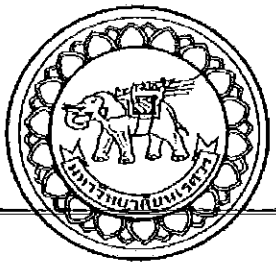
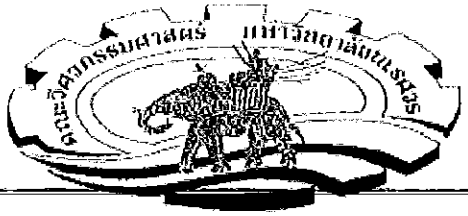


การบีบอัดข้อมูลภาพโดยวิธี LDT

Image compression by using the LDT method

นายกริชกร บุญเรือง รหัส 44362515
นายภูริพงศ์ ดอกเกี๋ยง รหัส 44362739

ปริญญานิพนธ์เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2547



ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การปิบอัดข้อมูลภาพโดยวิธี LDT
ผู้เสนอโครงงาน	นายกริชกร บุญเรือง รหัส 44362515
	นายภูริพงศ์ ดอกเกี๋ยง รหัส 44362739
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.สุชาติ แย้มเม่น
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2547

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

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

..... กรรมการ
(ดร.พนมขวัญ ริยะมงคล)

..... กรรมการ
(อาจารย์แสงชัย มังกรทอง)

หัวข้อโครงงาน	การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
ผู้ดำเนินโครงงาน	นายกริชกร บุญเรือง รหัส 44362515
	นายภูริพงศ์ ดอกเกี๋ยง รหัส 44362739
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์.ดร.สุชาติ แย้มเม่น
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2547

บทคัดย่อ

โครงงานนี้เป็นการศึกษาและพัฒนาโปรแกรมการบีบอัดข้อมูลภาพ โดยวิธีการแปลงแอลดีที ขั้นตอนการบีบอัดข้อมูลภาพประกอบด้วยนำภาพมาทำการแปลงแอลดีทีและเข้ารหัสแบบฮัฟฟ์แมน เพื่อจัดเก็บในรูปของไฟล์ไบนารี่ที่มีขนาดเล็กกว่าไฟล์ต้นฉบับ และหาอัตราการบีบอัดข้อมูลภาพเพื่อวัดการประสิทธิภาพการบีบอัดด้วยวิธีแอลดีที นอกจากนี้ยังทำการเปรียบเทียบคุณภาพของภาพที่ได้รับหลังจากการคลายการบีบอัดในรูปแบบค่า Signal to Noise Ratio (SNR) และ Peak Signal to Noise Ratio (PSNR)

จากผลการทดลองการบีบอัดข้อมูลภาพจำนวนสามภาพ คือ ภาพนีโม ภาพตา และภาพลีน่า พบว่าอัตราการบีบอัดข้อมูลของภาพตามีค่าสูงสุดเท่ากับ 1.9 ซึ่งมีค่า SNR เท่ากับ 77.2 dB และ PSNR เท่ากับ 42.4 dB

Project Title	**Image Data Compression by using the LDT method**
Name	Mr. Krichakon Boonruang ID. 44362515
	Mr. Phuripong Dokkiang ID. 44362739
Project Advisor	Assistant Professor Suchart Yammen
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic Year	2005

ABSTRACT

This project is to study and develop a program for image compression by using the linear decomposition transform (LDT). The method for compressing image in the three following steps. First, an image is transformed by using the LDT. Second, the transformed image is encoded by using the Huffman coding to keep the compressed image in term of a binary file. Third, the binary file is used for determining its compression ratio. In addition, the quality of the decompressed image is compared with the original image in regard to a Signal to Noise Ratio (SNR) and a Peak Signal to Noise (PSNR).

From the experimental result with three image compression, it has been found that the maximum value of the compression ratio is 1.9 for the eye image whose SNR is 77.2 dB and PSNR is 42.4 dB.

กิตติกรรมประกาศ

การจัดทำโครงงานในครั้งนี้ สำเร็จลุล่วงไปได้ด้วยดี ต้องขอกราบขอบพระคุณ คุณพ่อ คุณแม่ สำหรับการให้การสนับสนุน และความช่วยเหลือในด้านต่างๆ ผู้ช่วยศาสตราจารย์ ดร. สุชาติ แย้มเม่นอาจารย์ที่ปรึกษาโครงงาน ที่ให้ความดูแล เอาใจใส่และให้คำแนะนำในการทำงานเป็นอย่างดี ขอขอบพระคุณท่านคณะกรรมการสอบทุกท่านที่ได้สละเวลาอันมีค่า ครูอาจารย์ทุกท่านที่ประสิทธิ์ประสาทวิชาความรู้ให้ตลอดมา ตลอดจนเพื่อนๆทุกคนที่ได้ให้กำลังใจในการทำโครงงานนี้ตลอดมา

นายกริชกร บุญเรือง
นายภูริพงศ์ ดอกเกี๋ยง

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ทุกวันนี้คอมพิวเตอร์ได้เข้ามามีบทบาทในการทำงานมากขึ้น โดยมีการประยุกต์การใช้งานด้านต่างๆรวมทั้งด้านของการสื่อสารข้อมูล และใช้ในการบันทึกข้อมูลต่างๆเป็นต้น โดยข้อมูลในปัจจุบันนี้มีมากมายหลายแบบทั้งข้อมูลภาพ ข้อมูลเสียง ข้อมูลตัวหนังสือ ซึ่งต่างก็มีคุณสมบัติแตกต่างกันออกไป ข้อมูลภาพก็เป็นข้อมูลชนิดหนึ่งที่มีความสำคัญต่อการทำงานในปัจจุบันมาก เพราะเป็นข้อมูลที่สื่อความหมายได้ดี มีความสวยงาม แต่ข้อมูลภาพนั้นก็มีข้อเสียคือในการจัดเก็บนั้นใช้เนื้อที่ในการจัดเก็บมาก เมื่อข้อมูลมีขนาดใหญ่ทำให้ยากต่อการจัดเก็บและสื่อสารกันระหว่างเครือข่าย เพราะขนาดข้อมูลที่ใหญ่นั่นเอง ทางคณะผู้เสนอโครงงานจึงมีแนวคิดเพื่อที่จะหาทางย่อขนาดของข้อมูลภาพให้มีขนาดเล็กลง ทางคณะผู้เสนอโครงงานจึงได้นำเทคนิค LDT (Linear Decomposition Transform) [1] มาใช้ในการย่อขนาดของข้อมูลให้มีขนาดเล็กลง โดยแนวคิดที่ใช้จะเน้นไปที่ภาพขาวดำเพื่อเป็นแนวทางให้ผู้ที่สนใจได้พัฒนาต่อไป

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาพื้นฐาน LDT (Linear Decomposition Transform)
2. ทำการลดขนาดข้อมูลภาพให้มีขนาดเล็กลงกว่าภาพเดิม

1.3 ขอบข่ายของโครงงาน

1. ศึกษาเรื่อง LDT
2. ศึกษาการนำเทคนิค LDT ไปใช้ในการออกแบบเพื่อการบีบอัดข้อมูลภาพ

1.4 ขั้นตอนการดำเนินโครงงาน

1.4.1 การศึกษา

ตารางที่ 1.1 การศึกษา

กิจกรรม	เดือน - ปี		
	ม.ค. 47	ก.พ. 47	มี.ค. 47
ศึกษาเรื่อง LDT	←		
ศึกษาการนำเทคนิค LDT ไปใช้เพื่อบีบอัดข้อมูลภาพ		←	
ศึกษาการจัดเก็บข้อมูลภาพขาวดำ			→

1.4.2 การปฏิบัติงาน

ตารางที่ 1.2 การปฏิบัติงาน

กิจกรรม	เดือน – ปี							
	ก.พ. 47	มี.ค. 47	เม.ย. 47	พ.ค. 47	มิ.ย. 47	ก.ค. 47	ส.ค. 47	ก.ย. 47
1. หาข้อมูล	←	→						
2.ออกแบบ		←	→					
3. พัฒนา			←	→				
4. ทดสอบ					←	→		
5.สรุปผล							←	→

1.5 ผลที่คาดว่าจะได้รับ

1. ได้รับความรู้เกี่ยวกับ
   - เรื่อง LDT
   - เรื่องการจัดเก็บข้อมูลภาพขาวดำ
   - การบีบอัดข้อมูลภาพโดยใช้เทคนิค LDT
2. สามารถที่จะบีบอัดข้อมูลภาพให้มีขนาดเล็กลงได้
3. เพื่อให้ผู้ที่สนใจศึกษาและพัฒนาต่อไป

1.6 รายละเอียดงบประมาณ

1. ค่าถ่ายเอกสาร	500	บาท
2. ค่าอุปกรณ์คอมพิวเตอร์	1,500	บาท
รวมทั้งสิ้น	**2,000**	**บาท**

(สองพันบาทถ้วน)

บทที่ 2
หลักการและทฤษฎี

2.1 การลดขนาดข้อมูลภาพ

ในช่วงหลายปีที่ผ่านมานี้ ความต้องการของการลดขนาดข้อมูลภาพมีเพิ่มขึ้นอย่างต่อเนื่อง ตัวอย่างเช่น การบีบอัดขนาดข้อมูลภาพ ที่เป็นส่วนสำคัญของการใช้งานคอมพิวเตอร์มัลติมีเดีย (Computer Multimedia) การประชุมด้วยภาพระยะไกล (Televideo Confcrencing) ภาพถ่ายทางการแพทย์ (Medical Image) การส่งโทรสาร (Facsimile) และอื่นๆอีกมากที่ต้องใช้ความสามารถในการจัดการ จัดเก็บและการส่งข้อมูลภาพ ขบวนการลดขนาดข้อมูลจะเป็นสิ่งที่จำเป็นทั้งสิ้น

จากการที่ข้อมูลภาพซึ่งเป็นสองมิติ (2-D) ที่ได้มาจากการสุ่ม (Sampling) และควอนไตซ์ (Quantization) ของวัตถุที่แสงตกกระทบ เข้ามาเป็นข้อมูลดิจิตอลในคอมพิวเตอร์นั้น ข้อมูลของภาพที่ได้จะมีปริมาณมาก ซึ่งเป็นอุปสรรคในการจัดเก็บ การประมวลผลและการสื่อสารข้อมูลภาพ

ขบวนการลดขนาดข้อมูลภาพ (Image Compression) จะเป็นการลดจำนวนของข้อมูลที่จะใช้แทนภาพนั้นๆลง โดยมีหลักการคือ การตัดข้อมูลส่วนที่เกินความจำเป็นออกไป จึงทำให้ข้อมูลภาพที่เหลือนั้นลดลงได้ โดยจะกระทำก่อนที่ข้อมูลภาพจะถูกจัดเก็บในอุปกรณ์บันทึกข้อมูล หรือก่อนที่จะใช้สื่อสารและในเวลาต่อมาเมื่อต้องการใช้งาน ข้อมูลจะถูกแปลงผกผันออกมาเป็นภาพเดิมหรือเป็นภาพที่ใกล้เคียงกับภาพเดิมเพื่อนำไปใช้ต่อไป

ขบวนการลดขนาดข้อมูล (Data Compression) จะหมายถึงขบวนการที่ใช้ในการทำให้ข้อมูล (Data) ที่ต้องใช้แทนข่าวสารหนึ่งนั้นลดลง ซึ่งสามารถเปรียบเทียบข้อมูลได้กับตัวหนังสือที่จะสื่อความหมายถึงเนื้อหาสาระภายในหนังสือเล่มหนึ่งๆนั้นเอง ในกรณีของหนังสือสองเล่มที่แต่งโดยคนละคนแต่มีเนื้อหาเหมือนกันนั้น จำนวนตัวอักษร (ข้อมูล) ที่ใช้ในการบอกเล่าจะไม่เท่ากันก็ได้ นั้นก็แสดงว่าหนังสือที่ผู้แต่งใช้จำนวนตัวอักษรที่มากว่าจะต้องมีคำหรือประโยคบางประโยคที่เกินความจำเป็น เช่น อาจเป็นประโยคที่บอกเล่าถึงสิ่งที่ได้กล่าวมาแล้วก่อนหน้านี้ หรือคำ/ประโยค ที่ไม่ได้สื่อความหมายใดๆ การลดขนาดข้อมูลภาพก็เช่นเดียวกัน ข้อมูลของระดับความสว่างบนจุดภาพแต่ละจุดร่วมกันเพื่อสื่อถึงความหมายของภาพ ก็จะมีส่วนที่เกินความจำเป็นที่สามารถตัดออกไปได้ ถ้ากำหนดให้ n_1 และ n_2 เป็นจำนวนของข้อมูลที่ใช้ในการสื่อความหมายของภาพๆหนึ่ง อัตราส่วนการลดขนาดข้อมูล (Data compression ratio) จะคำนวณได้คือ

$$\text{Data compression ratio} = \frac{n_1}{n_2} \qquad (2\text{-}1)$$

และอีกวิธีหนึ่งในการวัดค่าการลดขนาดของข้อมูลภาพที่นิยมใช้กันคือ การวัดจำนวนบิต (Bit) ของข้อมูลที่ต้องใช้แทนระดับความสว่างของจุดภาพใดๆของภาพๆนั้น (Bit Per Pixel : bpp) โดยทั่วไปแล้วภาพระดับสีเทา 256 ระดับจะต้องใช้จำนวนบิตข้อมูลต่อหนึ่งจุดภาพเท่ากับ 8 บิต (8 bpp) เมื่อนำมาผ่านขบวนการลดข้อมูลแล้วจำนวนบิตที่ต้องใช้แทนระดับความสว่างนี้อาจลดลง อาจเหลือเพียง 1.1 bpp ได้โดยที่ระดับความสว่างยังคงมีได้ 256 ระดับเท่าเดิม

2.2 การลดขนาดข้อมูลแบบไม่มีความผิดพลาด

ในการประยุกต์ใช้ขบวนการลดขนาดข้อมูลภาพกับข้อมูลบางชนิด จะมีความจำเป็นที่จะต้องใช้ขบวนการลดขนาดข้อมูลภาพแบบที่ไม่มีความผิดพลาด (Loss-less Compression) เช่น การเก็บเอกสารทางการแพทย์หรือ เอกสารหลักฐานทางธุรกิจ ซึ่งเป็นข้อกำหนดทางกฎหมายที่ต้องใช้ขบวนการลดขนาดข้อมูลแบบที่ไม่มีความผิดพลาด หรืออีกตัวอย่างเช่น ในการประมวลผลข้อมูลภาพถ่ายจากดาวเทียม LANDSAT ซึ่งเป็นข้อมูลที่ได้มาด้วยค่าใช้จ่ายที่มากจึงไม่ควรให้การเก็บข้อมูลนี้เกิดความผิดพลาดขึ้น หรือในการนำไปใช้ในการเก็บข้อมูลของภาพถ่าย X-ray ซึ่งความผิดพลาดที่เกิดขึ้นจะส่งผลต่อความแม่นยำในการวินิจฉัยโรคลดลง จากตัวอย่างข้างต้นนี้จำเป็นต้องใช้ขบวนการลดขนาดข้อมูลแบบไม่มีความผิดพลาด

สำหรับขบวนการพื้นฐานของ Loss-less Compression ที่ใช้กันอยู่ในปัจจุบัน จะใช้ค่าอัตราการลดขนาดข้อมูล ประมาณ 2 ถึง 10 เท่า ซึ่งสามารถนำไปใช้กับข้อมูลภาพขาวดำหรือข้อมูลภาพระดับความสว่าง (Binary Image) ได้

จากหัวข้อที่ผ่านมา ขบวนการของ Loss-less Compression จะประกอบขึ้นจากขั้นตอนสองขั้นตอนคือ (1) ขบวนการแปลงข้อมูลภาพใหม่ เพื่อเป็นการลด Inter pixel Redundancy และ (2) ขบวนการเข้ารหัสข้อมูล เพื่อที่จะลด Coding Redundancy หรือจากรูปของขบวนการ source Coding

2.3 การลดขนาดข้อมูลแบบที่มีความผิดพลาด

วิธีการของ Lossy Compression นี้จะไม่เหมือนกับขบวนการดังที่ผ่านมาแล้ว โดยมันจะยอมลดความถูกต้องของภาพผลลัพธ์ที่ได้เพื่อแลกกับอัตราการลดขนาดข้อมูลที่เพิ่มขึ้น ทั้งนี้ขึ้นอยู่กับว่าในการใช้งานของข้อมูลภาพจะสามารถยอมรับความผิดพลาดได้มากเพียงใด โดยทั่วไปแล้วการใช้ขบวนการของ Lossy Compression จะสามารถใช้ลดขนาดข้อมูลภาพระดับสีเทา (Gray level image) ได้ถึง 30 เท่า โดยภาพที่ได้ยังคงใช้สื่อความหมายได้และที่ค่าอัตราส่วนการลดข้อมูลที่ 10-20 เท่า

ภาพที่ได้จะเหมือนกับภาพต้นแบบ เมื่อเปรียบเทียบกับการลดขนาดข้อมูลด้วยขบวนการ Loss-less Compression ซึ่งสามารถลดขนาดข้อมูลได้เพียง 2-3 เท่า

2.4 การเข้ารหัสแบบฮัฟฟ์แมน (Huffman Coding)

วิธีการที่ง่ายที่สุดในการทำ Loss-less Compression คือ จะเป็นการลดเพียงแต่ส่วนของ Coding Redundancy เท่านั้น ซึ่งเป็นความซ้ำซ้อนที่เกิดขึ้น เมื่อมีการใช้รหัสเลขฐานสอง ในการแทนความหมาย ระดับความสว่างของจุดภาพต่างๆ

การเข้ารหัสข้อมูลของระดับความสว่างที่จะทำให้ลดขนาดข้อมูลให้มีขนาดน้อยลงนั้น สามารถทำได้หลายวิธีด้วยกัน ซึ่งวิธีหนึ่งที่มีประสิทธิภาพคือ Huffman Coding เป็นวิธีที่นิยมใช้ในการลดขนาดข้อมูลในส่วนของ Coding Redundancy เนื่องจากมันเป็นวิธีที่จะให้รหัสที่เป็นตัวแทนของข้อมูลที่ต้องการได้สั้นที่สุด สำหรับวิธีการของ Huffman Coding จะสามารถทำได้โดยมีขั้นตอนดังนี้

ในขั้นแรกจะเป็นขบวนการที่แรกว่า Source Reduction โดยการนำค่าของความน่าจะเป็นในการที่จะพบ ค่าระดับความสว่างต่างๆ (P_r) ในข้อมูลภาพ นำค่าของความน่าจะเป็นนี้มาจัดเรียงกันตามลำดับจากมากไปหาน้อย จากนั้นก็จะทำการรวมค่าความน่าจะเป็นของระดับความสว่าง ที่น้อยที่สุดสองลำดับเข้าด้วยกันเป็นค่าเดียว จากนั้นจะทำการเรียงค่าความน่าจะเป็นนี้ใหม่และกระทำซ้ำเช่นนี้ต่อไปจนหมด

ตารางที่ 2.1 แสดงขั้นตอนการจัดกลุ่ม

ข้อมูลเริ่มต้น		การจัดกลุ่มข้อมูล			
ความสว่าง	ความน่าจะเป็น	1	2	3	4
a2	0.4	0.4	0.4	0.4	0.6
a6	0.3	0.3	0.3	0.3	0.4
a1	0.1	0.1	0.2	0.3	
a4	0.1	0.1	0.1		
a3	0.06	0.1			
a5	0.04				

ดังตารางที่ 2.1 แสดงถึงวิธีการของ Source Reduction ทางด้านซ้ายมือจะเป็นค่าของระดับความสว่างต่างๆ เรียงลงมาตามค่าของความน่าจะเป็น จากมากไปหาน้อย ในการทำ Source Reduction ครั้งที่ 1 ค่าที่น้อยที่สุดของความน่าจะเป็นสองค่า คือ 0.06 และ 0.04 จะถูกรวมเข้า

