การหาระยะลึกจากภาพ 2 มิติ

Find Depth from 2 Dimension Image

นายบรรเจิด ประทุมหอม รหัส 44362648

นายภาณุพันธ์ ดับปากพิง รหัส 44362713

นายอนุสรณ์ นาคคงคำ รหัส 44362820

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2547

ใบรับรองโครงงานวิจัย

หัวข้อโครงงาน	การหาระยะลึกจากภาพ 2 มิติ		
ผู้ดำเนินโครงงาน	นายบรรเจิด	ประทุมหอม	รหัส 44362648
	นายภาณุพันธ์	ดับปากพิง	รหัส 44362713
	นายอนุสรณ์	นาคคงคำ	รหัส 44362820
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. สุชาติ แย้มเม่น		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2547		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิจัย

.....ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. สุชาติ แย้มเม่น)

.....กรรมการ
(อาจารย์ ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ
(อาจารย์พงศ์พันธ์ กิจสนาโยธิน)

หัวข้อโครงงาน	การหาระยะลึกจากภาพ 2 มิติ	
ผู้ดำเนินโครงงาน	นายบรรเจิด ประทุมหอม	รหัส 44362648
	นายภาณุพันธ์ ดับปากพิง	รหัส 44362713
	นายอนุสรณ์ นาคกงคำ	รหัส 44362820
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. สุชาติ	แย้มเม่น
สาขาวิชา	วิศวกรรมคอมพิวเตอร์	
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์	
ปีการศึกษา	2547	

บทคัดย่อ

โครงงานนี้เป็นการศึกษาและพัฒนาโปรแกรมคอมพิวเตอร์สำหรับหาค่าระยะลึกของวัตถุที่สนใจในภาพ 2 มิติที่ได้รับจากกล้องดิจิตอล ซึ่งระยะลึกนี้คำนวณโดยใช้วิธี Active Triangulation ของ Emanuele Trucco และ Alessandro Verri ค่าระยะลึกตั้งแต่ 40 เซนติเมตร ถึง 255 เซนติเมตร ที่ได้จากการคำนวณของโปรแกรมมีค่าความคลาดเคลื่อนเฉลี่ย 0.87 เปอร์เซ็นต์ และ ระยะลึกตั้งแต่ 260 เซนติเมตร ถึง 300 เซนติเมตร มีความคลาดเคลื่อนเฉลี่ย 4.77 เปอร์เซ็นต์

Project Tiltle	Find Depth from 2 Dimension Image		
Name	Mr. Bunjerd	Pratumhom	ID 44362648
	Mr. Panupun	Dabpakping	ID 44362713
	Mr. Anusorn	Nakkongkum	ID 44362820
Project Advisor	Assistant Professor Suchart	Yammen , Ph.d	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic Year	2004		

ABSTRACT

This project is to study and develop a computer program for determining a depth of the interest object in the two 2-D images obtained from the digital cameras. The depth is computed by using the active triangulation method proposed by Emanuele Trucco and Alessandro Verri. The result have been shown that the mean error of the program is 0.87 [%] for the depth case of 40 [cm] through 255 [cm] ,and is 4.77 [%] for the depth case of 260 [cm] through 300 [cm].

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้เกิดขึ้นได้เนื่องจากการทำงานร่วมกันในหลายๆส่วน บุคคลแรกที่ต้องกล่าวถึงคือ ผู้ช่วยศาสตราจารย์ ดร.สุชาติ แย้มเม่น อาจารย์ที่ปรึกษาที่ให้ความเอาใจใส่ แนะนำและช่วยเหลือเสมอ รวมถึงอาจารย์ท่านอื่นๆ ที่มิได้กล่าวถึง ที่ได้คอยแนะนำ และให้คำปรึกษาจนคลายความข้องใจ ซึ่งต้องขอบพระคุณเป็นอย่างมาก

และต้องขอขอบพระคุณบุคคลที่สำคัญที่สุดที่ทำให้พวกข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดูพวกข้าพเจ้ามาเป็นอย่างดี พร้อมทั้งให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจ เอาใจใส่อย่างเต็มที่ ในทุกๆด้านอันหาที่เปรียบมิได้ พวกข้าพเจ้าขอระลึกในพระคุณอันสุดประมาณและขอกราบขอบคุณมา ณ ที่นี้

บรรเจิด ประทุมหอม
ภาณุพันธ์ ดับปากพิง
อนุสรณ์ นาคคงคำ

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ในปัจจุบันมีงานบางอย่างที่มนุษย์ไม่สามารถที่จะปฏิบัติได้ เนื่องจากอาจได้รับความเสี่ยงต่อสุขภาพ จึงมีการคิดค้น พัฒนา และสร้างเครื่องจักรกลที่มาทำงานที่มีความเสี่ยงเช่นนี้ หุ่นยนต์เป็นจักรกลที่มีความฉลาด สามารถทำงานได้ด้วยตัวเอง เป็นสิ่งประดิษฐ์ที่มนุษย์สร้างขึ้นมาเพื่อทำงานแทนมนุษย์ในงานที่เสี่ยงอันตราย ในการพัฒนาให้หุ่นยนต์มีประสิทธิภาพในการทำงานจำเป็นต้องพัฒนาระบบการมองเห็นของหุ่นยนต์ เพราะระบบการมองเห็นจะให้ข้อมูลซึ่งช่วยในการทำงานด้านต่างๆ เช่น การแบ่งแยกวัตถุ การรู้จำรู้จักวัตถุ รวมทั้งการคาดคะเนระยะทาง

โครงงานการหาระยะลึกจากภาพ 2 มิตินี้จัดทำขึ้นเพื่อศึกษาและพัฒนาวิธีการ หาระยะทางจากกล้องถึงวัตถุเป้าหมายในภาพ อันเป็นพื้นฐานของการคาดคะเนระยะทางในระบบการมองเห็นของหุ่นยนต์ โดยระบบการมองเห็นของหุ่นยนต์นี้จะประกอบด้วยกล้องดิจิตอล 2 ตัว ทำหน้าที่เป็นตัวรับภาพเข้ามาแล้วมีโปรแกรมคอมพิวเตอร์ ที่ทำหน้าที่ตรวจหาวัตถุที่สนใจในภาพแล้วทำการคำนวณหาระยะทางจากตัวกล้องถึงวัตถุที่สนใจ

ผลที่คาดว่าจะได้รับจากโครงงานนี้คือสามารถนำเอาเทคนิควิธีการหาระยะลึกไปใช้ในระบบการมองเห็นของหุ่นยนต์ เพื่อช่วยในการนำทางหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง

1.2 วัตถุประสงค์ของโครงงาน

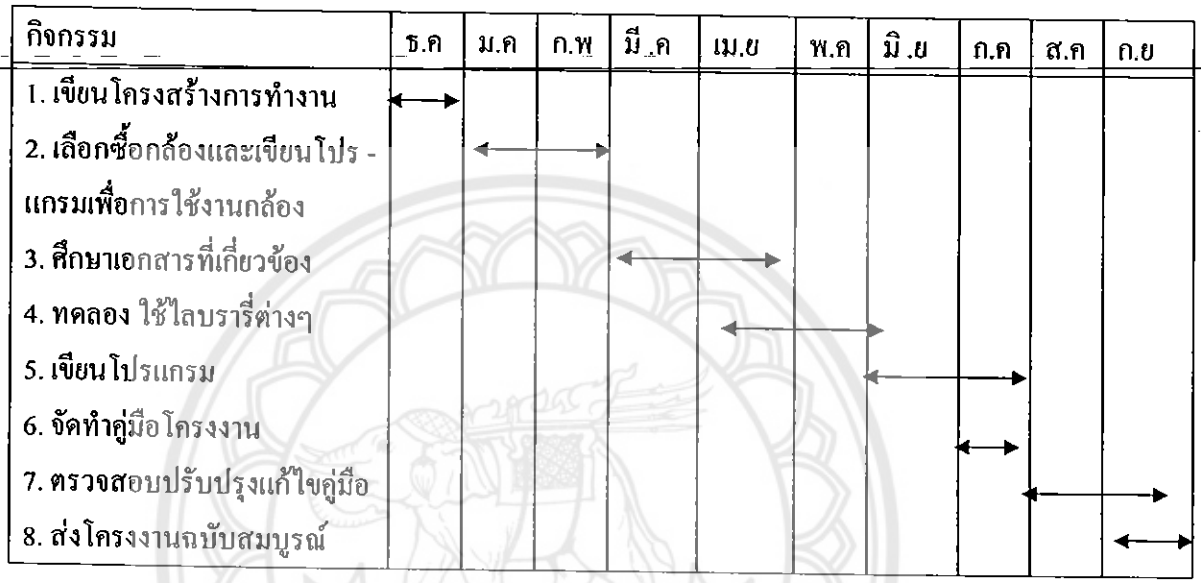
1. เพื่อศึกษากระบวนการหาระยะลึกของวัตถุจากภาพที่ได้จากกล้องดิจิตอล
2. เพื่อศึกษาการเขียนโปรแกรมเพื่ออ่านภาพจากกล้องดิจิตอล
3. เพื่อศึกษาการเขียนโปรแกรมด้วย Visual C++
4. เพื่อพื้นฐานในการศึกษาทางด้าน Stereo Vision
5. เพื่อเป็นพื้นฐานในการพัฒนาทางด้านระบบการนำทางหุ่นยนต์

1.3 ขอบข่ายของโครงงาน

1. ระยะลึกที่หาได้มีความคลาดเคลื่อนไม่เกิน 5 เปอร์เซ็นต์
2. โครงงานนี้จะหาระยะลึกจากกล้องถึงวัตถุที่กำหนดเท่านั้น

1.4 ขั้นตอนการดำเนินงาน

โครงงานนี้เริ่มด้วยการเลือกซื้อกล้องและอุปกรณ์ประกอบที่มีความเหมาะสมกับโครงงาน หลังจากนั้นจึงศึกษากระบวนการรับภาพและประมวลภาพที่ได้จากกล้องดิจิตอล แล้วจึงศึกษาเอกสารที่เกี่ยวข้องกับการหาระยะลึกจากภาพ สุดท้ายจะเป็นการเขียนโปรแกรมเพื่อการใช้งานด้วย Visual C++



1.5 ผลที่คาดว่าจะได้รับ

1. โปรแกรมคอมพิวเตอร์ที่สามารถทำการหาระยะลึกได้
2. ความรู้และความเข้าใจในการเขียนโปรแกรมเพื่ออ่านภาพจากกล้องดิจิตอล
3. ความรู้และทักษะในการเขียนโปรแกรมด้วย VisualC++

1.6 งบประมาณที่ต้องใช้

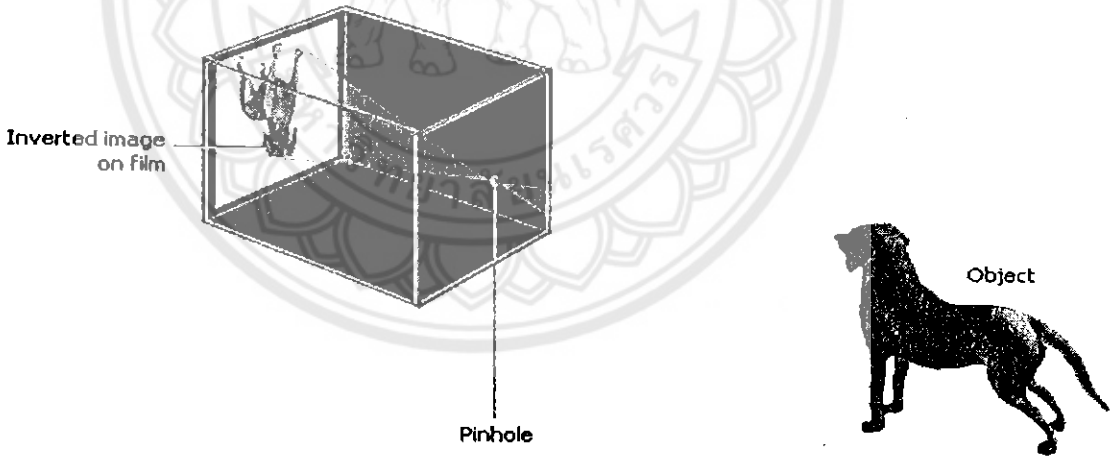
1. ค่าเอกสาร	300	บาท
2. ค่าวัสดุอุปกรณ์	500	บาท
3. ค่าจัดทำรูปเล่มโครงงาน	200	บาท
4. ค่ากล้องดิจิตอล 2 ตัว	2000	บาท
รวมค่าใช้จ่าย	3000	บาท (สามพันบาทถ้วน)

บทที่ 2
ทฤษฎีพื้นฐาน

ความรู้พื้นที่สำคัญในการหาระยะลึกจากภาพจากกล้องดิจิตอล คือการเข้าใจแบบจำลองทางเรขาคณิตของกล้อง ในบทนี้จะกล่าวถึง โครงสร้างและค่าพารามิเตอร์ของกล้องที่จำเป็นในกระบวนการหาระยะลึก ถัดจากนั้นจะอธิบายถึงวิธีการในการหาระยะลึก การใช้งาน DirectShow API ซึ่งเป็นเครื่องมือที่ใช้ในการอ่านภาพจากกล้อง ดิจิตอล และสุดท้ายจะอธิบายถึง OpenCV Library ซึ่งเป็นเครื่องมือที่สำคัญในการหาค่าคุณสมบัติของกล้อง และการประมวลผลภาพ

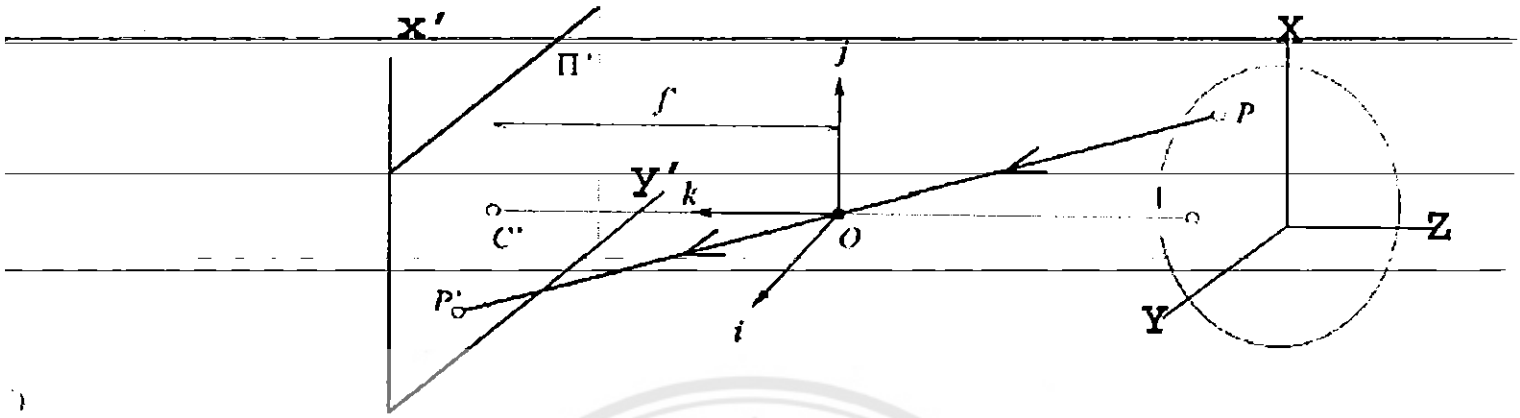
2.1 กล้องรูเข็ม

ในการอธิบายถึงการทำงานของกล้องนั้น นิยมใช้แบบจำลองที่เรียกว่า แบบจำลองกล้องรูเข็ม (Pinhole camera model) ซึ่งได้มาจากการทำงานของกล้องรูเข็ม ลักษณะและการทำงานของกล้องรูเข็มแสดงดังรูปที่ 2.1



รูปที่ 2.1 ลักษณะและการทำงานของกล้องรูเข็ม

ลักษณะของกล้องรูเข็มจะเป็นการนำเอากล่องมาเจาะรูที่ด้านหนึ่งให้เล็กที่สุด และใส่ฉากรับภาพไว้ที่ด้านในที่ตรงกันข้ามกับรูที่เจาะไว้ ภาพที่ปรากฏจะมีลักษณะกลับหัว ซึ่งสามารถสร้างแบบจำลองของกล้องรูเข็มทางเรขาคณิตได้ดังรูปที่ 2.2



รูปที่ 2.2 แสดงแบบจำลองทางเรขาคณิตของกล้องรูเข็ม

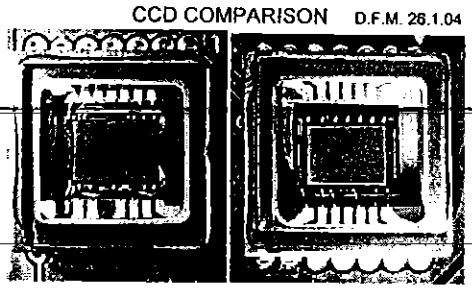
จากภาพที่ 2.2 กำหนดให้จุด O เป็นจุดศูนย์กลางของการฉาย(Projection center) จุด P คือจุดใดๆในโลกที่มีพิกัดเป็น (X,Y,Z) และจุด P' ซึ่งมีพิกัดเป็น (x',y') คือภาพฉายของจุด บนระนาบ Π' คือฉากรับภาพ f' คือระยะระหว่างจุดศูนย์กลางของการฉาย(O) และฉากรับภาพ(Π') เราสามารถแสดงความสัมพันธ์ระหว่างจุด P' และจุด P [1] ได้ดังนี้

$$x' = f'(X/Z) \quad (2.1)$$

$$y' = f'(Y/Z) \quad (2.2)$$

2.2 กล้องดิจิตอล

โครงสร้างการทำงานของดิจิตอลสามารถอธิบายโดยใช้แบบจำลองของกล้องรูเข็ม โดยที่ฉากรับภาพจะเป็น CCDs ซึ่งย่อมาจาก Charge Coupled Devices เป็นอุปกรณ์อิเล็กทรอนิคส์ที่ประกอบขึ้นจากสารกึ่งตัวนำชิ้นเล็กๆจำนวนมาก ที่มีความไวต่อแสงและสามารถแปลงความเข้มแสงเป็นสัญญาณทางไฟฟ้า จำนวนชิ้นส่วนสารกึ่งตัวนำนี้จะแสดงถึงประสิทธิภาพของภาพของ CCDs นอกจากจะใช้ในกล้องดิจิตอลแล้ว CCDs ใช้ในกล้องดูดาวที่ส่งไปกับยานอวกาศ สแกนเนอร์ เครื่องอ่านบาร์โค๊ดอีกด้วย รูปที่ 2.3 แสดงตัวอย่าง CCDs



รูปที่ 2.3 ตัวอย่าง CCDs

2.3 ค่าพารามิเตอร์ของกล้อง

ในหัวข้อที่ 2.1 ได้แสดงสมการที่แสดงความสัมพันธ์ระหว่างจุดใน 3 มิติของระบบพิกัดฉากของโลก กับจุด 2 มิติในระบบพิกัดฉากของภาพ สมการนี้จะสามารถนำมาใช้ได้จริงถ้าระยะทางทั้งหมดอ้างอิงจากระบบพิกัดฉากของกล้อง และระบบพิกัดฉากของภาพมีจุดกำเนิดอยู่ที่จุดสำคัญ (Principal point- คือจุดที่ แกนของระบบพิกัดของกล้องตัดกับฉากรับภาพ) แต่ในทางปฏิบัติแล้ว ระบบพิกัดฉากของโลกและระบบพิกัดฉากของภาพมีความสัมพันธ์กันโดยกลุ่มของค่าทางกายภาพจำนวนหนึ่ง เช่นระยะโฟกัสของเลนส์ ขนาดของพิกเซลล์ ตำแหน่งของจุดสำคัญ ตำแหน่งและทิศทางของกล้อง เราสามารถแยกค่าทางกายภาพเหล่านี้ออกเป็น 2 กลุ่มคือ ค่าพารามิเตอร์ภายใน(Intrinsic Parameter) ที่แสดงความสัมพันธ์ระหว่างระบบพิกัดฉากของกล้องกับระบบพิกัดฉากของภาพ เราเรียกเมทริกซ์ที่แสดงค่าคุณสมบัติภายในเหล่านี้ว่า เมทริกซ์การวัดกล้อง (Camera Calibration Matrix) ส่วนค่าพารามิเตอร์ภายนอกที่แสดงความสัมพันธ์ระหว่างระบบพิกัดฉากของโลกที่หยุดนิ่ง กับระบบพิกัดฉากของกล้อง เป็นการบอกตำแหน่งและทิศทางของกล้องในอวกาศ สามารถเรียกได้อีกอย่างว่า การเคลื่อนที่ของกล้อง (Camera Motion)

เพื่อให้การพิจารณาการทำงานสามารถทำได้ง่าย ในการคำนวณจะละความจริงที่ว่า กล้องประกอบด้วยเลนส์ จะไม่สนใจจุดที่อยู่ระหว่างจุดศูนย์กลางของการฉายกับฉากรับภาพ เราจะกลับมาพิจารณาผลของเลนส์ที่มีต่อภาพในภายหลัง

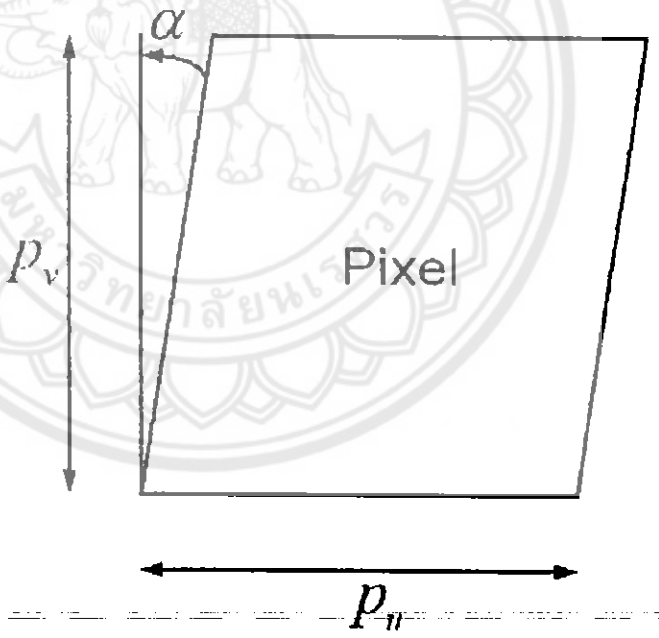
2.4 เมทริกซ์การวัดกล้อง(Camera Calibration Matrix)

กำหนดแทนเมทริกซ์การวัดของกล้องด้วย K ประกอบด้วยค่าพารามิเตอร์ภายในของกล้องในกระบวนการสร้างภาพ เมทริกซ์นี้ใช้ในการแปลงไปและกลับระหว่างระบบพิกัดฉากของกล้องและระบบพิกัดฉากของภาพ [2]

$$K = \begin{pmatrix} \frac{f}{p_u} & (\tan\alpha)\frac{f}{p_v} & u_0 \\ 0 & \frac{f}{p_v} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.3)$$

ระยะโฟกัส f ทำหน้าที่เป็นตัวแปรขยาย ในกล้องปกติระยะโฟกัสมักจะมีค่าไม่เท่ากับ 1 และระยะโฟกัสอาจจะมีการเปลี่ยนแปลงระหว่างกระบวนการสร้างภาพ และเมื่อมีการเปลี่ยนแปลงระยะโฟกัส ก็จะต้องมีการวัดค่าคุณสมบัติภายในของกล้องอีกครั้ง

ค่าตัวแปร p_u และ p_v แสดงถึงความกว้างและความยาวของพิกเซลล์ของรูปภาพ $c = [u_0, v_0]^T$ คือจุดสำคัญ(Principal Point)และ α แสดงถึงมุมของการหมุน(Skew angle)ของพิกเซลล์ รูปที่ 2.4 แสดงความหมายของตัวแปรต่างๆ



รูปที่ 2.4 อธิบายความหมายของค่าพารามิเตอร์ภายใน

เพื่อให้สามารถเข้าใจได้ง่าย จึงกำหนดค่าตัวแปรของเมทริกซ์การวัดใหม่ดังนี้

$$K = \begin{pmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

เมื่อ f_u และ f_v คือระยะโฟกัสวัดในความกว้างและความสูงของพิกเซลล์ s แสดงถึงการบิดของพิกเซลล์ และอัตราส่วน $f_u : f_v$ จะแสดงถึงอัตราส่วนของกล้อง และในทางปฏิบัติจะกำหนด ให้ $s = 1$

เราจะใช้เมทริกซ์การวัดในการแปลงจุดในระบบพิกัดฉากของกล้องมายังระบบพิกัดฉากของภาพ เมื่อกำหนดให้ m คือจุดในระบบพิกัดฉากภาพ และ M คือจุดในระบบพิกัดฉากกล้อง และ K คือเมทริกซ์การวัดของกล้อง เราสามารถแสดงกระบวนการแปลงจากระบบพิกัดฉากกล้องไปยังระบบพิกัดฉากของภาพด้วยสมการดังต่อไปนี้

$$m = KM \tag{2.5}$$

2.5 การเคลื่อนที่ของกล้อง(Camera Motion)

การเคลื่อนที่ของพิกัดฉากของกล้องในฉาก 3 มิติสามารถแสดงได้ด้วยเมทริกซ์การหมุน (Rotation Matrix) R และเวคเตอร์การเลื่อนที่ (Translation Vector) t การเคลื่อนที่ของกล้องจากระบบพิกัดฉาก C_1 ไปยังระบบพิกัดฉาก C_2 สามารถแสดงได้โดย [1]

$$C_2 = \begin{pmatrix} R & t \\ 0_3^T & 1 \end{pmatrix} C_1 \tag{2.6}$$

เมื่อ R คือเมทริกซ์การหมุนที่มีมิติ 3x3 และ t คือเวคเตอร์การเลื่อนที่ในทิศทาง X, Y และ Z การเคลื่อนที่ของจุดในฉากจะเท่ากับการย้อนกลับของกล้อง แสดงได้ดังสมการด้านล่าง

$$M_2 = \begin{pmatrix} R^T & -R^T t \\ 0_3^T & 1 \end{pmatrix} M_1 \tag{2.7}$$

เมื่อเรารวมสมการต่างเข้าด้วยกัน เราสามารถแสดงการฉายภาพได้ดังนี้[1]

