื่อสารแบบทิศทางเดียวจากอุปกรณ์จับภาพดวงตาไปยัง Application โดยการสื่อสารจะเป็นดังนี้

1) การติดต่อระหว่าง Application และตัวหุ่นยนต์ จะเป็นการสื่อสาร 2 Channel ซึ่ง Channel แรกจะเป็นการติดต่อด้วย WiFi Infrastructure Mode เพื่อส่งข้อมูลองศาและคำสั่งผ่าน Protocol ที่ได้คิดขึ้นซึ่งทำงานบน TCP และ Channel ที่สองจะใช้ในการส่งภาพจากตัวหุ่นยนต์กลับมายัง Application ด้วย Wireless Communication Radio Frequency 2.4 GHz แบบ AV
2) การติดต่อระหว่างกล้องจับภาพดวงตาและ Application จะติดต่อผ่านสาย USB 2.0

ในด้านของความเร็วในการสื่อสาร ความเร็วในการส่งข้อมูลของ M03 WiFi Module ซึ่งรองรับมาตรฐาน 802.11b ซึ่ง M03 WiFi Module มีความล่าช้าในการสื่อสารที่การส่งข้อมูลแบบ Serial Communication ซึ่ง WiFi Module รองรับที่ความเร็ว 1200 – 115200 bps ซึ่งการส่งข้อมูลด้วย Protocol ที่ทางกลุ่มผู้พัฒนาคิดขึ้นนั้นมีขนาดเล็กโดยมีขนาดไม่ถึง 100 Bytes ดังนั้นการส่งข้อมูลแบบ Serial Communication จึงเหมาะสมที่จะใช้ในการพัฒนา

3.5 การทำงานของระบบ

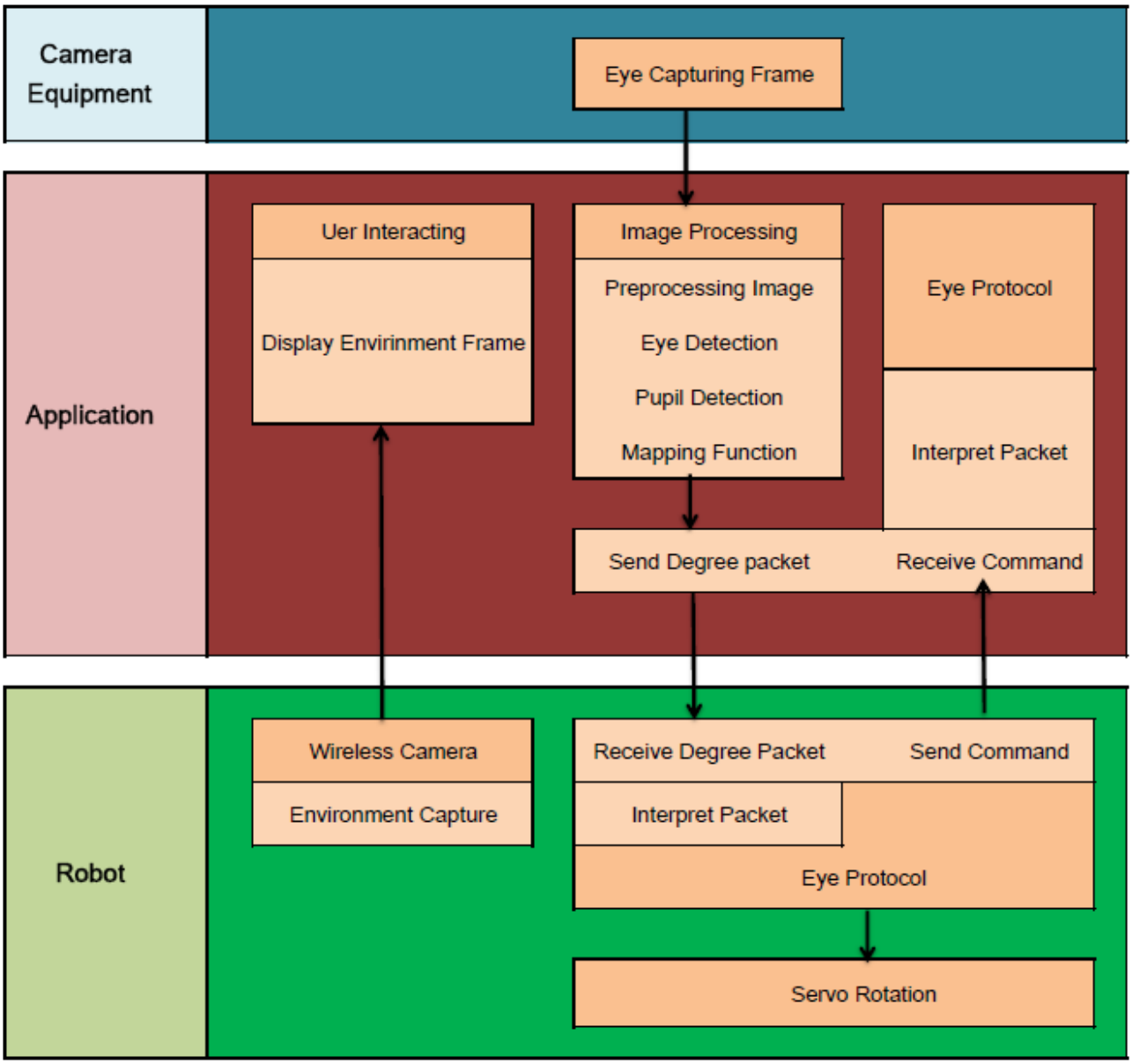
การทำงานของระบบจะแบ่งเป็น 3 ส่วนหลักคือ

1) Application บนคอมพิวเตอร์
2) ตัวหุ่นยนต์ติดตั้งกล้องถ่ายสภาพแวดล้อม

3) หมวกติดตั้งกล้องจับภาพดวงตา

โดยมีระบบการทำงานเป็นดังภาพที่ 3.32 มีการทำงานดังนี้

1) Application ทำงานหลักๆคือการประมวลผลภาพเพื่อหาตำแหน่งของดวงตาและรูม่านตา, ทำนายมุมการมองของดวงตา, รับส่งข้อมูลกับตัวหุ่นยนต์เพื่อควบคุมการทำงาน โดยมีขั้นตอนการทำงานดังนี้
   - ดึงภาพจากหมวกติดตั้งกล้องจับภาพดวงตา จากนั้นจะทำการประมวลผลภาพเพื่อหาตำแหน่งของดวงตาและรูม่านตา
   - ทำการคำนวณเพื่อหาทิศทางการมองของดวงตา
   - ติดต่อการใช้งานกับ User และแสดงผลลัพธ์ด้วย Graphic User Interface
   - ติดต่อการรับส่งข้อมูลกับตัวหุ่นยนต์ผ่านระบบ Network
2) ตัวหุ่นยนต์
   - ติดต่อรับส่งข้อมูลองศาจาก Application ผ่านระบบ Network
   - ควบคุมการหมุนของ Servo Motor ให้หมุนไปยังมุมที่ต้องการได้ในแกน X, Y
   - กล้องถ่ายสภาพแวดล้อมและส่งภาพกลับไปยัง Application ผ่าน Wireless Communication 2.4 GHz
3) หมวกติดตั้งกล้องจับภาพดวงตาจะทำการ ดึงภาพดวงตาของผู้ใช้งานและทำการส่งภาพไปยัง Application ผ่านสาย USB 2.0



ภาพที่ 3.32 การทำงานของระบบ