ด้วยกันเป็น 0.1 เพื่อแสดงถึงความน่าจะเป็นที่จะพบระดับความสว่างทั้งสองนี้ แล้วนำมาจัดเรียงใหม่ในช่องของ Source Reduction ครั้งที่ 1 จากนั้นจะกระทำเช่นนี้ต่อไปเรื่อยๆจนกระทั่งเหลือรหัสเพียงสองกลุ่มเท่านั้น

ขั้นที่สองของการทำ Huffman Coding คือการกำหนด รหัสที่ใช้แทนข้อมูลระดับความสว่างนี้ใหม่ โดยเริ่มจากข้อมูลที่ถูกจัดกลุ่มแล้ว (ทางด้านขวา) ย้อนกลับไปยังข้อมูลเริ่มต้น ดังแสดงในตารางที่ 2.2

ตารางที่ 2.2 แสดงขั้นตอนการกำหนดรหัสแทนข้อมูล

ข้อมูลเริ่มต้น			การจัดกลุ่มข้อมูล			
ข้อมูล	ความน่าจะเป็น	รหัส	1	2	3	4
a2	0.4	1	0.4 1	0.4 1	0.4 1	0.6 0
a6	0.3	00	0.3 00	0.3 00	0.3 00	0.4 1
a1	0.1	011	0.1 011	0.2 010	0.3 01	
a4	0.1	0100	0.1 0100	0.1 011		
a3	0.06	01010	0.1 0101			
a5	0.04	01011				

โดยเริ่มจากระดับความสว่างที่ถูกจัดให้เหลือเพียงสองกลุ่ม จะถูกแทนที่ด้วยรหัสเลขฐานสองคือ “0” และ “1”

ซึ่งจะใช้รหัส “0” แทนกลุ่มของข้อมูลที่ค่าความน่าจะเป็น 0.6 และ “1” แทนกลุ่มของข้อมูลที่ค่าความน่าจะเป็น 0.4 (สำหรับการแทนรหัสของกลุ่มของข้อมูลระดับความสว่างนี้ จะสามารถสลับกันได้ ระหว่าง “0” และ “1”) จากกลุ่มของระดับความสว่างที่มีค่าความน่าจะเป็นเท่ากับ 0.6 ซึ่งได้มาจากการรวมกันของกลุ่มข้อมูลระดับความสว่าง ดังนั้นเพื่อที่จะแยกความแตกต่างของระดับความสว่างทั้งสองนี้จึงใช้เลขฐานสอง “ 0 ” และ “ 1 ” โดยการต่อเพิ่มเข้าไปกับรหัสที่ได้กำหนดให้ก่อนหน้านี้ และกระทำต่อไปจนกระทั่งย้อนกลับไปถึงระดับความสว่างเริ่มต้นทางด้านซ้ายมือ จากรหัสที่หาได้ตามขบวนการของ Huffman Coding ตามตัวอย่างข้างต้นนี้จะได้ค่าของความยาวรหัสเฉลี่ย $\left(L_{avg}\right)$ คือ

$$L_{avg} = 1(0.4)+2(0.3)+3(0.1)+4(0.1)+5(0.06)+5(0.04) \quad (2\text{-}2)$$

$$= 2.2 \text{ Bit}$$

วิธีการของ Huffman Coding นี้จะเป็นวิธีการที่มีประสิทธิภาพมากในการสร้างรหัสแทนข้อมูลที่ต้องการ หลังจากที่หารหัสของ Huffman ได้แล้ว ขบวนการเข้ารหัสข้อมูลหรือถอดรหัสข้อมูลสามารถทำได้โดยการเปิดตารางเทียบข้อมูลที่ต้องการกับรหัสที่ได้

ข้อมูลที่เข้ารหัสแบบ Huffman นี้จะมีลักษณะพิเศษคือ ในการถอดรหัสข้อมูลจะสามารถทำได้โดยที่ไม่ต้องมีข้อมูลอ้างอิง ขั้นระหว่างรหัสแต่ละตัว รหัสของ Huffman นี้จะสามารถแยกออกมาได้โดยไม่ซ้ำกัน และจะแยกรหัสนี้ได้เพียงวิธีเดียวเท่านั้น จากการถอดรหัสจากซ้ายไปขวา ตัวอย่าง จากรหัส Huffman ที่ได้ในตารางที่ 2.2 ถ้าข้อมูลที่ผ่านการเข้ารหัสแล้วมีดังนี้ " 010100111100 " จะพบว่าถ้าอ่านจากซ้ายมาขวาแล้ว รหัสตัวแรกที่สามารถอ่านได้ คือ "0101" ซึ่งแทนข้อมูลของระดับความสว่างที่ a_3 รหัสตัวต่อไปที่มีคือ " 011 " ซึ่งเป็นรหัสของข้อมูล a_1 ทำต่อไปจนหมดจะได้ว่าข้อมูลของรหัสนี้คือ $a_3a_1a_2a_2a_6$ ซึ่งจากตัวอย่างนี้จะเห็นได้ว่าในการถอดรหัสของ Huffman จะสามารถทำได้แบบเดียวเท่านั้น วิธีการเข้ารหัสแบบ Huffman นี้เป็นวิธีการที่มีประสิทธิภาพมากที่สุด จึงนิยมใช้กันอย่างทั่วไป

2.5 แนวคิดการแปลงรูปแอลดีที (Linear Decomposition Transform)

กระบวนการ Wavelet เป็นกระบวนการที่สำคัญในการจัดการกับสัญญาณที่ทำให้การส่งสัญญาณนั้นมีคุณภาพ โดยการเปลี่ยนแปลงกระบวนการ wavelet จะถูกจำแนกสัญญาณไปเป็น linear combination ของผลรวมเชิงเส้นสัญญาณพื้นฐาน (Bases signals)กับค่าสัมประสิทธิ์ที่มีลักษณะเฉพาะในการประยุกต์ใช้งานด้านต่างๆ มากมายอาทิเช่น การลดสัญญาณรบกวน การบีบอัดข้อมูล และการตรวจจับสัญญาณที่เปลี่ยนแปลงฉับพลัน เป็นต้น

ในระดับหนึ่งของการแปลงรูปเวฟเล็ต จะเริ่มต้นด้วยการแยกสัญญาณนำเข้าออกเป็นสองชุดสัญญาณย่อยที่มีความยาวของแต่ละชุดสัญญาณย่อยเท่ากับเครื่องหนึ่งของความยาวสัญญาณนำเข้าสัญญาณชุดแรกมาจากสัญญาณนำเข้าที่มีเวลาเป็นเลขคู่และผ่านตัวกรองความถี่ต่ำที่ไม่แปรเปลี่ยนตามเวลาทำให้ได้สัญญาณแบบหยาบ (Coarse signal) ชุดแรกนี้บรรจุข้อมูลเฉพาะความถี่ต่ำในขณะที่สัญญาณชุดสองมาจากสัญญาณนำเข้าที่มีเวลาเป็นเลขคี่และผ่านตัวกรองความถี่สูงที่ไม่แปรเปลี่ยนตามเวลาทำให้ได้สัญญาณแบบละเอียด (Detail signal) ชุดสองนี้บรรจุข้อมูลเฉพาะความถี่สูง โดยทั่วไปตัวกรองสัญญาณที่ใช้มีความยาวที่แน่ชัดและรวมไปถึงฮาร์เวฟเล็ต (Haar wavelet) เม็กซิกันแฮทเวฟเล็ต (Mexican hat wavelet) และคอบีเชส์เวฟเล็ต (Daubechies wavelet) ณ ขั้นตอนการแปลงรูปเวฟเล็ตดังที่ได้กล่าวแล้วข้างต้นอาจเรียกอีกอย่างหนึ่งว่าสภาวะการวิเคราะห์ (Analysis stage) ในการแปลงรูปเวฟเล็ตจะต้องเลือกตัวกรองสัญญาณความถี่ต่ำและตัวกรองสัญญาณความที่สูงที่ทำให้ขบวนการแยกองค์ประกอบสัญญาณมีอินเวอรส์ (Invertible decomposition process) กล่าวคือ สัญญาณนำเข้าสามารถนำกลับคืนดั้งเดิมมาได้จากสัญญาณแบบ

หยาบและสัญญาณแบบละเอียด ขบวนการนี้อาจเรียกอีกอย่างหนึ่งว่าสภาวะสังเคราะห์ (Synthesis stage)

การแปลงรูปแอลดีที (Linear Decomposition) กับสัญญาณหนึ่งมิติ โดยทั่วไปการแปลงรูปแอลดีทีมีคุณลักษณะสมบัติแตกต่างจากการแปลงรูปเวฟเล็ตในสามประการดังต่อไปนี้คือ

- การแปลงรูปแอลดีที (LDT) จะใช้ตัวกรองสัญญาณที่ไม่แปรเปลี่ยนตามเวลาเพียงตัวเดียว
- ตัวกรองสัญญาณที่ไม่แปรเปลี่ยนตามเวลาของการแปลงรูปแอลดีทีจะปรับเปลี่ยนตามสัญญาณนำเข้า จึงทำให้ศักยภาพการแปลงรูปแอลดีทีมีประสิทธิภาพเหนือกว่าการแปลงรูปเวฟเล็ตในแง่ของการประยุกต์ใช้งานด้านการบีบอัดข้อมูลและการลดสัญญาณรบกวน [1]
- ค่าพารามิเตอร์ของตัวกรองสัญญาณจะถูกเลือกมาจากการใช้เกณฑ์ต่ำสุดของขนาดแอลพีนอร์ม (/p norm criterion)

2.6 พื้นฐานของการแปลงรูปแอลดีที

กำหนดให้ x (n) เป็นสัญญาณแบบไม่ต่อเนื่อง (Discrete signal) ที่มี x (n) = 0 สำหรับ n< 0 และสัญญาณนำเข้าที่จะถูกแปลรูปแอลดีทีมีค่าเป็น

$$\{x(0), x(1),x(2),\ldots,x(N-1)\} \tag{2-3}$$

ในที่นี้ N เป็นจำนวนเต็มคู่สมมุติว่าสัญญาณนำเข้าในสมการ (2-3) ประกอบด้วยผลรวมของสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า (Log term trend) {c (n)} และสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ (Short term trend) {d (n)} โดยที่แต่ละสัญญาณมีจำนวนข้อมูล N/2 การแปลงรูปแอลดีทีของ x(n) ถูกกำหนดนิยามดังแสดงไว้ในรูป 2.1 ได้ว่า

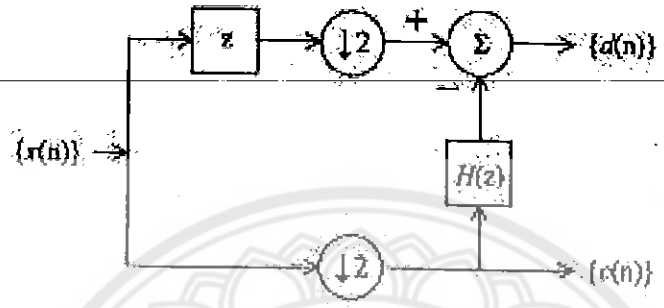
$$(\{c(n)\}, \{d(n)\}) = T(x(n)\}) \tag{2-4}$$

โดยที่ข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้าซึ่งได้มาจากการสุ่มตัวอย่างแบบลงด้วยสองช่วงเวลาค่าเป็น

$$c(n)=x(2n) \text{ สำหรับ } n = 0,1,\ldots, (N/2) - 1 \tag{2-5}$$

และข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ซึ่งได้มาจากการหาผลต่างของข้อมูลระหว่างการสุ่มตัวอย่างแบบลงด้วยสองช่วงเวลาของข้อมูลนำเข้ากับค่าประมาณการของข้อมูลดังกล่าว มีค่าเป็น

$$d(n) = x(2n+1) - \sum_{k=-q}^{q-1} h(k)x(2n-2k) \text{ สำหรับ } n = 0, 1, \ldots, (N/2)-1 \qquad (2\text{-}6)$$



รูป 2.1 พื้นฐานการแปลงรูปแอลดีที

คุณลักษณะของสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ {d(n) } จะถูกกำหนดโดยการเลือกค่าพารามิเตอร์ {h (k)} จำนวน 2q ตัวของตัวกรองสัญญาณดังแสดงไว้ในสมการ (2-6) จากรูป 2.1 เราพบว่าค่าของพารามิเตอร์ {h (k)} สามารถถูกเขียนให้อยู่ในรูปแบบการแปลงรูปซี (Z - Transform) ได้ ดังนี้คือ

$$H(z) = \sum_{k=-q}^{q-1} h(k)z^k \qquad (2\text{-}7)$$

ในความสนใจของผู้ทำการทดลอง จะเลือกค่าจำนวนพารามิเตอร์ 2q ตัวที่มีจำนวนค่อนข้างน้อยกว่าจำนวนข้อมูลสัญญาณนำเข้ามากนั่น คือ q<N และ q เป็นจำนวนเต็มบวกที่น้อย

2.7 การเลือกค่าพารามิเตอร์ของตัวกรองสัญญาณแบบไม่แปรเปลี่ยนตามเวลา

เพื่อเพิ่มประสิทธิภาพการลดสัญญารบกรวน จะเลือกค่าพารามิเตอร์ของตัวกรองสัญญาณ {h(k)} ที่ทำให้ขนาดแอลพีนอร์ม (*lp* Norm) สัญญาณที่พฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ {d (n)} ดังแสดงไว้ในสมการ (2-6) มีค่าน้อยที่สุด กล่าวคือ

$$\min_{h(k)\in R}\left[f(h(-q),.....,h(q-1))\right]=\min_{h(k)\in R}\left(\sum_{n=0}^{(N/2)-1}\left|x(2n+1)-\sum_{k=-q}^{q-1}h(k)x(2n-2k)\right|^{p}\right)^{1/p}$$

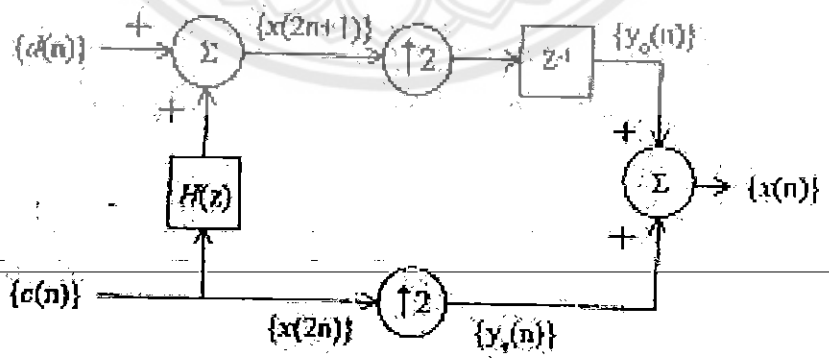
$$=f(h^{0}(-q),...,h^{0}(q-1)) \qquad (2\text{-}8)$$

ในการวิจัยนี้จะทดลองเฉพาะสองกรณีคือ p= 2 และคำตอบค่าพารามิเตอร์ที่ถูกเลือก $h^0(-q),\ldots,h^0(q-1)$ ได้มาจากวิธี Perturbation

2.8 ส่วนกลับของการแปลงรูปแอลดีที

สัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า {c (n)} และสัญญาณเที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ {d (n)} สามารถนำมาผ่านส่วนกลับของการแปลงรูปแอลดีทีเพื่อให้ได้สัญญาณนำเข้า {x (n)} ซึ่งส่วนกลับของการแปลงรูปแอลดีทีดังแสดงไว้รูป 2.2 ถูกนิยามได้ว่า

$$\{x(n)\} = T^{-1}(\{c(n)\}, \{d(n)\}) \qquad (2\text{-}9)$$



รูป 2.2 ส่วนกลับของการแปลงรูปแอลดีทีพื้นฐาน

จากรูป 2.2 เมื่อข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า c (n) ผ่านการสุ่มตัวอย่างแบบขึ้นด้วยสองช่วงเวลาจะได้ค่าผลลัพธ์ข้อมูลสัญญาณ ye (n) ที่มีค่าเป็น

$$y_e(n) = \begin{cases} c(n/2), n = 0,2,4...,(N/2) \\ 0, n = 1,3,5,...,(N/2)-1 \end{cases} \quad (2\text{-}10)$$

นำค่าข้อมูล c (n) ในสมการ (2-5) แทนลงในสมการ (2-10) จะได้ว่า

$$y_e(n) = \begin{cases} x(n), n = 0,2,4...,(N/2) \\ 0, n = 1,3,5,...,(N/2)-1 \end{cases} \quad (2\text{-}11)$$

จากสมการ (2-5) และ (2-6) จะได้ข้อมูลสัญญาณนำเข้าสำหรับช่วงเวลาเลขคี่กลับคืนมาดังแสดงไว้ในรูป 2.2 ที่มีค่าเป็น

$$x(2n-1) = d(n) + \sum_{k=-q}^{q-1} h(k)c(n-k) \text{ สำหรับ } n = 0, 1,..., (N/2-1) \quad (2\text{-}12)$$

ต่อจากนั้นนำข้อมูลสัญญาณนี้ผ่านการสุ่มตัวอย่างแบบขึ้นด้วยสองช่วงเวลาและเลื่อนช่วงเวลาไปทางซ้ายหนึ่งหน่วยจะได้ข้อมูลสัญญาณ yo (n) ดังนี้คือ

$$y_0(n) = \begin{cases} 0, n = 0,2,4...,(N/2) \\ x(n), n = 1,3,5,...,(N/2)-1 \end{cases} \quad (2\text{-}13)$$

โดยการใช้ผลรวมกันสมการ (2-11) และ (2-13) จะได้สัญญาณนำเข้ากลับคืนมาเป็น

$$y_0(n) + y_e(n) = x(n) \quad (2\text{-}14)$$

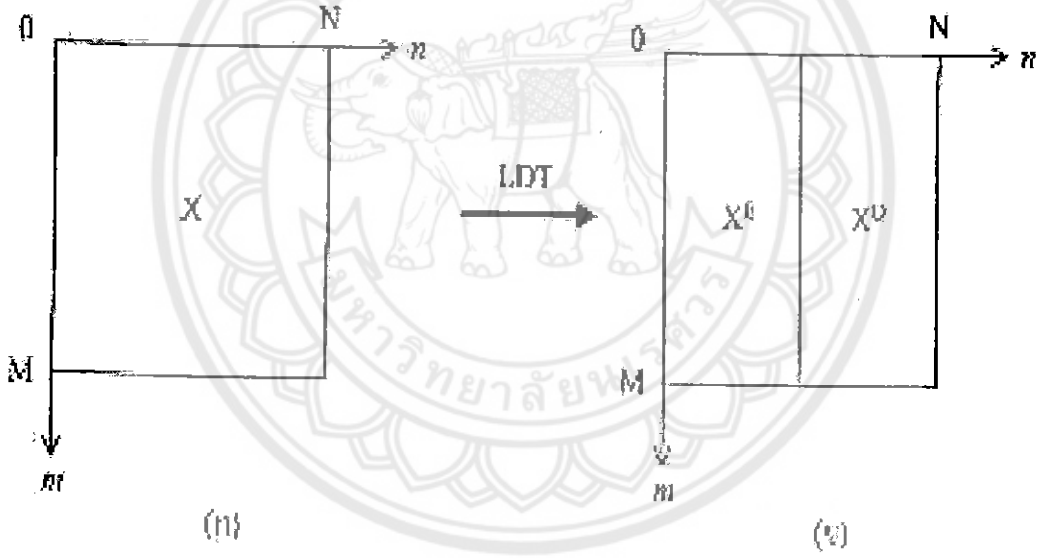
2.9 พื้นฐานของการแปลงรูปแอลดีทีสำหรับสองมิติ