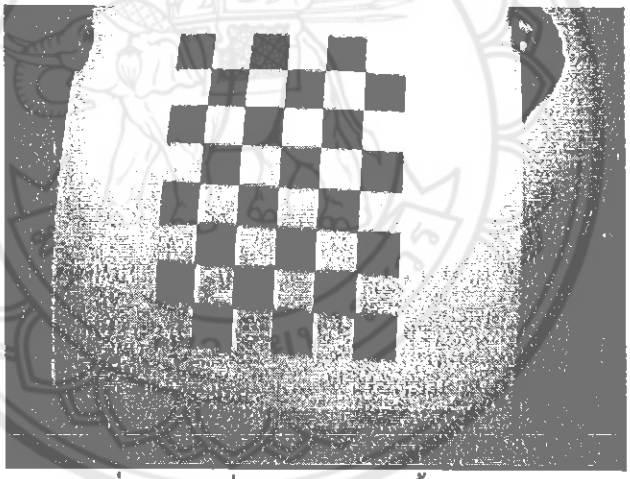
$$m = K\begin{bmatrix} R & t \end{bmatrix} M \tag{2.8}$$

$$m = PM \tag{2.9}$$

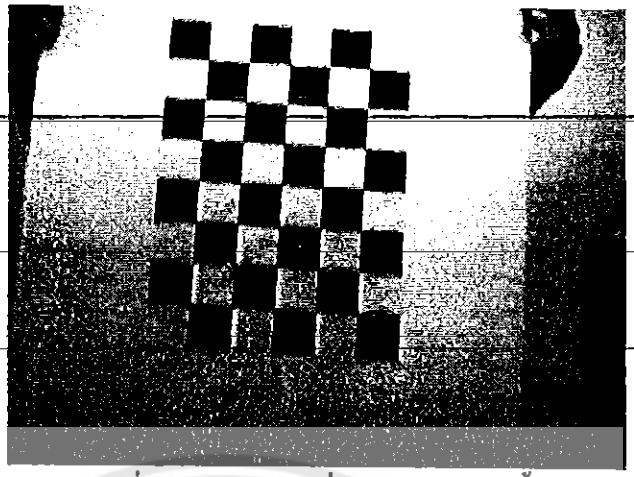
$$P = K\begin{bmatrix} R & t \end{bmatrix} \tag{2.10}$$

2.6 ค่าสัมประสิทธิ์ความบิดเบี้ยวของเลนส์(Lens distortion coeficient)

ภาพที่ได้จากกล้องที่ประกอบด้วยด้วยเลนส์ จะประกอบด้วยความผิดพลาดที่เกิดจากเลนส์ ภาพที่ 2.5 แสดงภาพที่ได้จากกล้องที่ประกอบด้วยความผิดพลาดที่เกิดจากเลนส์ จะสังเกตเห็นความบิดเบี้ยวเนื่องจากเลนส์ทำให้ขอบของกระดานหมากฮอสมองเห็นเป็นเส้นโค้ง และภาพที่ 2.6 เป็นภาพเดียวกับภาพที่ 2.5 แต่ได้รับการปรับแก้ความบิดเบี้ยวของเลนส์แล้ว



รูปที่ 2.5 ภาพที่แสดงความบิดเบี้ยวของเลนส์



ภาพ 2.6 ภาพที่ได้รับการแก้ไขเนื่องจากความบิดเบี้ยวของเลนส์

ความผิดพลาดเนื่องจากความบิดเบี้ยวของเลนส์สามารถอธิบายได้ด้วยสมการ [2]

$$x_d = x + x(k_1 r^2 + k_2 r^4) + (2p_1 xy + p_2(r^2 + 2x^2)) \quad (2.11)$$

$$y_d = y + y(k_1 r^2 + k_2 r^4) + (2p_1 xy + p_2(r^2 + 2y^2)) \quad (2.12)$$

$$r^2 = x^2 + y^2 \quad (2.13)$$

เมื่อ x_d, y_d คือ พิกัดของจุดของภาพเนื่องมาจากความบิดเบี้ยวของเลนส์

x, y คือ พิกัดของภาพที่ได้รับการแก้ไขแล้ว

k_1, k_2, p_1, p_2 คือ สัมประสิทธิ์ความบิดเบี้ยวของเลนส์

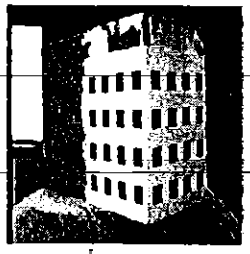
เมื่อเราทราบสัมประสิทธิ์ของเลนส์เราสามารถทำการปรับแก้ความผิดพลาดเนื่องจากความบิดเบี้ยวของเลนส์ได้ และจะทำให้เราสามารถได้จุดของภาพที่มีความถูกต้องมากขึ้น

2.7 การหาค่าพารามิเตอร์ต่างๆของกล้อง(Camera Calibration)

ในการทำงานทางด้านระบบการมองเห็นด้วยคอมพิวเตอร์มักเกี่ยวข้องกับการใช้งานค่าพารามิเตอร์ต่างๆของกล้อง จึงได้มีผู้พยายามคิดหาวิธีการ ในการหาค่าพารามิเตอร์ต่างๆของกล้องออกมาหลายวิธี ซึ่งสามารถแบ่งได้เป็น 2 ประเภท คือ [2]

1. การวัดด้วยเทคนิคการถ่ายภาพ ทำได้โดยการจับภาพวัตถุที่ใช้ในการวัด(Calibration Object)ที่ทราบตำแหน่งในพิกัด 3 มิติเป็นอย่างละเอียด กระบวนการวัดสามารถทำได้อย่างมี

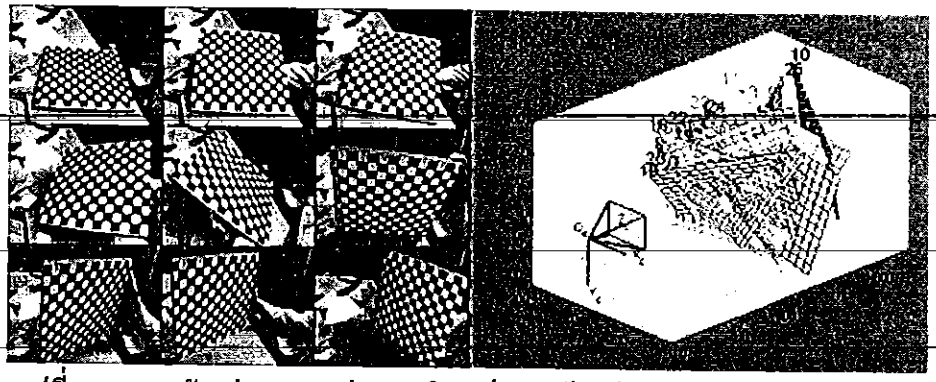
ประสิทธิภาพ วัตถุที่ใช้ในการวัดมักจะประกอบด้วยระนาบ 2 หรือ 3 ระนาบที่ตั้งฉากซึ่งกันและกัน เทคนิคการวัดด้วยวิธีนี้ต้องการสภาพแวดล้อมที่มีความเที่ยงตรงสูง จึงมีค่าใช้จ่ายในการวัดมาก



รูปที่ 2.7 วัตถุที่ใช้ในการวัดค่ากล้อง

2. การวัดด้วยตัวเอง (Self- Calibration) เทคนิคของวิธีการนี้ไม่จำเป็นต้องใช้วัตถุในการวัด ใช้เพียงการเคลื่อนที่กล้องไปในฉากที่หยุดนิ่ง จุดต่างๆในภาพจะสร้างเงื่อนไข ที่ทำให้เราสามารถคำนวณหาค่าพารามิเตอร์ภายในของกล้องได้ ซึ่งมาจากเงื่อนไขที่ว่า ถ้าภาพที่ได้จากกล้องตัวเดียวกันต้องมีค่าพารามิเตอร์ภายในเท่ากัน ในกระบวนการวัดต้องใช้ภาพอย่างน้อย 3 ภาพในการหาค่าพารามิเตอร์ภายในและภายนอกของกล้องที่จะช่วยในกระบวนการสร้างกลับของภาพ เทคนิควิธีการนี้มีความยืดหยุ่นสูง แต่ผลลัพธ์ที่ได้ยังไม่สามารถเชื่อถือได้

ในปี พ.ศ. 2541 Zhengyou Zhang ได้นำเสนอวิธีการใหม่ในการหาค่าพารามิเตอร์ต่างๆของกล้องในบทความที่ชื่อว่า A Flexible New Technique for Camera Calibration ซึ่งวิธีการที่นำเสนอ จะเป็นการนำเอาเทคนิควิธีการวัดด้วยเทคนิคการถ่ายภาพ และการวัดด้วยตัวเองมาใช้ คือ จะจับภาพของกระดานหมากฮอสในมุมต่างๆอย่างน้อย 2 ภาพ แล้วนำมาคำนวณหาค่าพารามิเตอร์ต่างๆของกล้อง ซึ่งมีข้อกำหนดว่ากระดานหมากฮอสหรือกล้องอย่างใดอย่างหนึ่งสามารถเคลื่อนที่ได้อย่างอิสระโดยที่เราไม่จำเป็นต้องทราบตำแหน่งที่แน่นอนทั้งของกล้องและกระดานหมากฮอสเลย ซึ่งวิธีการนี้ให้ผลลัพธ์ที่มีความถูกต้องในระดับที่สามารถนำไปใช้งานได้

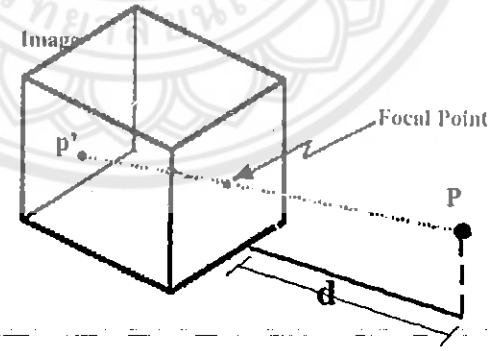


รูปที่ 2.8 แสดงตัวอย่างการหาค่าพารามิเตอร์ของกล้องด้วยวิธีของ Zhengyou Zhang

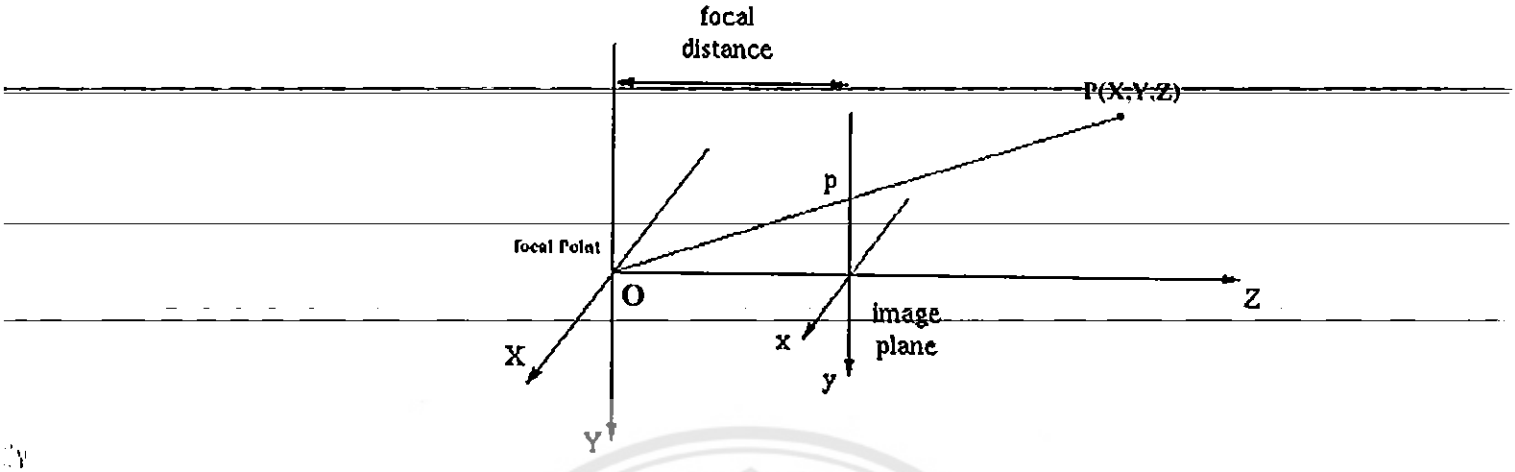
เนื่องจากประสิทธิภาพและความยืดหยุ่นของวิธีการนี้ จึงได้มีการนำไปใช้สร้างโปรแกรมสำหรับหาค่าพารามิเตอร์ของกล้องหลายโปรแกรม ในโครงงานนี้ได้เลือกใช้ฟังก์ชั่นใน OpenCV ในการหาค่าพารามิเตอร์ของกล้อง

2.8 วิธีการหาระยะลึกโดยใช้กล้อง 2 ตัว

พิจารณาจากรูปที่ 2.9 แสดงการทำงานของกล้องรูเข็มจะเห็นว่าระยะลึกที่ต้องการหาคือระยะห่างระหว่างจุด P กับระนาบด้านหน้าของกล้องที่มีจุดโฟกัสอยู่ด้วย จากรูปที่ 2.9 d คือระยะลึกของจุด P

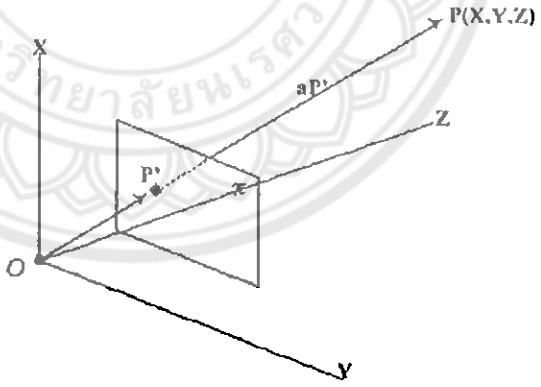


รูปที่ 2.9 แสดงระยะลึกของภาพของกล้องรูเข็ม



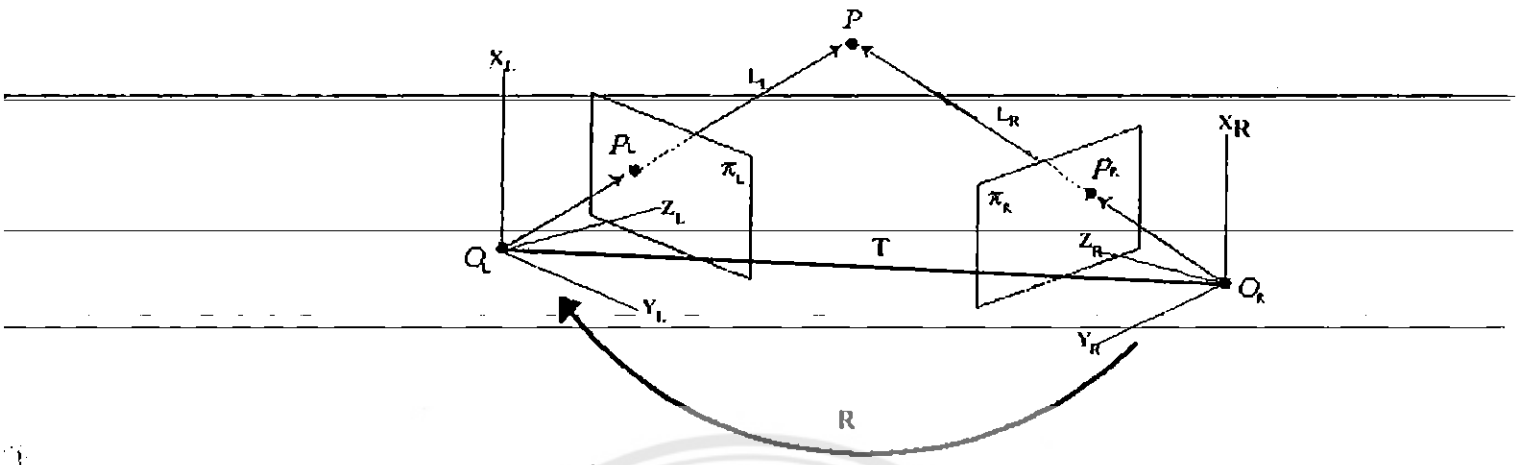
รูปที่ 2.10 แสดงความสัมพันธ์ระหว่างพิกัดของจุดและระยะลึก

รูปที่ 2.10 แสดงระบบพิกัดฉากของกล้องที่มีจุดโฟกัสอยู่ที่จุดกำเนิด และฉากรับภาพขนานกับระนาบ XY อยู่ในตำแหน่งที่ Z มากกว่า 0 และพิกัดของ P คือ (X,Y,Z) จากรูปจะเห็นว่าระยะลึกของจุด P คือ Z ดังนั้นในระบบที่เอาระบบพิกัดฉากของกล้องเป็นระบบอ้างอิง ระยะลึกสามารถหาได้จากองค์ประกอบทางแกน Z ของพิกัดของจุดที่เราสนใจ



รูปที่ 2.11 แสงรังสีที่ใช้ในการหาระยะลึก

การหาพิกัดของจุดที่เราสนใจ จากข้อมูลของตำแหน่งในระบบพิกัดฉากของภาพของจุดนั้นเรียกว่าการสร้างกลับของภาพ(Image Reconstruction) เมื่อ P' คือจุดที่อยู่บนฉากรับภาพ π และ P คือจุดที่เราสนใจ การสร้างกลับของภาพจากกล้องตัวเดียวจะทำให้เราได้เส้นตรงที่มีสมการเป็น $a\overline{OP'}$ เมื่อ $a \in R$ ดังนั้นเราจึงต้องใช้กล้องอย่างน้อย 2 ตัวในการหาพิกัดของจุดที่เราสนใจ



รูปที่ 2.12 แสดงการหาพิกัดของจุดที่สนใจด้วยกล้อง 2 ตัว

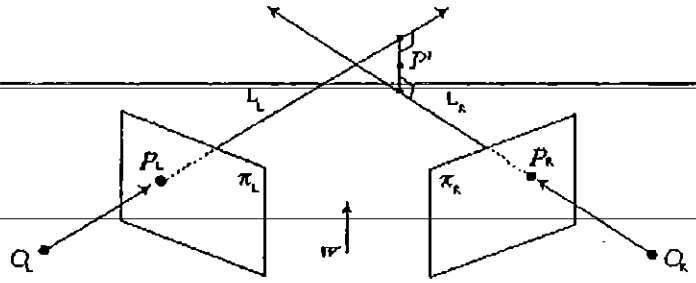
รูปที่ 2.12 แสดงระบบที่ประกอบด้วยกล้อง 2 ตัว จากรูปจะเห็นว่า เราสามารถหาตำแหน่งของจุด P ได้จากการหาจุดตัดของเส้นตรงที่ลากผ่านจุด O_L และ p_L และเส้นตรงที่ผ่านจุด O_R และ p_R เมื่อ กำหนดว่า พิกัดของกล้องตัวซ้ายเป็นพิกัดอ้างอิง และกำหนดว่า R คือเมทริกซ์ที่แสดงการหมุนของระบบพิกัดฉากของกล้องด้านขวาเมื่อเทียบกับพิกัดฉากของกล้องด้านซ้าย และ T คือ เวคเตอร์ที่แสดงการเลื่อนที่ของจุดกำเนิดของระบบพิกัดฉากของกล้องด้านขวาเทียบกับพิกัดฉากของกล้องด้านซ้าย เราสามารถแสดงได้ว่าเส้นตรง L_L ได้ด้วย $a\overrightarrow{O_L p_L}$ และ แทนเส้นตรง L_R ด้วย $T+b(R^T\overrightarrow{O_R p_R})$

ในทางปฏิบัติแล้วในการหาค่าตำแหน่งของจุดของวัตถุในภาพมักจะมีความผิดพลาดรวมอยู่ด้วย ดังนั้นเส้นทั้งสองจึงอาจจะไม่ตัดกัน ดังแสดงในภาพที่ 2.13 ในการคำนวณหาพิกัดของจุดของวัตถุที่สนใจ จะหาจากค่าเฉลี่ยจากจุดที่เส้นตรงทั้งสองที่อยู่ใกล้กันที่สุด ในรูปที่ 2.13 คือจุด P' ซึ่งเป็นจุดกึ่งกลางของส่วนของเส้นตรงที่ขนานกับเวกเตอร์ w เชื่อมเส้นตรง L_L และ L_R เข้าด้วยกัน โดยที่ w คือเวกเตอร์ที่ตั้งฉากกับเส้นตรงทั้งสอง เราสามารถหาค่าจุดบนเส้นตรง L_L และ L_R ที่อยู่ใกล้ที่สุดได้จากแก้ระบบสมการ[3]

$$T = a\overrightarrow{O_L p_L} - b(R^T)\overrightarrow{O_R p_R} + c(\overrightarrow{O_L p_L} \times (R^T)\overrightarrow{O_R p_R}) \qquad (2.14)$$

และสามารถหาจุด P' ที่เป็นจุดที่เป็นค่าเฉลี่ยได้จาก[3]

$$P' = \frac{(a\overrightarrow{O_L p_L} + T + b(R^T)\overrightarrow{O_R p_R})}{2} \qquad (2.15)$$



รูปที่ 2.13 แสดงการหาพิกัดของจุดที่สนใจด้วยกล้อง 2 ตัวในทางปฏิบัติ

เมื่อรู้ค่าคุณสมบัติของกล้อง สามารถหาค่า R และ T ได้ดังนี้[3]

$$R = R_R R_L^T \tag{2.16}$$

$$T = T_L - R^T T_R \tag{2.17}$$

เมื่อ R_L คือเมทริกซ์การหมุนของกล้องด้านซ้าย

R_R คือเมทริกซ์การหมุนของกล้องด้านขวา

T_L คือเวคเตอร์การเลื่อนที่ของกล้องด้านซ้าย

T_R คือเมทริกซ์การเลื่อนที่ของกล้องด้านขวา

2.9 DirectShow API

ไมโครซอฟต์ไดเรกโชว์ เอพีไอ (Microsoft DirectShow API) เป็นสถาปัตยกรรมสำหรับสื่อข้อมูลบนแพลตฟอร์มไมโครซอฟต์วินโดว์ (Microsoft Windows Platform) โดยเป็นส่วนหนึ่งของไมโครซอฟต์ไดเรกเอ็กซ์ เอพีไอ(Microsoft DirectX API) ซึ่งเราสามารถนำไดเรกโชว์มาใช้ในการเขียนโปรแกรมเพื่อเล่นหรือเก็บภาพจากสื่อมัลติมีเดียได้ เราสามารถเขียนแอพพลิเคชันโดยใช้ไดเรกโชว์ เช่น โปรแกรมเล่นหรือตัดต่อวิดีโอ โปรแกรมจับภาพจากกล้องดิจิตอล

ไดเรกโชว์ใช้หลักการของ COM(Component Object Model) ดังนั้นในการใช้งานไดเรกโชว์จึงต้องมีความเข้าใจการเขียนโปรแกรมแบบ COM Client

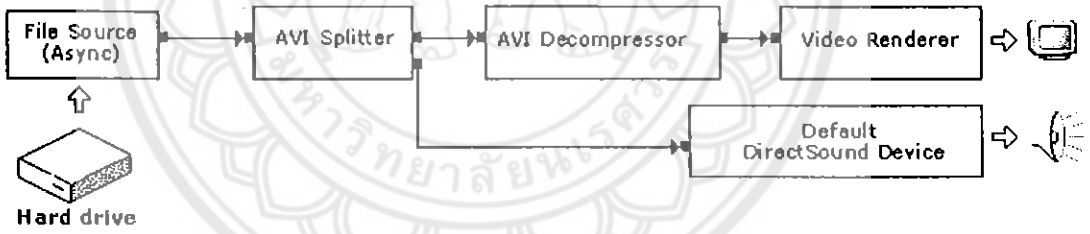
2.9.1 สถาปัตยกรรมของไดเรคโชว์

โครงสร้างการทำงานของไดเรคโชว์เกิดจากเอาหน่วยของซอฟแวร์(Software Component)ที่เรียกว่า ฟิลเตอร์(Filter) มาต่อกัน ฟิลเตอร์คือหน่วยของโปรแกรมที่ทำการจัดการกับข้อมูลในรูปแบบสื่อมัลติมีเดีย ตัวอย่างของงานที่ทำโดย ไดเรคโชว์ฟิลเตอร์

- อ่านไฟล์จากดิสก์
- จับภาพวิดีโอ จากอุปกรณ์จับภาพ เช่น กล้องดิจิตอล การ์ดวิดีโอ
- จัดการกับข้อมูลที่มีการบีบอัด เช่น MPEG-1 Video
- ส่งข้อมูลไปแสดงผลยังอุปกรณ์แสดงผล เช่น การ์ดแสดงผล การ์ดเสียง

ฟิลเตอร์สามารถรับข้อมูลเข้าและส่งข้อมูลออก ตัวอย่างเช่น ฟิลเตอร์สำหรับคลายการบีบอัดข้อมูล MPEG-1 (Decode MPEG-1 Video) ข้อมูลเข้าคือ สายข้อมูล MPEG-1 และข้อมูลออกคือ ชุดของข้อมูลวิดีโอที่ได้รับการคลายการบีบอัดแล้ว

ในไดเรคโชว์แอพพลิเคชันจะจัดการกับงานต่างๆ โดยการนำเอาฟิลเตอร์มาต่อเข้าด้วยกัน ดังนั้นข้อมูลที่ออกจากฟิลเตอร์หนึ่งจะเข้าสู่ฟิลเตอร์ตัวอื่น กลุ่มของฟิลเตอร์ที่มาเชื่อมต่อเข้าด้วยกันเรียกว่า ฟิลเตอร์กราฟ (Filter Graph) ตัวอย่าง ฟิลเตอร์กราฟที่ใช้ในการเล่นกลับ ไฟล์ AVI