กำหนดให้ X (m, n) เป็นข้อมูลภาพสองมิติขนาด MxN ที่มี M และ N เป็นจำนวนเต็มบวก โดยที่ X (m, n) สำหรับ m< 0 และ n<0 การแปลงรูปแอลดีทีภาพสองมิติที่นำเสนอประกอบด้วยสองขั้นตอนดังต่อไปนี้ คือ เริ่มด้วยการแปลงรูปแอลดีทีในแต่ละแถวของภาพ X จะได้ผลลัพธ์สองภาพย่อย X และ X ดังแสดงไว้ในรูป 2.3 (ข) ที่มีค่าข้อมูลภาพทั้งสองเป็น

$$X^{E}(m,n) = X(m,2n) \tag{2-15}$$

และ

$$X^{0}(m,n) = X(m,2n+1) - \sum_{k=-q}^{q-1} h_h(k) X(m,2n-2k) \tag{2-16}$$



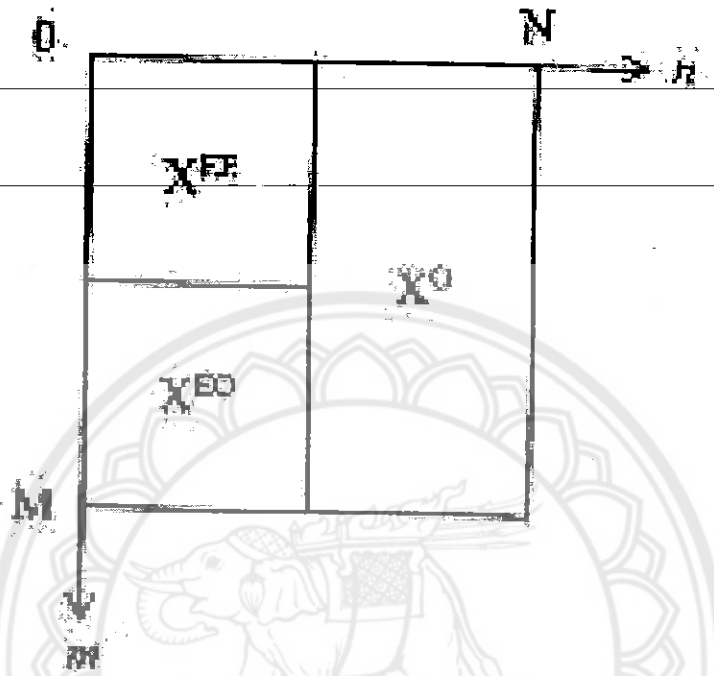
รูป 2.3 พื้นฐานการแยกองค์ประกอบภาพโดยใช้วิธีแอลดีทีตามแนวนอน

ต่อจากนั้นทำการแปลงรูปแอลดีทีในแต่ละหลักของภาพ X^{E} ที่ได้จากขั้นตอนแรก จะพบว่าการแยกองค์ประกอบของภาพออกเป็นสามภาพย่อยดังแสดงไว้ในรูป 2.4 มีค่าข้อมูลภาพย่อย X^{EE} และ X^{EO} มีค่าเป็น

$$X^{EE}(m,n) = X(2m,2n) \tag{2-17}$$

และ

$$X^{EO}(m,n) = X(2m+1,2n) - \sum_{k=-q}^{q-1} h_h(k)X(2m-2k,2n) \qquad (2\text{-}18)$$



รูป 2.4 พื้นฐานการแยกองค์ประกอบภาพโดยใช้วิธีแอลดีที

ถ้ากำหนดให้ $T_{2D}(\bullet)$ เป็นสัญญากรการแปลงแอลดีที่สำหรับภาพสองมิติ X แล้วจะพบว่าการแปลงรูปแอลดีทีสำหรับภาพสองมิติมีค่าเป็น

$$\left[X^{EE}, X^{EO}, X^{O}\right] = T_{2D}(X) \qquad (2\text{-}19)$$

โดยข้อมูลภาพย่อย X^{EE} และ X^{EO} สามารถหาได้จากสมการ (2-17) และ (2-18) ตามลำดับ ส่วนข้อมูลย่อย X^{O} ถูกกำหนดได้ตามความสัมพันธ์ดังสมการ (2-16) เมื่อนำข้อมูลภาพย่อยทั้งสามารถต่อกันดังแสดงไว้ในรูป 2.4 จะเห็นได้ว่าภาพย่อย X^{EE} แสดงถึงข้อมูลภาพ X ที่ได้มาจากการสุ่มตัวอย่างแบบลงด้วยสองช่วงเวลาเท่านั้น สำหรับภาพย่อย X^{O} และ X^{EO} ได้มากจากผลต่างการประมาณค่าซึ่งผ่านตัวกรองสัญญาณหนึ่งมิติ $\{h_h(k)\}$ และ $\{h_v(k)\}$ ตามลำดับการเลือกพารามิเตอร์ของตัวกรองสัญญาณทั้งสองใช้แนวทางเดียวกับที่ได้กล่าวมาแล้วในหัวข้อที่ 2.8

2.10 ส่วนกลับของการแปลงรูปแอลดีทีสำหรับภาพสองมิติ

เมื่อนำข้อมูลภาพย่อย X^{EE} และ X^{EO} แทนลงในสมการ (2-17) และ (2-18) ตามลำดับจะพบว่าข้อมูลภาพดั้งเดิม $X(2m,2n)$ และ $X(2m+1,2n)$ กลับคืนมามีค่าเป็น

$$X(2m,2n) = X^{EE}(m,n) \tag{2-20}$$

และ

$$X(2m+1,2n) = X^{EO}(m,n) + \sum_{k=-q}^{q-1} h_h(k) X^{EE}(m-k,n) \tag{2-21}$$

ต่อมานำข้อมูลภาพย่อย X^{O} แทนลงในสมการ (2-20) จะพบว่าข้อมูลภาพดั้งเดิม $X(m,2n+1)$ กลับคืนมา มีค่าเป็น

$$X(m,2n+1) = X^{O}(m,n) + \sum_{k=-q}^{q-1} h_h(k) X(m,2n-2k) \tag{2-22}$$

ต่อจากนั้นนำข้อมูลภาพดั้งเดิม $\{X(2m,2n)\}$จากสมการ (2-20) และ $\{X(2m+1,2n)\}$ จากสมการ (2-21) ผ่านขบวนการส่วนกลับการแปลงแอลทีดีในแต่ละหลักของภาพ ทำให้ได้ข้อมูลภาพดังเดิม {X (m,2n)} และนำข้อมูลภาพ {(m,2n+1)} มาทำส่วนกลับการแปลงแอลดีทีตามแนวระดับจะทำให้ได้ข้อมูลภาพดั้งเดิม X (m, n) กลับคืนมาเหมือนเดิมทุกประการ ถ้ากำหนดให้ $T_{2D}^{-1}(\bullet)$ เป็นสัญญากรส่วนกลับของการแปลงแอลดีทีสำหรับภาพย่อยสองมิติของ X^{EE}, X^{EO} และ X^{O} แล้วจะพบว่าส่วนกลับการแปลงรูปแอลดีทีสำหรับภาพย่อยสองมิติดังกล่าวมีค่าเป็น

$$X = T_{2D}^{-1}(X^{EE}, X^{EO}, X^{O}) \tag{2-23}$$

โดยที่ข้อมูลภาพย่อย X^{EE} ลงแทนในสมการ (2-20) ทำให้ได้ข้อมูลภาพย่อย X (2m,2n) ส่วนข้อมูลย่อย X^{EO} และ X^{EE} ลงแทนในสมการ (2-21) ทำให้ได้ข้อมูลภาพย่อย X(2m,1,2n) ต่อมาหามูลภาพย่อย X(m,2n) ซึ่งได้มาจากผลรวมของการสุ่มตัวอย่างแบบขึ้นของข้อมูลภาพย่อย X(2m,,2n) และ X(2m,1,2n) ขั้นตอนต่อไปนำข้อมูลย่อย X(m,,2n) กับข้อมูลย่อย X^{O} แทนลงในสมการ (2-22) ทำให้ได้ข้อมูลภาพย่อย X(m,,2n+1) ขั้นตอนสุดท้ายได้รับภาพดั้งเดิม X ซึ่งได้มาจากผลผลรวมของการสุ่มตัวอย่างแบบขึ้นของข้อมูลภาพย่อย X(m, 2n) และ X(m,2n +n) จะเห็นได้ว่าไม่ว่าจะเลือกพารามิเตอร์กรองสัญญาณ $\{h_h(k)\}$ และ $\{h_v(k)\}$ มีค่าเท่าใดก็ตาม ส่วนกลับของการแปลงรูปแอลดีทีสำหรับภาพสองมิติดังแสดงไว้ในสมการ (2-23) สามารถหาค่าได้เสมอ

บทที่ 3
วิธีการดำเนินการทดลอง

การดำเนินการในการทดลองเพื่อลดขนาดข้อมูลภาพโดยใช้แนวคิด LDT มีรายละเอียดและวิธีการดำเนินการดังนี้

3.1 ออกแบบขั้นตอนในการเขียนโปรแกรม

ในการทำการทดลองเพื่อลดขนาดข้อมูลโดยใช้วิธี LDT นั้นผู้ทำการทดลองได้ออกแบบขั้นตอนในการเขียนโปรแกรมเพื่อความเป็นระเบียบและเพื่อง่ายต่อการแก้ไขและปรับปรุงโดยจะแยกการทำงานแต่ละขั้นตอนดังนี้

ขั้นตอนที่ 1

- นำภาพต้นแบบมาเข้ากระบวนการลดขนาดขั้นแรกโดยใช้วิธี LDT

ขั้นตอนที่ 2

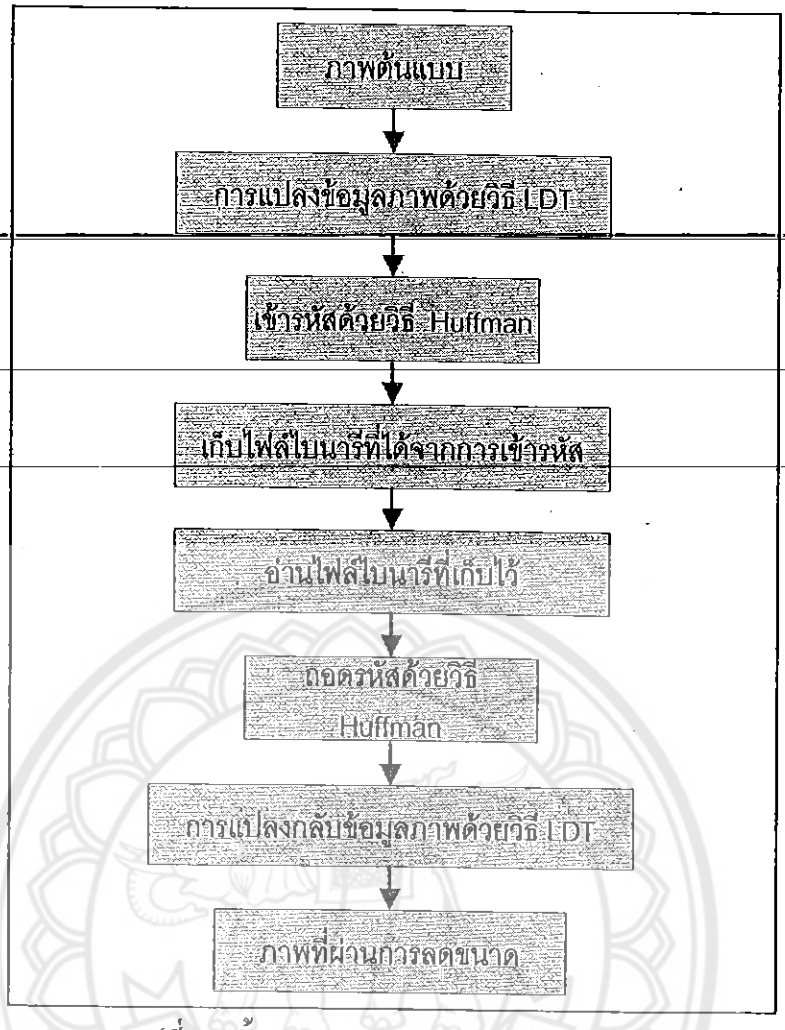
- นำเอาท์พุทที่ได้มาเข้ากระบวนการ Huffman Encoding ซึ่งจะได้ไฟล์ภาพที่ผ่าน การลดขนาด โดยจะนำไปหา Data Compression Ratio

ขั้นตอนที่ 3

- นำภาพที่ผ่านการลดขนาดมาเข้ากระบวนการการแปลงกลับ โดยนำมาเข้า Huffman Decoding

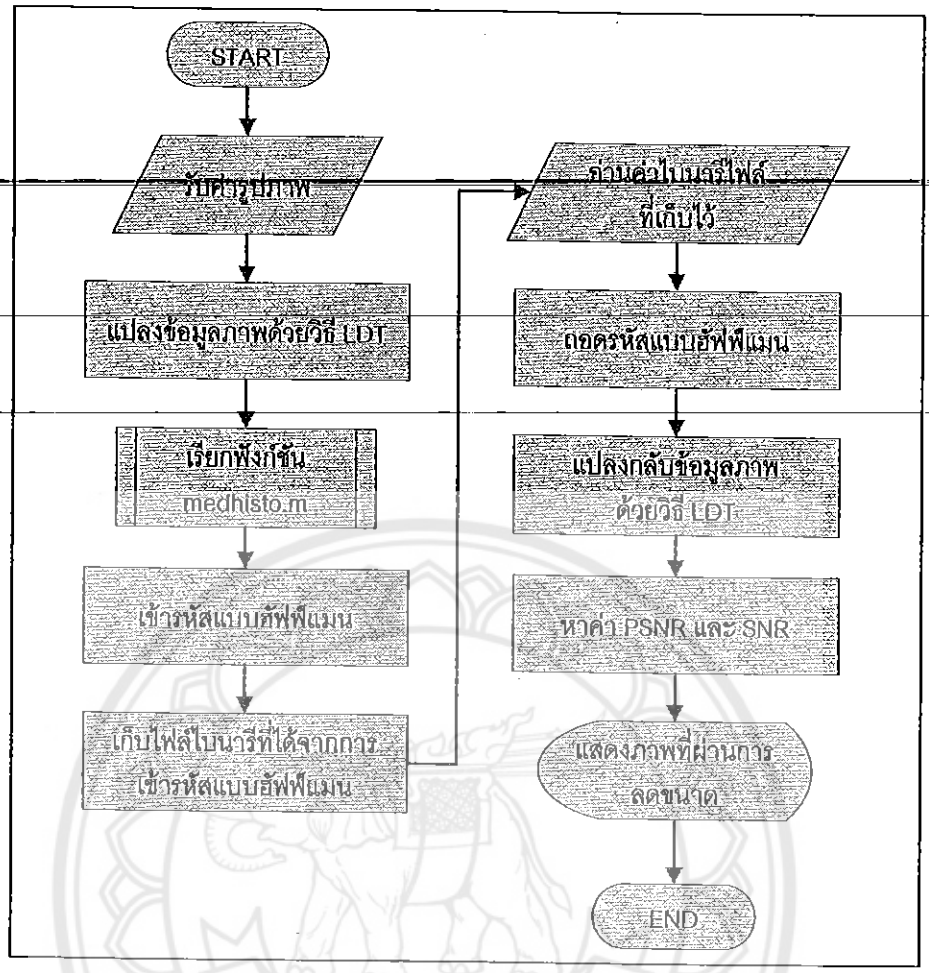
ขั้นตอนที่ 4

- นำมาเข้า LDT^{-1} ซึ่งจะได้ภาพที่มีลักษณะเหมือนภาพต้นแบบโดยมีความผิดเพี้ยนเกิดขึ้นเล็กน้อย โดยจะมีการหา PSNR และ SNR เพื่อเปรียบเทียบภาพต้นแบบกับภาพที่ผ่านการลดขนาดและนำมาแปลงกลับ



รูปที่ 3.1 ขั้นตอนการทำงานของโปรแกรม

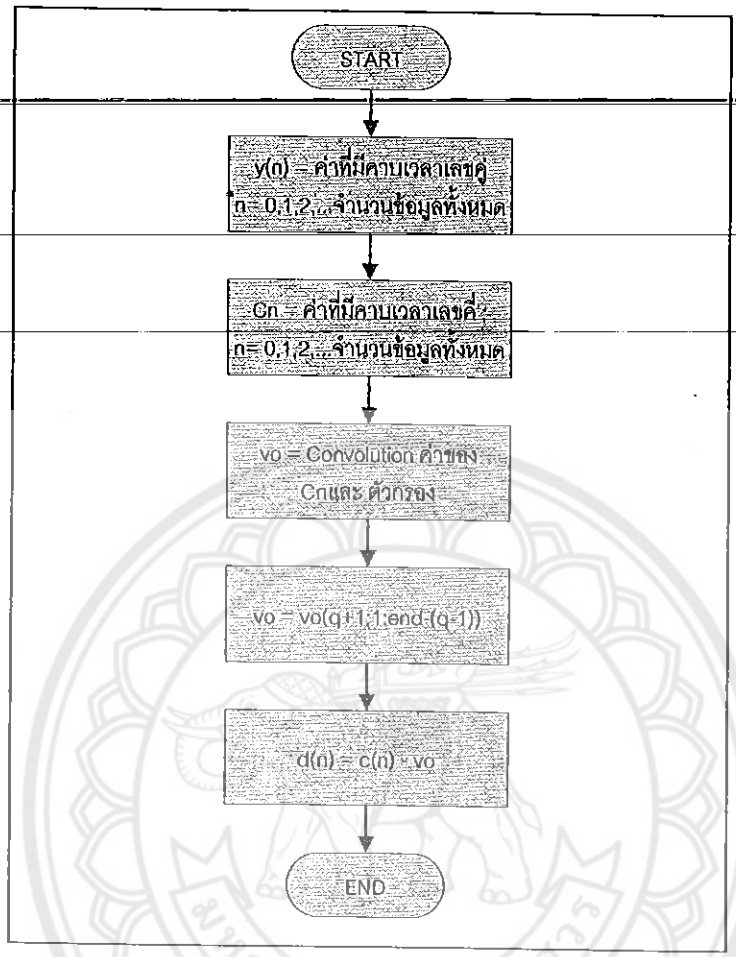
จากรูปที่ 3.1 เป็นขั้นตอนการเขียนโปรแกรมเพื่อลดขนาดข้อมูลภาพโดยเริ่มแรกจะนำภาพต้นแบบมาแปลงข้อมูลด้วยวิธีแอลดีทีจากนั้นจะนำภาพที่ผ่านการแปลงข้อมูลมาทำการเข้ารหัสด้วยวิธี ฮัฟฟ์แมนโดยผลที่ได้จะเป็นข้อมูลแบบไบนารีซึ่งจะถูกนำมาเก็บเป็นไฟล์ที่ถูกลดขนาดแล้ว จากนั้นจะนำไฟล์ไบนารีที่เก็บนั้นมาทำการถอดรหัสด้วยวิธีฮัฟฟ์แมนแล้วจึงนำมาแปลงผกผันกลับด้วยวิธีแอลดีทีก็จะได้ภาพที่ผ่านการลดขนาดและแปลงกลับมาให้ได้ภาพที่มีลักษณะเหมือนภาพต้นฉบับที่มีความผิดเพี้ยนไปจากภาพเดิมเล็กน้อย



รูปที่ 3.2 flow chart ขั้นตอนการทำงานของโปรแกรม

จากรูปที่ 3.2 เป็นการแสดงถึงขั้นตอนการทำงานของโปรแกรมเพื่อลดขนาดข้อมูลภาพด้วยวิธีแอลดีที โดยเริ่มจากโปรแกรมรับค่าที่อ่านข้อมูลของรูปภาพเข้ามาและนำข้อมูลนั้นเข้าแปลงด้วยวิธีแอลดีจากนั้นจะทำการหาค่าความถี่ของค่าที่เหมือนกันของข้อมูลด้วยการเรียกฟังก์ชัน medhisto.m เพื่อเป็นการเตรียมข้อมูลเพื่อจะทำการเข้ารหัสฮัฟฟ์แมนซึ่งจะได้ไฟล์ที่เป็นไบนารีไฟล์ จากนั้นโปรแกรมจะอ่านค่าไบนารีไฟล์นั้นเพื่อทำการถอดรหัสฮัฟฟ์แมนจากนั้นจะทำการแปลงผกผันกลับด้วยวิธีแอลดีที จากนั้นโปรแกรมจะนำค่าที่ได้ไปคำนวณหาค่า PSNR และ SNR จากนั้นจะนำค่าที่ได้แสดงออกทางจอภาพพร้อมกับภาพต้นฉบับและภาพที่ทำการแปลงกลับของภาพที่ผ่านการลดขนาด ซึ่งรายละเอียดการทำงานของฟังก์ชันต่างๆที่ใช้ในการลดขนาดข้อมูลภาพนั้นจะแสดงรายละเอียดในส่วนของการเขียนฟังก์ชันนั้นๆ

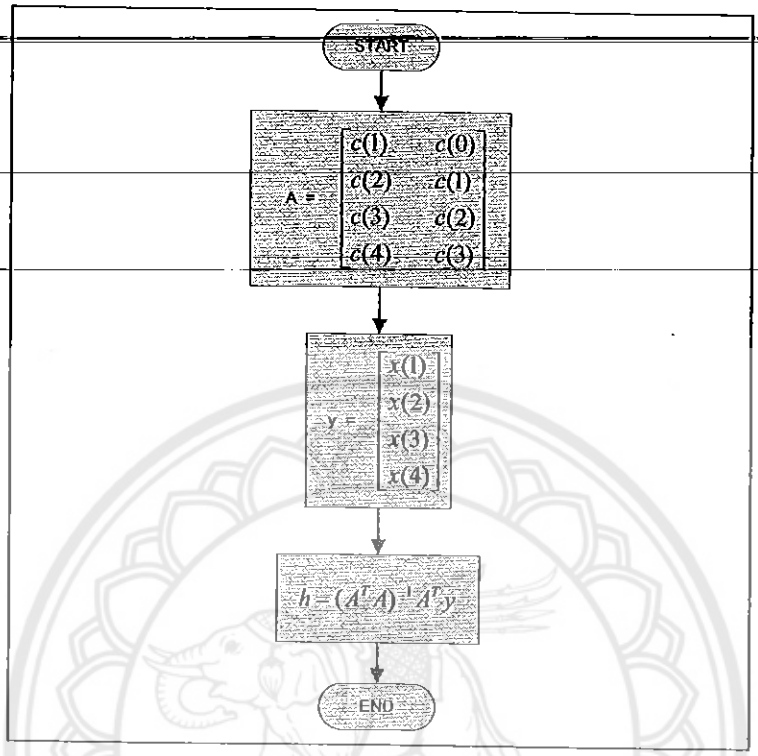
ฟังก์ชัน pkldt.m เป็นฟังก์ชันสำหรับการแยกองค์ประกอบของภาพด้วยวิธี LDTแบบ 1 มิติ



รูปที่ 3.3 flow chart การแยกองค์ประกอบด้วยวิธี LDT แบบ 1 มิติ

จากรูปที่ 3.3 ฟังก์ชัน pkldt.m เป็นฟังก์ชันการแยกองค์ประกอบด้วยวิธีแอลดีทีแบบ 1 มิติ คือฟังก์ชันจะอ่านข้อมูลทั้งหมดและทำการเก็บค่าข้อมูลในตำแหน่งคี่ไว้ในตัวแปล c(n) โดย n เป็นจำนวนเต็มตั้งแต่ 0 ถึง (จำนวนข้อมูลทั้งหมด/2)-1 ซึ่งตัวแปร c(n) เป็นตัวแปรแบบอาเรย์ จะถูกกำหนดให้เป็นตัวแปรที่แทนค่าสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้าและและฟังก์ชันจะหาค่า d(n) ซึ่งคือสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่โดยนำค่าตัวแปรc(n) ลบด้วยค่า convolution ของตัวแปร c(n) กับ ตัวกรองสัญญาณที่หาได้จากฟังก์ชัน genH.m ซึ่งรายละเอียดของฟังก์ชัน genH.m จะได้กล่าวต่อไป

ไฟล์ genH.m เป็นฟังก์ชันที่หาค่าพารามิเตอร์ของตัวกรองสัญญาณที่ใช้สำหรับการลดขนาดข้อมูลภาพด้วยวิธี LDT



รูปที่ 3.4 flow chart การหาฟิลเตอร์ของการแยกองค์ประกอบด้วยวิธี LDT

ฟังก์ชัน genH.m จะเป็นฟังก์ชันที่คำนวณหาค่าพารามิเตอร์ของตัวกรองสัญญาณ โดยจะเป็นการหาค่าตามสมการ $h = (A^T A)^{-1} A^T y$ โดยตัวแปรต่างๆมีรายละเอียดดังนี้

ให้ X(N) แทนข้อมูลทั้งหมด , N คือจำนวนข้อมูลทั้งหมด และ n คือจำนวน 0 ถึง(N/2)-1 ค่าพารามิเตอร์ที่ต้องการนั้นเป็นค่าที่ต้องการนำมา convolution กับข้อมูล c(n) เพื่อที่จะให้ค่าผลต่างของ c(n) และ c(n) นั้นเข้าใกล้ 0 มากที่สุด จากสมการ (2-6)

$$d(n) = x(2n+1) - \sum_{k=-q}^{q-1} h(k)x(2n-2k) \text{ สำหรับ } n = 0, 1,\ldots, (N/2)-1$$

จากสมการสามารถแจกแจงได้ดังนี้

n=0 ; $d(0) = X(1) - \{h(-1)c(1) + h(0)c(0)\}$

n=1 ; $d(1) = X(3) - \{h(-1)c(2) + h(0)c(1)\}$

n=2 ; $d(2) = X(5) - \{h(-1)c(3) + h(0)c(2)\}$

.

.

n=(N/2)-1 ; $d(n) = X(2n+1) - \{h(-1)c(n+1) + h(0)c(n)\}$

ซึ่งเราจะสามารถเขียนให้อยู่ในรูปเมตริกซ์ได้ดังนี้

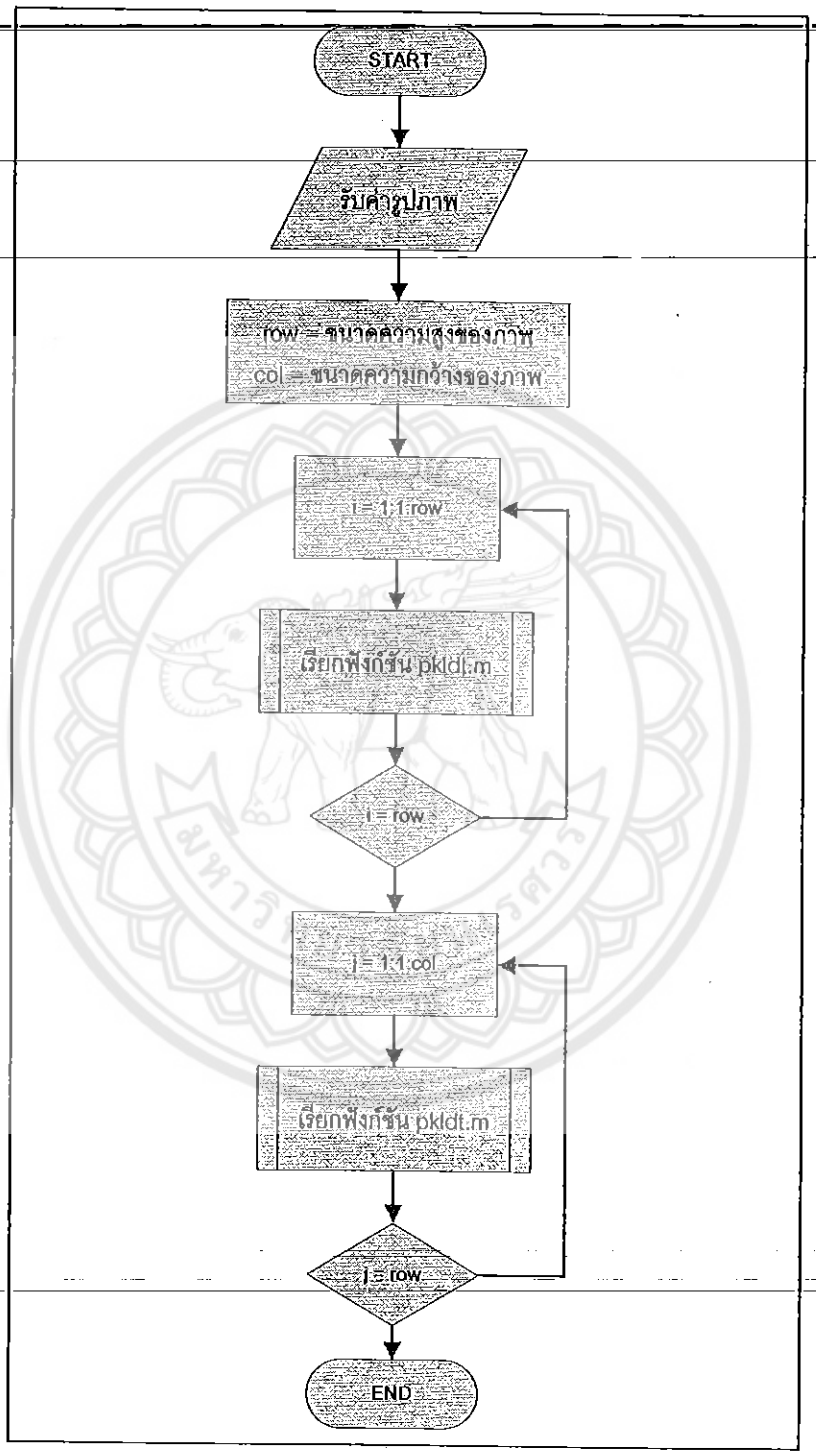
$$\begin{bmatrix} d(0) \\ d(1) \\ d(2) \\ . \\ . \\ d(n) \end{bmatrix} = \begin{bmatrix} X(1) \\ X(3) \\ X(5) \\ . \\ . \\ X(2n+1) \end{bmatrix} - \begin{bmatrix} h(-1)c(1) + h(0)c(0) \\ h(-1)c(2) + h(0)c(1) \\ h(-1)c(3) + h(0)c(2) \\ . \\ . \\ h(-1)c(n+1) + h(0)c(n) \end{bmatrix}$$

$$= \begin{bmatrix} x(1) \\ x(3) \\ x(5) \\ . \\ . \\ x(2n+1) \end{bmatrix} - \begin{bmatrix} c(1) & c(1) \\ c(2) & c(2) \\ c(3) & c(3) \\ . & . \\ . & . \\ c(n+1) & c(n) \end{bmatrix} \begin{bmatrix} h(-1) \\ h(0) \end{bmatrix}$$

ถ้าให้ $d = \begin{bmatrix} d(0) \\ d(1) \\ d(2) \\ . \\ . \\ d(n) \end{bmatrix}$, $y = \begin{bmatrix} X(1) \\ X(3) \\ X(5) \\ . \\ . \\ X(2n+1) \end{bmatrix}$, $A = \begin{bmatrix} c(1) & c(1) \\ c(2) & c(2) \\ c(3) & c(3) \\ . & . \\ . & . \\ c(n+1) & c(n) \end{bmatrix}$, $h = \begin{bmatrix} h(-1) \\ h(0) \end{bmatrix}$

ซึ่งโปรแกรมจะนำค่าตัวแปรเหล่านี้แทนค่าลงในสมการ $h = (A^TA)^{-1} A^Ty$ เพื่อคำนวณหาค่าพารามิเตอร์ของตัวกรองสัญญาณต่อไป

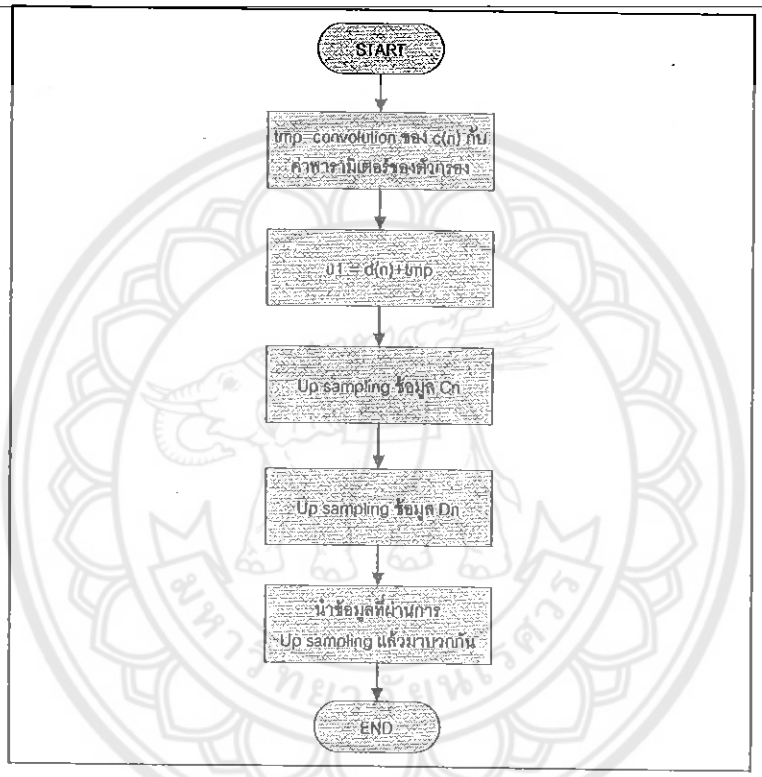
ไฟล์ pkld2d.m เป็นฟังก์ชันแยกองค์ประกอบของภาพด้วยวิธี LDT โดยเป็นการแยกองค์ประกอบแบบ 2 มิติ



รูปที่ 3.5 flow chart การแยกองค์ประกอบด้วยวิธี LDT แบบ 2 มิติ

จากรูปที่ 3.5 ฟังก์ชัน pkld2d.m เป็นฟังก์ชันการแปลงแอลดีทีสองมิติ โดยฟังก์ชันนี้จะรับค่าข้อมูลภาพเข้ามาแล้วหาขนาดของภาพ โดยตัวแปร row = ขนาดความสูงของภาพ ตัวแปร col = ขนาดความกว้างของภาพ จากนั้นฟังก์ชันนี้จะเรียกใช้ฟังก์ชัน pkldt.m เพื่อทำการแยกองค์ประกอบด้วยวิธี LDT ตามแถวทีละแถวจนครบทุกแถวและจากนั้นจะเรียกฟังก์ชัน pkldt.m อีกครั้งเพื่อทำการแยกองค์ประกอบด้วยวิธี LDT ตามหลักทีละหลักจนครบ

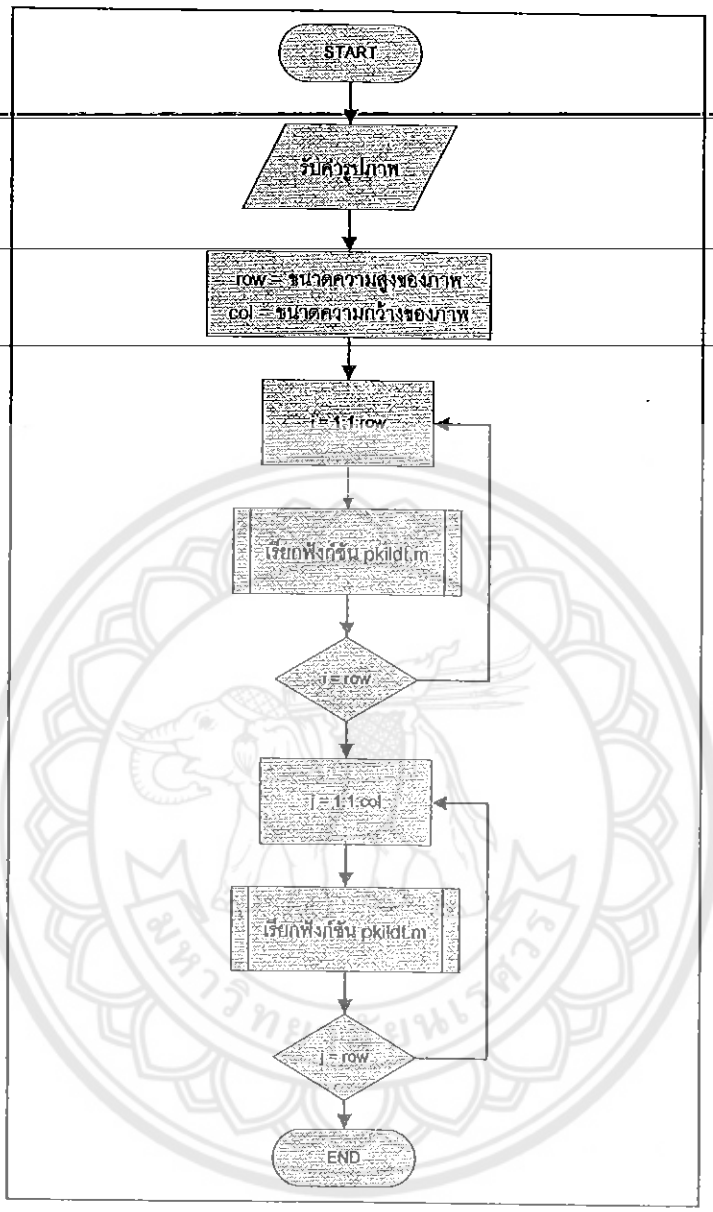
ไฟล์ pkildt.m เป็นการแปลงกลับด้วยวิธี LDT แบบ 1 มิติ



รูปที่ 3.6 flow chart การการแปลงกลับของวิธี LDT แบบ 1 มิติ

จากรูปที่ 3.6 เป็นการทำงานของฟังก์ชัน pkildt.m โดยฟังก์ชันจะอ่านค่า c(n) , d(n) และค่าพารามิเตอร์ของตัวกรองสัญญาณที่ได้จากการแปลงแอลดีทีและจะทำการ convolution ค่าตัวแปร c(n) กับ ค่าพารามิเตอร์ของตัวกรอง ซึ่งค่าที่ได้นั้นจะนำมาบวกกับ d(n) และจะทำการ Up sampling c(n) และ d(n) และจะนำค่าทั้งสองมาบวกกัน