รูปที่ 2.14 ฟิลเตอร์กราฟสำหรับเล่นกลับไฟล์ AVI

File Source Filter จะทำการอ่านไฟล์ AVI จากฮาร์ดดิสก์

AVI Spliter Filter จะทำการแบ่งข้อมูลออกเป็น 2 สายคือ ข้อมูลวิดีโอที่ได้รับการบีบอัด และข้อมูล ออดิโอ

AVI Decompressor Filter จะทำการคลายข้อมูลวิดีโอที่ได้รับการบีบอัดได้ผลลัพธ์เป็นข้อมูลของวิดีโอแต่ละเฟรม

Video Renderer Filter จะทำการวาดข้อมูลภาพแต่ละเฟรมลงบนหน้าจอ

Default DirectSound Device Filter จะทำการแสดงเสียงออกทางลำโพง

แอพพลิเคชั่นไม่สามารถจัดการกับกระแสข้อมูลในฟิลเตอร์กราฟได้โดยตรงแต่สามารถควบคุมฟิลเตอร์เหล่านี้ได้โดยผ่านโครงสร้างโปรแกรมระดับสูง (High Level Component) ที่เรียกว่า Filter Graph Manager ซึ่งแอพพลิเคชั่นสามารถสั่งให้ฟิลเตอร์กราฟเริ่มต้นทำงานหรือหยุดทำงานได้โดยผ่าน Filter Graph Manager นอกจากนี้แอพพลิเคชั่นสามารถรู้เหตุการณ์ที่เกิดภายในฟิลเตอร์กราฟโดยผ่านMessage ที่ส่งผ่าน Filter Graph Manager

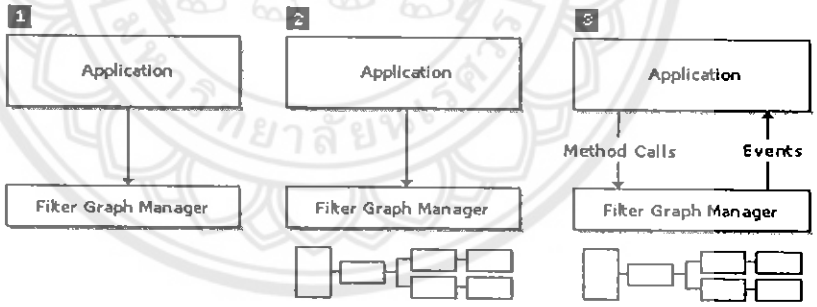
นอกจากช่วยในการควบคุมการทำงานของฟิลเตอร์กราฟแล้ว Filter Graph Manager ยังช่วยในการสร้างฟิลเตอร์กราฟอีกด้วย

2.9.2 ขั้นตอนการสร้างแอพพลิเคชันด้วย ไดเรคโชว์

ก่อนที่จะทำการสร้างแอพพลิเคชั่นด้วยไดเรคโชว์ต้องทำการติดตั้งไดเรคเอ็กซ์ เอสดีเค (DirectX SDK)ก่อน รายละเอียดการติดตั้งไดเรคเอ็กซ์ สามารถอ่านได้ที่ภาคผนวก ก

ขั้นตอนในการสร้างโปรแกรมที่ใช้ไดเรคโชว์มีอยู่ 3 ขั้น

1. สร้างอินสแตน(instance) ของ Filter Graph Manager
2. ใช้ Filter Graph Manager ในการสร้างฟิลเตอร์กราฟ
3. เริ่มต้นการทำงานของฟิลเตอร์กราฟ



รูปที่ 2.15 ขั้นตอนการสร้างแอพพลิเคชันด้วยไดเรคโชว์

ตัวอย่างโปรแกรมสำหรับเล่นกลับไฟล์ AVI

1. ใช้ฟังก์ชั่น CoInitialize เพื่อเป็นการบอกคอมพิวเตอร์จะมีการใช้ COM ในโปรแกรม

```
HRESULT hr = CoInitialize(NULL);
if (FAILED(hr))
{
    // Add error-handling code here.
}
```

2. การสร้าง Instance ของ FilterGraph Managerด้วยฟังก์ชั่น CoCreateInstance

```
IGraphBuilder *pGraph;
HRESULT hr = CoCreateInstance ( CLSID_FilterGraph,
                                NULL,
                                CLSCTX_INPROC_SERVER,
                                IID_IGraphBuilder,
                                (void **)&pGraph
                                )
```

3. ใช้ Filter Graph Manager ที่ได้สร้างออบเจ็กท์ IMediaControl สำหรับควบคุมการทำงานของฟิลเตอร์กราฟและสร้างออบเจ็กท์ IMediaEventสำหรับรับ Massage จากฟิลเตอร์กราฟ

```
IMediaControl  *pControl;
IMediaEvent  *pEvent;
hr = pGraph->QueryInterface(IID_IMediaControl, (void  **)&pControl);
hr = pGraph->QueryInterface(IID_IMediaEvent, (void  **)&pEvent);
```

4. การสร้างฟิลเตอร์กราฟสำหรับเล่นกลับไฟล์ AVI ซึ่งในตัวอย่างจะเป็นการสร้างแบบอัตโนมัติ

```
hr = pGraph->RenderFile("C:\\Example.avi", NULL);
```

5. เริ่มต้นการทำงานของฟิลเตอร์กราฟด้วยออบเจ็กท์ IMediaControl ที่สร้างขึ้น

```
hr = pControl -> Run();
```

6. รอจนกว่าจะเกิดเหตุการณ์จบไฟล์ด้วยออบเจ็กท์ IMediaEvent

```
long evCode  =  0;
pEvent -> WaitForCompletion(INFINITE, &evCode);
```

7. เมื่อจบการทำงานของโปรแกรมต้องมีการคืนหน่วยความจำให้กับระบบ

```
pControl ->Release();
pEvent->Release();
pGraph->Release();
CoUnitialize();
```

ตัวอย่างโปรแกรมที่สมบูรณ์

```
#include <dshow.h>
void  main(void)
{
        IGraphBuilder *pGraph  =  NULL;
        IMediaControl  *pControl  = NULL;
        IMediaEvent  *pEvent  = NULL;
        //Initialize the COM library
        HRESULT hr  = Coinitialize(NULL);
        if(FAILED(hr))
        {
            printf("Error – Could not initialize COM library");
            return;
        }
        //Create the filter graph manager and query for interfaces.
            hr  =  CoCreateInstance(CLSID_FilterGraph,  NULL,
                                    CLSCTX_INPROC_SERVER, IID_IGraphBuilder,
                                    (void  **) &pGraph);
        if(FAILED(hr))
        {
            printf("Error – Could not create the Filter Graph Manager.");
            return;
        }
```

```
hr = pGraph->QueryInterface(IID_IMediaConTrol, (void **)&pControl);
hr = pGraph->QueryInterface(IID_IMediaEvent, (void **)&pEvent);

//Build the graph. IMPORTANT: Change this string to file on your system.
hr = pGraph->RenderFile("C:\\Example.avi", NULL);

if(SUCCEEDED(hr))
{
    //Run the graph
    hr = pControl->Run();
    if(SUCEEDED(hr))
    {
        //Wait for complete
        long evCode ;
        pEvent->WaitForCompletion(INFINITE, &evCode);
    }
}

pControl->Release();
pEvent->Release();
pGraph->Release();
CoUninitialize();
}
```

2.9.3 ไดเรคโชว์ฟิลเตอร์

ไดเรคโชว์ฟิลเตอร์คือ COM Object ที่ทำหน้าที่อย่างใดอย่างหนึ่ง ไดเรคโชว์ประกอบด้วยฟิล-เตอร์มากมายสำหรับให้แอพพลิเคชั่นเรียกใช้งาน นอกจากนี้ผู้ใช้ยังสามารถสร้างฟิลเตอร์ขึ้นใช้ได้เอง โดยการ สืบทอดจากฟิลเตอร์ที่มีอยู่แล้ว

ฟิลเตอร์จะประกอบด้วย pin ที่ใช้สำหรับส่งผ่านข้อมูลจากฟิลเตอร์หนึ่งไปยังอีกฟิลเตอร์หนึ่ง pin เป็น COM object สามารถแบ่งได้ 2 ชนิดคือ

1. อินพุทพิน(Input pin)จะทำหน้าที่รับข้อมูลเข้าสู่ฟิลเตอร์
2. เอาท์พุทพิน(Output pin) จะทำหน้าที่ส่งข้อมูลออกจากฟิลเตอร์

ฟิลเตอร์บางชนิดอาจจะประกอบด้วยอินพุทพินเพียงอย่างเดียว เช่น Renderer Filter หรือบางชนิดอาจจะมีเพียงเอาท์พุทพินเพียงอย่างเดียวเช่น Source Filter หรืออาจจะประกอบด้วยทั้งอินพุทและเอาท์พุทพิน เช่น Transform Filter ฟิลเตอร์อาจจะประกอบด้วยอินพุทหรือเอาท์พุทมากกว่า 1 ตัว

เราสามารถแบ่งฟิลเตอร์ในไดเรคโชว์ออกเป็นกลุ่มได้ดังนี้

Source Filter

ใช้สำหรับนำข้อมูลเข้าสู่ฟิลเตอร์กราฟ ซึ่งข้อมูลอาจจะมาจาก ไฟล์ในดิสก์ , ระบบเครือข่าย, กล้องหรือการ์ดสำหรับจับภาพจากวิดีโอ แต่ละ Source Filter สามารถจัดการกับข้อมูลต่างชนิดจากแหล่งข้อมูลเดียวกัน

Transform Filter

ทำหน้าที่รับข้อมูลทางอินพุทพิน ทำการประมวลผล และทำการส่งผลลัพธ์ออกทางเอาท์พุทพิน

Renderer Filter

ฟิลเตอร์นี้จะอยู่ในตำแหน่งท้ายสุดในฟิลเตอร์กราฟ ทำหน้าที่รับข้อมูลทางอินพุทพินแล้วแสดงออกสู่ผู้ใช้ เช่น ฟิลเตอร์สำหรับวาดภาพวิดีโอออกทางหน้าจอ

Splitter Filter

ทำหน้าที่แยกข้อมูลที่รับเข้ามาทางอินพุทพินออกเป็นข้อมูลตั้งแต่ 2 สายทางด้านเอาท์พุทพิน เช่น AVI Spliter Filter ทำหน้าที่แยกข้อมูลภาพและเสียงออกจากไฟล์ AVI

Mux Filter

ทำหน้าที่ตรงกันข้ามกับ Splitter Filter คือทำหน้าที่รวมข้อมูลตั้ง 2 สายขึ้นไปให้ออกมาเป็นข้อมูลสายเดียว เช่น AVI Mux รับข้อมูลวิดีโอ และเสียง นำมารวมกันและส่งผลลัพธ์เป็นข้อมูลในรูปแบบ AVI ออกไป

2.9.4 Filter Graph Manager

filter Graph Manager คือ COM object ที่ทำหน้าที่สร้างและควบคุมการทำงานของฟิลเตอร์กราฟ เป็นศูนย์กลางการทำงานของไดเรคโชว์ แอพพลิเคชั่นใช้ Filter Graph Manager ในการสร้าง ควบคุมการทำงานและติดต่อกับฟิลเตอร์กราฟ นอกจากนี้ Filter Graph Manager ยังเป็นตัวช่วยทำให้ฟิลเตอร์ต่างๆในฟิลเตอร์กราฟสามารถทำงานสอดคล้องกัน ทำการส่งข้อมูลเกี่ยวกับเหตุการณ์ที่เกิดภายในฟิลเตอร์กราฟไปสู่แอพพลิเคชั่น และยังประกอบด้วยฟังก์ชั่นที่ควบคุมการทำงานของฟิลเตอร์กราฟอีกมากมาย

Filter Graph Manager ประกอบด้วย Interface ที่สำคัญดังนี้

IFilterGraph Interface

IFilterGraph Interface ประกอบด้วยฟังก์ชันที่เกี่ยวข้องกับการสร้างฟิลเตอร์กราฟ เช่น

- AddFilter ทำหน้าที่เพิ่ม ฟิลเตอร์เข้าไปยังฟิลเตอร์กราฟ
- RemoveFilter ทำหน้าที่นำเอาฟิลเตอร์ที่กำหนดออกจากฟิลเตอร์กราฟ
- EnumFilter ทำหน้าที่ตรวจสอบจำนวนและชนิดของฟิลเตอร์ในฟิลเตอร์กราฟ
- ConnectDirect ทำหน้าที่นำพินที่กำหนดมาต่อกัน
- Disconnect ทำหน้าที่ตัดการเชื่อมต่อระหว่างพินที่กำหนด

IMediaControl Interface

ประกอบด้วยชุดฟังก์ชั่นสำหรับควบคุมการกระแสข้อมูลในฟิลเตอร์กราฟ ประกอบด้วย การเริ่มต้นการทำงานของฟิลเตอร์กราฟ(Run) หยุดการทำงานชั่วคราว(Pause) หยุดการทำงานของฟิลเตอร์กราฟ(Stop)

IMediaEvent Interface

ประกอบด้วยฟังก์ชั่นสำหรับคืนค่าข้อมูลเกี่ยวกับเหตุการณ์(Event) แอพพลิเคชั่นสามารถใช้ Interface นี้ในการตอบสนองต่อเหตุการณ์ที่เกิดภายในฟิลเตอร์กราฟ เช่น การเล่นกลับไฟล์มาถึงจุดสุดท้ายของไฟล์แล้ว หรือเกิดความผิดพลาดในการ เรนเดอร์ภาพบนหน้าจอ เป็นต้น ตัวอย่างฟังก์ชั่นใน IMediaEvent Interface

- GetEvent ทำหน้าที่คืนค่าข้อมูลของเหตุการณ์จาก EventQueue

2.9.5 การจับภาพวิดีโอในไดเรคโชว์ (Video Capture in DirectShow)

ความหมายของการจับภาพวิดีโอในไดเรคโชว์ คือแอพพลิเคชั่นที่ข้อมูลวิดีโอได้มาจากอุปกรณ์ฮาร์ดแวร์ อุปกรณ์สำหรับการจับภาพประกอบด้วย กล้องดิจิตอล ,TV tuner card , Video tape recorder ซึ่งแอพพลิชั่นในการจับภาพวิดีโอนี้สามารถที่จะบันทึกข้อมูลวิดีโอที่ได้ลงดิสก์หรือนำแสดงแบบทันที

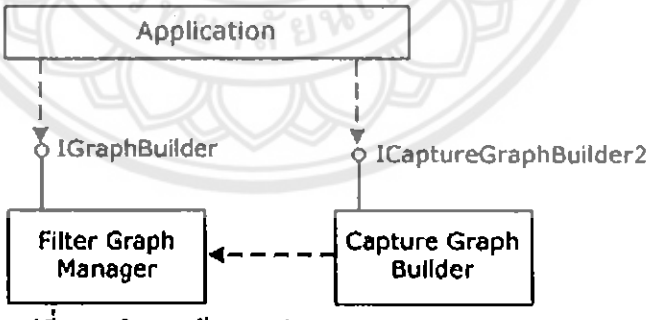
2.9.5.1 ฟิลเตอร์กราฟสำหรับการจับภาพวิดีโอ

ฟิลเตอร์กราฟที่ทำหน้าที่จับภาพวิดีโอหรือออดิโอ เรียกว่า Capture Filter Graph ซึ่งจะมีความซับซ้อนมากกว่าฟิลเตอร์กราฟที่ใช้ในการเล่นกลับไฟล์วิดีโอ เพื่อทำให้การสร้าง Capture Filter Graph ง่ายขึ้น ไดเรคโชว์จึงได้สร้างอินเตอร์เฟซสำหรับช่วยในการสร้าง Capture Filter Graph ชื่อว่า ICaptureGraphBuilder2 ที่ประกอบด้วยฟังก์ชั่นสำหรับช่วยในการสร้าง และควบคุมการทำงานของ Capture Filter Graph

โครงสร้างการทำงานของ ICaptureGraphBuilder ,Filter Graph Manager และ แอพพลิเคชัน

ขั้นตอนในการสร้างและใช้งานฟิลเตอร์กราฟ

1. สร้างอินสแตนของ ICaptureGraphBuilder2
2. สร้างอินสแตนของ Filter Graph Manager
3. ทำการเชื่อมโยงระหว่างอินสแตนของ ICaptureGraphBuilder2 และ Filter Graph Manager โดยการใช้ฟังก์ชั่น SetFilterGraph ของ ICaptureGraphBuilder2



รูปที่ 2.16 โครงสร้างการทำงานของ ICaptureGraphBuilder

ตัวอย่างฟังก์ชั่นที่ทำการสร้าง Capture Filter Graph

```
HRESULT InitCaptureGraphBuilder(    IGraphBuilder **ppGraph,
                                    ICaptureGraphBuilder2 **ppBuilder )
{
    if (!ppGraph || !ppBuilder)
    {
        return E_POINTER;
    }

    IGraphBuilder *pGraph = NULL;
    ICaptureGraphBuilder2 *pBuilder = NULL;

    // Create the Capture Graph Builder.
    HRESULT hr = CoCreateInstance(CLSID_CaptureGraphBuilder2,
                                  NULL,
                                  CLSCTX_INPROC_SERVER,
                                  IID_ICaptureGraphBuilder2,
                                  (void **) &pGraph );

    if (SUCCEEDED (hr))
    {
        // Create the Filter Graph Manager
        hr = CoCreateInstance(  CLSID_FilterGraph,
                                0,
                                CLSCTX_INPROC_SERVER,
                                IID_IGraphBuilder,
                                (void **) &pGraph );
```

```
if (SUCCEEDED(hr))
{
    // Initialize the Capture Graph Builder.
    pBuilder->SetFiltergraph(pGraph);

    //Return both interface pointer to the caller.
    *ppBuilder = pBuilder;
    *ppGraph = pGraph;
    return S_OK;
  }
  else
  {
    pBilder->Release();
  }
 }
return hr; //Failed
}
```

2.9.5.2 อุปกรณ์ที่ใช้ในการจับภาพวิดีโอในไดเรคโชว์

ไดเรคโชว์รองรับอุปกรณ์จับภาพที่ไดรเวอร์ของอุปกรณ์นั้นรองรับมาตรฐาน WDM(Windows Driver Model) และมาตรฐาน VFW (Video For Windows)

ในไดเรคโชว์ อุปกรณ์จับภาพที่มีไดรเวอร์ตามโครงสร้าง WDM แต่ละตัวจะปรากฏเหมือนฟิลเตอร์ตัวหนึ่งในฟิลเตอร์กราฟ WDM Video Capture Filter จะเป็นกลุ่มของฟิลเตอร์ที่ทำหน้าติดต่อกับอุปกรณ์จับภาพแต่ละชนิด ในขั้นตอนการใช้งานนั้น WDM Video Capture Filter จะทำการปรับค่าตัวเองบนพื้นฐานของลักษณะของไดรเวอร์และชื่อของฟิลเตอร์ที่ใช้ จะปรากฏตามชื่อของไดรเวอร์ เราจะไม่พบฟิลเตอร์ที่เรียกว่า WDM Video Capture Filter

นอกจากนี้ ไดเรคโชว์ยังรองรับอุปกรณ์จับภาพที่ใช้มาตรฐาน VFW (Video For Windows) Driver ซึ่งปัจจุบันไดรเวอร์ชนิดนี้เหลือใช้งานไม่มากนัก โดยการใช้งานจะทำผ่าน VFW Filter

2.9.5.3 ฟิลเตอร์สำหรับจับภาพในไดเรคโชว์

Capture Filter มีลักษณะบางประการที่แตกต่างจากฟิลเตอร์ชนิดอื่น แต่ในการใช้งานเราจะใช้ Capture Graph Builder ซ่อนรายละเอียดต่างๆเหล่านั้น เพื่อช่วยให้เข้าใจการทำงานของ Capture Graph มีรายละเอียดที่ต้องทำความเข้าใจดังนี้

กลุ่มของพิน(Pin Categories)

Capture Filter มักจะมีเอาท์พุทพินอย่างน้อย 2 ชนิดที่ทำการส่งออกข้อมูลชนิดเดียวกัน เช่น ส่งข้อมูลสำหรับการแสดงผล และพินสำหรับส่งข้อมูลสำหรับการจับภาพ อย่างไรก็ตาม ข้อมูลบางชนิดอาจจะไม่เหมาะที่จะแบ่งแยกตามพิน แทนที่จะแบ่งแยกตามหน้าที่ แต่จะดีกว่าถ้าเราจะระบุพินโดยใช้ GUID ที่เรียกว่า pin category

พินสำหรับแสดงผล และพินสำหรับการจับภาพ

อุปกรณ์ในการจับภาพบางส่วนจะมีการแยกข้อมูลเอาท์พุท สำหรับการแสดงผลและเอาท์พุทสำหรับการจับภาพ พินที่ใช้แสดงผลจะถูกใช้สำหรับการเรนเดอร์ภาพบนจอภาพ แต่พินที่ใช้สำหรับการจับภาพจะถูกใช้สำหรับการบันทึกข้อมูลวิดีโอเป็นไฟล์บนดิสก์ ความแตกต่างระหว่างพินที่ใช้แสดงผลและพินที่ใช้ในการจับภาพ

1. พินที่ใช้แสดงผล จับเฟรมที่ต้องการจาก พินที่ใช้ในการจับภาพ
2. แต่ละเฟรมจากพินที่ใช้ในการจับภาพจะใช้ช่วงเวลาของสายข้อมูลเมื่อเฟรมถูกจับ แต่พินที่ใช้สำหรับแสดงผล จะไม่ต้องใช้เวลาตรงนั้น

สาเหตุที่พินสำหรับแสดงผลไม่ต้องใช้ช่วงเวลา คือ การแสดงผล มี latecy เพียงเล็กน้อยเท่านั้น

Upstream WDM Filter

อุปกรณ์ที่ใช้ไดรเวอร์ตามโครงสร้าง WDM Driver อาจต้องการฟิลเตอร์เพิ่มเติมเพื่อควบคุมการทำงานของอุปกรณ์นั้น ประกอบด้วย

1. TV Tuner Filter สำหรับควบคุมการเลือกช่อง analog TV
2. TV Audio Filter สำหรับการตั้งค่า ออดิโอสำหรับ analog TV
3. Analog Video Crossbar Filter สำหรับการค้นช่องสัญญาณวิดีโอ และออดิโอผ่านอุปกรณ์ฮาร์ดแวร์ เช่น อุปกรณ์ที่มีหลายอินพุท (S-Video, Video Composite การ์ดวิดีโอที่มีช่องสัญญาณวิดีโอมากกว่าหนึ่งช่อง) Crossbar Filter ช่วยให้แอพพลิเคชั่นสามารถเลือกรับข้อมูลจากช่องสัญญาณเหล่านั้นได้

มีฟิลเตอร์หลายชนิดที่แสดงถึงอุปกรณ์ตัวเดียวกัน แต่เรียกใช้ฟังก์ชันที่ต่างกันในอุปกรณ์ตัวนั้น ในการที่พินมาเชื่อมต่อกันนั้นมิได้พิจารณาถึงชนิดของข้อมูลที่ไหลผ่านพินที่ต่อกัน แต่จะพิจารณาถึงค่า GUID ที่เรียกว่า mediums ค่า GUID นี้จะบอกถึง ไดรเวอร์ขนาดเล็ก(mini driver) ตัวอย่างเช่น TV Tuner Filter และ Video Capture Filter ต่างใช้ GUID ตัวเดียวกัน จึงทำให้สามารถสร้างกราฟได้อย่างถูกต้อง

2.9.5.4 การเลือกอุปกรณ์จับภาพ

ในการค้นหาอุปกรณ์จับภาพวิดีโอที่ติดตั้งอยู่ภายในเครื่องคอมพิวเตอร์ ไดเรคโชว์จะใช้อินเตอร์เฟซที่ชื่อว่า ICreateDevEnum ซึ่งอินเตอร์เฟซนี้จะประกอบด้วยฟังก์ชั่นที่คืนค่าเป็นรายการของอุปกรณ์ และรายละเอียดฟิลเตอร์ที่รองรับการทำงานของอุปกรณ์ตัวนั้น ขั้นตอนการค้นหาอุปกรณ์ด้วย ICreateDevEnum มีขั้นตอนดังนี้

1. สร้างอินสแตนของ ICreateDevEnum ด้วย CoCreateInstance
2. สร้างออบเจ็คที่แสดงรายการของอุปกรณ์โดยการเรียกใช้ฟังก์ชั่น CreateClass Enumerator โดยการใช้ CLSID ตามหมวดหมู่ที่ต้องการ ในกรณีของการค้นหาอุปกรณ์สำหรับจับภาพวิดีโอ จะใช้ CLSID_VideoInputDeviceCategory ซึ่งฟังก์ชั่นนี้จะคืนค่าเป็น IEnumMoniker ซึ่งเป็นอินเตอร์เฟซพอยน์เตอร์ใช้สำหรับการค้นหารายละเอียดของอุปกรณ์แต่ละตัว

```
IEnumMoniker *pEnum = NULL;
//Create the System Evice Enumerator.
HRESULT hr = CoCrateInstance(   CLSID_SystemDeviceEnum,
                                NULL,
                                CLSCTX_INPROC_SERVER,
                                IID_ICreateDevEnum,
                                Reinterpret<void**> (&pDevEnum)
                            );
```

```
if (SUCCEEDED(hr))
{
    //Create an enumerator for the video capture category.
    hr = pDevEnum->CrateClassEnumerator( CLSID_VideoInputDeviceCategory,
                                         &pEnum,
                                         0
                                         );
}
```