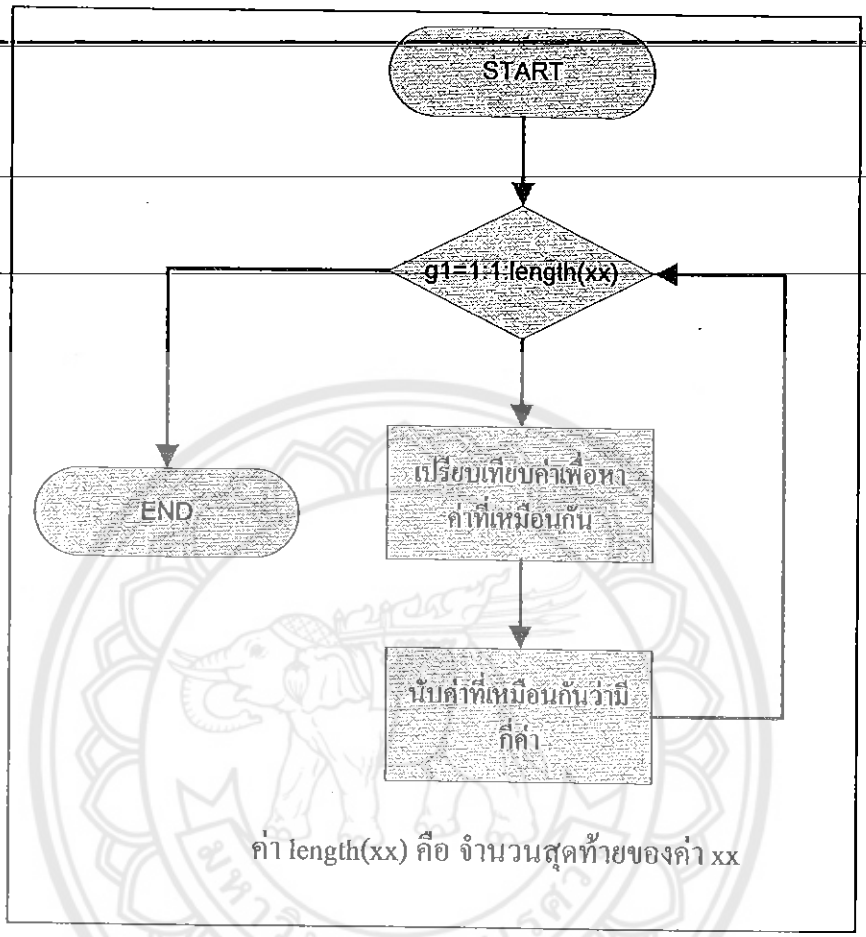
ไฟล์ pkildt2d.m เป็นการแปลงกลับด้วยวิธี LDT แบบ 2 มิติ



รูปที่ 3.7 flow chart การแปลงกลับของวิธี LDT แบบ 2 มิติ

จากรูปที่ 3.7 ฟังก์ชัน pkild2d.m เป็นฟังก์ชันการแปลงผกผันของแอลดีทีสองมิติ โดยฟังก์ชันนี้จะรับค่าข้อมูลภาพเข้ามาแล้วหาขนาดของภาพ โดยตัวแปร row = ขนาดความสูงของภาพ ตัวแปร col = ขนาดความกว้างของภาพ จากนั้นฟังก์ชันนี้จะเรียกใช้ฟังก์ชัน pkildt.m เพื่อทำการแปลงกลับด้วยวิธี LDT ตามหลักทีละหลักจนครบทุกหลักและจากนั้นจะเรียกฟังก์ชัน pkildt.m อีกครั้งเพื่อทำการแปลงกลับด้วยวิธี LDT ตามแถวทีละแถวจนครบ

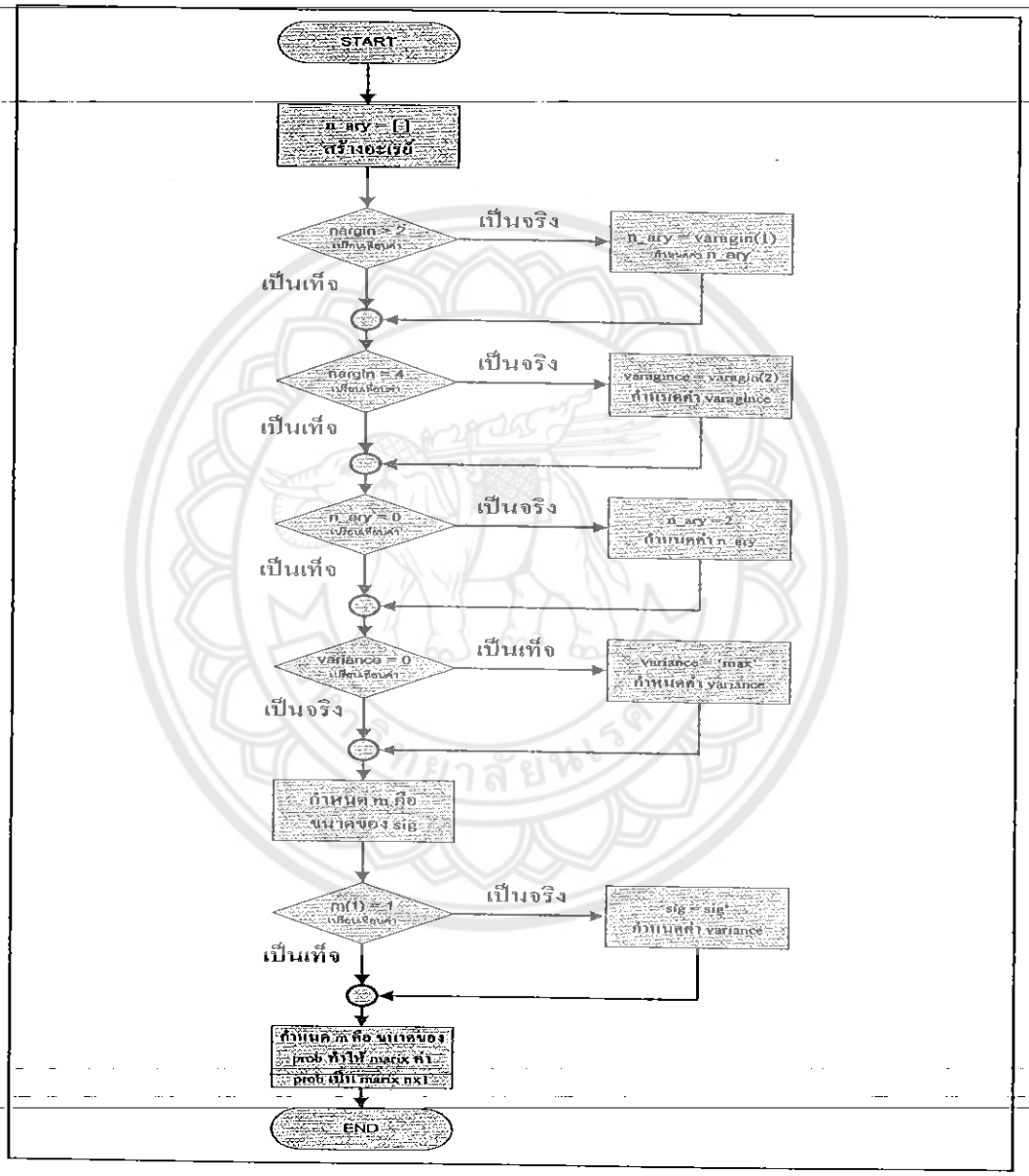
ฟังก์ชั่น medhisto.m เป็นฟังก์ชั่นที่ทำหน้าที่ในการหาค่าความถี่ของข้อมูล (สามารถดูฟังก์ชั่นได้จากภาคผนวก)



รูปที่ 3.8 Flowchart ฟังก์ชั่นการหาค่าความถี่ของข้อมูล

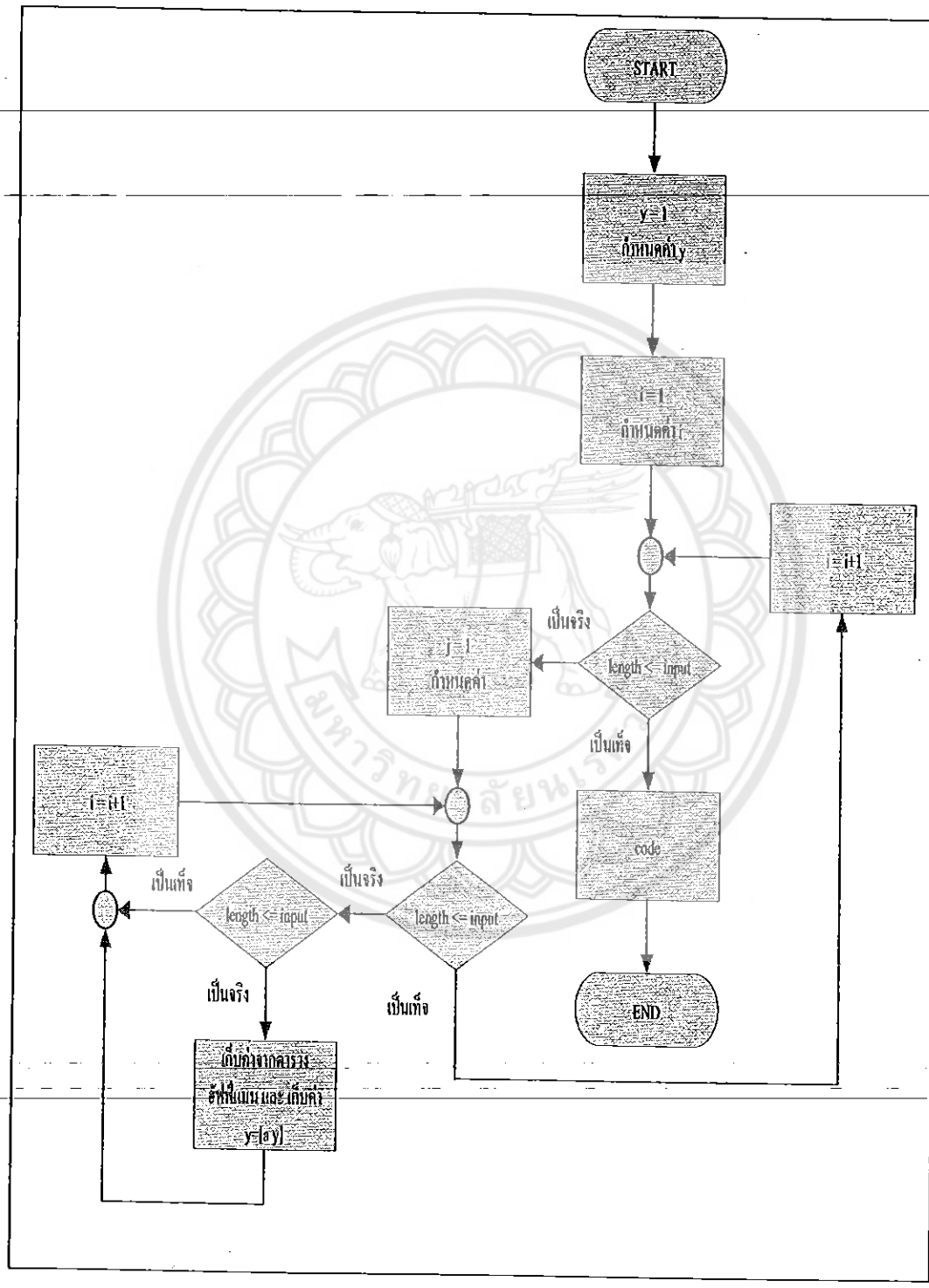
จากรูปที่ 3.8 ฟังก์ชัน medhisto.m จะทำงานโดยรับค่าข้อมูลเข้ามาแล้วทำการเปรียบเทียบข้อมูลทีละค่าเพื่อหาว่าข้อมูลที่มีค่าซ้ำกันมีอะไรบ้างและมีจำนวนเท่าไร

ฟังก์ชั่น huffmantable.m เป็นฟังก์ชั่นที่ทำหน้าที่ในการสร้างตารางรหัส ตามวิธีของการเข้ารหัสแบบฮัฟฟ์แมน โดยจะเริ่มจากการสร้างแผนภูมิฮัฟฟ์แมนก่อน ทำให้ได้โครงสร้างของแผนภูมิฮัฟฟ์แมน โดยเก็บไว้ในตัวแปรอาเรย์ จากนั้นจะทำการแปลงแผนภูมิฮัฟฟ์แมนให้เป็นรหัสฮัฟฟ์แมนโดยจะทำการเก็บไว้ในตัวแปรอาเรย์ เพื่อใช้ในการเข้ารหัสแบบฮัฟฟ์แมนและถอดรหัสแบบฮัฟฟ์แมนต่อไป



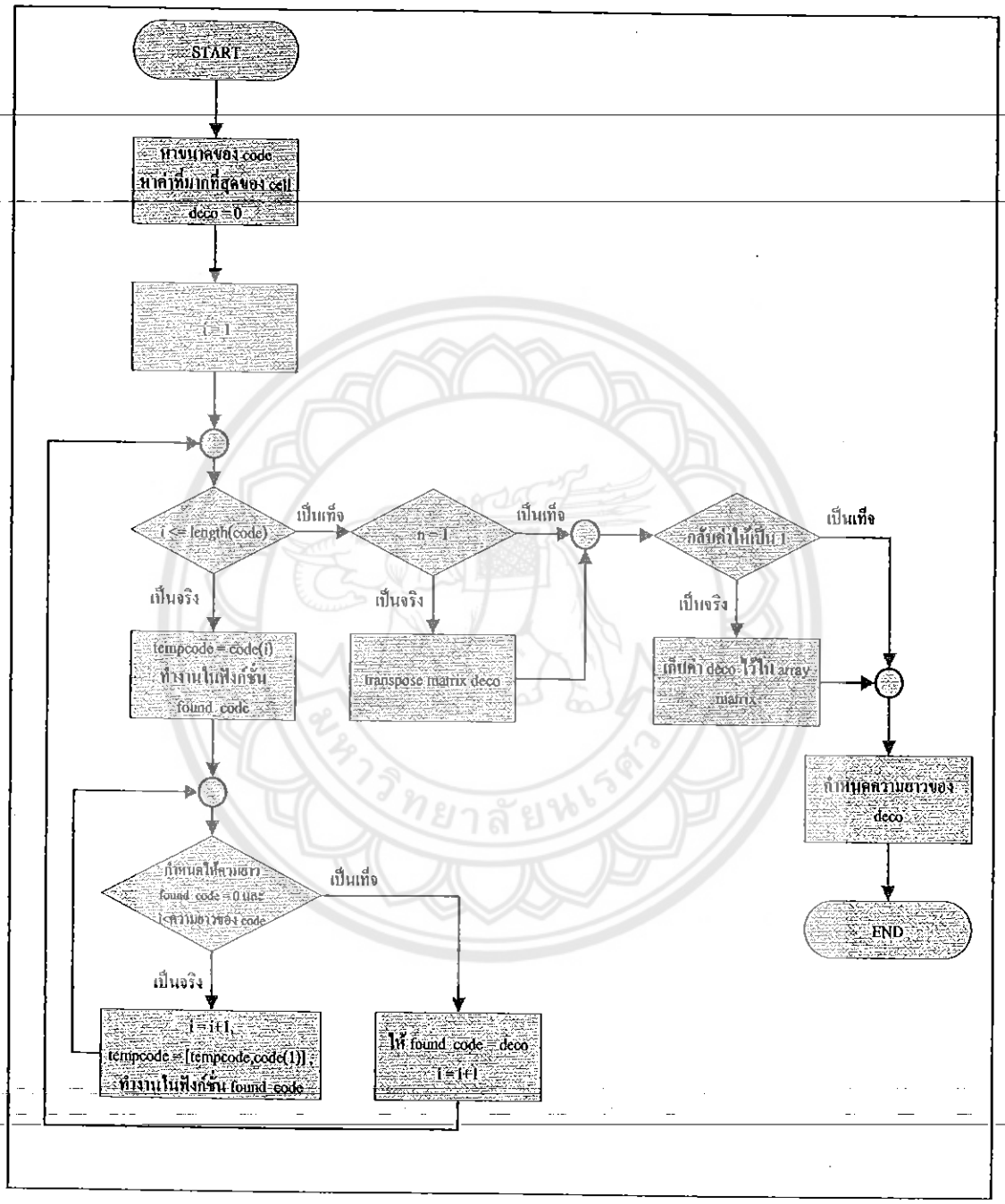
รูปที่ 3.9 Flowchart ฟังก์ชั่นสร้างตารางรหัสฮัฟฟ์แมน

ฟังก์ชั่น encodehuffman.m เป็นทำหน้าที่ในการเขียนข้อมูลที่ละบิตโดยจะรวมให้ครบ 8 บิตก่อนแล้วจึงทำการเขียน เนื่องจากรหัสฮัฟฟ์แมนมีความยาวของบิตที่แปรแปลี่ยนได้หรือมีความยาวของจำนวนบิตที่ไม่เท่ากัน

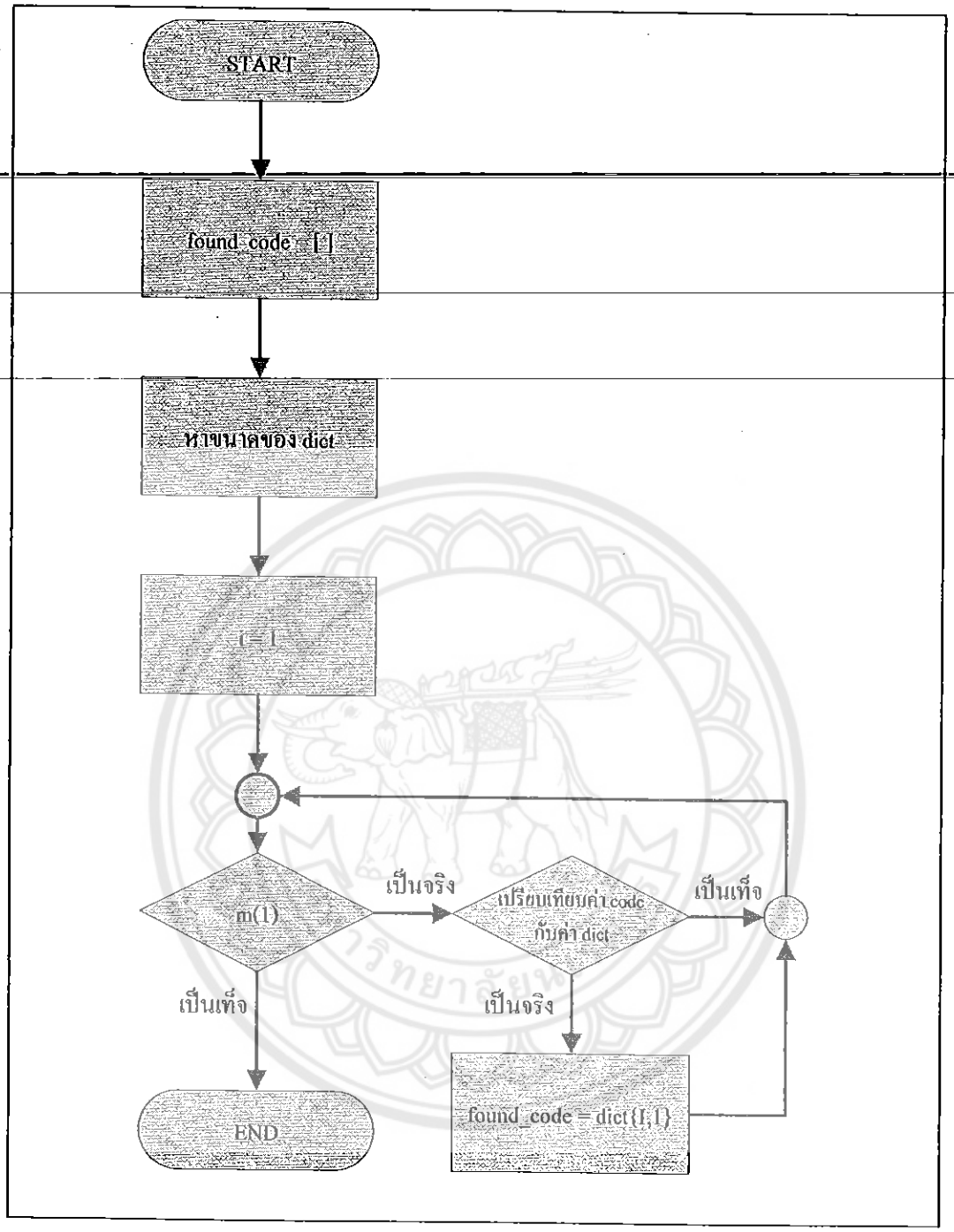


รูปที่ 3.10 Flowchart ฟังก์ชั่นการเข้ารหัสแบบฮัฟฟ์แมน

ฟังก์ชั่น decodehuffman.m เป็นทำหน้าที่ในการแปลงรหัสที่เข้ารหัสไว้แล้ว เทียบกับตารางฮัฟฟ์แมนและจะได้ค่าที่ถอดรหัสออกมา