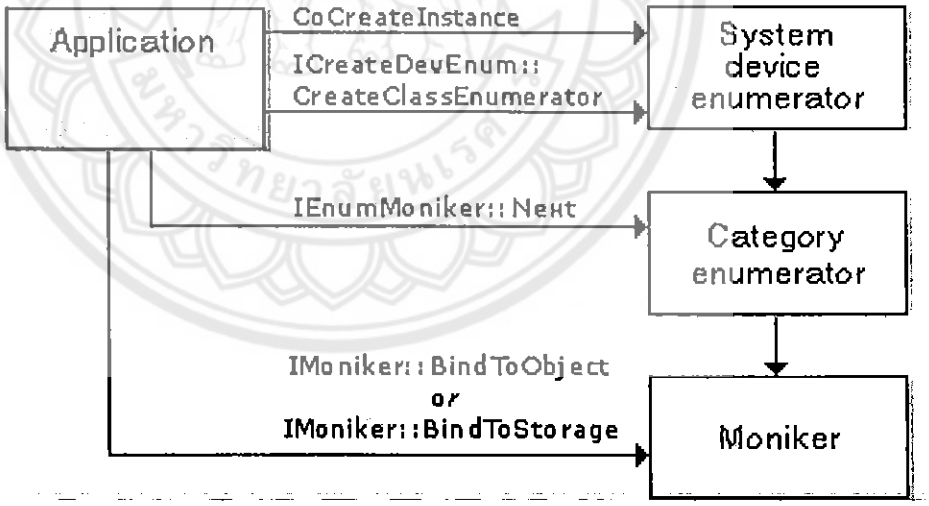
3. ใช้ฟังก์ชั่น Next ใน IEnumMoniker เพื่อดึงเอา IMoniker แต่ละตัวออกมา ซึ่ง IMoniker จะเป็น COM objetc ที่เก็บข้อมูลเกี่ยวกับออบเจ็คอื่น ซึ่งในที่นี้ก็คือ อุปกรณ์ที่ติดตั้งอยู่ในคอมพิวเตอร์แต่ละตัว เมื่อฟังก์ชั่น Next คืนค่าเป็น S_FALSE แสดงว่าได้แสดงอุปกรณ์ออกมาหมดแล้ว

4. ใช้ IMoniker ที่ได้จากขั้นตอนที่แล้วในการค้นหา ชื่อที่เรียกใช้ง่าย(Friendly Name), รายละเอียดของอุปกรณ์แต่ละตัว(Description), และตำแหน่งที่อยู่ของอุปกรณ์ที่ใช้แบ่งแยกอุปกรณ์ที่เป็นชนิดเดียวกัน(DevicePath) ด้วยฟังก์ชั่น BindToStorage ใน IMoniker ซึ่งฟังก์ชั่นนี้จะคืนค่าเป็น IPropertyBag ที่เป็นอินเตอร์เฟซสำหรับการอ่านค่ารายละเอียด ซึ่งเราจะได้ค่ารายละเอียดออกมาด้วยการใช้ฟังก์ชั่น Read ใน IPropertyBag ตัวอย่างคำสั่งในการอ่านค่ารายละเอียดของอุปกรณ์ขึ้นมาแสดงใน ลิสท์บอกซ์

```
HWND hList; // Handle to to list box
IMoniker *pMoniker = NULL;
while (pEnum->Next(1, &pMoniker, NULL) == S_OK)
{
    IPropertyBag *pPropBag;
    hr = pMoniker->BindToStorage(0, 0, IID_IPropertyBag, (void**) (&pPropBag));
    if(FAILED(hr))
    {
        pMoniker->Release();
        continue; // Skip this one, maybe the next one will work
    }
    // Find the description or friendly name.
    VARIANT varName;
    VariantInit(&varName);
    hr = pPropBag->Read("Description", &varName, 0);
    if(FAILED(hr))
    {
        hr = pPropBag->Read("FriendlyName", &varName, 0);
    }
    if(SUCCEEDED(hr))
    {
        USES_CONVERSION;
        (long)SendMessage( hList, LB_ADDSTRING,0,
                           (LPARAM)OLE2T(varName.bstrVal));
        VariantClear(&varName);
    }
    pPropBag->Release();
    pMoniker->Release();
}
}
```

5. ดึงค่าฟิลเตอร์ที่ใช้ในการจัดการกับอุปกรณ์ด้วยการเรียกใช้ฟังก์ชั่น BindToObject ใน IMoniker ใน IMoniker แล้วอาจจะทำการ เพิ่มฟิลเตอร์ตัวนี้เข้าสู่ฟิลเตอร์กราฟด้วยฟังก์ชั่น AddFilter ใน Filter Graph Manager ตัวอย่างคำสั่ง

```
IBaseFilter *pCap = NULL;
hr = pMoniker->BindToObjext(        0,
                                    0,
                                    IID_IBaseFilter,
                                    (void**)&pCap
                                );
if(SUCCEEDED(hr))
{
        hr = m_pGraph->AddFilter(pCap, "Capture Filter");
}
```



รูปที่ 2.17 ขั้นตอนการค้นหาอุปกรณ์จับภาพที่อยู่ในเครื่องคอมพิวเตอร์

2.9.5.5 การทำงานกับอุปกรณ์ Crossbar

ถ้าการ์ดจับภาพวิดีโอมีมากกว่าหนึ่งอินพุท หรือรองรับการทำงานที่ใช้อุปกรณ์หลายตัวสำหรับข้อมูลหนึ่งอัน แล้ว ฟิลเตอร์กราฟต้องใช้ Analog Video Crossbar Filter ในการควบคุมการทำงานของอุปกรณ์นั้น ICaptureGraphBuilder2 จะเพิ่มฟิลเตอร์ตัวนี้โดยอัตโนมัติ ถ้าเราต้องการ ซึ่งอาจจะรับข้อมูลต่อจาก Capture Filter ฟิลเตอร์กราฟอาจจะประกอบด้วย Crossbar Filter มากกว่า 1 ฟิลเตอร์ขึ้นอยู่กับอุปกรณ์ที่เราต้องการควบคุม

โครงสร้างของอุปกรณ์ที่มีลักษณะเป็น Crossbar จะเป็นเหมือนกล่องที่หลายอินพุทและหลายเอาท์พุท

รูปที่ 2.18โครงสร้างของอุปกรณ์ Crossbar

ในการใช้อุปกรณ์ ที่มีลักษณะเป็น Crossbar นั้น เราจะต้องทำการจับคู่อินพุทและเอาท์พุทเพื่อให้ข้อมูลจากอินพุทที่กำหนดไหลออกทางเอาท์พุทที่ต้องการ เช่น อุปกรณ์ Crossbar อาจจะประกอบด้วย ช่อง TV , วิดีโอ, วิทยุ ถ้าสมมติว่าช่องเอาท์พุทมีเพียงช่องเดียวเราก็ต้องเลือกว่าเราต้องการอะไร จะดู TV ดูวิดีโอ หรือจะฟังวิทยุ อินเตอร์เฟซที่ช่วยในการเชื่อมระหว่างอินพุทและเอาท์พุทที่ต้องการคือ IAMCrossbar ซึ่งประกอบด้วยฟังก์ชั่นที่สำคัญดังนี้

- CanRoute รับค่าอินพุทพินและเอาท์พุทพินของCrossbar Filter ที่ต้องการ และจะคืนค่าเป็น S_OK ถ้าอินพุทพินและเอาท์พุทพินนั้นสามารถที่เชื่อมกัน และจะคืนค่าเป็น S_FALSE ถ้าพินทั้งสองไม่สามารถเชื่อมกันได้
- Get_CrossbarPinInfo ฟังก์ชั่นนี้จะคืนค่าเป็นข้อมูลของพินที่เราส่งเข้าไปในฟังก์ชั่น
- Get_IsRoutedTo ฟังก์ชั่นนี้จะรับค่าเป็น เอาท์พุทพินและคืนค่าเป็นอินพุทพินที่กำลังต่อกับเอาท์พุทพินที่เราส่งเข้าไปในฟังก์ชั่น
- Get_PinCount ฟังก์ชั่นนี้จะคืนค่าเป็นจำนวนอินพุทพินและเอาท์พุทพินของอุปกรณ์ Crossbar
- Route จะทำหน้าที่เชื่อมอินพุทพินเข้ากับเอาท์พุทพินที่กำหนด

ในการค้นหา Crossbar Filter ที่อยู่ใน Capture Filter Graph ต้องใช้ฟังก์ชั่น FindInterface ใน ICaptureGraphBuilder2 ซึ่งมันจะทำการค้นหาฟิลเตอร์ที่รองรับการทำงานของ IAMCrossbar

ตัวอย่างรหัสคำสั่งที่ใช้ในการค้นหา ฟิลเตอร์ที่ต้องการ

```
IAMCrossbar *pXBar1 = NULL;
hr = pBuilder->FindInterface(&LOOK_UPSTREAM_ONLY, NULL,
                             pSrc, IID_IAMCrossbar, (void**)&pXBar1 );
if(SUCCEEDED(hr))
{
    //Found one crossbar. Get its IBaseFilter interface.
    IBaseFilter *pFilter = NULL;
    hr = pXBar1->QueryInterface( IID_IBaseFilter, (void**)&pFilter );
    if(SUCCEEDED(hr))
    {
    //Search upstream for another crossbar.
    IAMCrossbar *pXBar2 = NULL;
    hr = pBuilder->FindInterface( &LOOK_UPSTREAM_ONLY, NULL,
                                  pFilter, IID_IAMCrossbar, (void**)&pXBar2 );
    if(SUCCEEDED(hr))
    {
            /*...*/
            pXBar2->Release();
    }
}
pXBar1->Release();
}
```

เมื่อเราได้พอยน์เตอร์ของ IAMCrossbar มาแล้วเราสามารถดึงข้อมูลเกี่ยวกับ Crossbar Filter นั้นโดยใช้ฟังก์ชั่นภายใน IAMCrossbar

2.9.6 การประมวลผลภาพแต่ละเฟรมในไดเรคโชว์

สายข้อมูลวิดีโอประกอบด้วย ภาพหลายภาพมาต่อเนื่องกันตามลำดับเวลา การเข้าถึงภาพแต่ละภาพในสายข้อมูลเพื่อที่จะประมวลภาพในระดับพิกเซลล์นั้น ไดเรคโชว์ได้สร้างฟิลเตอร์เพื่อช่วยในการทำงานนี้ชื่อว่า Sample Grabber Filter ซึ่งเป็น Transform Filter ชนิดหนึ่งที่มี หนึ่งอินพุทพินและหนึ่งเอาท์พุทพิน ซึ่งเราสามารถเพิ่มฟิลเตอร์ตัวนี้เข้าไปโดยที่จะไม่มีผลกระทบใดๆต่อข้อมูลที่ไหลผ่านตัวมัน ซึ่งแอพพลิเคชั่นสามารถรับข้อมูลรูปภาพออกมาได้โดยผ่าน ISampleGrabber ซึ่งวิธีการที่จะดึงข้อมูลรูปภาพมาประมวลผลในแอพพลิเคชันมี 2 วิธีคือการอ่านข้อมูลจากบัฟเฟอร์ของ ISampleGrabber หรือ ทำการส่งฟังก์ชั่น callback เพื่อทำการประมวลผลเข้าไป ประกอบด้วยฟังก์ชั่นที่สำคัญดังนี้

- SetOneShot เป็นตัวที่ใช้กำหนดว่าเมื่อฟิลเตอร์จับภาพได้หนึ่งภาพแล้วจะสั่งให้ฟิลเตอร์กราฟหยุดทำงานหรือไม่ ถ้าเซตค่าอินพุทเป็นTRUEฟิลเตอร์กราฟจะหยุดทำงานหลังจากได้ภาพแล้ว
- SetMediaType กำหนดชนิดของข้อมูลที่จะเข้ามาทางอินพุทพินของ ISampleGrabber
- GetConnectedMediaType คืนค่าชนิดของข้อมูลมีเดียของอินพุทพิน
- SetBufferSamples ทำหน้าที่กำหนดว่าเมื่อมีข้อมูลมีเดียไหลผ่านตัวมันให้ทำการเก็บข้อมูลนั้นลงบัฟเฟอร์
- GetCurrentBuffer ฟังก์ชั่นนี้จะทำหน้าที่คืนสำเนา(Copy)ของข้อมูลในบัฟเฟอร์ที่เป็นข้อมูลล่าสุดที่จับได้โดยฟิลเตอร์
- SetCallback เป็นฟังก์ชั่นที่กำหนดว่า จะให้ ISampleGrabber เรียกใช้ฟังก์ชั่น callback ฟังก์ชั่นจากที่ใด

ขั้นตอนการใช้งาน ISampleGrabber

1. สร้างอินสแตนของ Sample Grabber Filter แล้วเพิ่มมันสู่ฟิลเตอร์กราฟ

```
IBaseFilter *pGrabberF = NULL;
hr = CoCreateInstance (CLSID_SampleGrabber, NULL, CLSCTX_INPROC_SERVER,
                       IID_IBaseFilter, (void**)&pGrabberF );
if(FAILED(hr)){ //Return an error. }
hr = pGraph->AddFilter( pGrabberF, "Sample Grabber");
if(FAILED(hr))
{ //Return an error. }
```

2. สร้าง ISampleGrabber จาก Sample Grabber Filter ที่เราสร้างขึ้น

```
ISampleGrabber *pGrabber;
pGrabber->QueryInterface (IID_ISampleGrabber, (void**)&pGrabber);
```

3. ทำการกำหนดค่าชนิดข้อมูลของ ISampleGrabber ที่เราสร้างขึ้นด้วยฟังก์ชั่น SetMediaType

```
AM_MEDIA_TYPE mt;
ZeroMemory(&mt, sizeof(AM_MEDIA_TYPE));
mt.majortype = MEDIATYPE_Video;
mt.subtype = MEDIASUBTYPE_RGB24;
hr = pGrabber->SetMediaType(&mt);
```

ตัวอย่างต่อไป จะแสดงการกำหนดค่าชนิดของมีเดียตามจำนวนบิตของ การ์ดแสดงผล

```
HDC hdc = GetDC(NULL);
int iBitDepth = GetDeviceCaps(hdc, BITSPIXEL);
Release(NULL, hdc);
mt.majortype = MEDIATYPE_Video;

switch (iBitDepth){
 case 8:
    mt.subtype = MEDIATYPE_Video; break;
case 16:
    mt.subtype = MEDIASUBTYPE_RGB555; break;
case 24:
    mt.subtype = MEDIATYPE_Video; break;
case 32:
    mt.subtype = MEDIASUBTYPE_RGB32; break;
default:
    return E_FAIL; }
hr = pGrabber->SetMediaType(&mt);
hr = pGrabber->SetMediaType(&mt);
```

4. สร้างฟิลเตอร์กราฟ แล้วทำการเชื่อม Source Filter เข้ากับ Sample Grabber Filter

```
IBaseFilter *pSrc;
hr = pGraph->AddSourceFilter(wszFileName, "Source"), &pSrc);
if(FAILED(hr))
{
    //Return an error code.
}
hr = ConnectFilters(pGraph, pSrc, pGrabberF);
```

5. สั่งให้กราฟเริ่มต้นทำงาน ซึ่งการทำงานของ Sample Grabber สามารถทำงานได้ 2 ทางคือ

   1. บัฟเฟอร์ Sample Grabber Filter จะทำสำเนาข้อมูลมีเดียก่อนที่ส่งไปยังฟิลเตอร์ตัวถัดไป

   2. โดยการใช้ callback ที่สร้างโดยแอพพลิเคชันส่งเข้าไปเพื่อจัดการกับภาพแต่ละภาพ

ตัวอย่างการใช้บัฟเฟอร์

```
hr = pGrabber->SetOneShot(TRUE);
hr = pGrabber->SetBufferSamples(TRUE);
pControl->Run(); // Run the graph.
pEvent->WaitForCompletion(INFINITE, &evCode); // Wait till it's done.
```

6. ดึงแต่ละภาพออกมาใช้งานในโหมดบัฟเฟอร์ เราจะใช้คำสั่ง GetCurrentBuffer ในการทำสำเนาข้อมูลในบัฟเฟอร์ออกมายังหน่วยความที่กำหนด

```
long cbBuffer = 0;
hr = pGrabber->GetCurrentBuffer(&cbBuffer, NULL);
char *pBuffer = new char[cbBuffer];
if (!pBuffer)
{
    // Out of memory. Return an error code.
}
hr = pGrabber->GetCurrentBuffer(&cbBuffer, (long*)pBuffer);
```

ตัวอย่างรหัสคำสั่ง เพื่อหารูปแบบของข้อมูลในบัฟเฟอร์

```
AM_MEDIA_TYPE mt;
hr = pGrabber->GetConnectedMediaType(&mt);
if (FAILED(hr))
{
        //Return error code.
}
//Examne the format block.
VIDEOFOHEADER *pVih;
if (    (mt.formattype == FORMAT_VideoInfo) &&
        (mt.cbFormat >= sizeof(VIDEOINFOHEADER)) &&
        (mt.pbFormat != NULL))
{
        pVih = (VIDEPINFOHEADER*)mt.pbFormat;
}
else
{
        //Wrong format. Free the format block and return an error.
        FreeMediaType(mt);
        return VFW_E_INVALIDMEDIATYPE;
}
// You can use the media type to access the BITMAPINFOHEADERE information.
// For example, the following code draws the bitmap using GAI:
SetDIBitsToDevice( hdc, 0, 0, pVih->bmiHeader.biWidth, pVih->bmiHeader.biHeigh,
                   0, 0, 0, pVih->bmiHeader.biHeigh, pBuffer,
                   (BITMAPINFO*)&pVih->bmiHeader, DIB_RGB_COLOR );
// Free the format block when you are done:
FreeMediaType(mt);
```

2.10 OpenCV Library

OpenCV คือ ไลบรารี่ที่เขียนด้วยภาษา C และ C++ อีกเล็กน้อย ประกอบด้วยโครงสร้างข้อมูล ฟังก์ชั่นและคลาสที่ทำงานทางด้านระบบการมองเห็นด้วยคอมพิวเตอร์ OpenCV เป็น Open Source ที่สามารถทำงานได้ทั้ง Windows Platform และ Linux Platform OpenCV พัฒนาโดยบริษัท อินเทล คอร์ปเปอเรชั่น (Intel Corporation) OpenCV เป็นไลบรารี่ที่ได้รับความนิยมในการทำงานทางด้านระบบการมองเห็นด้วยคอมพิวเตอร์ (Computer Vision) จึงมีกระดานข่าวสำหรับแลกเปลี่ยนความคิดและสอบถามปัญหาในการใช้งาน OpenCV ซึ่งอยู่ที่ http://groups.yahoo.com/group/OpenCV/ โดยก่อนที่จะเข้าใช้ได้ต้องทำการสมัครสมาชิกก่อน ซึ่งปัจจุบัน(2548) มีสมาชิกเกือบหนึ่งหมื่นคน

2.10.1 โครงสร้างข้อมูลใน OpenCV

2.10.1.1 IplImage

เป็นโครงสร้างข้อมูลพื้นฐานสำหรับเก็บข้อมูลทั่วไปของภาพและข้อมูลแต่ละพิกเซลล์ มีโครงสร้างดังนี้

```
typedef struct _IplImage
{
    int nSize;
    int ID;
    int nChannel;
    int alphaChannel;
    int Depth;
    char colorModel[4];
    char channelSeq[4];
    int dataOrder;
    int origin;
    int align;
    int width;
    int height;
    struct _IplROI *roi;
    struct _IplImage *maskROI;
```

```
    void *imageId;
    struct _IplTileInfo *tileInfo;
    int imageSize;
    char *imageData;
    int widthStep;
    int BorderMode[4];
    int BorderConst[4];
    char *imageDataOrigin;
} IplImage;
```

imageData

จะเป็นตัวแปรที่เก็บค่าพอยเตอร์ชี้ไปยังหน่วยความจำที่เก็บข้อมูลแต่ละไบต์ของภาพ

origin

จะเป็นข้อมูลที่บอกว่าภาพมีจุดเริ่มต้นที่มุมบนซ้าย (origin = 0) หรือมุมล่างซ้าย (origin = 1)

depth

ความลึกบิตของแต่ละชั้นสีในพิกเซลล์ ประกอบด้วย

- IPL_DEPTH_8U แต่ละชั้นสีมีขนาด 8 บิต
- IPL_DEPTH_16S แต่ละชั้นสีมีขนาด 16 บิต
- IPL_DEPTH_32S แต่ละชั้นสีมีขนาด 32 บิต
- IPL_DEPTH_32F แต่ละชั้นสีมีขนาด 32 บิต

widthStep

ขนาดของข้อมูลแต่ละแถวในรูปภาพในหน่วย Byte

width

ความกว้างของภาพในหน่วย พิกเซลล์

height

ความยาวของรูปภาพในหน่วย พิกเซลล์

nChannel

จำนวนชั้นของสีในรูปภาพ มีได้ตั้ง 1 – 4 ชั้น

nSize

ขนาดของโครงสร้างข้อมูลนี้

roi

เป็นตัวที่บอกว่าโครงสร้างข้อมูลนี้เก็บบางส่วนของภาพตามขนาดที่กำหนด

ImageSize

ขนาดของรูปภาพได้จาก width× height

ตัวอย่างการอ้างถึงแต่ละพิกเซลล์ในรูปภาพสามารถทำได้ดังนี้

```
for(i=0; i<image1->height; i+=10)
{
  for(j=(image1->widthStep)*i; j<(image1->widthStep)*(i+1);
      j+=image1->nChannels)
  {
   image1->imageData[j]   = (char)255;//red
   image1->imageData[j+1] = 0;//green
   image1->imageData[j+2] = 0;//blue
  }
}
```

2.10.1.2 CvPoint

ใช้สำหรับแสดงข้อมูลพิกัดในระนาบ 2 มิติ

```
typedef struct CvPoint
{
        int x;
        int y;
} CvPoint;
```

2.10.1.3CvPoint2D32f

ใช้สำหรับแสดงจุดใน 2 มิติ ด้วยจำนวนจริง

```
type struct CvPoint2D32f
{
    float  x;
    float  y;
} CvPoint2D32f
```

2.10.1.4 CvPoint3D32f

ใช้สำหรับจุดใน 3 มิติด้วยจำนวนจริง

```
typedef struct CvPoint3D32f
{
    float x;
    float y;
    float z;
} CvPoint3D32f;
```

2.10.1.5 CvSize

ใช้สำหรับแสดงขนาดของรูปสี่เหลี่ยม

```
typedef  struct CvSize
{
    int width;
    int height;
}  CvSize;
```

2.10.2 ฟังก์ชั่นพื้นฐานใน OpenCV

2.10.2.1 cvCreateImage

เป็นฟังก์ชั่นที่จะทำหน้าที่จองหน่วยความจำให้กับรูปภาพตามขนาดที่ต้องการ มีโครงสร้างดังนี้

```
IplImage* cvCreateImage( CvSize size, int depth, int channels );
```

size

ขนาดของรูปภาพที่ต้องการในหน่วยพิกเซลล์

depth

ความลึกบิตของแต่ละชั้นสี ประกอบด้วย

- IPL_DEPTH_8U – 8 บิตจำนวนเต็มไม่มีเครื่องหมาย
- IPL_DEPTH_8S - 8 บิตจำนวนเต็มมีเครื่องหมาย
- IPL_DEPTH_16S - 16 บิตจำนวนเต็มมีเครื่องหมาย
- IPL_DEPTH_32S - 32บิตจำนวนเต็มมีเครื่องหมาย
- IPL_DEPTH_32F - จำนวนทศนิยม
- IPL_DEPTH_64F - จำนวนจริง

channels

จำนวนชั้นของสีที่ต้องการ

2.10.2.2 cvReleaseImage

คืนค่าหน่วยความจำที่เก็บข้อมูลภาพ

```
void ReleaseImage( IplImage ** image);
```

image

ภาพที่ต้องการคืนหน่วยความจำ

2.10.2.3 cvFlip

สำหรับกลับด้านของรูปภาพ มีโครงสร้างดังนี้

```
void  cvFlip( const CvArr* A, CvArr* B=0, int flip_mode=0);
```

A

ภาพตั้งต้น

B

ภาพผลลัพธ์

flip_mode

วิธีในการกลับภาพ มีดังต่อไปนี้

B(i,j)=A(rows(A)-i-1,j) if flip_mode = 0

B(i,j)=A(i,cols(A)-j-1) if flip_mode > 0

B(i,j)=A(rows(A)-i-1,cols(A)-j-1) if flip_mode < 0

2.10.3 คลาสในการหาค่าพารามิเตอร์ของกล้อง

OpenCV ได้สร้างคลาสสำหรับทำการหาค่าพารามิเตอร์ของกล้อง และได้ประกาศไว้ในไฟล์ cvaux.h และ cvCalibFilter.cpp คลาสนี้มีชื่อว่า cvCalibFilter ซึ่งมีฟังก์ชั่นที่น่าสนใจดังนี้

2.10.3.1 SetEtalon

คือการกำหนดค่าเบื้องต้นสำหรับการใช้งานคลาสนี้

```
virtual bool SetEtalon( CvCalibEtalonType etalonType, double* etalonParams, int
                        pointCount = 0,  CvPoint2D32f* points = 0 );
```

etalonType

กำหนดชนิดของกระดานหมากฮอสที่ใช้ในตาราง

etalonParams

เป็นอาเรย์ที่มีสมาชิก 3 ตัว โดยที่

etalonParams[0] คือ ความกว้างของกระดานหมากฮอส(ช่อง)

etalonParams[1] คือ ความยาวของกระดานหมากฮอส(ช่อง)

etalonParams[2] คือ ความยาวของแต่ละช่องของกระดานหมากฮอสในหน่วย เซนติเมตร

pointCount และ points

ในโครงงานนี้ใช้ค่าปกติของมัน

CvCalibEtalonType

typedef enum CvCalibEtalonType

{

CV_CALIB_ETALON_USER = -1,

CV_CALIB_ETALON_CHESSBOARD = 0,

CV_CALIB_ETALON_CHECKERBOARD =CV_CALIB_ETALON_CHESSBOARD

}CvCalibEtalonType;

โดยในโครงงานนี้ค่านี้จะเท่ากับ CV_CALIB_ETALON_CHESSBOARD ตลอด

2.10.3.2 SetCameraCount

กำหนดจำนวนกล้องที่จะวัด

virtual void SetCameraCount(int cameraCount);

cameraCount

จำนวนกล้องที่จะวัด ในโครงงานนี้ cameraCount = 2

2.10.3.3 SetFrames

กำหนดจำนวนภาพที่จะใช้ในการวัดและเริ่มต้นทำการวัดค่าพารามิเตอร์ของกล้อง

virtual bool SetFrames(int totalFrames);

totalFrames

จำนวนภาพที่ใช้ในการวัด เมื่อคลาสได้รับภาพจากฟังก์ชัน push ครบตามจำนวนแล้ว จึงจะเริ่มทำการคำนวณเพื่อหาค่าพารามิเตอร์ของกล้อง ต้องมากกว่า 5 ภาพ

2.10.3.4 stop

หยุดทำการวัดค่าพารามิเตอร์ของกล้อง

virtual void Stop(bool calibrate = false);

2.10.3.5 IsCalibrated

ตรวจสอบว่าอ๊อบเจ็คนี้ได้ทำการคำนวณหาค่าพารามิเตอร์ของกล้องแล้วหรือไม่

```
bool IsCalibrated();
```

2.10.3.6 FindEtalon

ทำการตรวจสอบว่า อาเรย์ของภาพที่ส่งเข้าไป สามารถค้นหาจุดมุมของกระดานหมากฮอสได้ครบหรือไม่ สามารถแสดงจุดที่คลาสหาพบได้ โดยการใช้ฟังก์ชั่น DrawPoints วาดวงกลมรอบจุดที่พบ

```
virtual bool FindEtalon( IplImage** imgs );
```

imgs

อาเรย์ของภาพที่ได้จากกล้อง

2.10.3.7 Push

เก็บภาพไว้สำหรับการคำนวณหาค่าพารามิเตอร์ของกล้อง โดยภาพนี้ต้องสามารถหาจุดมุมของกระดานหมากฮอสได้ครบด้วย ซึ่งสามารถตรวจสอบโดยใช้ฟังก์ชั่น FindEtalon

```
virtual bool Push( const CvPoint2D32f** points = 0 );
```

2.10.3.8 DrawPoints

วาดวงกลมรอบจุดมุมในกระดานหมากฮอสที่พบ ถ้าพบจุดทั้งหมดจะแสดงผลลัพธ์ดังภาพที่ 2.15 มีโครงสร้างดังนี้

```
virtual void DrawPoints( IplImage** dst );
```

dst

อาเรย์ของภาพที่จะวาดวงกลมลงไป

รูปที่ 2.19 แสดงผลลัพธ์ของการใช้ฟังก์ชั่น DrawPoints

2.10.3.9 GetCameraParams

สำหรับดึงค่าของค่าพารามิเตอร์ของกล้องออกมาโดยข้อมูลที่ได้จะเก็บอยู่ในโครงสร้างข้อมูล

CvCamera virtual const CvCamera* GetCameraParams(int idx = 0) const;

idx

ลำดับกล้องที่ต้องการรู้ค่าพารามิเตอร์

2.10.3.10 CvCamera

โครงสร้างข้อมูลสำหรับเก็บค่าพารามิเตอร์ของกล้อง มีโครงสร้างดังนี้

```
typedef struct CvCamera
{
        float   imgSize[2]; /* size of the camera view, used during calibration */
        float   matrix[9]; /* intinsic camera parameters:  [ fx 0 cx; 0 fy cy; 0 0 1 ] */
        float   distortion[4]; /* distortion coefficients - two coefficients for
                        radial distortion  and another two for tangential: [ k1 k2 p1 p2 ] */
        float   rotMatr[9];
        float   transVect[3]; /* rotation matrix and transition vector relatively to some
                                reference point in the space. */
}CvCamera;
```

imgSize[2]

ขนาดความกว้างและความยาวของภาพในหน่วยพิกเซลล์

matrix[9]

เมทริกซ์การวัดของกล้อง

distortion[4]

ค่าสัมประสิทธิ์ความบิดเบี้ยวของเลนส์

rotMatr[9]

เมทริกซ์การหมุนของกล้อง

transVect[3]

เวคเตอร์การเลื่อนที่ของกล้อง

บทที่ 3
โครงสร้างและการใช้งานโปรแกรม

เนื่องจากในกระบวนการหาระยะลึกจากภาพ 2 มิติ ประกอบด้วยงานที่สำคัญ 3 อย่างคือ

1. การหาค่าพารามิเตอร์ของกล้อง
2. การหาจุดที่ตรงกันในภาพจากกล้องทั้งสอง
3. การคำนวณหาระยะลึก

ดังนั้นเพื่อให้การใช้งานได้ง่าย เราจึงแบ่งโปรแกรมออกเป็น 3 ส่วนคือ

1. Preview เป็นส่วนที่ใช้ในการทดสอบการจับภาพ และการหาจุดที่ตรงกันในภาพ
2. Calib เป็นส่วนที่จะทำการหาค่าพารามิเตอร์ต่างๆของกล้องแล้วเก็บค่าเหล่านั้นในรูปแบบของไฟล์เอกสาร
3. FindDepth เป็นส่วนที่ทำหน้าที่หาระยะลึกของภาพแบบเวลาจริง โดยการใช้ค่าพารามิเตอร์ของกล้องจากไฟล์ที่ได้จากส่วนของ Calib

3.1 การเลือกกล้อง

เนื่องจากในโครงงานนี้เราได้ใช้การ์ดวิดีโอที่มี 4 ช่อง ในโปรแกรมจึงมีให้เลือกว่า กล้องด้านซ้ายและด้านขวาจะรับมาจากช่องสัญญาณใด

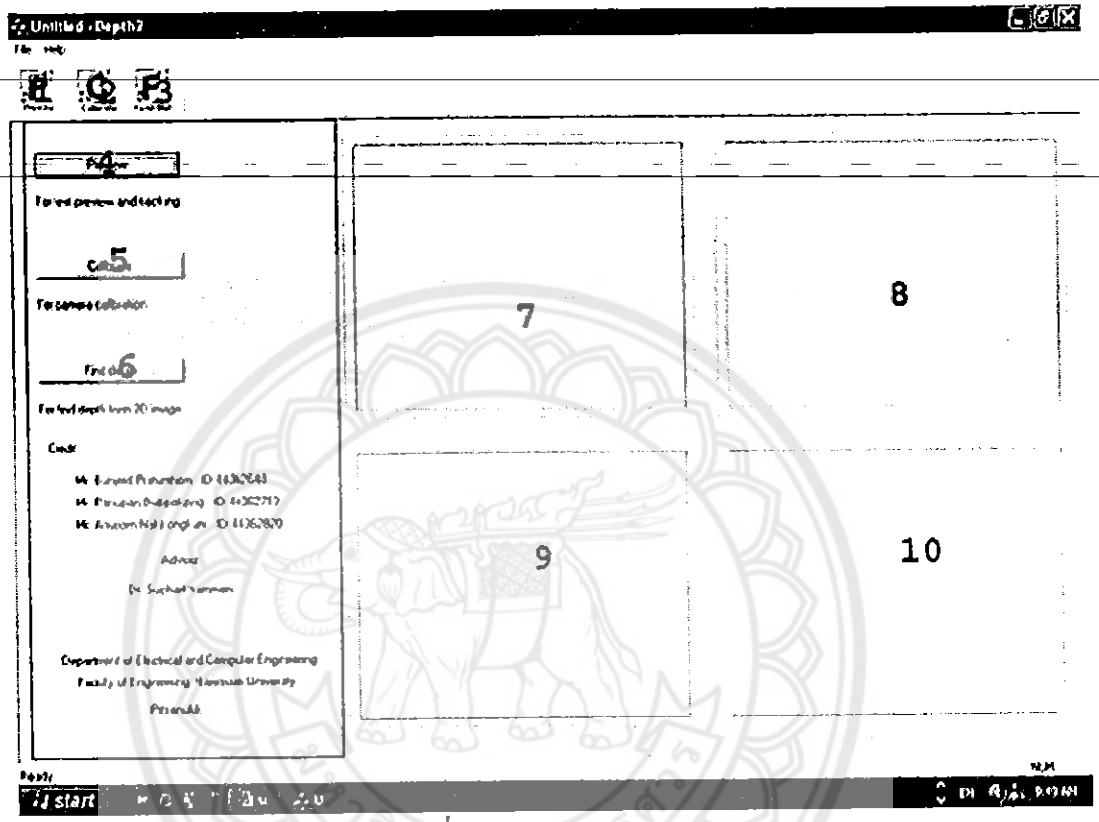
3.2 การหาจุดเป้าหมายในภาพ

ในการทำโครงงานนี้เราได้ใช้จุดมุมของตาราง ดังแสดงในภาพ เป็นจุดเป้าหมายที่เราต้องการหาระยะลึก โดยเราได้ดัดแปลงจากฟังก์ชั่น cvFindChessBoardCornerGuesses ใน OpenCV แล้วนำค่าที่ได้ส่งเข้าฟังก์ชั่นสำหรับหาระยะลึกต่อไป

รูปที่ 3.1 การหาจุดที่ตรงกันในภาพ

3.3 การใช้งานโปรแกรม

3.3.1 เริ่มต้นการทำงาน



รูปที่ 3.2 หน้าแรกของโปรแกรม

หมายเลข 1 สำหรับเข้าสู่การทำงานในส่วนของ Preview

หมายเลข 2 สำหรับเข้าสู่การทำงานในส่วนของ Calib

หมายเลข 3 สำหรับเข้าสู่การทำงานในส่วนของ FindDepth

หมายเลข 4 สำหรับเข้าสู่การทำงานในส่วนของ Preview

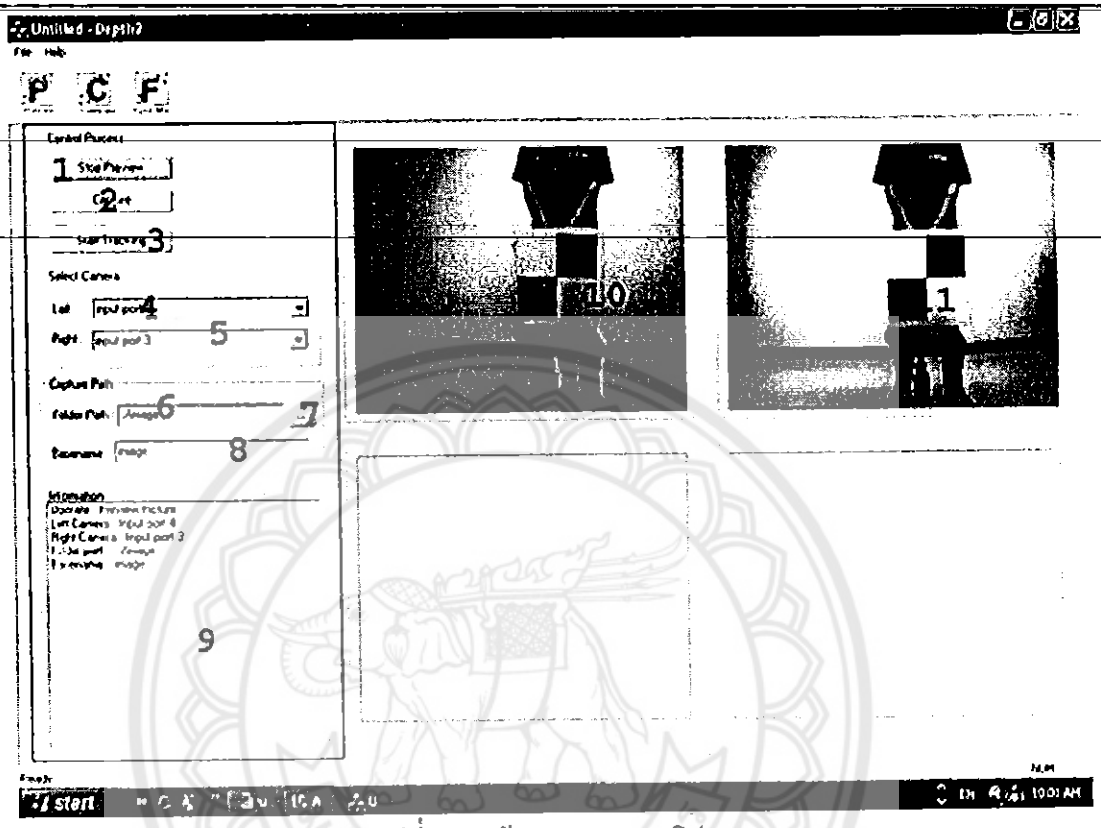
หมายเลข 5 สำหรับเข้าสู่การทำงานในส่วนของ Calib

หมายเลข 6 สำหรับเข้าสู่การทำงานในส่วนของ FindDepth

หมายเลข 7,8 เป็นฉากสำหรับแสดงภาพจากกล้องด้านซ้ายและขวา ตามลำดับ

หมายเลข 9,10 เป็นฉากสำหรับแสดงภาพที่จะจับจากกล้องด้านซ้ายและด้านขวา ตามลำดับ สำหรับบันทึกเป็นไฟล์

3.3.2 การทำงานในส่วนของ Preview



รูปที่ 3.3 หน้า Preview ของโปรแกรม

หมายเลข 1 สำหรับเปิดและปิดการแสดงภาพ

หมายเลข 2 สำหรับบันทึกรูปภาพที่แสดงอยู่เป็นไฟล์ภาพบิตแมบ ตามโฟลเดอร์ในหมายเลข 6 และตามชื่อฐานในหมายเลข 8

หมายเลข 3 สำหรับแสดงการตรวจหาจุดที่ตรงกันใน 2 ภาพ คือจุดมุมตรงกลางของตาราง

หมายเลข 4 สำหรับเลือกกล้องด้านซ้าย

หมายเลข 5 สำหรับเลือกกล้องด้านขวา

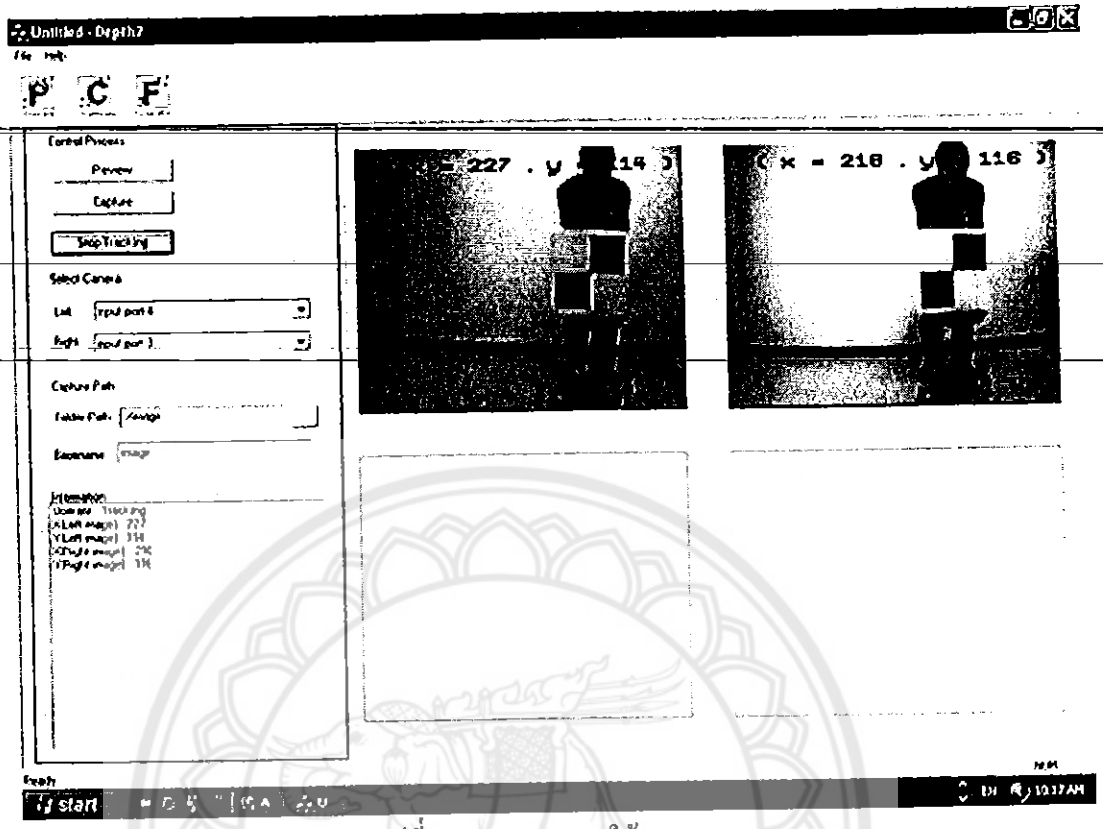
หมายเลข 6 แสดงที่อยู่ของโฟลเดอร์สำหรับการบันทึกภาพ

หมายเลข 7 แสดงรายการโฟลเดอร์ให้เลือก

หมายเลข 8 แสดงชื่อฐานสำหรับบันทึกภาพที่จับได้

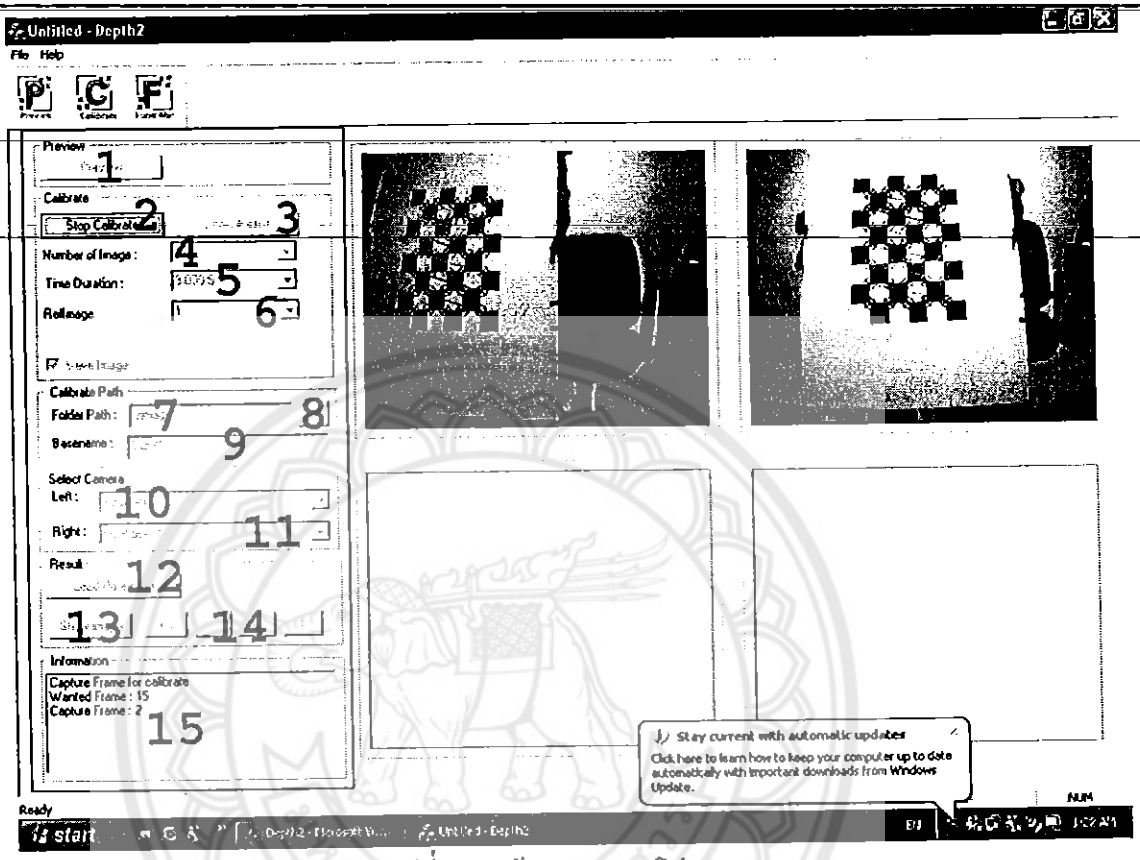
หมายเลข 9 แสดงข้อมูลทั่วไป

หมายเลข 10-11 ภาพจากกล้องด้านซ้ายและด้านขวาตามลำดับ



รูปที่ 3.4 แสดงการหาใช้งาน Tracking

3.3.3 การใช้งานในส่วน Calib



รูปที่ 3.5 หน้า Calib ของโปรแกรม

หมายเลข 1 สำหรับเปิดและปิดการแสดงภาพจากกล้อง(ทดสอบการทำงานของกล้อง)

หมายเลข 2 สำหรับเริ่มต้นและจบการหาค่าพารามิเตอร์ของกล้อง

หมายเลข 3 สำหรับแสดงค่าพารามิเตอร์ของกล้องที่วัดได้ครั้งล่าสุด

หมายเลข 4 สำหรับเลือกจำนวนภาพที่จะใช้ในการวัด

หมายเลข 5 สำหรับเลือกคาบเวลาในเก็บภาพของ CalibFilter

หมายเลข 6 สำหรับเลือกว่าจะใช้ภาพใดเป็นภาพอ้างอิง

หมายเลข 7 แสดงที่อยู่ของโฟลเดอร์ที่จะเก็บภาพและไฟล์ที่เก็บค่าพารามิเตอร์ของกล้อง

หมายเลข 8 จะแสดงรายการโฟลเดอร์ให้เลือก

หมายเลข 9 จะแสดงชื่อฐานสำหรับการบันทึกภาพ

หมายเลข 10 สำหรับเลือกกล้องด้านซ้าย

หมายเลข 11 สำหรับเลือกกล้องด้านขวา

หมายเลข 12 สำหรับอ่านข้อมูลไฟล์ที่เก็บค่าพารามิเตอร์ของกล้องตามโฟลเดอร์และชื่อฐานที่เราเลือกไว้

หมายเลข 13 แสดงรูปภาพที่อยู่ในโฟลเดอร์ที่เราเลือกตามชื่อฐานที่เรากำหนด

หมายเลข 14 สำหรับเลื่อนดูภาพ

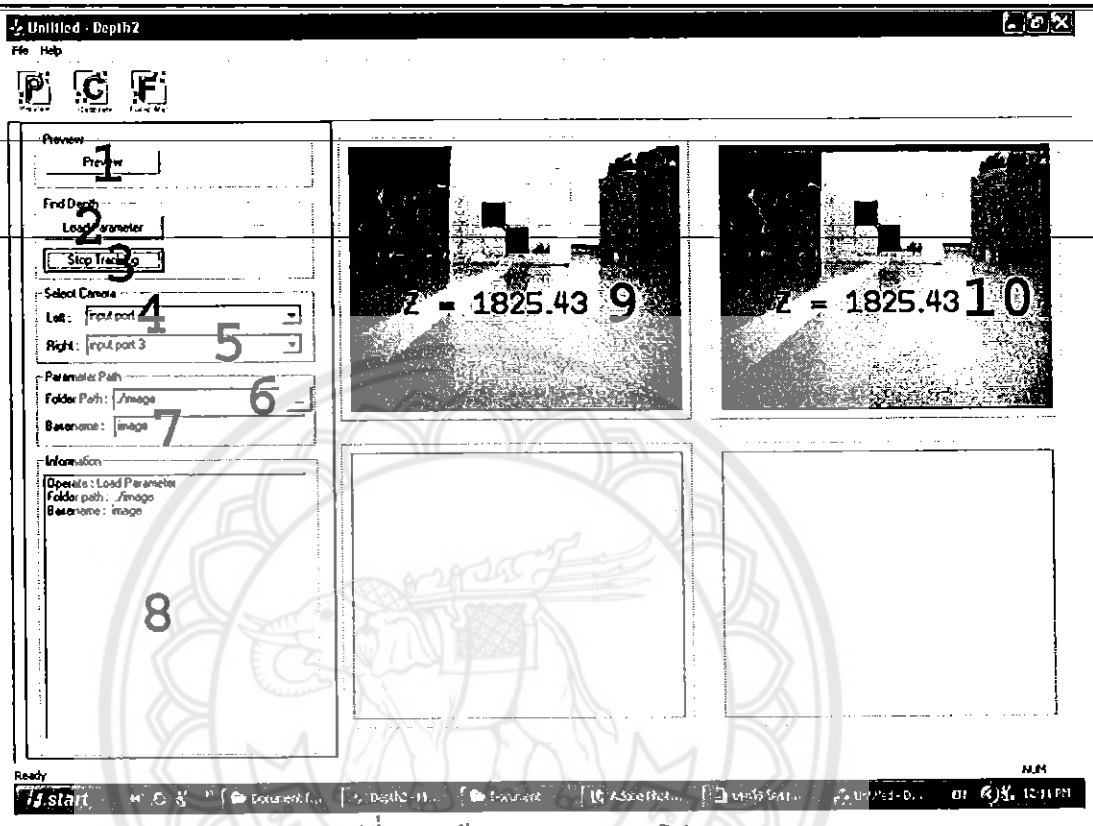
หมายเลข 15แสดงข้อมูลต่างๆ

หมายเลข 16 – 17 แสดงภาพที่ใช้ในการวัด

ขั้นตอนการหาค่าพารามิเตอร์ของกล้อง

1. เลือกจำนวนภาพที่จะใช้ในการวัด(หมายเลข 4)
2. เลือกคาบเวลาที่ต้องการ(หมายเลข 5)
3. เลือกว่าจะใช้ภาพใดเป็นภาพอ้างอิง(หมายเลข 6)
4. เลือกว่าจะบันทึกภาพที่จับได้หรือไม่
5. เริ่มต้นทำการวัดโดยการกดที่ปุ่มหมายเลข 2
6. ทำการเคลื่อนที่ภาพกระดานหมากฮอสต่อหน้ากล้อง ถ้าโปรแกรมจับภาพได้ จะแสดงเป็นภาพกระพริบ สังเกตุว่าต้องการอีกกี่ภาพ ที่หมายเลข 19
7. เมื่อได้ภาพครบตามจำนวน หมายเลข 19 จะแสดงข้อความ On process calibrate
8. รอจนมีแมสเสจบ็อกแสดงข้อความว่า Calibrate complete
9. กดที่ปุ่มหมายเลข 2 เพื่อหยุดการหาค่าพารามิเตอร์ของกล้อง
10. ถ้าต้องการดูผลลัพธ์ของการหาค่าพารามิเตอร์ ให้กดที่ปุ่มหมายเลข 3
11. ถ้าต้องการดูภาพที่จับได้ให้กดที่ปุ่มหมายเลข 17 และเลื่อนดูภาพที่กลุ่มของปุ่มหมายเลข 18