รูปที่ 3.11 Flowchart ฟังก์ชั่นการถอดรหัสแบบฮัฟฟ์แมน



รูปที่ 3.12 Flowchart ฟังก์ชั่นการถอดรหัสแบบฮัฟฟ์แมน (ต่อ)

บทที่ 4

ผลการทดลอง

4.1 บทนำ

ในบทนี้จะกล่าวถึงการนำหลักการทฤษฎีพื้นฐานและขั้นตอนการดำเนินการที่กล่าวมาแล้วในบทที่ 2 และ 3 มาใช้ในการทดลองลดขนาดข้อมูลภาพ โดยมีรายละเอียดการแปลง LDT ของภาพและผลอัตราการบีบอัดข้อมูลภาพที่ได้ ค่า PSNR และ SNR ที่แสดงถึงคุณภาพของภาพที่ผ่านการลดขนาด

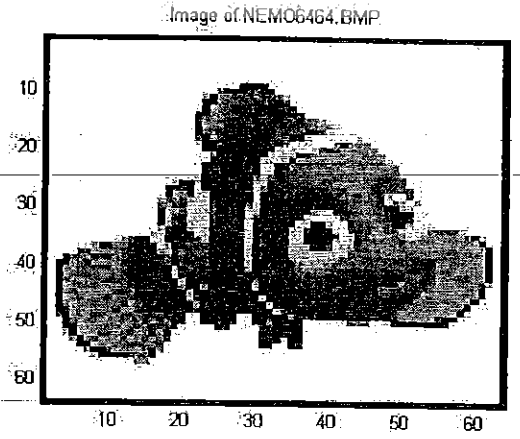
ขั้นตอนในการทดลองที่ทางผู้จัดทำพัฒนาขึ้นนี้ จะทำโดยใช้โปรแกรม Matlab® V7.0.0.1 ในบทนี้จะแสดงเฉพาะสมการคณิตศาสตร์ ขบวนการที่ใช้ในการทดลองและผลการทดลองที่ได้ ส่วนรายละเอียดของโปรแกรมจะแสดงไว้ภายในภาคผนวก

4.2 การแปลงและแปลงผกผันด้วยวิธี LDT

ในหัวข้อนี้ จะเป็นการแสดงการแปลงและแปลงผกผันด้วยวิธี LDT การทดลองในส่วนนี้จะทำโดยใช้โปรแกรม Matlab® V7.0.0.1 เป็นเครื่องมือ ซึ่งการทดลองจะแสดงให้เห็นถึงกระบวนการแปลง LDT ข้อมูลภาพและแสดงค่าคุณสมบัติของภาพที่ผ่านการลดขนาด

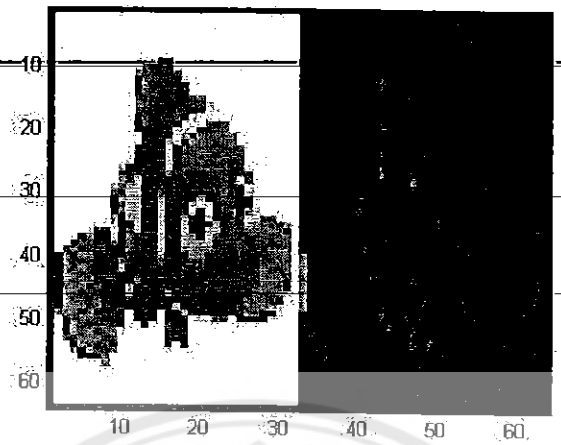
4.2.1 การแปลงข้อมูลภาพด้วยวิธี LDT

ข้อมูลภาพเป็นข้อมูลของสัญญาณที่อยู่ในสองมิติ คือ ในแกนของระยะในแนวนอน (X) และระยะในแกนตั้ง (Y) สำหรับข้อมูลภาพในโปรแกรม Matlab V 7.0.0.1 จะถูกเก็บอยู่ในรูปของเมตริกซ์สองมิติ จากภาพต้นแบบ ขนาด 64X64 Pixel ดังรูปที่ 4.1



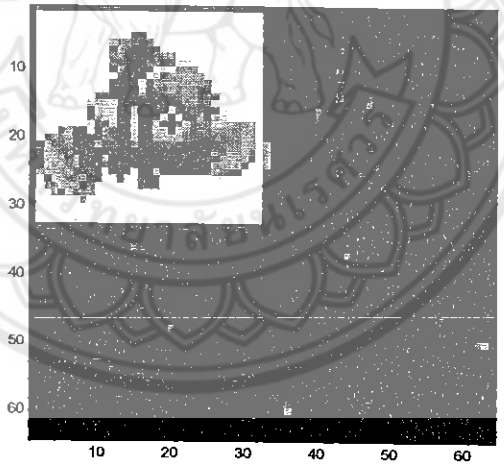
รูปที่ 4.1 ภาพต้นแบบขนาด 64X64 Pixel

การแปลงข้อมูลภาพด้วยวิธี LDT นั้นจะทำการแยกองค์ประกอบของภาพโดยเลือกข้อมูลของแต่ละแถวตามแนวนอนมาผ่านการแปลง LDT จนครบทุกแถว ดังแสดงในรูปที่ 4.2



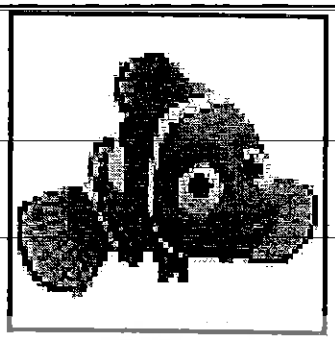
รูปที่ 4.2 ภาพที่ได้จากการแปลง LDT ของเส้นภาพตามแนวนอน

จากนั้นข้อมูลที่ผ่านการแยกองค์ประกอบตามแนวนอนแล้ว จะผ่านการแยกองค์ประกอบด้วยวิธี LDT ตามแนวตั้งอีกครั้ง จึงได้ผลลัพธ์ดังรูปที่ 4.3



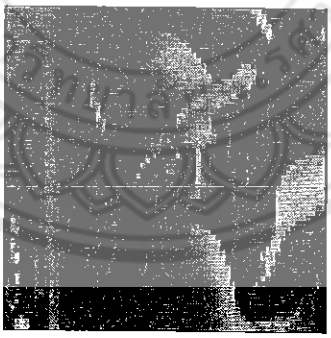
รูปที่ 4.3 ภาพที่ผ่านการแปลง LDT ของเส้นภาพตามแนวนอนและแนวตั้ง

ในขั้นตอนการทดลองต่อไปนี้ ทางผู้จัดทำได้ใช้ภาพต้นแบบคือ NEMO6464.BMP ขนาด 5176 bytes , EYE6464.TIF ขนาด 5486 bytes และ LENA6464.BMP ขนาด 5176 bytes ซึ่งมีความละเอียดของภาพเท่ากันคือ 64X64 Pixel ดังแสดงในรูปที่ 4.4 , 4.5 และ 4.6 ตามลำดับ

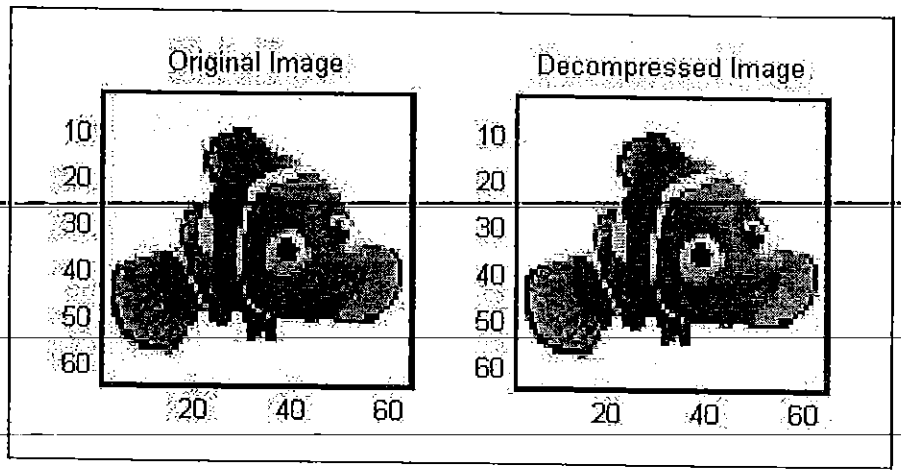


รูปที่ 4.4 ภาพต้นฉบับ NEMO6464.BMP ความละเอียด 64X64 Pixel

รูปที่ 4.5 แสดงภาพต้นฉบับ EYE6464.TIF ความละเอียด 64X64 Pixel

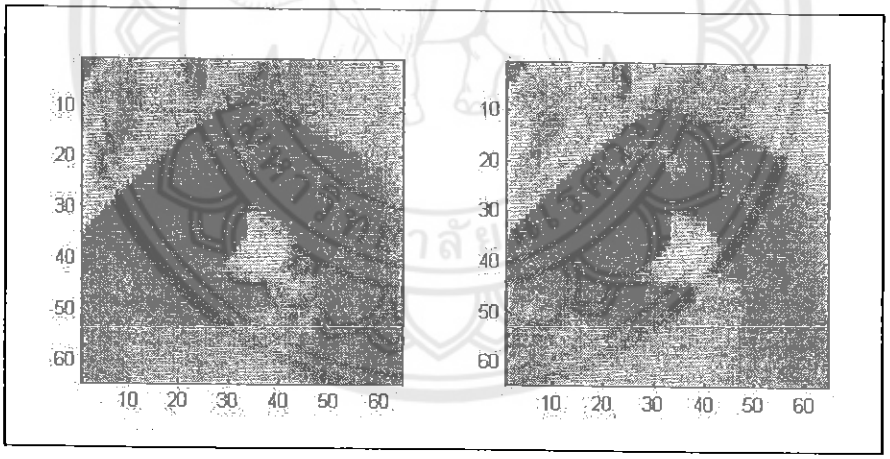


รูปที่ 4.6 แสดงภาพต้นฉบับ LENA6464.BMP ความละเอียด 64X64 Pixel



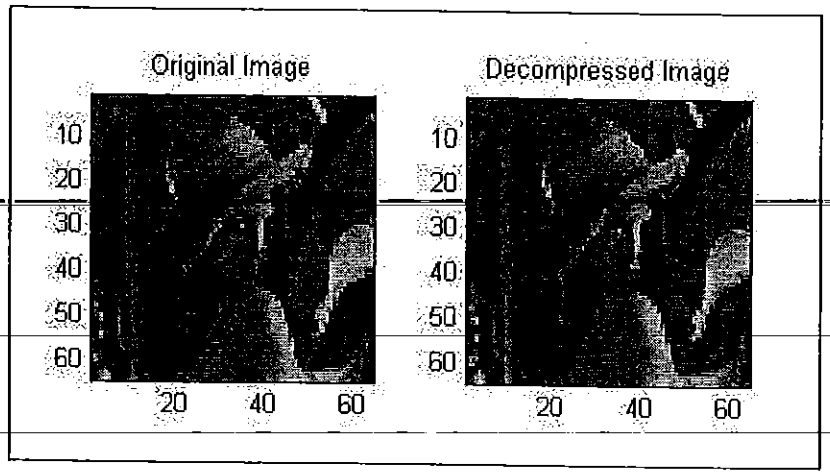
รูปที่ 4.7 แสดงภาพ NEMO (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.7 แสดงถึง: ภาพต้นฉบับ NEMO.BMP ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5176 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.8 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 28.2 SNR = 62.1 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน



รูปที่ 4.8 แสดงภาพ EYE (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.8 แสดงถึง: ภาพต้นฉบับ EYE.TIF ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5486 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.9 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 42.4 SNR = 77.2 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน



รูปที่ 4.9 แสดงภาพ LENA (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.9 แสดงถึง: ภาพต้นฉบับ LENA.BMP ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5176 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.5 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 41.5 SNR = 74.9 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน

จากผลที่ได้ของการบีบอัดข้อมูลภาพตามขั้นตอนที่ได้กล่าวมาแล้วนั้นแสดงดังรูปที่ 4.7 , รูปที่ 4.8 และรูปที่ 4.9 สามารถเขียนเป็นตารางที่ 4.1 ได้ดังนี้

ตารางที่ 4.1 แสดงค่า CR, SNR, PSNR ของภาพที่ทำการทดลองทั้งหมด

ภาพที่ทดสอบ	CR	SNR (dB)	PSNR (dB)
ภาพ NEMO ขนาด 64x64 pixel	1.8	62.1	28.2
ภาพ EYE ขนาด 64x64 pixel	1.9	77.2	42.4
ภาพ lena ขนาด 64x64 pixel	1.5	74.9	41.5

บทที่ 5
สรุปผลการทดลอง

จากการทดลองเพื่อลดขนาดข้อมูลภาพโดยใช้ฟังก์ชันการแยกองค์ประกอบตามวิธี LDT ฟังก์ชันการเข้ารหัสและถอดรหัสแบบฮัฟฟ์แมน ฟังก์ชันการแปลงกลับของ LDT กับภาพต้นแบบจำนวน 3 ภาพ คือ ภาพนีโม (NEMO6464.BMP) , ภาพตา (EYE6464.TIF) และภาพลีน่า (LENA6464.BMP) ซึ่งมีความละเอียด 64X64 พิกเซล โดยใช้วิธี LDT ตามที่ได้อธิบายไว้ในวิธีการดำเนินการทดลองในบทที่ 3 พบว่าอัตราการบีบอัดข้อมูลภาพของภาพนีโมเท่ากับ 1.8 ภาพตาเท่ากับ 1.9 และภาพลีน่าเท่ากับ 1.5 นอกจากนี้เมื่อทำการคลายการบีบอัดภาพทั้ง 3 พบว่าภาพตามีคุณภาพดีที่สุดเนื่องจากมีค่า PSNR และ SNR สูงสุด แต่อย่างไรก็ตามภาพทั้งสามนั้นไม่มีคุณลักษณะเหมือนภาพต้นแบบเนื่องจากภาพทั้งสามหลังจากถูกแปลงรูปโดย LDT แล้วนำค่าสัมประสิทธิ์ของ LDT ไปทำให้เป็นจำนวนเต็มก่อนเข้ารหัสแบบฮัฟฟ์แมน

เอกสารอ้างอิง

[1] James A.Cadzaw and Suchart Yammen. Data Adaptive Linear Decomposition Transform. Vanderbilt University:2002

[2] รศ.ดร. มนัส สังวรศิลป์. คู่มือการใช้งาน MATLAB ฉบับสมบูรณ์. นนทบุรี: สำนักพิมพ์อินโฟเพรส. 2543.

ภาคผนวก
โปรแกรมการบีบอัดภาพด้วยวิธี LDT
ฟังก์ชั่นที่ใช้ในโปรแกรม

โปรแกรมการบีบอัดข้อมูลภาพด้วยวิธี LDT

```
%=========================================================
%  การบีบอัดข้อมูลภาพโดยใช้วิธี  LDT
%  Image Compresstion Using by LDT
%
%  File name : hufftest.m
%  By : Mr.Krichakon Boonruang          ID 44362515
%       Mr.Puripong  Dokkiang           ID 44362739
%=========================================================
clear all
close all
nsig = double(imread('nemo6464.bmp','bmp'));     % การับค่าจากไฟล์ข้อมูลภาพ
%=========================================================
[QQ,hh,hv]=pkldt2d(nsig,2);                      % เรียกใช้ฟังก์ชัน LDT
[row,col] = size(nsig);                          % อ่านค่าขนาดของไฟล์ภาพ
%=========================================================
nesig = QQ(:);
[M N] = size(nsig);
input = nesig;
new = reshape(input,1,M*N);
sig = round(new)+512;
histo = medhisto(sig);
[I J V] = find(histo);
p = V/sum(V);                    %หาความน่าจะเป็น
symbols = J;
actualsig = sig;
dict = huffmantable(symbols,p);  % สร้างตารางฮัฟฟ์แมน
% Encode of Huffman code=================================================
input = actualsig;
code = encodehuffman(input,dict);
```

```
% Save Files=============================================================
fid = fopen('compress1.eb','wb');        % การอ่านไฟล์ที่ชื่อ compress1.ebj เป็นไฟล์ไบนารี่
fwrite(fid,code,'ubit1');                % เขียนข้อมูลลงไฟล์
fclose(fid);                             % ปิดไฟล์
[a b] = size(code);
%read in the same file==================================================
fid  = fopen('compress1.eb','rb');       % เปิดไฟล์
data = fread(fid,[a b],'ubit1');         % อ่านข้อมูลในไฟล์
fclose(fid);                             % ปิดไฟล์
% Show size files=======================================================
AD=dir('nemo6464.bmp')
BD=dir('compress1.eb')
% Decode of Huffman code ================================================
deco = decodehuffman(data, dict);
deco=round(deco)-512;
desig = reshape(deco,M,N);
% =======================================================================
r=pkildt2d(desig,hh,hv);
% Check Error============================================================
ER = (nsig-r);
ERR = ER^2;
nesig = nsig^2;
Error = (sum(ERR(:)))/M*N;
SNR = 10*log10((sum(nesig(:))/M*N)/Error)
PSNR = 10*log10(max(nesig(:))/Error)
%=======================================================================
% Display(Show) =========================================================
figure(1)
subplot(2,2,1),image(nsig),colormap(gray(256)),title('Picture')
subplot(2,2,2);image(QQ);colormap(gray(256)),title('LDT')
subplot(2,2,3),image(r),colormap(gray(256)),title('Inverse')
```