3.3.4 การใช้งานส่วน FindDepth



รูปที่ 3.6 หน้า FindDepth ของโปรแกรม

หมายเลข 1 สำหรับเปิดและปิดการแสดงภาพ เพื่อการทดสอบการทำงานของกล้อง

หมายเลข 2 สำหรับอ่านข้อมูลค่าพารามิเตอร์ของกล้องเพื่อใช้ในการหาระยะลึก ตามโฟลเดอร์และชื่อฐานที่เลือก

หมายเลข 3 สำหรับเริ่มและจบการหาระยะลึก

หมายเลข 4 สำหรับเลือกกล้องด้านซ้าย

หมายเลข 5 สำหรับเลือกกล้องด้านขวา

หมายเลข 6 แสดงโฟลเดอร์ที่เป็นที่เก็บไฟล์ที่เก็บค่าพารามิเตอร์ของกล้องที่จะใช้ในการหาค่าระยะลึก

หมายเลข 7 ชื่อฐานของไฟล์ที่เก็บค่าพารามิเตอร์ของกล้อง

หมายเลข 8 แสดงพิกัดของจุดเป้าหมายของภาพทางด้านและด้านขวา

หมายเลข 9 - 10 แสดงค่าระยะลึกที่โปรแกรมหาได้ ตามค่าตัวแปรที่กำหนดให้

ขั้นตอนในการหาระยะลึก

1. เลือกโฟลเดอร์และกำหนดชื่อฐานสำหรับไฟล์ที่เก็บค่าพารามิเตอร์ของกล้อง
2. กดที่ปุ่มหมายเลข 2 เพื่ออ่านข้อมูลค่าพารามิเตอร์ของกล้องจากไฟล์ที่กำหนด
3. เริ่มต้นทำการหารระยะลึกโดยการกดที่ปุ่มหมายเลข 3
4. ถ้าต้องการหยุดทำการหาระยะลึก ให้กดที่ปุ่มหมายเลข 3 อีกครั้ง
5. ระยะลึกจะแสดงตามหมายเลข 9

3.4 คลาสต่างๆภายในโปรแกรม

3.4.1 CVidCapture

เป็นคลาสที่ทำหน้าที่ติดต่อและควบคุมการทำงานในส่วนของการจับภาพจากกล้องดิจิตอล ประกอบด้วยฟังก์ชั่นและตัวแปรดังนี้ ดังนี้

ฟังก์ชั่น

- Init() สำหรับการกำหนดค่าในการเริ่มต้นทำงาน
- Connect() สำหรับติดต่อกับการ์ดวีดีโอ
- Disconnect() สำหรับยกเลิกการติดต่อกับการ์ดวีดีโอ
- SampleCB (double sampleTimeSec, IMediaSample* mediaSample) จะเป็นฟังก์ชั่นที่มีการเรียกทุกครั้งเมื่อมีการจับภาพได้ ใช้สำหรับการประมวลผลภาพแต่ละเฟรม และจะมีการเรียก callback function เพื่อการประมวลผลภาพแต่ละเฟรม
- StartImageCap() สำหรับเริ่มต้นการจับภาพอย่างต่อเนื่อง
- Stop() สำหรับหยุดการจับภาพอย่างต่อเนื่อง

ตัวแปร

- CVVIDCAP_CALLBACK fCaptureCallback คือตัวแปรที่เป็น callback function ที่จะมีการเรียกใช้ใน ฟังก์ชั่น SampleCB()
- CALLBACKSTRUCT CallBackData คือ โครงสร้างข้อมูลที่เก็บข้อมูลที่จะเป็นอินพุทของ fCaptureCallback

3.4.2 callback function

โครงสร้างของ callback function

```
typedef bool (*CVVIDCAP_CALLBACK)(CALLBACKSTRUCT *CBdata);
```

3.4.3 CALLBACKSTRUCT

โครงสร้างข้อมูล สำหรับส่งเป็นอินพุทของ callback function

```
struct CALLBACKSTRUCT
{
        IplImage* LeftImage; //ภาพที่ได้จากกล้องด้านซ้าย
        IplImage* RightImage;//ภาพที่ได้จากกล้องตัวขวา
        PREVIEWSTRUCT PreviewData; //ใช้ในการแสดงผลออกทางหน้าจอ
        CCalibCamera CalibrateData;//อ๊อบเจ็คที่ใช้ในการหาค่าพารามิเตอร์ของ
กล้อง
        int TimeDuration;//คาบเวลาในการประมวลผล
        BOOL bSaveImg;//เลือกว่าจะมีการบันทึกภาพหรือไม่
        //finddepth
        CCalcDepth CalcDepthData;//อ๊อบเจ็คที่ใช้ในการหาระยะลึก
};

struct PREVIEWSTRUCT
{
        int LeftID;//ค่า ID ของ static ที่ใช้ในการแสดงภาพทางซ้าย
        int RightID;//ค่า ID ของ static ที่ใช้ในการแสดงภาพทางขวา
        CWnd* Parent;//เจ้าของ static ที่ใช้ในการแสดงผล
};
```

3.4.4 CCalibCamera

เป็นคลาสที่ทำหน้าที่ในการหาค่าพารามิเตอร์ของกล้อง ประกอบด้วยตัวแปรและฟังก์ชั่นดังนี้

ฟังก์ชั่น

- Init() สำหรับกำหนดค่าเริ่มต้นในการทำงาน
- CheckChessboardImage(IplImage **images) สำหรับตรวจสอบภาพกระดานหมากฮอสว่าสามารถพบจุดมุมครบทุกจุดหรือไม่
- DrawPoint(IplImage** images) สำหรับวาดจุดลงบนภาพตามจุดมุมของกระดานหมากฮอส
- PushImage(IplImage** images) สำหรับเก็บภาพเพื่อการคำนวณหาค่าพารามิเตอร์ของกล้อง
- IsCalibrated() สำหรับตรวจสอบว่าอ๊อบเจ็คได้ทำการคำนวณหาค่าพารามิเตอร์ของกล้องแล้วหรือไม่
- SaveParam() สำหรับบันทึกค่าพารามิเตอร์ของกล้องเป็นไฟล์เอกสาร
- LoadParam() สำหรับอ่านค่าพารามิเตอร์ของกล้องจากไฟล์เอกสาร

ตัวแปร

- char path[256] ข้อความที่แสดงที่อยู่ที่ใช้ในการบันทึกและอ่านข้อมูลค่าพารามิเตอร์ของกล้องจากๆฟล์เอกสาร
- int LongPerSquare ความยาวของกระดานหมากฮอสแต่ละช่อง
- int nColumn จำนวนคอลัมน์ของกระดานหมากฮอส
- int nRow จำนวนแถวของกระดานหมากฮอส
- int nRefImage ลำดับของภาพอ้างอิง
- int nFrame จำนวนภาพที่จะใช้ในการคำนวณหาค่าพารามิเตอร์ของกล้อง

3.4.4 CCalcDepth

คลาสสำหรับคำนวณหาระยะลึกจากภาพ ประกอบด้วย ฟังก์ชั่นและตัวแปรดังนี้

ฟังก์ชั่น

- Calculate(int leftx , int lefty ,int rightx, int righty) คำนวณหาระยะลึก โดยที่รับอินพุทเป็นจุดที่ตรงกันของภาพทางด้านซ้ายและด้านขวา

ตัวแปร

- CCalibCamera CameraParameter เป็นตัวแปรที่เก็บค่าตัวแปรที่จะใช้ในการคำนวณหาระยะลึก

บทที่ 4

ผลการทดลอง

ในการทำโครงงานได้แบ่งการทดลองออกเป็น 3 ส่วนคือ

1. หาจุดที่ตรงกันจากภาพที่ได้จากกล้องดิจิตอล
2. หาค่าพารามิเตอร์ของกล้อง
3. หาระยะลึก

4.1 การหาจุดที่ตรงกันในภาพ

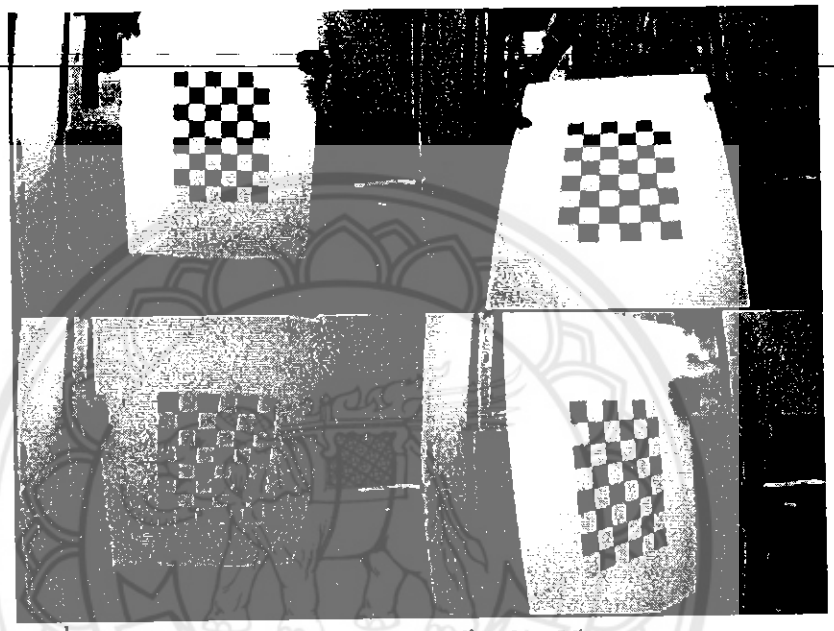
จากการทดลองหาจุดมุมของตารางดังแสดงในรูปที่ 4.1 พบว่าโปรแกรมสามารถหาได้เป็นอย่างดี แต่พบปัญหาคือ

- ไม่สามารถหาจุดมุมได้ในระยะทางที่น้อยกว่า 40 เซนติเมตร ดังนั้นการทดสอบระยะลึกจึงเริ่มต้นที่ 40 เซนติเมตร
- ต้องทำการปรับเปลี่ยนขนาดของตารางให้เข้ากับระยะทางมิฉะนั้นจะไม่สามารถหาจุดมุมได้

รูปที่ 4.1 การทดลองหาจุดมุมของตาราง

4.2 การหาค่าพารามิเตอร์ของกล้อง

เพื่อให้ได้ค่าที่มีความเที่ยงตรง จึงได้มีการทดลองหาค่าพารามิเตอร์ของกล้องหลายๆครั้งแล้วนำค่าที่ได้มาหาค่าเฉลี่ย ตารางที่ 4.1 – ตารางที่ 4.6 แสดงค่าพารามิเตอร์ที่ได้จากการหาพารามิเตอร์ของกล้องตามวิธีของ Zhengyou Zhang ดังแสดงในรูปที่ 4.2



รูปที่ 4.2 แสดงการหาค่าคุณสมบัติของกล้อง โดยวิธีของ Zhengyou Zhang

ตารางที่ 4.1 อัตราส่วนของระยะโฟกัสต่อความยาว และอัตราส่วนระยะโฟกัสต่อความกว้างของกล้องด้านซ้าย

การทดลองครั้งที่	ผลการทดลอง	
	f_u	f_v
1	333.44003	339.43646
2	331.91600	338.67943
3	332.44057	339.39023
4	320.72382	327.81330
5	345.72427	351.91798
6	331.34588	338.57859
7	374.49523	384.09675
8	335.98609	342.32706
9	330.78971	337.79718
10	329.83438	336.67635
11	328.56743	335.10259
12	336.03075	341.96716
13	338.01161	345.18080
14	341.41661	351.51748
15	328.44989	335.67472
16	332.10537	339.55194
17	350.50553	359.07111
18	327.18416	333.97295
19	322.26959	330.49749
20	340.93549	346.42526
ค่าเฉลี่ย	335.6086	342.7837

ตารางที่ 4.2 อัตราส่วนของระยะโฟกัสต่อความยาว และอัตราส่วนระยะโฟกัสต่อความกว้างของกล้องด้านขวา

การทดลองครั้งที่	ผลการทดลอง	
	f_u	f_v
1	339.57027	346.72602
2	331.92393	338.83532
3	335.28887	342.78334
4	332.48961	339.85247
5	349.87052	357.09924
6	332.77227	340.00945
7	343.45523	348.79429
8	338.61114	346.06918
9	330.66305	336.36412
10	325.36318	332.17835
11	327.55214	334.11442
12	329.15871	336.50494
13	337.98571	344.71548
14	338.19097	347.15433
15	335.47501	342.80729
16	331.37793	338.62465
17	346.82995	356.93733
18	339.80482	347.40784
19	313.71092	323.30956
20	326.52677	334.27758
ค่าเฉลี่ย	334.3311	341.7283

ตารางที่ 4.3 จุดสำคัญของกล้องด้านซ้าย

การทดลองครั้งที่	ผลการทดลอง	
	u_0	v_0
1	189.28828	111.49827
2	188.44225	112.95948
3	183.69101	113.56991
4	188.05278	117.44486
5	190.44618	114.52373
6	188.97049	108.21316
7	213.58930	117.27325
8	185.84812	112.94944
9	171.34233	110.95534
10	183.50654	111.49808
11	181.85069	125.49762
12	193.45990	107.91610
13	195.57583	114.14443
14	192.95410	116.38108
15	183.59048	114.14962
16	194.88811	115.47480
17	187.62527	120.53859
18	180.56031	116.46743
19	177.49478	110.84482
20	176.07526	106.66982
ค่าเฉลี่ย	187.3626	113.9485

ตารางที่ 4.4 จุดสำคัญของกล้องด้านขวา

การทดลองครั้งที่	ผลการทดลอง	
	u_0	v_0
1	163.72734	127.77007
2	161.29569	123.94338
3	160.05209	133.03858
4	160.89031	128.81866
5	173.20581	126.59597
6	163.61101	127.01021
7	171.71733	113.41074
8	167.93217	129.13640
9	152.93921	123.18782
10	162.82978	124.89195
11	155.51519	126.45672
12	173.52874	128.09874
13	168.43198	127.73591
14	161.66939	129.58937
15	159.01290	127.82112
16	166.59780	131.52486
17	179.83797	135.36602
18	158.53570	127.69122
19	138.24789	128.16402
20	162.60659	139.15225
ค่าเฉลี่ย	163.1092	127.9702

ตารางที่ 4.5 สัมประสิทธิ์ความบิดเบี้ยวของเลนส์ของกล้องด้านซ้าย

การทดลองครั้งที่	ผลการทดลอง			
	k_1	k_2	p_1	p_2
1	-0.41375	0.17186	0.00095	-0.00389
2	-0.39879	0.18602	-0.00050	-0.00199
3	-0.37578	0.13656	-0.00037	0.00182
4	-0.42383	0.19838	-0.00204	-0.00589
5	-0.39063	0.12344	-0.00137	-0.00024
6	-0.40206	0.17879	0.00122	-0.00271
7	-0.35020	-0.05964	-0.00115	-0.00918
8	-0.36121	-0.05954	-0.00153	0.00181
9	-0.43516	0.26162	0.00043	0.00095
10	-0.34264	-0.14484	-0.00091	0.00193
11	-0.34072	0.10831	-0.00543	0.00686
12	-0.35535	0.09296	0.00380	0.00105
13	-0.42843	0.20193	-0.00100	-0.00791
14	-0.56256	0.64063	-0.00156	-0.01542
15	-0.37988	0.14124	0.00049	-0.00011
16	-0.43237	0.25123	-0.00111	-0.00725
17	-0.39986	0.10097	0.00331	-0.00253
18	-0.38133	0.13333	0.00094	0.00113
19	-0.42466	0.13387	-0.00216	-0.00220
20	-0.44328	0.31899	0.00621	-0.00048
ค่าเฉลี่ย	-0.40211	0.1558	-0.000089	-0.0022

ตารางที่ 4.6 สัมประสิทธิ์ความบิดเบี้ยวของเลนส์ของกล้องด้านขวา

การทดลองครั้งที่	ผลการทดลอง			
	k_1	k_2	p_1	p_2
1	-0.41786	0.23128	0.00165	-0.00180
2	-0.45025	0.41636	-0.00285	-0.00038
3	-0.40210	0.17646	0.00073	-0.00131
4	-0.42080	0.22787	0.00160	-0.00035
5	-0.41608	0.23139	-0.00270	-0.00444
6	-0.44721	0.44771	-0.00073	-0.00140
7	-0.39028	0.36623	-0.00189	0.00329
8	-0.39696	0.24802	-0.00186	-0.00417
9	-0.42374	0.43988	-0.00407	0.00770
10	-0.38527	0.19002	-0.00130	0.00106
11	-0.37262	0.04201	-0.00040	0.00338
12	-0.38170	-0.10064	-0.00124	-0.00651
13	-0.45221	0.34297	-0.00332	-0.00374
14	-0.42741	0.23484	-0.00180	-0.00058
15	-0.41612	0.20900	-0.00078	-0.00067
16	-0.43553	0.37591	-0.00109	-0.00241
17	-0.47183	0.44227	0.00250	-0.01074
18	-0.43593	0.33477	0.00140	-0.00308
19	-0.52430	0.60142	-0.01608	0.02124
20	-0.38170	-0.02780	-0.01057	0.00092
ค่าเฉลี่ย	-0.4225	0.2715	-0.0021	-0.000199

4.3 การหาระยะลึก

ระยะลึก จะหาจากระยะทางระหว่าง จุดมุมของตาราง ดังแสดงในภาพที่ 4.1 และกล้องดิจิตอล

ตารางที่ 4.7 ผลทดลองการหาระยะลึก

ระยะทาง(mm)	ค่าที่วัดได้(mm)	ความผิดพลาด(%)
400	406	1.5000
450	456	1.3333
500	504	0.8000
550	555	0.9091
600	598	0.3333
650	649	0.1538
700	697	0.4286
750	745	0.6667
800	793	0.8750
850	847	0.3529
900	889	1.2222
950	948	0.2105
1000	986	1.4000
1050	1027	2.1905
1100	1093	0.6364
1150	1146	0.3478
1200	1180	1.6667
1250	1247	0.2400
1300	1296	0.3077
1350	1347	0.2222
1400	1384	1.1429
1450	1431	1.3103
1500	1480	1.3333
1550	1541	0.5806

ตารางที่ 4.7(ต่อ) ผลทดลองการหาระยะลึก

ระยะทาง(mm)	ค่าที่วัดได้(mm)	ความผิดพลาด(%)
1600	1576	1.5000
1650	1649	0.0606
1700	1688	0.7059
1750	1757	0.4000
1800	1759	2.2778
1850	1816	1.8378
1900	1864	1.8947
1950	1923	1.3846
2000	1965	1.7500
2050	2064	0.6829
2100	2101	0.0476
2150	2147	0.1395
2200	2212	0.5455
2250	2264	0.6222
2300	2280	0.8696
2350	2381	1.3191
2400	2383	0.7083
2450	2460	0.4082
2500	2520	0.8000
2550	2551	0.0392
2600	2662	2.3846
2650	2764	4.3019
2700	2849	5.5185
2750	2850	3.6364
2800	2909	3.8929
2850	2909	2.0702
2900	3057	5.4138

ตารางที่ 4.7(ต่อ) ผลทดลองการหาระยะลึก

ระยะทาง(mm)	ค่าที่วัดได้(mm)	ความผิดพลาด(%)
2950	3207	8.7119
3000	3210	7.0000

ความผิดพลาดเฉลี่ย ในระยะ 400 – 2550 มิลลิเมตร 0.87 %

ความผิดพลาดเฉลี่ย ในระยะ 2600 – 3000 มิลลิเมตร 4.77 %

บทที่ 5
สรุป

5.1 สรุป

เป้าหมายของโครงงานนี้คือการหาวิธีและเขียนโปรแกรมเพื่อการการหาระยะลึกจากภาพจากกล้องดิจิตอล ซึ่งในโครงงานได้สร้างคลาสใน Visual C++ 3 คลาสคือคลาส CVidCapture เพื่ออ่านภาพจากกล้องดิจิตอล CCalibCamera เพื่อหาค่าพารามิเตอร์ของกล้องดิจิตอล และ CCalcDepth เพื่อคำนวณหาระยะลึก นอกจากนี้ยังสร้างฟังชั่น TrackCorner เพื่อการค้นหาจุดมุมของกระดานหมากฮอสเพื่อใช้ค้นหาจุดที่ตรงกันในภาพจากกล้องทั้งสอง จากการหาค่าพารามิเตอร์ของกล้องและใช้ค่าคุณสมบัตินั้นในการคำนวณหาระยะลึก พบว่าในช่วงระยะ 40 – 255 เซนติเมตร ค่าที่ได้มีความคลาดเคลื่อนโดยเฉลี่ย 0.87 % ระยะ 260 - 300 เซนติเมตร ค่าความคลาดเคลื่อนโดยเฉลี่ย 4.77 % ซึ่ง

5.2 ข้อเสนอแนะ

- ในการพัฒนาต่อควรจะมีการพัฒนาวิธีการในการหาจุดที่ตรงกันจากกล้องทั้งสองให้ดีขึ้นเพื่อให้สามารถตรวจหาวัตถุต่างๆได้

- สามารถใช้คลาส CCalibCamera และ CCalDepth ในโปรแกรมทางด้านระบบการมองเห็นด้วยคอมพิวเตอร์โปรแกรมอื่นได้ เนื่องจากมีการเขียนให้เป็นอิสระจากตัวโปรแกรมหลัก

เอกสารอ้างอิง

[1] David A.Forsyth and Jean Ponce. Computer Vision A MODERN APPROACH. New Jersey : Pearson Education International. 2003.

[2] Zhengyou Zhang. "A Flexible New Technique for Camera Calibration" [Online]. Available: http://research.microsoft.com/~zhang/Papers/TR98-71.pdf

[3] Chris Yang. "Three-dimension Reconstruction from Stereo Extrinsic and Camera Intrinsic Properties" [Online]. Available: http://member.rogers.com/yanger/project.html.

[4] นิรุธ อำนวยศิลป์. คู่มือการเขียนโปรแกรม Microsoft VisualC++ Version6.กรุงเทพมหานคร: บริษัท ส.เอเชียเพรส(1989)จำกัด. 2542.

[5] Intel Corperation. "OpenCV" [Online]. Available:http://www.sourceforge.net/project/opencv-library/opencvman_old.pdf.

ภาคผนวก

ภาคผนวก ก
การติดตั้งไดเรคเอ็กซ์

ขั้นตอนการติดตั้ง Direct SDK ประกอบด้วยขั้นตอนดังนี้

1. ดาวน์โหลด DirectX SDK (ในการทำโครงใช้ DirectX SDK 9.0) จากเวบไซต์ www.microsoft.com แล้ว ค้นหาคำว่า directx sdk

รูปที่ 1 การค้นหา directx sdk

2. เลือกลิงค์ที่ต้องการ

รูปที่ 2 เลือกลิงค์ไปยัง DirectX SDK ที่ต้องการ

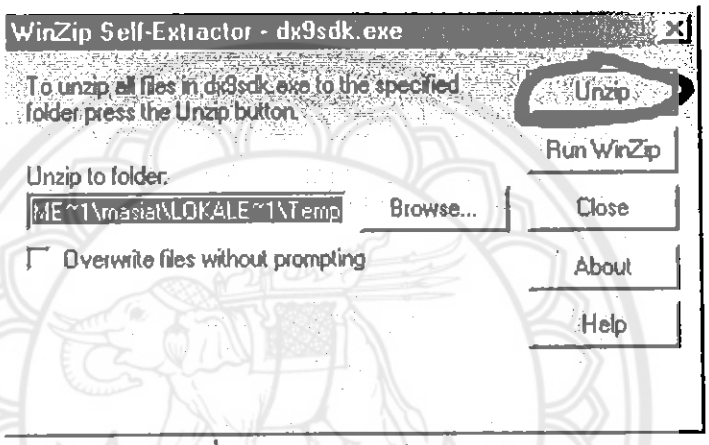
3. เลือกดาวน์โหลด

รูปที่ 3 เลือกดาวน์โหลด เพื่อดาวน์โหลด DirectX SDK

4. ผลลัพธ์ที่ได้จากการดาวน์โหลด

รูปที่4 ผลลัพธ์ที่ได้จากดาวน์โหลด DirectX SDK

5. ทำการขยายไฟล์ที่ดาวน์โหลดมา

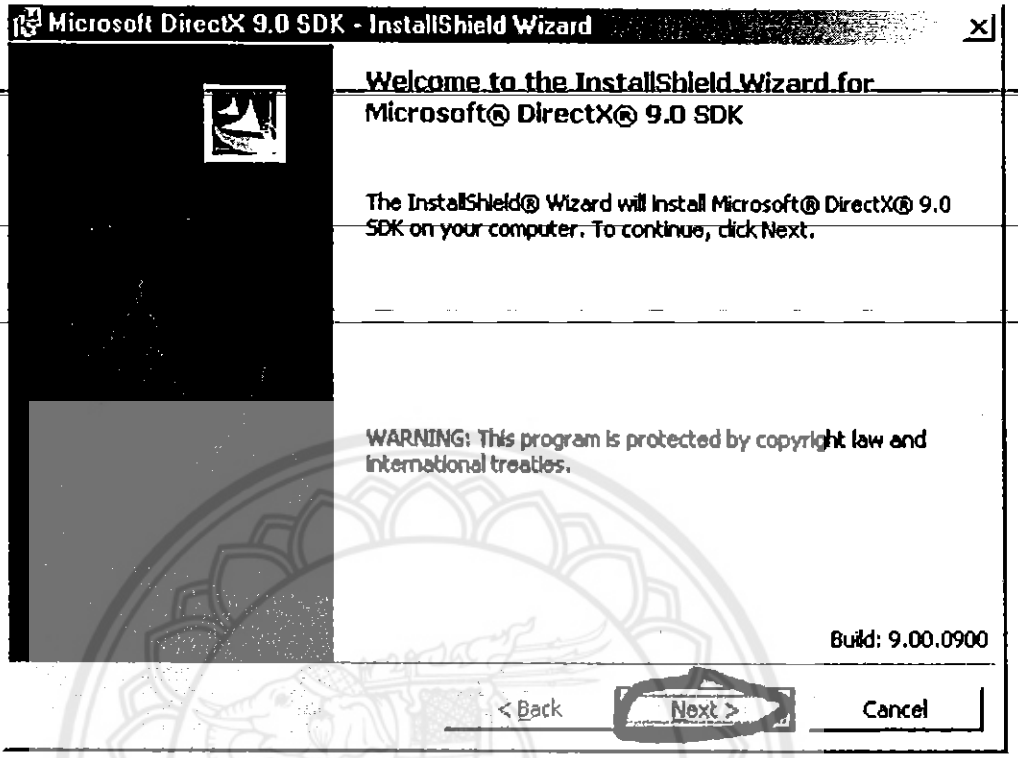


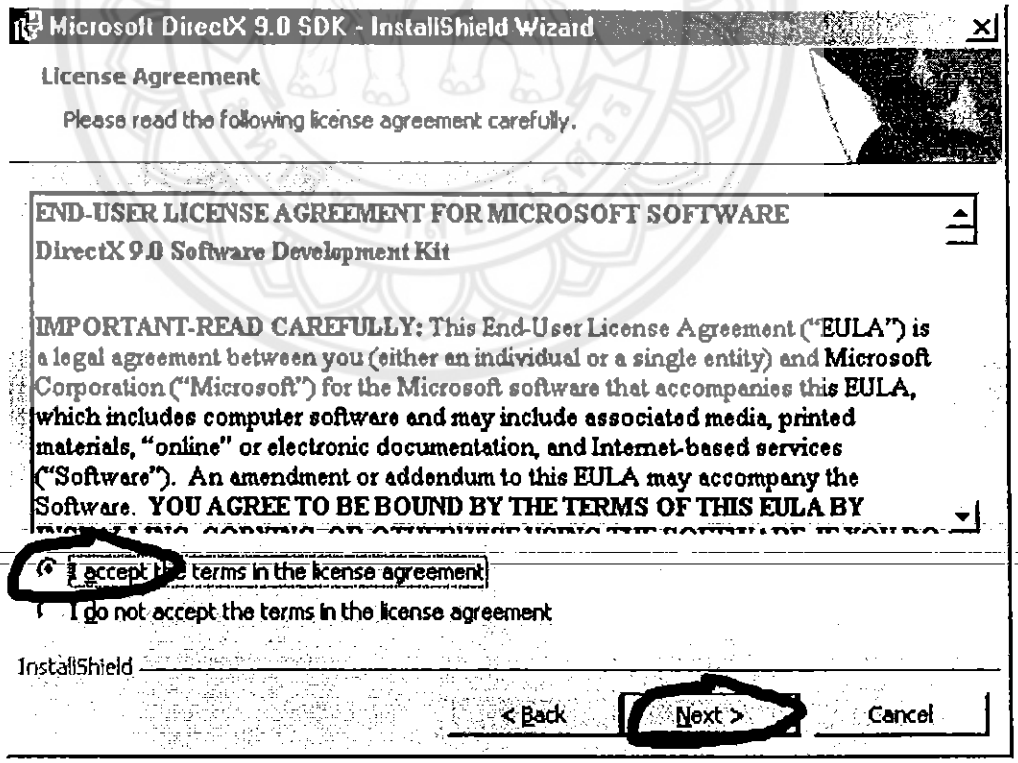
รูปที่5 การขยายไฟล์ dx9sdk.exe

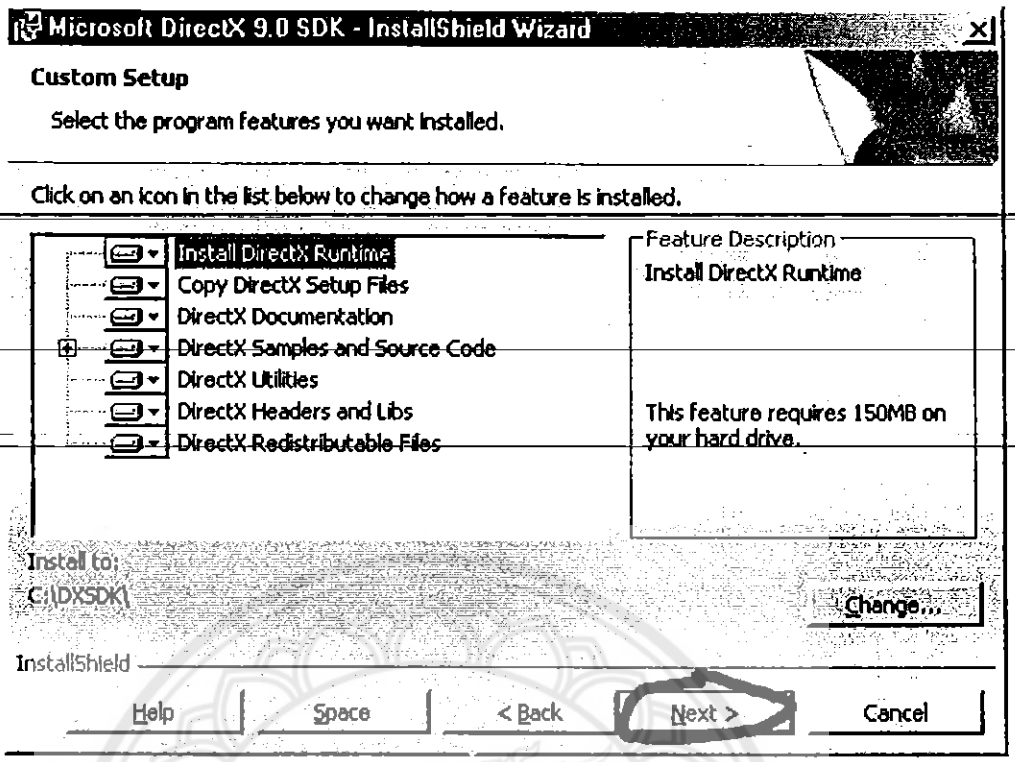
6. เข้าไปในโฟลเดอร์ที่ได้จากการขยาย แล้วคลิกที่ไฟล์ setup.exe

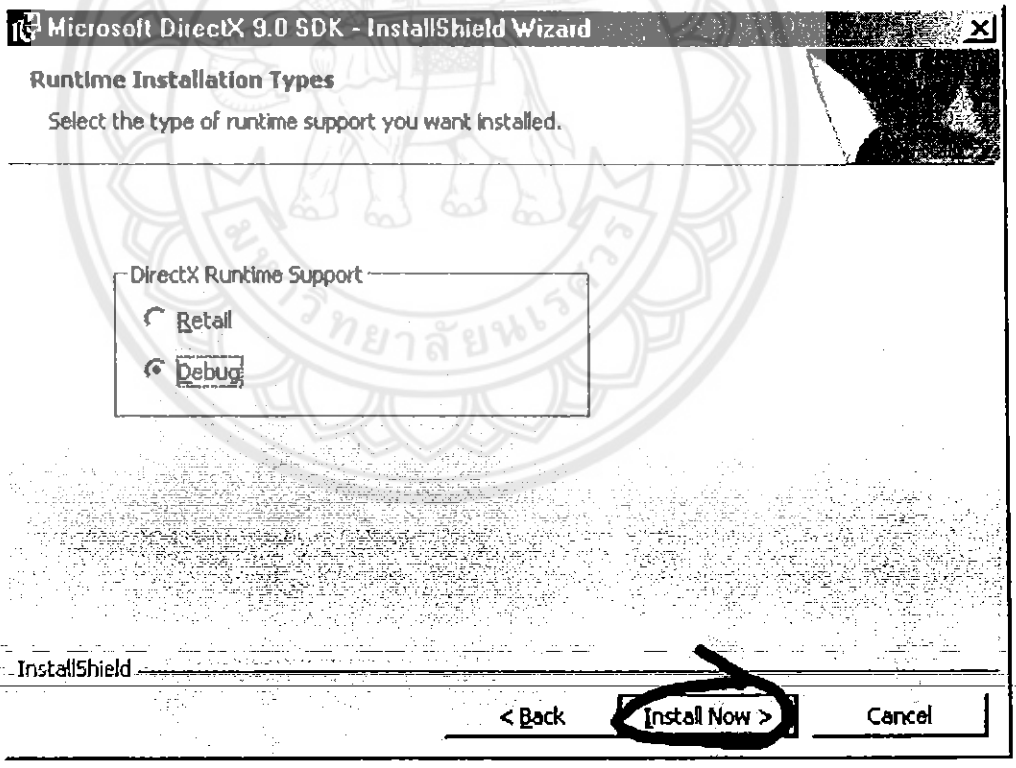
รูปที่ 6 ไฟล์ setup.exe ที่ได้จากการขยายไฟล์ dx9sdk.exe

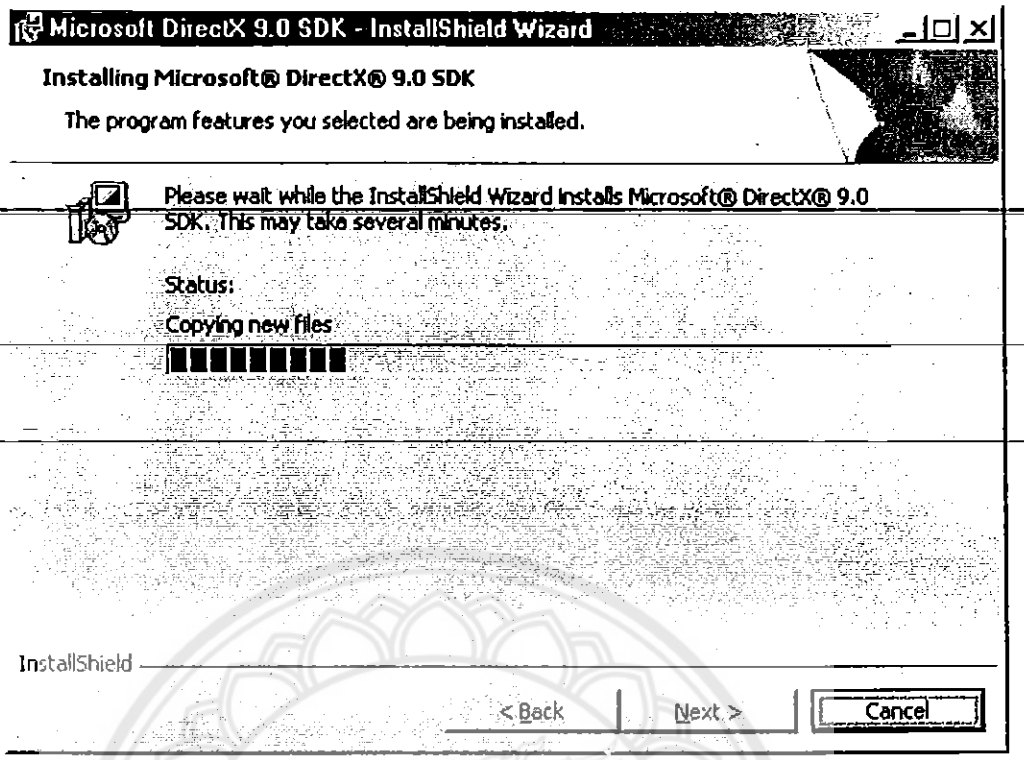
7. ทำการติดตั้งตามขั้นตอน

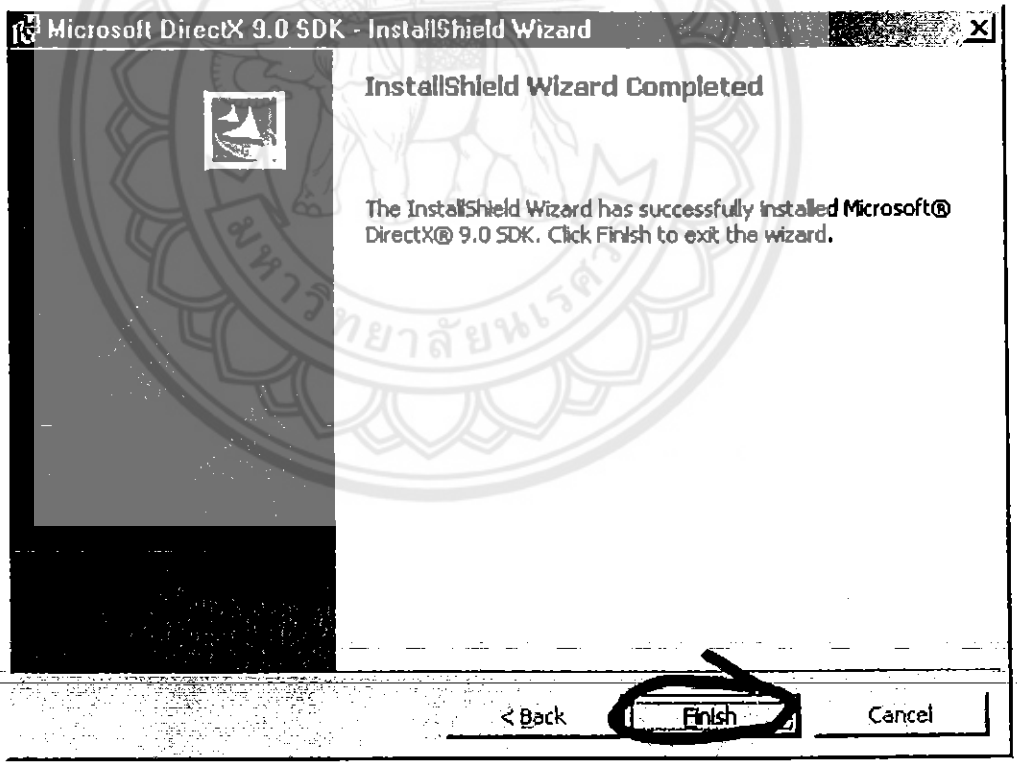


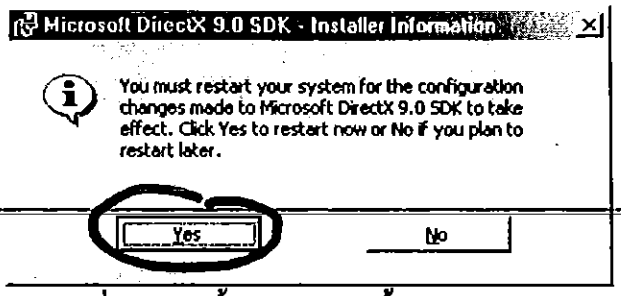



Microsoft DirectX 9.0 SDK - InstallShield Wizard
Custom Setup
Select the program features you want installed.
Click on an icon in the list below to change how a feature is installed.
Install DirectX Runtime
Copy DirectX Setup Files
DirectX Documentation
DirectX Samples and Source Code
DirectX Utilities
DirectX Headers and Libs
DirectX Redistributable Files
Feature Description
Install DirectX Runtime
This feature requires 150MB on your hard drive.
Install to:
C:\DXSDK\
Change...
InstallShield
Help
Space
< Back
Next >
Cancel


Microsoft DirectX 9.0 SDK - InstallShield Wizard
Runtime Installation Types
Select the type of runtime support you want installed.
DirectX Runtime Support
Retail
Debug
InstallShield
< Back
Install Now >
Cancel


Microsoft DirectX 9.0 SDK - InstallShield Wizard
Installing Microsoft® DirectX® 9.0 SDK
The program features you selected are being installed.
Please wait while the InstallShield Wizard installs Microsoft® DirectX® 9.0 SDK. This may take several minutes.
Status:
Copying new files
InstallShield
< Back
Next >
Cancel

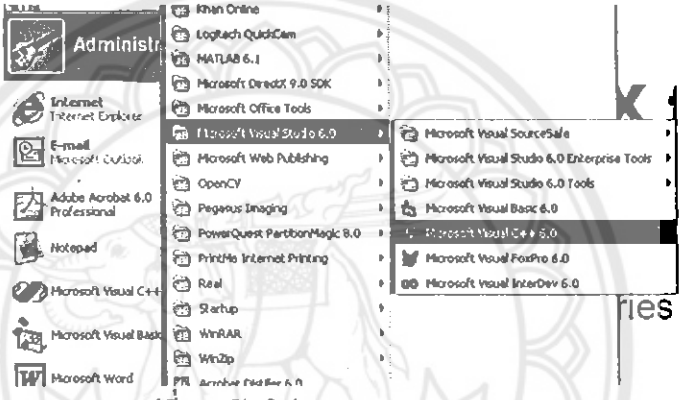

Microsoft DirectX 9.0 SDK - InstallShield Wizard
InstallShield Wizard Completed
The InstallShield Wizard has successfully installed Microsoft® DirectX® 9.0 SDK. Click Finish to exit the wizard.
< Back
Finish
Cancel



รูปที่ 7 แสดงขั้นตอนการติดตั้ง DirectX SDK

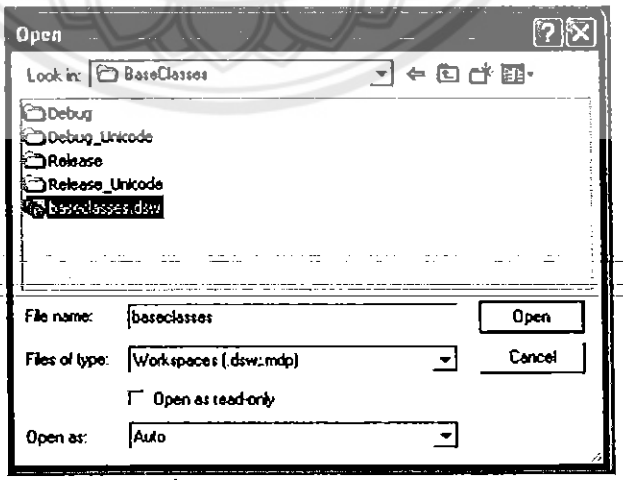
8. ทำการคอมไพล์ BaseClass เพื่อการใช้งานดังนี้

8.1 เปิดโปรแกรม Visual C++ 6.0



รูปที่ 8 เปิดโปรแกรม Visual C++ 6.0

8.2 เปิดไฟล์ BaseClasses.dsw จากโฟลเดอร์ C:\DXSDK\Samples\C++\DirectShow\BaseClass



รูปที่ 9 เปิดไฟล์ baseclasses.dsw

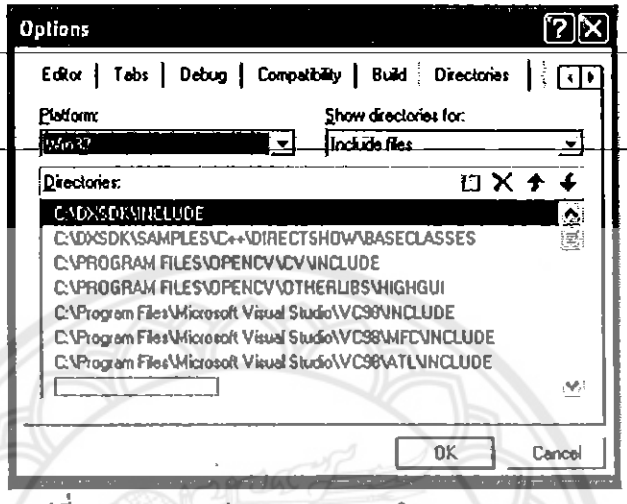
8.2 ทำการ Build ทั้ง Debug mode และ Release mode

9. เซตค่าการใช้งาน Visual C++ ให้สามารถใช้งาน Direct SDK ได้

9.1 ไปที่เมนู Tool->Options->Directories-> Include Files แล้วเพิ่มพาร์ทดังต่อไปนี้

C:\DXSDK\INCLUDE

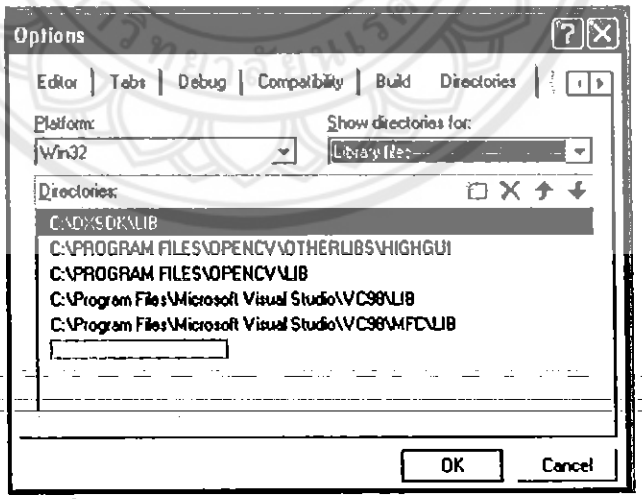
C:\DXSDK\SAMPLES\C++\DIRECTSHOW\BASECLASSESES



รูปที่ 10 การเซตค่า Include files ใน Visual C++ 6.0

9.2 ไปที่ 9.1 ไปที่เมนู Tool->Options->Directories-> Library Files แล้วเพิ่มพาร์ท ดังต่อไปนี้

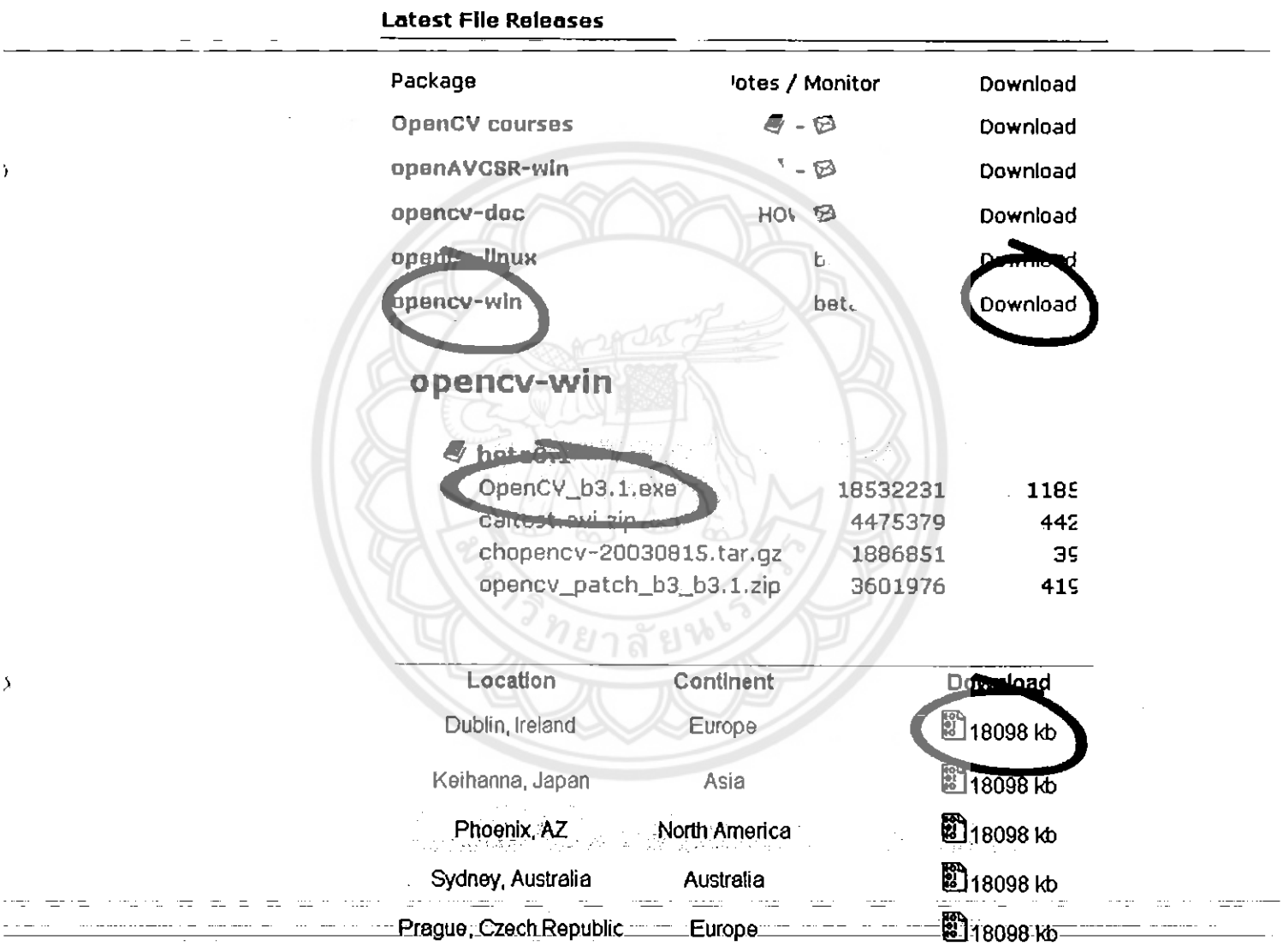
C:\DXSDK\LIB



รูปที่ 11 การเซตค่า Library files ใน Visual C++ 6.0

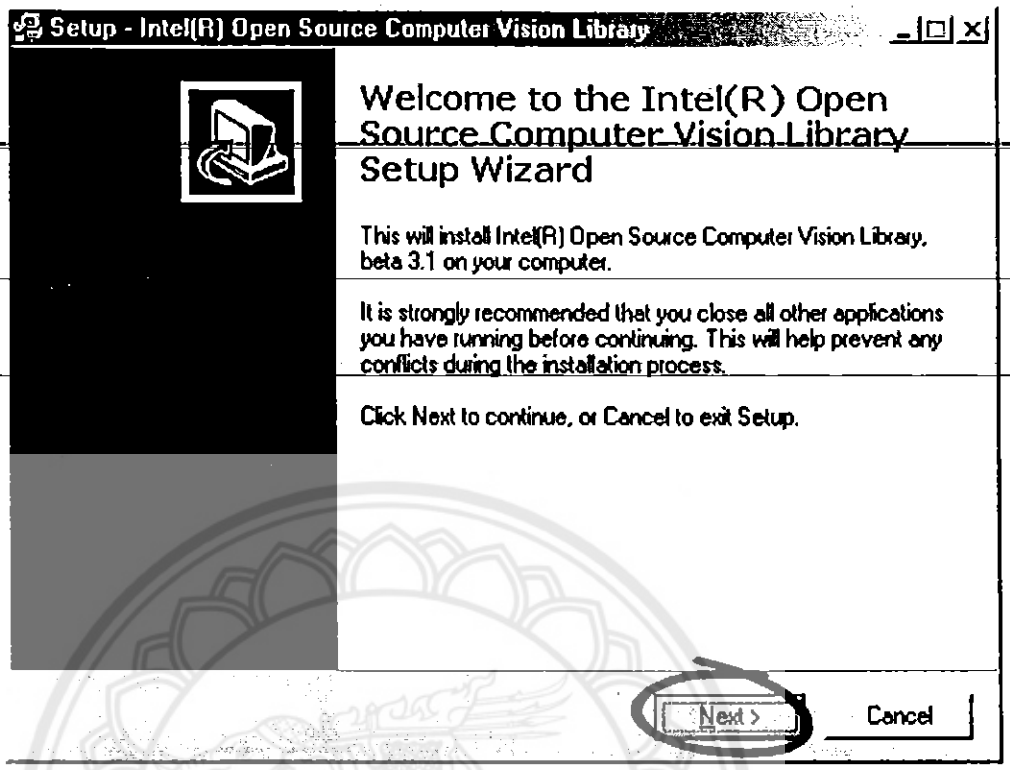
ภาคผนวก ข
การติดตั้งและการใช้งาน OpenCV

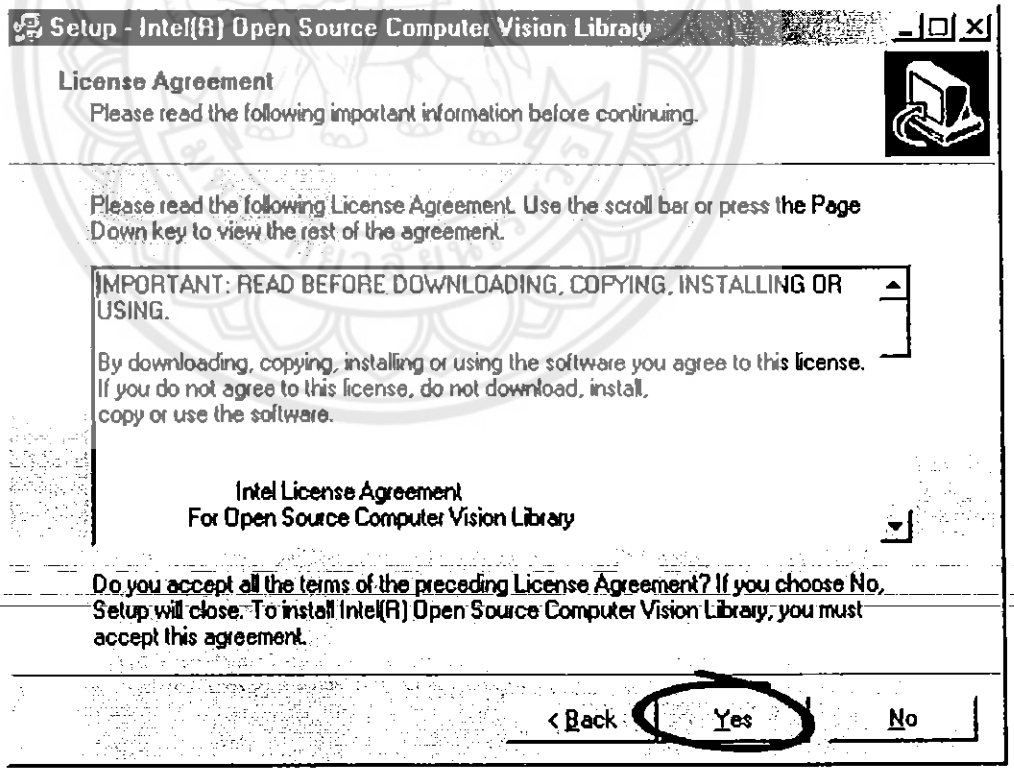
1. ดาวน์โหลด OpenCV ที่ เวบไซต์ http://www.sourceforge.net/projects/opencvlibrary ตามขั้นตอนดังรูป