ฟังก์ชั่นที่ใช้ในโปรแกรม

```
%=========================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%
%  File name : pkldt.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong Dokkiang          ID 44362739
%=========================================
function [cn,dn] = pkldt(xn,hh)
   q = length(hh)/2;
   y  = xn(2:2:end);
   cn = xn(1:2:end);
   v0 = conv(cn,hh);
   v0 = v0(q+1:1:end-(q-1));
   dn = y-v0;
```

```
% ==========================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี  LDT
%  Image Compresstion Using by LDT
%
% File name : pkldt2d.m
% By : Mr.Krichakon Boonruang          ID 44362515
%       Mr.Puripong  Dokkiang          ID 44362739
% ==========================================================
function [J,hh,hv] = pkldt2d(I,q)
[row,col] = size(I);
hh = genH(I,q);
state1=[];
for i=1:1:row
  [cn,dn] = pkldt(I(i,:),hh);
  tmp = [cn dn];
  state1 = [state1 ; tmp];
end
hv= genH(state1.',q);
state2=[];
for i=1:1:col
   [cn,dn] = pkldt((state1(:,i)).',hv);
   tmp = [cn dn];
   state2 = [state2 tmp.'];
end
J=state2;
```

```
%=========================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%
%  File name : pkildt.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=========================================================
function yn = pkildt(cn, dn, hn)
  q=(length(hn)/2);
  tmp = conv(cn,hn);
  tmp = tmp(q+1:1:end-(q-1));
  u1 = dn+tmp;
  v1 = zeros(1,2*length(dn));
  v0 = zeros(1,2*length(cn));
  v1(1:2:end) = u1;
  v0(1:2:end) = cn;
  v1=[0 v1];
  v1=v1(1:1:end-1);
  yn = v1 + v0;
```

```
%=========================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%
%  File name : pkild2d.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=========================================================
function I = pkildt2d(J,hh,hv)
[row,col] = size(J);
state1 = [];
for i=1:1:row
    tmp = (J(:,i)).';
    cn  = tmp(1:1:row/2);
    dn  = tmp((row/2)+1:1:end);
    out = pkildt(cn,dn,hv);
    state1 = [state1 out.'];
end
state1;
state2=[];
for i=1:1:col
  tmp = state1(i,:);
  cn = tmp(1:1:col/2);
  dn = tmp((col/2)+1:1:end);
  out = pkildt(cn,dn,hh);
  state2 = [state2 ; out];
end
I = state2;
```

```
%=============================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%  File name : genH.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=============================================
function hh = genH(I,q)
  [row,col] = size(I);
  A = [];    Y = [];
  tmp = [0];
  for p=1:1:row
    tt  = I(p,:);
    tcn = tt(1:2:end);
    tdn = tt(2:2:end);
    for i = 1:1:length(tcn)
      for k = -q:1:(q-1)
        index=i-k;
        if index > 0 & index <= length(tcn)
          tmp = [tmp tcn(index)];
        else
          tmp = [tmp 0];
        end
      end
      A = [A ; tmp(2:end)];
      Y = [Y;tdn(i)];
      tmp=0;
    end
  end
  A;  Y;
  hh = inv(A.'*A)*(A.'*Y);
  hh = hh.';
```

```
%=============================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%  File name : medhisto..m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=============================================================
function h = medhisto(xx);
h = [];
for gl = 1: 1: length(xx)              การกำหนดค่า g1
  [indx,indy,val] =  find(xx==gl);     การนำค่า xx มาเปรียบเทียบกับค่า g1
  h          =  [h sum(val)];          การนับค่าที่เหมือนกันว่ามีกี่ค่า
end
```

```
%=================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%  File name : huffmanrable.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=================================================
function dict = huffmantable(sig,prob);
%=================================================
% FUNCTION dict = huffmantable(sig,prob);
% This function is Table of huffman code
% sig is input
% prob is Probability distribution
% dict is Table of huffman code
%=================================================
n_ary = [];
variance = '';
msg=nargchk(2,4, nargin);
if nargin > 2
        n_ary = varargin{1};
end
if nargin == 4
        variance = varargin{2};
end
if isempty(n_ary)
    n_ary = 2; % default value is binary encryption
end
if ( variance )
        % if variance contains a non-null string do nothing
else
   variance = 'max'; % default is maximum variance Huffman code
end
```

```
% Make sure that internally all vectors are represented as column vectors
m = size(sig);
if( m(1) == 1 )
   sig = sig';
end
m = size(prob);
prob = prob(:);

% Make sure that the input symbols are in a cell array format
if ~iscell(sig)
        [m,n] = size(sig);
        sig = mat2cell(sig, ones(1,m), ones(1,n)) ;
end

% Check if all the input symbols are either alphabets or numbers or a
% combination of the two
for i=1:length(sig)
        isalphanumeric(i) = ischar(sig{i}) || isnumeric(sig{i});
end

% Check if the each symbol in the first input is unique
for i = 1:length(sig)-1
        pilotpoint = sig{i};
        for j = i+1:length(sig)
                if length(pilotpoint) == length(sig(j)) && min(pilotpoint == sig{j})
                        error('comm:huffmandict:RepeatedSymbols', 'Source symbols repeat')
                end
        end
end
```

```
% Create tree nodes with the signals and the corresponding probabilities
huff_tree = struct('signal', [], 'probability', [],...
   'child', [], 'code', [], 'origOrder', -1);
for i=1:length( sig )
   huff_tree(i).signal = sig{i};
   huff_tree(i).probability = prob(i);
         huff_tree(i).origOrder = i;
end

% Sort the signal and probability vectors based on ascending order of
% probability
[s, i] = sort(prob);
huff_tree = huff_tree(i);
huff_tree = create_tree(huff_tree, n_ary, variance); % create a Huffman tree
[huff_tree,dict,avglen] = create_dict(huff_tree,{},0, n_ary); % create the codebook

% The next few lines of code are to sort the dictionary.
% If sorting based on original order then use dict{:,4}.
% If sorting based on the lenght of code, then use  dict{:,3}.
[dictsort,dictsortorder] = sort([dict{:,4}]);
finaldict = {};
for i=1:length(dictsortorder)
   finaldict{i,1} = dict{dictsortorder(i), 1};
   finaldict{i,2} = dict{dictsortorder(i), 2};
end
dict = finaldict;
```

```
%-------------------------------------------------------------------
%% Function: huff_tree
% Input: An array of structures to be arranged into a Huffman tree
% Utility: This is a recursive algorithm to create the Huffman Code
%       tree. This is a recursive function
function huff_tree = create_tree(huff_tree, n_ary, variance)

% if the length of huff_tree is 1, it implies there is no more than one
% node in the array of nodes. This is the termination condition for the
% recursive loop
if( length(huff_tree) <= 1)
   return;
end
% Combine the first n_ary (lowest probability) number of nodes under one
% parent node, remove these n_ary nodes from the list of nodes and add
% the new parent node that was just created
temp = struct('signal', [], 'probability', 0, ...
   'child', [], 'code', []);

for i=1:n_ary
   if( length(huff_tree) == 0), break; end
   temp.probability = temp.probability + huff_tree(1).probability; % for ascending order
   temp.child{i} = huff_tree(1);
         temp.origOrder = -1;
   huff_tree(1) = [];
end
if( strcmpi(variance, 'min') == 1 )
   huff_tree = insertMinVar(huff_tree, temp);
else
   huff_tree = insertMaxVar(huff_tree, temp);
end
```

```
% create a Huffman tree from the reduced number of free nodes
huff_tree = create_tree(huff_tree, n_ary, variance);
return;
%-------------------------------------------------------------------
%% This function will insert a node in the sorted list such that the
% resulting list will be sorted (ascending). If there exists node with the
% same probability as the new node, the new node  is placed after these
% same value nodes.
function huff_tree = insertMaxVar(huff_tree, newNode)
sortedOn = [huff_tree.probability];
i = 1;
while i <= length(huff_tree) && ...
      newNode.probability > huff_tree(i).probability
   i = i+1;
end
huff_tree = [huff_tree(1:i-1) newNode huff_tree(i:end)];

%-------------------------------------------------------------------
%% This function will insert a node in the sorted list such that the
% resulting list will be sorted (ascending). If there exist nodes with the
% same probability as the new node, the new node  is placed before these
% same value nodes.
function huff_tree = insertMinVar(huff_tree, newNode)
sortedOn = [huff_tree.probability];
i = 1;
while i <= length(huff_tree) && ...
      newNode.probability >= huff_tree(i).probability
   i = i+1;
end
huff_tree = [huff_tree(1:i-1) newNode huff_tree(i:end)];

%-------------------------------------------------------------------
```

```
%% This function does a pre-order traversal of the tree to create the codes
% for each leaf node. This is a recursive function
function [huff_tree,dict,total_wted_len] = create_dict(huff_tree,dict,total_wted_len, n_ary)
% Check if the current node is a leaf node. If it is, then add the signal on
% this node and its corresponding code to the dictionary global n_ary
if( length(huff_tree.child) == 0 )
   dict{end+1,1} = huff_tree.signal;
   dict{end, 2} = huff_tree.code;
   dict{end, 3} = length(huff_tree.code);
        dict{end, 4} = huff_tree.origOrder;
   total_wted_len = total_wted_len + length(huff_tree.code)*huff_tree.probability;
   return;
end
num_childrens = length(huff_tree.child);

for i = 1:num_childrens
   huff_tree.child{i}.code = [huff_tree(end).code, (num_childrens-i)];
   [huff_tree.child{i}, dict, total_wted_len] = ...
      create_dict(huff_tree.child{i}, dict, total_wted_len, n_ary);
end
```

```
%=========================================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%
%  File name : decodehuffman.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=========================================================
function deco = decodehuffman(comp, dict)
msg=nargchk(2,2, nargin);
[m,n] = size(comp);
isSigNonNumeric = max(cellfun('isclass', {dict{:,1}}, 'char') );
deco = {};
i = 1;
while(i <= length(comp))
   tempcode = comp(i);
   found_code = is_a_valid_code(tempcode, dict);
   while(length(found_code) == 0 && i < length(comp))
      i = i+1;
      tempcode = [tempcode, comp(i)];
      found_code = is_a_valid_code(tempcode, dict);
   end
         deco{end+1} = found_code;
   i=i+1;
end
if( n == 1 )      % if input was a column vector
   deco = deco';   % the decoded output should also be a column vector
end
if ( ~isSigNonNumeric )
         deco = cell2mat(deco);
end
deco = deco(length(deco):-1:1);
```

```
%-------------------------------------------------------------------
function found_code = is_a_valid_code(code, dict)
found_code = [];
m = size(dict);
for i=1:m(1)
  if ( isequal(code, dict{i,2}) )
    found_code = dict{i,1};
    return;
  end
end
```

```
%=============================================
%การบีบอัดข้อมูลภาพโดยใช้วิธี LDT
%  Image Compresstion Using by LDT
%
%  File name : encodehuffman.m
%  By : Mr.Krichakon Boonruang        ID 44362515
%       Mr.Puripong  Dokkiang         ID 44362739
%=============================================
    function code = encodehuffiman(input,dict);
    y = 0 ;
    for i = 1:1:length(input),
      for j = 1:1:length(dict),
        if input(i) == dict{j,1},
          a=dict{j,2};
          dict{j,1};
          y = [a y];
        end
      end
    end
    code = y(1:1:end-1) ;
```

ประวัติผู้เขียน

ชื่อ นาย กริชกร บุญเรือง

ภูมิลำเนา 837 ถ.บรมไตรโลกนารถ2 ต.ในเมือง อ.เมืองพิษณุโลก จ. พิษณุโลก 65000

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนพิษณุโลกพิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรี ชั้นปีที่ 4

สาขาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email : sillyjija@hotmail.com

ชื่อ นาย ภูริพงศ์ ดอกเกี๋ยง

ภูมิลำเนา 111 หมู่2 ต. ตาลชุม อ.ท่าวังผา จ.น่าน 55140

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนยุพราชวิทยาลัย
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรี ชั้นปีที่ 4

สาขาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email : bombjajaa@hotmail.com

การบีบอัดข้อมูลภาพโดยวิธี LDT

Image Data Compression by using the LDT method

กริชกร บุญเรือง และ ภูริพงศ์ คอกเกี๋ยง
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
มหาวิทยาลัยนเรศวร อ. เมือง จ.พิษณุโลก 65000

บทคัดย่อ

โครงงานนี้เป็นการศึกษาและพัฒนาโปรแกรมการบีบอัดข้อมูลภาพ โดยวิธีการแปลงแอลดีที ขั้นตอนการบีบอัดข้อมูลภาพประกอบด้วยนำภาพมาทำการแปลงแอลดีทีและเข้ารหัสแบบฮัฟฟ์แมน เพื่อจัดเก็บในรูปของไฟล์ไบนารี่ที่มีขนาดเล็กกว่าไฟล์ต้นฉบับ และหาอัตราการบีบอัดข้อมูลภาพเพื่อวัดการประสิทธิภาพการบีบอัดด้วยวิธีแอลดีที นอกจากนี้ยังทำการเปรียบเทียบคุณภาพของภาพที่ได้รับหลังจากการคลายการบีบอัดในรูปแบบค่า Signal to Noise Ratio (SNR) และ Peak Signal to Noise Ratio (PSNR)

จากผลการทดลองการบีบอัดข้อมูลภาพจำนวนสามภาพ คือ ภาพนีโม ภาพตา และภาพลีน่า พบว่าอัตราการบีบอัดข้อมูลของภาพตามีค่าสูงสุดเท่ากับ 1.9 ซึ่งมีค่า SNR เท่ากับ 77.2 dB และ PSNR เท่ากับ 42.4 dB

ABSTRACT

This project is to study and develop a program for image compression by using the linear decomposition transform (LDT). The method for compressing image in the three following steps. First, an image is transformed by using the LDT. Second, the transformed image is encoded by using the Huffman coding to keep the compressed image in term of a binary file. Third, the binary file is used for determining its compression ratio. In addition, the quality of the decompressed image is compared with the original image in regard to a Signal to Noise Ratio (SNR) and a Peak Signal to Noise (PSNR).

From the experimental result with three image compression, it has been found that the maximum value of the compression ratio is 1.9 for the eye image whose SNR is 77.2 dB and PSNR is 42.4 dB.

1. บทนำ

ทุกวันนี้คอมพิวเตอร์ได้เข้ามามีบทบาทในการทำงานมากขึ้น โดยมีการประยุกต์การใช้งานด้านต่างๆรวมทั้งด้านของการสื่อสารข้อมูล และใช้ในการบันทึกข้อมูลต่างๆเป็นต้น โดยข้อมูลในปัจจุบันนี้มีมากมายหลายแบบทั้งข้อมูลภาพ ข้อมูลเสียง ข้อมูลตัวหนังสือ ซึ่งต่างก็มีคุณสมบัติแตกต่างกันออกไป ข้อมูลภาพก็เป็นข้อมูลชนิดหนึ่งที่มีความสำคัญต่อการทำงานในปัจจุบันมาก เพราะเป็นข้อมูลที่สื่อความหมายได้ดี มีความสวยงาม แต่ข้อมูลภาพนั้นก็มีข้อเสียคือในการจัดเก็บนั้นใช้เนื้อที่ในการจัดเก็บมาก เมื่อข้อมูลมีขนาดใหญ่ทำให้ยากต่อการจัดเก็บและสื่อสารกันระหว่างเครือข่าย เพราะขนาดข้อมูลที่ใหญ่นั่นเอง ทางคณะผู้เสนอโครงงานจึงมีแนวคิดเพื่อที่จะหาทางย่อขนาดของข้อมูลภาพให้มีขนาดเล็กลง ทางคณะผู้เสนอโครงงานจึงได้นำเทคนิค LDT (Linear Decomposition Transform) มาใช้ในการย่อขนาดของข้อมูลให้มีขนาดเล็กลง โดยแนวคิดที่ใช้จะเน้นไปที่ภาพขาวดำเพื่อเป็นแนวทางให้ผู้ที่สนใจได้พัฒนาต่อไป

2. หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 การลดขนาดข้อมูลภาพ

ขบวนการลดขนาดข้อมูล (Data Compression) จะหมายถึงขบวนการที่ใช้ในการทำให้ข้อมูล (Data) ที่ต้องใช้แทนข่าวสารหนึ่งนั้นลดลง ซึ่งสามารถเปรียบเทียบข้อมูลได้กับตัวหนังสือที่ที่จะสื่อความหมายถึงเนื้อหาสาระภายในหนังสือเล่มหนึ่งๆนั่นเอง ในกรณีของหนังสือสองเล่มที่แต่งโดยคนละคนแต่มีเนื้อหาเหมือนกันนั้น จำนวนตัวอักษร (ข้อมูล) ที่ใช้ในการบอกเล่าจะไม่เท่ากันก็ได้ นั่นก็แสดงว่าหนังสือที่ผู้แต่งใช้จำนวนตัวอักษรที่มากว่าจะต้องมีคำหรือประโยคบางประโยคที่เกินความจำเป็น เช่น อาจเป็นประโยคที่บอกเล่าถึงสิ่งที่ได้กล่าวมาแล้วก่อนหน้านี้ หรือคำ/ประโยค ที่ไม่ได้สื่อความหมายใดๆ การลดขนาดข้อมูลภาพก็เช่นเดียวกัน ข้อมูลของระดับความสว่างบนจุดภาพแต่ละจุดร่วมกันเพื่อสื่อถึงความหมายของภาพ ก็จะมีส่วนที่เกินความจำเป็นที่สามารถตัดออกไปได้ ถ้ากำหนดให้ n_1 และ n_2 เป็นจำนวนของข้อมูลที่ใช้ในการสื่อความหมาย

ของภาพๆหนึ่ง อัตราส่วนการลดขนาดข้อมูล (Data compression ratio) จะคำนวณได้คือ

$$\text{Data compression ratio} = \frac{n_1}{n_2} \qquad (2\text{-}1)$$

2.2 แนวคิดการแปลงรูปแอลดีที

กระบวนการ Wavelet เป็นกระบวนการที่สำคัญในการจัดการกับสัญญาณที่ทำให้การส่งสัญญาณนั้นมีคุณภาพ โดยการเปลี่ยนแปลงกระบวนการ wavelet จะถูกจำแนกสัญญาณไปเป็น linear combination ของผลรวมเชิงเส้นสัญญาณพื้นฐาน (Bases signals) กับค่าสัมประสิทธิ์ ที่มีลักษณะเฉพาะในการประยุกต์ใช้งานด้านต่างๆ มากมาย

การแปลงรูปเวฟเล็ต จะเริ่มต้นด้วยการแยกสัญญาณนำเข้าออกเป็นสองชุดสัญญาณย่อย ที่มีความยาวของแต่ละชุดสัญญาณย่อยเท่ากับเครื่องหนึ่งของความยาวสัญญาณนำเข้าสัญญาณชุดแรก มาจากสัญญาณนำเข้าที่มีเวลาเป็นเลขคู่ และผ่านตัวกรองความถี่ต่ำที่ไม่แปรเปลี่ยนตามาเลาทำให้ได้สัญญาณแบบหยาบ (Coarse signal) ชุดแรกนี้บรรจุข้อมูลเฉพาะความถี่ต่ำในขณะที่สัญญาณชุดสองมาจากสัญญาณนำเข้าที่มีเวลาเป็นเลขคี่ และผ่านตัวกรองความถี่สูงที่ไม่แปรเปลี่ยนตามเวลาทำให้ได้สัญญาณแบบละเอียด (Detail signal) ชุดสองนี้บรรจุข้อมูลเฉพาะความถี่สูง

การแปลงรูปแอลดีที (Linear Decomposition) กับสัญญาณหนึ่งมิติ โดยทั่วไปการแปลงรูปแอลดีทีมีคุณลักษณะสมบัติแตกต่างจากการแปลงรูปเวฟเล็ตในสามประการดังต่อไปนี้คือ

- การแปลงรูปแอลดีที (LDT) จะใช้ตัวกรองสัญญาณที่ไม่แปรเปลี่ยนตามเวลาเพียงตัวเดียว
- ตัวกรองสัญญาณที่ไม่แปรเปลี่ยนตามเวลาของการแปลงรูปแอลดีทีจะปรับเปลี่ยนตามสัญญาณนำเข้า จึงทำให้ศักยภาพการแปลงรูปแอลดีทีมีประสิทธิภาพเหนือกว่าการแปลงรูปเวฟเล็ตในแง่ของการประยุกต์ใช้งานด้านการบีบอัดข้อมูลและการลดสัญญาณรบกวน
- ค่าพารามิเตอร์ของตัวกรองสัญญาณจะถูกเลือกมาจากการใช้เกณฑ์ต่ำสุดของขนาดแอลพีนอร์ม (/p norm criterion)

2.3 พื้นฐานในการแปลงรูปแอลดีที

กำหนดให้ x (n) เป็นสัญญาณแบบไม่ต่อเนื่อง (Discrete signal) ที่มี x (n) = 0 สำหรับ n< 0และสัญญาณนำเข้าที่จะถูกแปลงรูปแอลดีทีมีค่าเป็น

$$\{x(0), x(1), x(2), \ldots, x(N-1)\} \qquad (2\text{-}3)$$

ในที่นี้ N เป็นจำนวนเต็มคู่สมมุติว่าสัญญาณนำเข้าในสมการ (2-3) ประกอบด้วยผลรวมของสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า (Log term trend) {c (n)} และสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ (Short term trend) {d (n)} โดยที่แต่ละสัญญาณมีจำนวนข้อมูล N/2 การแปลงรูปแอลดีทีของ x(n) ถูกกำหนดนิยามดังแสดงไว้ในรูป 2.1 ได้ว่า

$$(\{c(n)\}, \{d(n)\}) = T(x(n)\}) \qquad (2\text{-}4)$$

โดยที่ข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้าซึ่งได้มาจากการสุ่มตัวอย่างแบบลงด้วยสองช่วงเวลาค่าเป็น

$$c(n)=x(2n) \text{ สำหรับ } n = 0,1,\ldots, (N/2)-1 \qquad (2\text{-}5)$$

และข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ซึ่งได้มาจากการผลต่างของข้อมูลระหว่างการสุ่มตัวอย่างแบบลงด้วยสองช่วงเวลาของข้อมูลนำเข้ากับค่าประมาณการของข้อมูลดังกล่าว มีค่าเป็น

$$d(n) = x(2n+1) - \sum_{k=-q}^{q-1} h(k)x(2n-2k) \text{ สำหรับ } n = 0, 1,\ldots, (N/2)-1 \qquad (2\text{-}6)$$