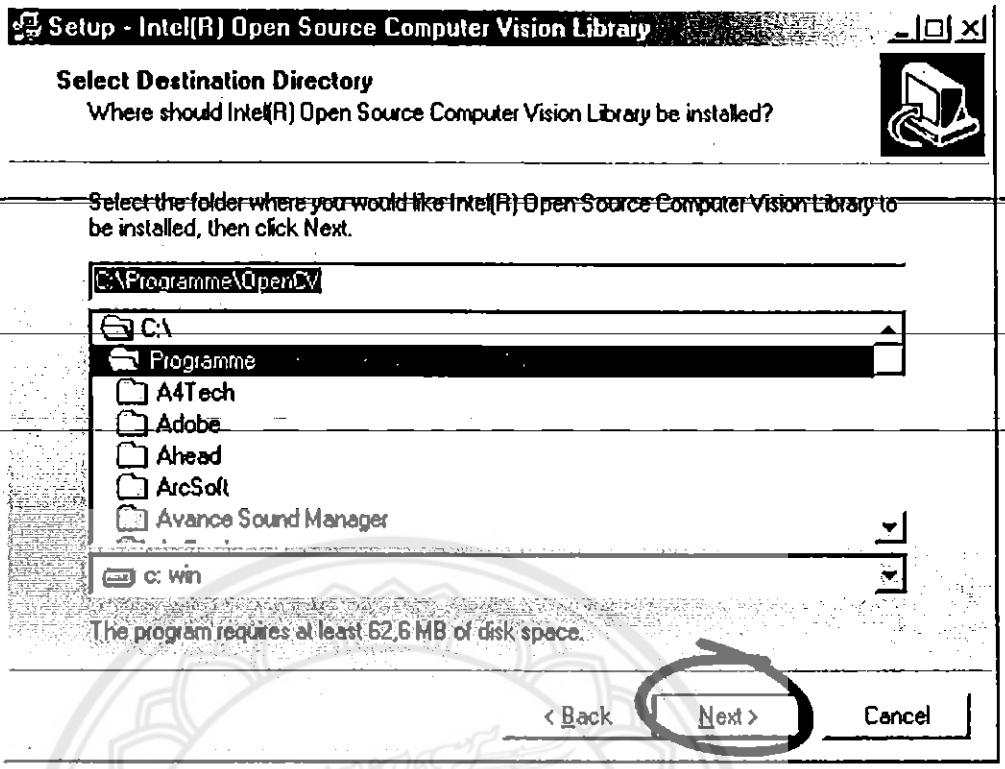


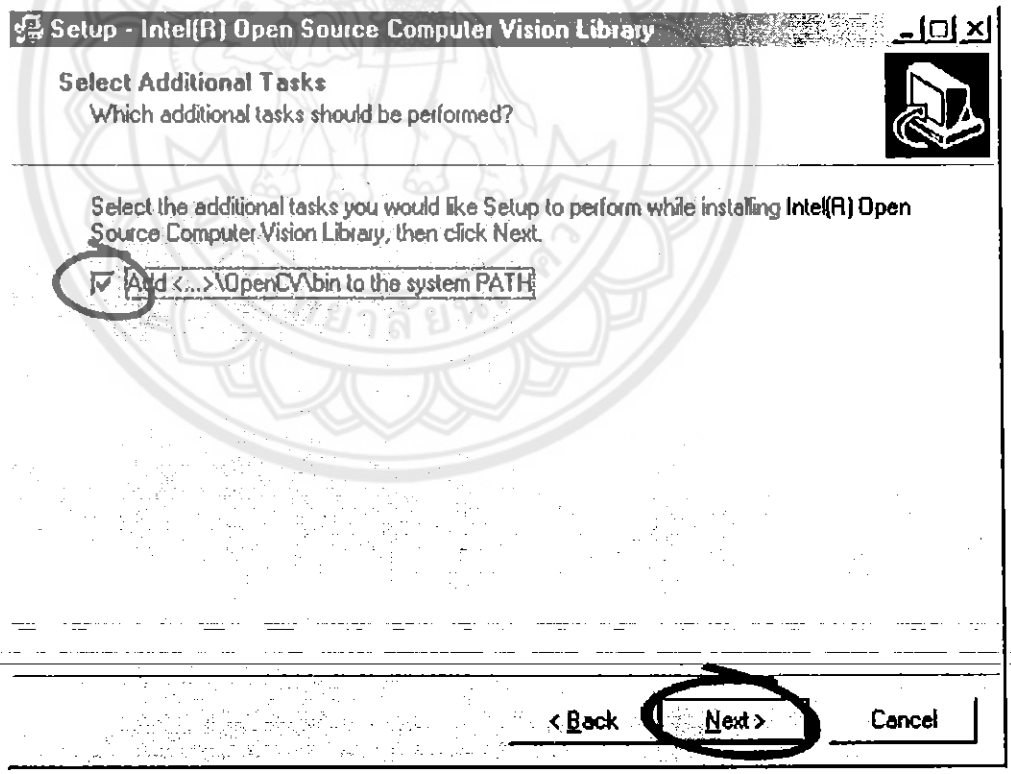
รูปที่ 1 ขั้นตอนการดาวน์โหลด OpenCV

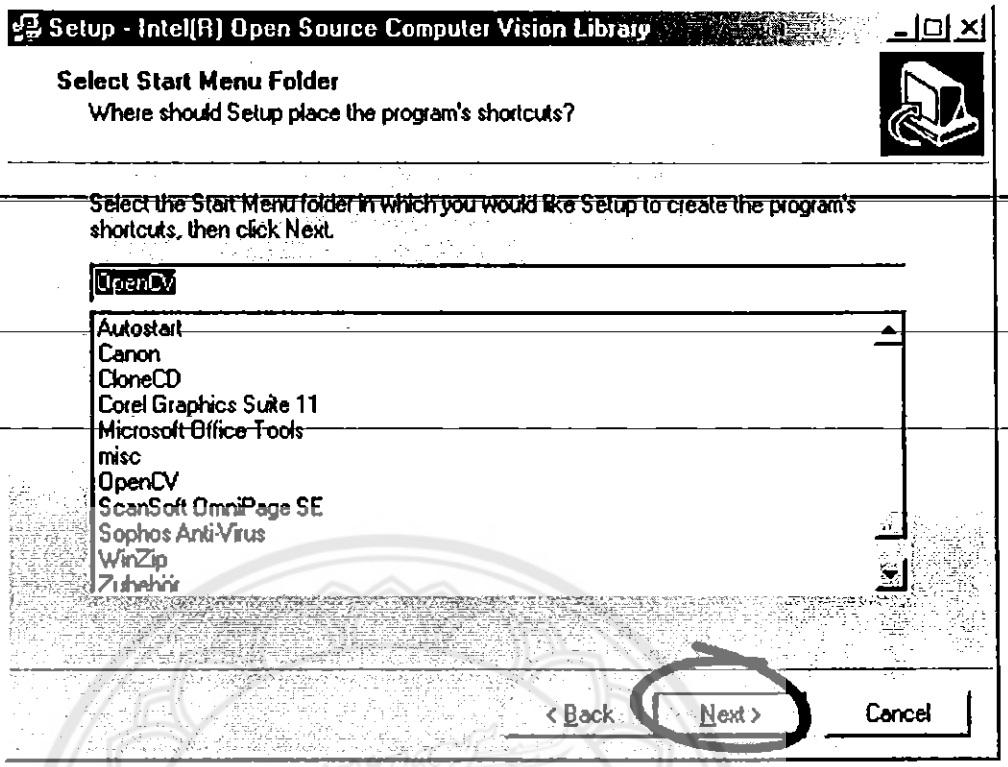
2. เมื่อได้ไฟล์ OpenCv_b3.1.exe มาแล้วให้ทำการติดตั้งดังขั้นตอนต่อไปนี้

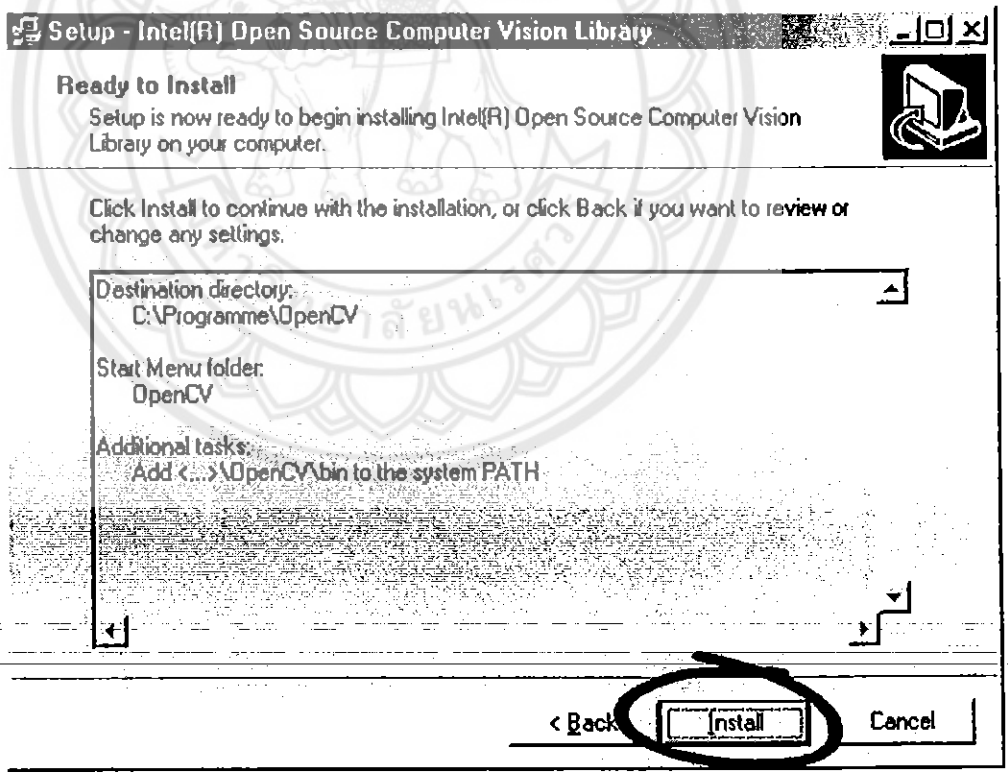


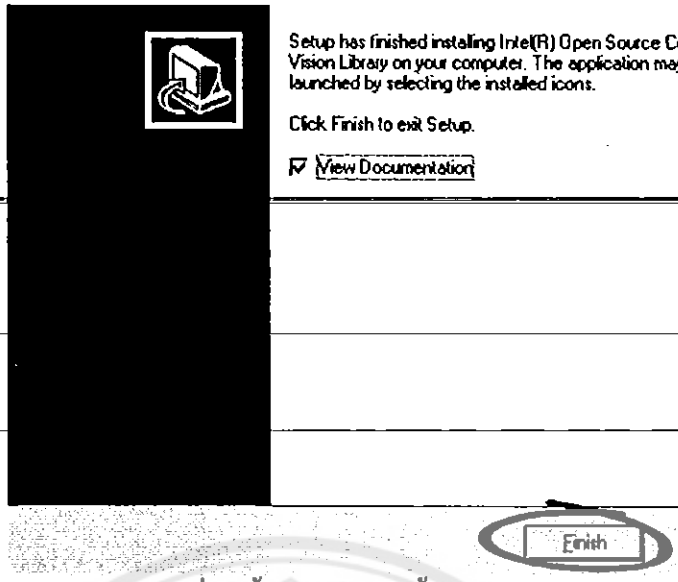



Setup - Intel(R) Open Source Computer Vision Library
Select Destination Directory
Where should Intel(R) Open Source Computer Vision Library be installed?
Select the folder where you would like Intel(R) Open Source Computer Vision Library to be installed, then click Next.
C:\Programme\OpenCV
C:\
Programme
A4Tech
Adobe
Ahead
ArcSoft
Avance Sound Manager
c: win
The program requires at least 62,6 MB of disk space.
< Back
Next >
Cancel


Setup - Intel(R) Open Source Computer Vision Library
Select Additional Tasks
Which additional tasks should be performed?
Select the additional tasks you would like Setup to perform while installing Intel(R) Open Source Computer Vision Library, then click Next.
Add <...>\OpenCV\bin to the system PATH
< Back
Next >
Cancel


Setup - Intel(R) Open Source Computer Vision Library
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Select the Start Menu folder in which you would like Setup to create the program's shortcuts, then click Next.
OpenCV
Autostart
Canon
CloneCD
Corel Graphics Suite 11
Microsoft Office Tools
misc
OpenCV
ScanSoft OmniPage SE
Sophos Anti-Virus
WinZip
< Back
Next >
Cancel


Setup - Intel(R) Open Source Computer Vision Library
Ready to Install
Setup is now ready to begin installing Intel(R) Open Source Computer Vision Library on your computer.
Click Install to continue with the installation, or click Back if you want to review or change any settings.
Destination directory:
C:\Programme\OpenCV
Start Menu folder:
OpenCV
Additional tasks:
Add <...>\OpenCV\bin to the system PATH
< Back
Install
Cancel



รูปที่ 2 ขั้นตอนการติดตั้ง OpenCV_b3.1.exe

3. ทำการ Build OpenCV เพื่อการใช้งาน ก่อนทำการ Build OpenCV ต้องทำการติดตั้ง DirectX SDK และเซตค่าเพื่อการใช้งานก่อนแล้ว ขั้นตอนการ Build OpenCV มีดังนี้

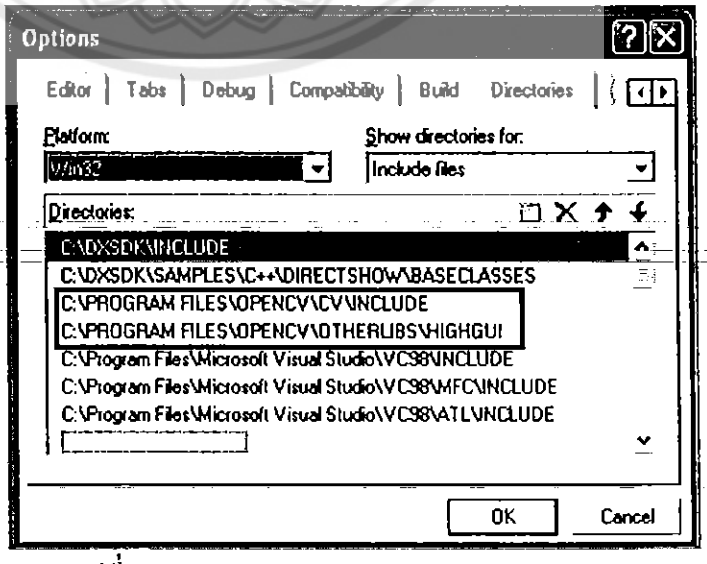
   3.1 เปิดไฟล์ OpenCV.dsw จากโฟลเดอร์ C:\Program File\OpenCV_dsw

   3.2 จากเมนูบาร์ เลือก Build->Batch Build->Build

4. การกำหนดค่าใน VisualC++ 6.0 เพื่อการใช้งาน OpenCV

   4.1 กำหนดค่า Include files

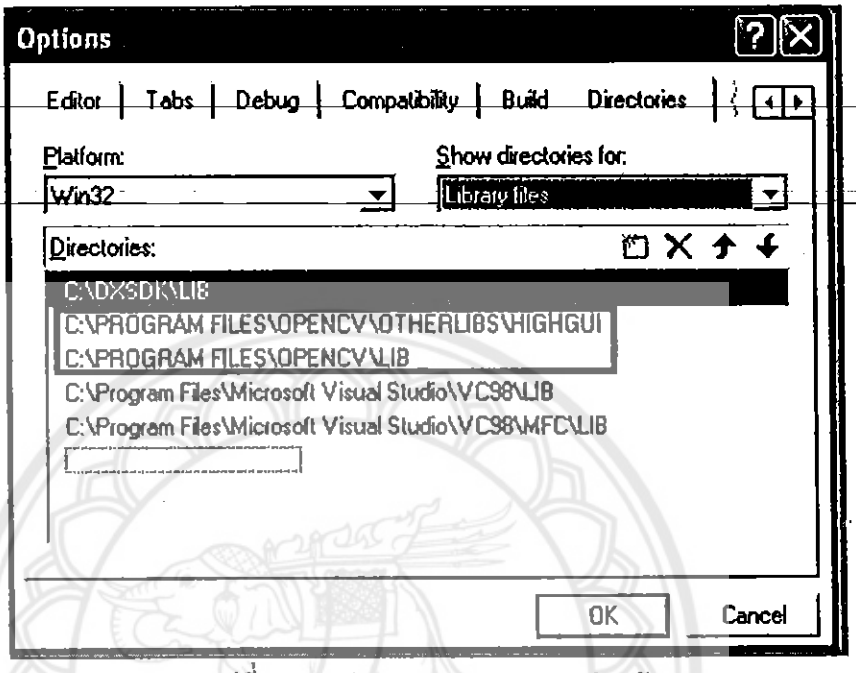
   กำหนดพาร์ทดังรูป



รูปที่ 3 การกำหนด Include files สำหรับ OpenCV

4.2 การกำหนดค่า library files

กำหนดพาร์ทดังรูป

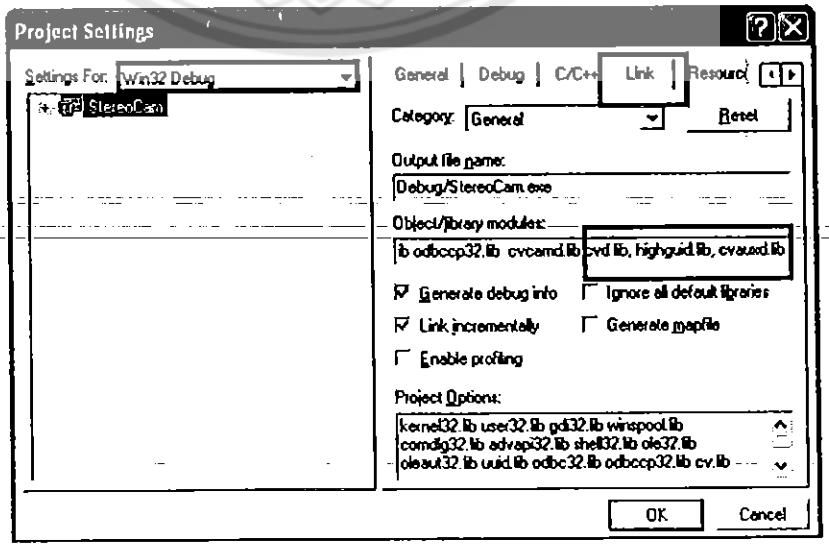


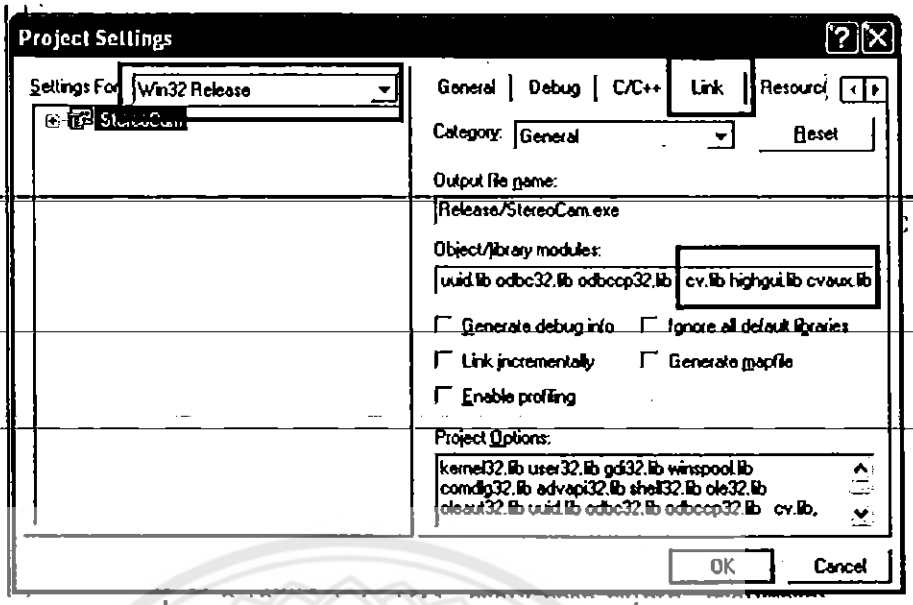
รูปที่ 4 การกำหนด Library files สำหรับ OpenCV

5. การเขียนโปรแกรมเพื่อการใช้งาน OpenCV

5.1 include file "cv.h, cvaux.h, highgui.h" เข้ามาในโปรเจค

5.2 จากเมนูบาร์ เลือก Project->Settings-> Links tab แล้วเซตค่าตามรูป





รูปที่ 5 แสดงการเซตค่า การเขียนโปรแกรมเพื่อใช้ OpenCV

ประวัติผู้เขียนโครงงาน

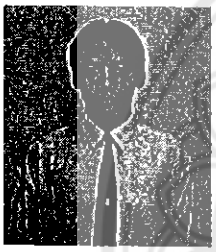
ชื่อ นาย บรรเจิด ประทุมหอม

ภูมิลำเนา 53/1 หมู่ 3 ต.แม่สลิด อ.บ้านตาก จ.ตาก 63120

ประวัติการศึกษา

- จบมัธยมศึกษาจาก โรงเรียนตากพิทยาคม
- ปัจจุบัน กำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : jerd_p@hotmail.com



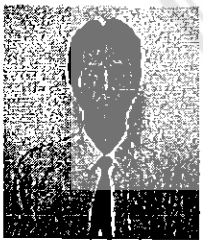
ชื่อ นาย ภาณุพันธ์ ดับปากพิง

ภูมิลำเนา 25/1 หมู่ 2 ต.ดอนทอง อ.เมือง จ.พิษณุโลก 65000

ประวัติการศึกษา

- จบมัธยมศึกษาจาก โรงเรียนพิษณุโลกพิทยาคม
- ปัจจุบัน กำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : tosuke_toro@hotmail.com



ชื่อ นาย อนุสรณ์ นาคคงคำ

ภูมิลำเนา 162 หมู่ 2 ต.พรหมพิราม อ.พรหมพิราม จ.พิษณุโลก 65150

ประวัติการศึกษา

- จบมัธยมศึกษาจาก โรงเรียนพิษณุโลกพิทยาคม
- ปัจจุบัน กำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : foto_toro@hotmail.com