2.4 ส่วนกลับการแปลงรูปแอลดีที

สัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า {c (n)} และสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบชั่วครู่ {d (n)} สามารถนำมาผ่านส่วนกลับของการแปลงรูปแอลดีทีเพื่อให้ได้สัญญาณนำเข้า {x (n)} ซึ่งส่วนกลับของการแปลงรูปแอลดีทีดังแสดงไว้รูป 2.2 ถูกนิยามได้ว่า

$$\{x(n)\} = T^{-1}(\{c(n)\}, \{d(n)\}) \qquad (2\text{-}9)$$

จากรูป 2.2 เมื่อข้อมูลสัญญาณที่มีพฤติกรรมแนวโน้มข้อมูลแบบช้า c (n) ผ่านการสุ่มตัวอย่างแบบขึ้นด้วยสองช่วงเวลาจะได้ค่าผลลัพธ์ข้อมูลสัญญาณ ye (n) ที่มีค่าเป็น

$$y_e(n) = \begin{cases} c(n/2), n = 0,2,4\ldots,(N/2) \\ 0, n = 1,3,5,\ldots,(N/2)-1 \end{cases} \qquad (2\text{-}10)$$

นำค่าข้อมูล c (n) ในสมการ (2-5) แทนลงในสมการ (2-10) จะได้ว่า

$$y_e(n) = \begin{cases} x(n), n = 0,2,4\ldots,(N/2) \\ 0, n = 1,3,5,\ldots,(N/2)-1 \end{cases} \qquad (2\text{-}11)$$

จากสมการ (2-5) และ (2-6) จะได้ข้อมูลสัญญาณนำเข้าสำหรับช่วงเวลาเฉพาะคี่กลับคืนมาดังแสดงไว้ในรูป 2.2 ที่มีค่าเป็น

$$x(2n-1) = d(n) + \sum_{k=-q}^{q-1} h(k)c(n-k) \quad \text{สำหรับ } n = 0, 1, \ldots, (N/2-1) \qquad (2\text{-}12)$$

ต่อจากนั้นนำข้อมูลสัญญาณนี้ผ่านการสุ่มตัวอย่างแบบขึ้นด้วยสองช่วงเวลาและเลื่อนช่วงเวลาไปทางซ้ายหนึ่งหน่วยจะได้ข้อมูลสัญญาณ yo (n) ดังนี้คือ

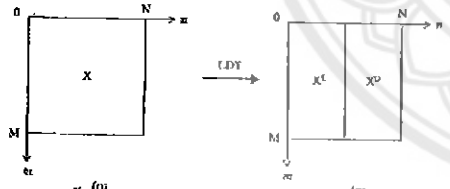
$$y_0(n) = \begin{cases} 0, n = 0,2,4\ldots,(N/2) \\ x(n), n = 1,3,5,\ldots,(N/2)-1 \end{cases} \qquad (2\text{-}13)$$

โดยการใช้ผลรวมกันสมการ (2-11) และ (2-13) จะได้สัญญาณนำเข้ากลับคืนมาเป็น

$$y_0(n) + y_e(n) = x(n) \qquad (2\text{-}14)$$

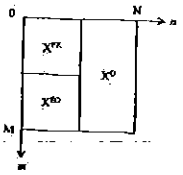
2.5 พื้นฐานการแปลงรูปแอลดีทีสองมิติ

กำหนดให้ X (m, n) เป็นข้อมูลภาพสองมิติขนาด MxN ที่มี M และ N เป็นจำนวนเต็มบวกโดยที่ X (m, n) สำหรับ m< 0 และ n<0 การแปลงรูปแอลดีทีภาพสองมิติที่นำเสนอประกอบด้วยสองขั้นตอน ดังต่อไปนี้ คือ เริ่มด้วยการแปลงรูปแอลดีทีในแต่ละแถวของภาพ X จะได้ผลลัพธ์สองภาพย่อย X และ X ดังแสดงไว้ในรูป 2.3 (ข)



รูป 2.3 พื้นฐานการแยกองค์ประกอบภาพโดยใช้วิธีแอลดีทีตามแนวระดับ

ต่อจากนั้นทำการแปลงรูปแอลดีทีในแต่ละหลักของภาพ X^E ที่ได้จากขั้นตอนแรก จะพบว่าการแยกองค์ประกอบของภาพออกเป็นสามภาพย่อยดังแสดงไว้ในรูป 2.4



รูป 2.4 พื้นฐานการแยกองค์ประกอบภาพโดยใช้วิธีแอลดีที

2.6 ส่วนกลับการแปลงรูปแอลดีทีสำหรับภาพสองมิติ

เมื่อนำข้อมูลภาพย่อย X^{EE} และ X^{EO} แทนลงในสมการ (2-17) และ (2-18) ตามลำดับจะพบว่าข้อมูลภาพดั้งเดิม $X(2m,2n)$ และ $X(2m+1,2n)$ กลับคืนมามีค่าเป็น

$$X(2m,2n) = X^{EE}(m,n) \qquad (2\text{-}20)$$

และ

$$X(2m+1,2n) = X^{EO}(m,n) + \sum_{k=-q}^{q-1} h_h(k) X^{EE}(m-k,n) \qquad (2\text{-}21)$$

ต่อมานำข้อมูลภาพย่อย X^O แทนลงในสมการ (2-20) จะพบว่าข้อมูลภาพดั้งเดิม $X(m,2n+1)$ กลับคืนมา มีค่าเป็น

$$X(m,2n+1) = X^O(m,n) + \sum_{k=-q}^{q-1} h_h(k) X(m,2n-2k) \qquad (2\text{-}22)$$

ต่อจากนั้นนำข้อมูลภาพดั้งเดิม $\{X(2m,2n)\}$ จากสมการ (2-20) และ $\{X(2m+1,2n)\}$ จากสมการ (2-21) ผ่านขบวนการส่วนกลับการแปลงแอลดีทีในแต่ละหลักของภาพ ทำให้ได้ข้อมูลภาพดังเดิม {X (m,2n)} และนำข้อมูลภาพ {(m,2n+1)} มาทำส่วนกลับการแปลงแอลดีทีตามแนวระดับจะทำให้ได้ข้อมูลภาพดั้งเดิม X (m, n) กลับคืนมาเหมือนเดิมทุกประการ ถ้ากำหนดให้ $T_{2D}^{-1}(\bullet)$ เป็นสัญญาการส่วนกลับของการแปลงแอลดีทีสำหรับภาพย่อยสองมิติของ X^{EE}, X^{EO} และ X^O แล้วจะพบว่าส่วนกลับการแปลงรูปแอลดีทีสำหรับภาพย่อยสองมิติดังกล่าวมีค่าเป็น

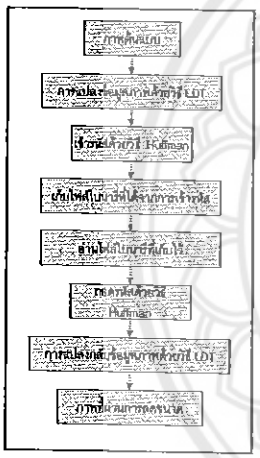
$$X = T_{2D}^{-1}(X^{EE}, X^{EO}, X^O) \qquad (2\text{-}23)$$

โดยที่ข้อมูลภาพย่อย X^{EE} ลงแทนในสมการ (2-20) ทำให้ได้ข้อมูลภาพย่อย X (2m,2n) ส่วนข้อมูลย่อย X^{EO} และ X^{EE} ลงแทนในสมการ (2-21) ทำให้ได้ข้อมูลภาพย่อย X(2m,1,2n) ต่อมาหามูลภาพย่อย X(m,2n) ซึ่งได้มาจากผลรวมของการสุ่มตัวอย่างแบบขึ้นของข้อมูลภาพย่อย X(2m,,2n) และ X(2m,1,2n) ขั้นตอนต่อไปนำข้อมูลย่อย X(m,,2n) กับข้อมูลย่อย X^O แทนลงในสมการ (2-22) ทำให้ได้ข้อมูลภาพย่อย X(m,,2n+1) ขั้นตอนสุดท้ายได้รับภาพดั้งเดิม X ซึ่งได้มาจากผลผลรวมของการสุ่มตัวอย่างแบบขึ้นของข้อมูลภาพย่อย X(m, 2n) และ X(m,2n +n) จะเห็นได้ว่าไม่ว่าจะเลือกพารามิเตอร์กรองสัญญาณ $\{h_h(k)\}$ และ $\{h_v(k)\}$ มีค่าเท่าใดก็ตาม ส่วนกลับของการแปลงรูปแอลดีทีสำหรับภาพสองมิติดังแสดงไว้ในสมการ (2-23) สามารถหาค่าได้เสมอ

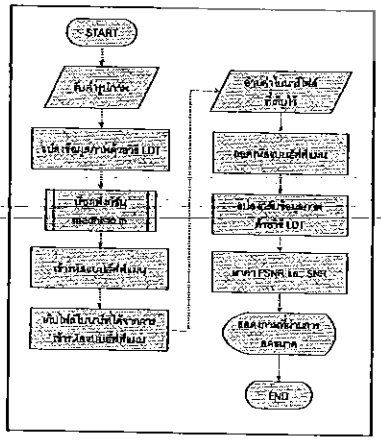
3. วิธีดำเนินการทดลอง

ในการทำการทดลองเพื่อลดขนาดข้อมูลโดยใช้วิธี LDT นั้นผู้ทำการทดลองได้ออกแบบขั้นตอนในการเขียนโปรแกรมเพื่อความเป็นระเบียบและเพื่อง่ายต่อการแก้ไขและปรับปรุงโดยจะแยกการทำงานแต่ละขั้นตอนดังนี้

- นำภาพต้นแบบมาเข้ากระบวนการลดขนาดขั้นแรกโดยใช้วิธี LDT
- นำเอาท์พุทที่ได้มาเข้ากระบวนการHuffman Encoding ซึ่งจะได้ไฟล์ภาพที่ผ่านการลดขนาด โดยจะนำไปหา Data Compression Ratio
- นำภาพที่ผ่านการลดขนาดมาเข้ากระบวนการการแปลงกลับ โดยนำมาเข้า Huffman Decoding
- และนำมาเข้า LDT^{-1} ซึ่งจะได้ภาพที่มีลักษณะเหมือนภาพต้นแบบ โดยจะมีการหา PSNR และ SNR เพื่อเปรียบเทียบภาพต้นแบบกับภาพที่ผ่านการลดขนาดและนำมาแปลงกลับ



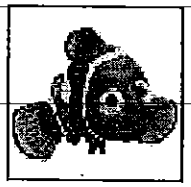
รูปที่ 3.1 แสดงขั้นตอนการทำงานของโปรแกรม



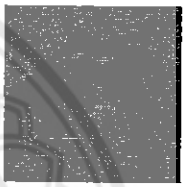
รูปที่ 3.2 flow chart แสดงขั้นตอนการทำงานของโปรแกรม

4.ผลการทดลอง

ในขั้นตอนการทดลองต่อไปนี้ ทางผู้จัดทำได้ใช้ภาพต้นแบบคือ NEMO6464.BMP ขนาด 5176 bytes , EYE6464.TIF ขนาด 5486 bytes และ LENA6464.BMP ขนาด 5176 bytes ซึ่งมีความละเอียดของภาพเท่ากันคือ 64X64 Pixel ดังแสดงในรูปที่ 4.1 , 4.2 และ 4.3 ตามลำดับ



รูปที่ 4.1 ภาพต้นฉบับ NEMO6464.BMP ความละเอียด 64X64 Pixel



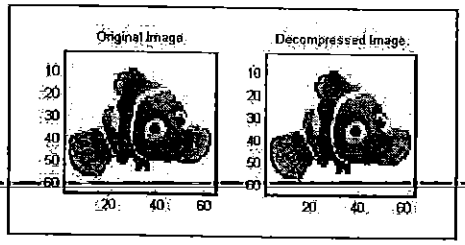
รูปที่ 4.2 แสดงภาพต้นฉบับ EYE6464.TIF ความละเอียด 64X64 Pixel

รูปที่ 4.3 แสดงภาพต้นฉบับ LENA6464.BMP ความละเอียด 64X64 Pixel

จากผลที่ได้ของการบีบอัดข้อมูลภาพตามขั้นตอนที่ได้กล่าวมาแล้วนั้น สามารถเขียนเป็นตารางที่ 4.1

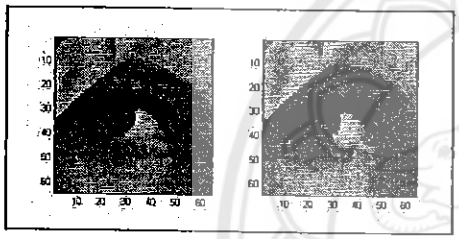
ตารางที่ 4.1 แสดงค่า CR, SNR, PSNR ของภาพที่ทำการทดลองทั้งหมด

ภาพที่ทดสอบ	CR	SNR (dB)	PSNR (dB)
ภาพ NEMO ขนาด 64x64 pixel	1.8	62.1	28.2
ภาพ EYE ขนาด 64x64 pixel	1.9	77.2	42.4
ภาพ lena ขนาด 64x64 pixel	1.5	74.9	41.5



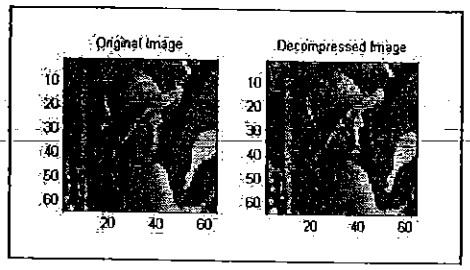
รูปที่ 4.4 แสดงภาพ NEMO (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.4 แสดงถึง: ภาพต้นฉบับ NEMO.BMP ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5176 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.8 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 28.2 SNR = 62.1 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน



รูปที่ 4.5 แสดงภาพ EYE (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.5 แสดงถึง: ภาพต้นฉบับ EYE.TIF ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5486 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.9 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 42.4 SNR = 77.2 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน



รูปที่ 4.6 แสดงภาพ LENA (ขนาด 64x64 Pixel) ทั้งภาพต้นฉบับและภาพที่ผ่านการลดขนาด

จากรูปที่ 4.6 แสดงถึง: ภาพต้นฉบับ LENA.BMP ความละเอียด 64X64 Pixel ซึ่งมีขนาด 5176 bytes และภาพที่ผ่านการลดขนาด ซึ่งนำมาแปลงผกผันให้ได้ข้อมูลภาพซึ่งจะได้อัตราการลดขนาดเท่ากับ 1.5 เท่า และค่าความผิดพลาดที่ได้ เมื่อเทียบกับภาพต้นแบบมีค่า PSNR = 41.5 SNR = 74.9 โดยภาพที่ผ่านการแปลงกลับนั้นมีความชัดเจนดี ไม่มีความสิ่งรบกวน

ในการทดลองในแต่ละหัวข้อ จะเป็นการหาอัตราการลดขนาดข้อมูลที่ได้ ค่าความผิดพลาดที่เกิดขึ้น ผลการทดลองที่ได้ดังได้กล่าวมาแล้วนั้นจะทำโดยใช้โปรแกรม Matlab V7.0.0.1 ซึ่งสะดวกต่อการคำนวณสมการทางคณิตศาสตร์ที่ซับซ้อนแต่มีข้อเสียตรงที่ใช้เวลาในการคำนวณนานนั่นเอง

สำหรับวิธีที่ใช้ในการทดลองจะเป็นดังต่อไปนี้

1. ข้อมูลภาพต้นแบบที่ใช้ทั้งหมด จะเป็นภาพระดับสีเทา 256 ระดับ (8 bpp) ความละเอียด 64X64 Pixel
2. วัดอัตราการลดขนาดข้อมูลที่ได้ ออกมาในหน่วยของเท่า
3. นำข้อมูลที่ถูกลดขนาดแล้วมาทำการขยาย เพื่อให้ได้กลับมาเป็นข้อมูลภาพ แล้ววัดค่าความผิดพลาดของข้อมูลภาพที่ได้นี้เปรียบเทียบกับภาพต้นแบบ

บทที่ 5 สรุปผลการทดลอง

จากการการทดลองเพื่อลดขนาดข้อมูลภาพโดยใช้ฟังก์ชันการแยกองค์ประกอบตามวิธี LDT ฟังก์ชันการเข้ารหัสและถอดรหัสแบบฮัฟฟ์แมน ฟังก์ชันการแปลงกลับของ LDT กับภาพต้นแบบจำนวน 3 ภาพ คือ ภาพนีโม (NEMO6464.BMP) , ภาพตา (EYE6464.TIF) และภาพลีน่า (LENA6464.BMP) ซึ่งมีความละเอียด 64X64 พิกเซลโดยใช้วิธี LDT ตามที่ได้อธิบายไว้ในวิธีการดำเนินการทดลองในบทที่ 3 พบว่าอัตราการบีบอัดข้อมูลภาพของภาพนีโมเท่ากับ 1.8 ภาพตาเท่ากับ 1.9 และภาพลีน่าเท่ากับ 1.5 นอกจากนี้เมื่อทำการคลายการบีบอัดภาพทั้ง 3 พบว่าภาพตามีคุณภาพดีที่สุดเนื่องจากมีค่า PSNR และ SNR สูงสุด แต่อย่างไรก็ตามภาพทั้งสามนั้นไม่มีคุณลักษณะเหมือนภาพต้นแบบเนื่องจากภาพทั้งสามหลังจากถูกแปลงรูปโดย LDT แล้วนำค่าสัมประสิทธิ์ของ LDT ไปทำให้เป็นจำนวนเต็มก่อนเข้ารหัสแบบฮัฟฟ์แมน

กิตติกรรมประกาศ

การจัดทำโครงงานในครั้งนี้ สำเร็จลุล่วงไปได้ด้วยดี ต้องขอกราบขอบพระคุณ คุณพ่อ คุณแม่ สำหรับการให้การสนับสนุนและความช่วยเหลือในด้านต่างๆ อาจารย์ สุชาติ แย้มแม่นอาจารย์ที่ปรึกษาโครงงาน ที่ให้ความดูแล เอาใจใส่และให้คำแนะนำในการทำงานเป็นอย่างดี ขอขอบพระคุณท่านคณะกรรมการสอบทุกท่านที่

ได้สละเวลาอันมีค่า ครูอาจารย์ทุกท่านที่ประสิทธิ์ประสาทวิชาความรู้ให้ตลอดมา ตลอดจนเพื่อนๆทุกคนที่ได้ให้กำลังใจในการทำโครงงานนี้ตลอดมา

เอกสารอ้างอิง

[1] James A.Cadzaw and Suchart Yammen. Data Adaptive Linear Decomposition Transform. Vanderbilt University:2002

[2] รศ.ดร. มนัส สังวรศิลป์. คู่มือการใช้งาน MATLAB ฉบับสมบูรณ์. นนทบุรี: สำนักพิมพ์อินโฟเพรส. 2543.

ประวัติผู้เขียนโครงงาน

นายกริชกร บุญเรือง ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

นายภูริพงศ์ ดอกเกี๋ยง ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร