การบีบอัดข้อมูลภาพโดยการเข้ารหัสแบบ พรีดิกทีฟ

IMAGE COMPRESSION USING PREDICTIVE CODING

นาย ชาญวิทย์ ปิ่นคำ รหัส 45380027
นาย ธนภูมิ อึ้งตระกูล รหัส 45380045
นาย เฉลิมเกียรติ ยั่งยืนเจริญสุข รหัส 45380175

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2548

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การบีบอัดข้อมูลภาพโดยการเข้ารหัสแบบ พรีดิกทีฟ
ผู้ดำเนินโครงงาน	นาย ชาญวิทย์ ปิ่นคำ รหัส 45380027
	นาย ธนภูมิ อึ้งตระกูล รหัส 45380045
	นาย เฉลิมเกียรติ ยั่งยืนเจริญสุข รหัส 45380175
อาจารย์ที่ปรึกษา	ดร.พนมขวัญ ริยะมงคล
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2548

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงการฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ

(ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุชาติ แย้มแม่น)

.....กรรมการ

(อาจารย์ศิริพร เดชะศิลารักษ์)

หัวข้อโครงงาน	การบีบอัดข้อมูลภาพโดยการเข้ารหัสแบบ พรีดิกทีฟ			
ผู้ดำเนินโครงงาน	นาย ชาญวิทย์	ปิ่นคำ	รหัส	45380027
	นาย ธนภูมิ	อึ้งตระกูล	รหัส	45380045
	นาย เฉลิมเกียรติ	ยั่งยืนเจริญสุข	รหัส	45380175
อาจารย์ที่ปรึกษา	ดร. พนมขวัญ	ริยะมงคล		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์			
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์			
ปีการศึกษา	2548			

.....

บทคัดย่อ

โครงงานเรื่อง การบีบอัดข้อมูลภาพโดยการเข้ารหัสแบบ พรีดิกทีฟ มีจุดประสงค์หลักคือต้องการลดขนาดภาพที่ต้องการจัดเก็บ โดยไม่ให้เกิดการสูญเสียของข้อมูลภาพ ทางผู้จัดทำได้พัฒนาขึ้นด้วยโปรแกรมแมทแล็ป โดยใช้วิธีการเข้ารหัสแบบ พรีดิกทีฟ และ การเข้ารหัสแบบ ฮัฟฟ์แมน ในการบีบอัดข้อมูล หลังจากผ่านโปรแกรมการเข้ารหัสแล้วภาพที่ได้จะมีขนาดเนื้อที่ในการจัดเก็บลดลง นอกจากนี้ค่า SNR และ ค่า PSNR มีค่าเข้าใกล้อนันต์ แสดงว่ารูปหลังการคลาย มีค่าเท่ากับรูปต้นแบบ

Project Title	**IMAGE COMPRESSION USING PREDICTIVE CODING**		
Name	Mr. Chanwit	Pinkome	ID. 45380027
	Mr. Thanapoom	Eungtrakool	ID. 45380045
	Mr. Chaloemkiat	Yangyunjaroensuk	ID. 45380175
Project Advisor	Panomkhown Riyamongkol , Ph.D.		
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic	2005		

ABSTRACT

The objective of image compression using predictive coding which is a lossless compression coding is reducing image properties. This project is developed from MATLAB Program by using Predictive coding and Huffman coding to compress images. The result of this project is the program which can compress image files. Moreover, the SNR and PSNR values are infinitive values. These would show that the decoded image is the same as the original one.

กิตติกรรมประกาศ

การจัดทำโครงงานสื่อการบีบอัดข้อมูลภาพโดยการเข้ารหัส พรีดิกทีฟ (Image compression using by predictive coding) สำเร็จลุล่วงไปได้ด้วยดี ผู้จัดทำได้รับความกรุณาอย่างยิ่งจาก ดร.พนม ขวัญ ริยะมงคล , ผู้ช่วยศาสตราจารย์ ดร.สุชาติ แย้มแม่น , อาจารย์ศิริพร เดชะศิลารักษ์ อาจารย์ที่ปรึกษาโครงงาน ที่กรุณาสละเวลา ความคิด ประสบการณ์ และคำปรึกษาอันมีค่า ทำให้คณะผู้จัดทำได้รับประสบการณ์ การทำงานอันมีค่าอย่างยิ่ง

ขอบคุณพี่ๆเพื่อนๆ ทุกคนที่คอยถามไถ่และช่วยเหลือทุกเรื่อง ทั้งในเรื่องการเรียนและการจัดทำโครงงานงานในครั้งนี้

ขอกราบขอบพระคุณบิดา มารดา ญาติพี่น้อง ที่ช่วยดูแล เป็นกำลังใจ และให้ความรักความอบอุ่น ตลอดเวลา โดยเฉพาะทุนทรัพย์จากบิดาและมารดาที่เอื้อหนุนตลอดมา

นาย ชาญวิทย์ ปิ่นคำ
นาย ธนภูมิ อึ้งตระกูล
นาย เฉลิมเกียรติ ยั่งยืนเจริญสุข

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ในปัจจุบันถือได้ว่าคอมพิวเตอร์กลายเป็นสิ่งสำคัญอีกอย่างหนึ่งในการดำรงชีวิตของมนุษย์เราไปแล้ว มนุษย์ใช้คอมพิวเตอร์ในการทำงานหลายอย่าง อาทิเช่น ทำเอกสาร , ส่งe-mail , เล่นอินเทอร์เนท , เก็บเอกสารที่สำคัญต่างๆเป็นต้น แต่ก็จะพบว่าปัญหาที่เกิดขึ้นกับคอมพิวเตอร์เสมอๆก็คือ การที่หน่วยความจำของคอมพิวเตอร์ไม่ไม่เพียงพอที่จะเก็บข้อมูลของผู้ใช้งานได้หมด ทางผู้จัดทำโครงงานได้เล็งเห็นถึงปัญหาตรงจุดนี้จึงได้จัดทำโครงงานขึ้นมา โครงงานนี้ได้แนวคิดมาจากความสนใจเนื้อที่ในการจัดเก็บขนาดของรูปภาพต่างๆ ซึ่งรูปภาพที่เราจัดเก็บนั้นจะมีความละเอียดต่างกัน ขึ้นอยู่กับประเภทของการจัดเก็บ เช่น PNG, TIF, TIFF, GIF, JPEG, bmp, dib เป็นต้น เมื่อรูปภาพที่จัดเก็บมีความละเอียดมากก็จำเป็นต้องใช้ขนาดหรือเนื้อที่ในการจัดเก็บมาก ซึ่งจะทำให้สิ้นเปลืองเนื้อที่ในการจัดเก็บ ดังนั้นเพื่อที่จะทำให้การจัดเก็บรูปภาพที่มีความละเอียดมาก แต่ใช้พื้นที่ในการจัดเก็บข้อมูลน้อยๆ โดยใช้วิธีเข้ารหัสแบบ พรีดิกทีฟ (predictive coding) มาช่วยในการบีบอัดข้อมูลภาพ เพื่อที่จะได้ภาพที่คมชัดเหมือนเดิมและมีขนาดไฟล์ที่เล็กลงด้วย ซึ่งจะทำให้เราประหยัดเนื้อที่ในการจัดเก็บภาพมากขึ้นอีกด้วย

1.2 วัตถุประสงค์ของโครงงาน

ในการจัดทำโครงงานชิ้นนี้ขึ้นมามีวัตถุประสงค์หลักๆ ดังนี้

1. เพื่อนำความรู้ที่ได้เรียนมาในรายวิชา image processing มาประยุกต์ใช้ให้เกิดประโยชน์ เนื่องจากทางผู้จัดทำโครงงานเห็นว่าสามารถนำความรู้ในเรื่อง การเข้ารหัสแบบ พรีดิกทีฟ ในรายวิชา image processing สามารถนำมาประยุกต์ใช้ได้
2. เพื่อจัดเก็บภาพที่ใช้เนื้อที่ในการจัดเก็บน้อยลงแต่ยังสามารถคงความคมชัดของภาพไว้เหมือนเดิม
3. เพื่อทำให้เกิดความรู้และทักษะในการใช้โปรแกรมที่เกี่ยวข้องกับโครงงาน เช่น โปรแกรม Matlab และ มีความเข้าใจในวิธีเข้ารหัสแบบ พรีดิกทีฟ (predictive coding) และ ฮัฟฟ์แมน (Huffman coding) มากขึ้น

1.3 ขอบเขตของโครงงาน

ในการจัดทำโปรแกรมเพื่อบีบอัดข้อมูลภาพนั้น จำเป็นใช้องค์ประกอบหลายๆด้านเพื่อประกอบการพัฒนา โดยมีดังนี้

1. ใช้โปรแกรม Matlab ในการเขียน การเข้ารหัสแบบ พรีดิกทีฟ (predictive coding) นำภาพมาทำการเข้าโปรแกรม การเข้ารหัสแบบ พรีดิกทีฟ (predictive coding) ติดตามผลที่ได้
2. ทำโปรแกรมโดยใช้ Huffman code เป็นตัวบีบอัดข้อมูลภาพ สังเกตผลที่ได้จากการใช้ Huffman code โปรแกรม
3. ทำการเปรียบเทียบผลที่ได้จากการเข้ารหัสแบบ พรีดิกทีฟ (predictive coding) และการใช้ Huffman code แล้วสรุปผล

1.4 ขั้นตอนการดำเนินงาน

ตารางที่ 1.1 แสดงขั้นตอนการดำเนินงานของโครงงาน

กิจกรรม	ม.ค.48	ก.พ.48	มี.ค.48	เม.ย.48	พ.ค.48	ก.พ.49	มี.ค.49	เม.ย.49	พ.ค.49
1. ศึกษาโปรแกรม Matlab	←	—	—	—	—	—	—	→	
2. ศึกษา Image compression	←	→							
3. ศึกษา Predictive coding และ Huffman coding		←	→						
4. ออกแบบรูปแบบการวิเคราะห์ข้อมูลภาพ			←	→					
5. เขียนโปรแกรมในการบีบอัดข้อมูลภาพ					←	→			
6. ทดสอบและแก้ไขการทำงานของโปรแกรม							←	→	
7. นำเสนอผลงานและจัดทำรายงาน									↔

1.5 ผลที่คาดว่าจะได้รับ

ผลที่คาดว่าจะได้รับจากโครงงานชิ้นนี้มีดังนี้

1. สามารถนำความรู้ที่ได้รับมาจากในชั้นเรียนมาประยุกต์ใช้ให้เกิดประโยชน์ได้
2. ได้ภาพที่ยังคงคุณภาพเหมือนกับภาพต้นแบบแต่ใช้เนื้อที่ในการจัดเก็บน้อยลงกว่าเดิม
3. โครงงานนี้จะสามารถเป็นแนวทางเพื่อให้ผู้ที่สนใจศึกษาการบีบอัดข้อมูลภาพแบบฮัฟฟ์แมนได้นำโปรแกรมไปพัฒนา หรือนำไปใช้ต่อไป

1.6 งบประมาณ

1. เอกสาร 3,000 บาท

บทที่ 2
หลักการและทฤษฎี

2.1 หลักการบีบอัดข้อมูลภาพ (Image Compression)

ความสนใจในการบีบอัดข้อมูลภาพเริ่มขึ้นเมื่อประมาณกว่า 35 ปีที่แล้วในโลกของอะนาลอก เพื่อลดแบนด์วิธในการส่งสัญญาณวีดีโอ โดยกระบวนการนี้ถูกเรียกว่า การบีบแบนด์วิธ (Bandwidth compression) จากนั้นเมื่อเทคโนโลยีด้านดิจิตอลพัฒนาขึ้นทำให้เกิดการบีบอัดข้อมูลทางดิจิตอลขึ้น และได้รับการพัฒนาเพิ่มขึ้นอย่างรวดเร็ว

หลักการในการบีบอัดข้อมูลภาพคือ พยายามลดหรือกำจัดส่วนของข้อมูลที่เกินความจำเป็น หรือซ้ำซ้อนกัน(Data redundancy) โดยยังคงข่าวสารไว้เหมือนเดิม ซึ่งจะทำให้ข้อมูลภาพมีขนาดลดลงจากเดิม และสามารถนำภาพกลับมาแสดงภายหลังโดยผ่านกระบวนการคลาย (Decompression)

ข้อมูลจริงมักถูกแปลงเป็นสำเนาแล้วส่งไปในลักษณะ Signals และ digitized ในการบันทึกรูปภาพใน memory นั้นจะใช้เนื้อที่ในการจัดเก็บมาก เทคนิคในการทำ source coding (เรียกว่า data compaction หรือ compression) สามารถใช้การเข้ารหัสและจะได้ข้อมูลใหม่ออกมา วิธีการทำ compression สามารถประยุกต์ใช้ได้ที่เห็นชัดนั้นมีอยู่ 2 ระบบคือ

1. ระบบ การสื่อสาร โดยการเข้ารหัสข้อมูลเราสามารถที่จะส่งสารสนเทศไปในเวลาที่สั้นกว่าเดิม ซึ่งเราสามารถส่ง Bandwidth เล็กๆ แทนที่จะส่งรหัสข้อมูลไปแทน
2. ระบบการบันทึก เราสามารถลดเนื้อที่ในการบันทึกข้อมูล

ในการบีบอัดรูปภาพนั้นจะมีความแตกต่างไปจากการบีบอัดข้อมูลในระบบเลขฐานสองแน่นอน โปรแกรมบีบอัดข้อมูลทั่วไปสามารถนำมาใช้ในการบีบอัดรูปภาพได้ แต่ผลที่ได้จากการใช้โปรแกรมรูปที่ได้จะมีประสิทธิภาพต่ำกว่า เพราะว่ารูปภาพจะมีคุณสมบัติเกี่ยวทางสถิติที่สามารถนำไปใช้โดยการเข้ารหัสตามชนิดของการออกแบบ อย่างไรก็ตามบางส่วนที่ดีของรูปต้องเสียหายไปในการบันทึกเป็น Bandwidth เล็กๆ หรือบันทึกในพื้นที่น้อยกว่า อย่างไรก็ตามเราสามารถใช้วิธี lossy compression ในพื้นที่ส่วนที่เสียหายไปได้

Lossless compression เป็นการบีบอัดข้อมูล และหลังทำการบีบอัดข้อมูลจะได้ข้อมูลที่เหมือนกับข้อมูลเดิม ในกรณีเมื่อมีการจัดการตารางข้อมูลเลขฐานสอง และเอกสารอื่นๆ เมื่อทำการบีบอัดข้อมูลแล้ว ผลที่ได้จากการทำ decompressed สำเนาข้อมูลนั้น จะได้ข้อมูลที่เหมือนข้อมูลจริงออกมา ในทางตรงข้าม รูปภาพและเพลงก็เหมือนกัน ไม่ต้องการทำสำเนา แต่ต้องการทำให้ใกล้เคียงกับต้นฉบับ ซึ่งความแตกต่างระหว่างรูปต้นฉบับกับรูปที่ทำการบีบอัดแล้วจะมีความเหมือนกันมาก

เทคนิคและวิธีการบีบอัดข้อมูลภาพ 2 มิติ สามารถแบ่งออกเป็น 2 ประเภทคือ

1) time-domain (หรือ space-domain)

2) transform-domain encoding

Time-domain เป็นเทคนิคที่ดูเหมือนเหมาะแก่การปฏิบัติการทำนายเปรียบเทียบ ที่ประกอบด้วยโครงสร้างที่เหมือนการปรับเดลต้าและความแตกต่างของการเปลี่ยนคลื่นชีพจร การศึกษาระบบส่วนมากจะอาศัยทฤษฎีการสื่อสารในอดีต ข้อมูลทฤษฎีแรกเริ่มถูกทำขึ้นโดย Elias เขาได้ตั้งชื่อวิธีการนี้ว่า "predictive coding" ใน predictive coding การถ่ายโอนข้อมูลที่ดีโดย predictor ซึ่งเป็นการเปลี่ยนข้อมูลจากต้นฉบับนั้นไปเป็นรูปแบบสัญลักษณ์และข้อมูลจะเป็นอิสระต่อกัน

Transform-domain เป็นวิธีการจำแนกค่าสัมประสิทธิ์ของข้อมูลตัวอย่างเป็นกลุ่มของฟังก์ชัน orthogonal และทำการส่งค่าสัมประสิทธิ์ไปทำงานต่อ ข้อมูลที่ถูกบีบอัดจะถูกรับโดยการกำจัดค่าสัมประสิทธิ์ที่ไม่สำคัญหรือโดยการลดระดับใน transform domain วิธีนี้เป็นวิธีที่เร็วดังนั้นจะใช้กับ computer ที่มีประสิทธิภาพสูง

ประเภทของการบีบอัดข้อมูลภาพแบ่งได้เป็น 2 แบบคือ

1. การบีบอัดข้อมูลแบบไม่มีการสูญเสีย (Lossless compression หรือ Bit-preserving หรือ Versible compression)

2. การบีบอัดข้อมูลแบบมีการสูญเสียบางส่วน (Lossy Compression หรือ Irreversible compression)

2.2 การเข้ารหัส (Coding)

กระบวนการที่กำหนดให้การจัดลำดับของเลขฐานสอง ไปเป็นสัญลักษณ์ เรียกว่า การเข้ารหัส (Coding) เซตของการจัดลำดับของเลขฐานสอง เรียกว่า รหัส และเลขที่เป็นเลขเฉพาะของเซต เรียก รหัสคำ (codeword)

การเข้ารหัสที่เราสนใจ คือ การเข้ารหัสเอโทรปี (Entropy coding) คือการลดความฟุ่งเฟือยด้วยวิธีการบีบอัดข้อมูลโดยไม่สนใจถึงความหมายของข้อมูล เพียงแต่มองข้อมูลเป็นบิต 0 หรือ 1 ซึ่งในการบีบอัดภาพที่นิยมอยู่ คือ การเข้ารหัสแบบฮัฟฟ์แมน (Huffman coding)

2.3 การเข้ารหัสแบบฟรีดิกทีฟ (Predictive Coding)

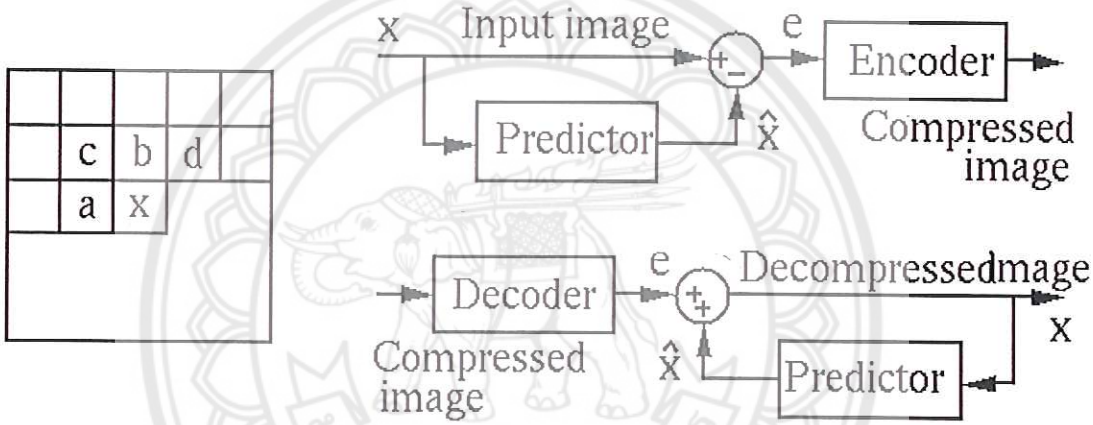
Predictive coding เป็นการเข้ารหัสรูปภาพอีกประเภทหนึ่งที่เป็นแบบ Time domain ที่จะได้รูปภาพที่มีเป็นอิสระต่อกันหลังจากการเข้ารหัสแล้ว และเป็นแบบ lossless ซึ่งจะได้ผลลัพธ์หลังจากการ decode แล้วจะเหมือนกับภาพก่อนการ encoding

โดยทั่วไปสัญลักษณ์จะมีความใกล้เคียงกันมาก และความแตกต่างนั้นจะเป็นส่วนที่เล็กมากๆ ที่เราให้แทนด้วยค่าพิกเซลล์ x ที่มีความผิดพลาดน้อยมากๆ หาได้จากตัวที่ล้อมรอบพิกเซลล์ x ทั้ง 4 โดยแทนค่าแต่ละด้านเป็น a, b, c และ d ตามลำดับ

$$e_1 = x - a$$
$$e_2 = x - b$$
$$e_3 = x - \frac{a+b+c+d}{4}$$

โดยทั่วๆไป Predicted value $\hat{x}_0$ จากพิกเซลล์ x_0 เป็น ผลรวมเชิงเส้นของค่าที่อยู่ติดกับบริเวณจุดพิกเซลล์ที่มีมาให้ a_i (i= 1, 2 , 3 ,.....,n)

$$\hat{x}_0 = \sum_{j=i}^{n} a_j x_j$$



รูปที่1 แสดงขั้นตอนในการเข้ารหัส

Entropy ของ histogram ของรูปภาพที่มีการเข้ารหัส จะมีขนาดเล็กกว่ารูปต้นฉบับ อย่างไรก็ตาม Huffman coding จะยังส่งผลกระทบต่อต้นฉบับเดิมอยู่

Optimal predictive coding

ความผิดพลาดหลักของ Predictive ซึ่งจะมีลักษณะความผิดพลาดเป็นสี่เหลี่ยมจตุรัส คือ

$$E(e^2) = E[(x_0 - \hat{x}_0)^2] = E(x_0^2) - 2E(x_0\hat{x}_0) + E(\hat{x}_0^2)$$

ที่ซึ่ง

$$\hat{x}_0 = \sum_{j=i}^{n} a_j x_j$$

การหาค่าสัมประสิทธิ์ของค่า a_i ได้จากค่าน้อยที่สุดของ $E(e^2) \rightarrow min$ ได้ว่า

$$\frac{\partial}{\partial a_i} E(e^2) = \frac{\partial}{\partial ai} E(\hat{x}_0^2) - 2\frac{\partial}{\partial a_i} E(x_0\hat{x}_0) = 0$$

จาก

$$2\frac{\partial}{\partial a_i}E(x_0\hat{x}_x) = 2E(x_0\frac{\partial}{\partial ai}\sum_{j=1}^{n}a_j x_j) = 2E(x_0 \cdot x_i) = 0$$

และ

$$\frac{\partial}{\partial a_i}E(\hat{x}_0{}^2) = 2E(\hat{x}_0\frac{\partial}{\partial ai}\hat{x}_0) = 2E(\sum_{j=1}^{n}a_j x_j \cdot x_i) = 2\sum_{j=1}^{n}a_j E(x_j \cdot x_i)$$

จะได้

$$E(x_0 \cdot x_1) = \sum_{j=1}^{n}a_j E(x_j \cdot x_i)$$

กำหนดให้

$$E(x_i x_j) = r_{ij}$$

ความสัมพันธ์ระหว่าง x_i กับ x_j สามารถคาดคะเนได้จากผลการทดลองหลายๆครั้ง

$$\hat{r}_{ij} = \hat{E}(x_i x_j) = \frac{1}{k}\sum_{l} x_i^{(l)} x_j^{(l)}$$

ค่าที่ทำนายจะดีที่สุดในค่านี้

$$r_{0i} = \sum_{j=1}^{n}a_j r_{ij} \qquad (j = 1,\ldots,n)$$

และสามารถแสดงในรูปแบบของเวกเตอร์

$$r_0 = Ra$$

ที่ซึ่ง

$$r_0 = \begin{bmatrix} r_{01} \\ \cdots \\ r_{0n} \end{bmatrix}, \qquad R = \begin{bmatrix} \cdots & \cdots & \cdots \\ \cdots & r_{ij} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$$

ดังนั้นค่าสัมประสิทธ์สามารถหาได้จาก

$$a = R^{-1}r_0$$

เราสามารถลดข้อผิดพลาดของข้อมูลที่เพิ่มขึ้นได้จากสมการดังต่อไปนี้

$$\sum_{j=1}^{n}a_j < 1$$

ตัวอย่าง

$$\hat{x}(i,j) = 0.9x(i,j-1)$$

$$\hat{x}(i,j) = 0.9x(i-1,j)$$

$$\hat{x}(i,j) = 0.7x(i,j-1) + 0.7x(i-1,j) - 0.5x(i-1,j-1)$$

$$\hat{x}(i,j) = \begin{cases} 0.9x(i,j-1) & if \quad \Delta h \le \Delta v \\ 0.9x(i-1,j) & else \end{cases}$$

ที่ซึ่ง

$$\Delta h = \left|x(i-1,j) - x(i-1,j-1)\right|, \quad \Delta v = \left|x(i,j-1) - x(i-1,j-1)\right|$$

2.4 การเข้ารหัสแบบฮัฟฟ์แมน (Huffman coding)

กระบวนการนี้ถูกพัฒนาขึ้นโดย เดวิท ฮัฟฟ์แมน(David Huffman) ซึ่งเป็นสมาชิกในทฤษฎีข่าวสารแรกของ โรเบิร์ต ฟาโน (Robert Fano) ที่ MIT การเกิดรหัสโดยการใช้กระบวนการนี้ จึงเรียกว่ารหัสของฮัฟฟ์แมน เป็นรหัสแบบ prepix code และเป็นสถานการณ์ที่ใช้แบบตัวอย่าง (เซตของความน่าจะเป็น)

2.4.1 กระบวนการของฮัฟฟ์แมนมีข้อสังเกตุพื้นฐาน 2 ข้อ คือ

1. ในรหัสข้อมูลปัจจุบัน สัญลักษณ์ที่มีอัตราการเกิดบ่อยครั้ง (มีความน่าจะเป็นในการเกิดสูง) จะมีจำนวนบิตของรหัสข้อมูลสั้นลง

2. ในรหัสข้อมูลปัจจุบัน สัญลักษณ์สองตัวที่มีการปรากฏน้อยที่สุดมีความยาวเท่ากัน

วิธีการสังเกตุ คือถ้าสัญลักษณ์มีอัตราการเกิดสูงขึ้น จะมีรหัสคำยาวกว่า จำนวนเฉลี่ยบิตต่อสัญลักษณีมากกว่า ในเงื่อนไขการเปลี่ยนตำแหน่ง ดังนั้นจึงมีการกำหนดให้ รหัสคำที่ยาวกว่าเป็นสัญลักษณ์ที่มีความถี่ในการเกิดสูงกว่า ไม่สามารถเป็น รหัสปัจจุบัน

ในการสังเกตุอีกวิธีหนึ่ง พิจารณาตามตำแหน่ง ในการคาดคะเน จากรหัสที่เข้ามาที่มีรหัสคำเหมือนกันมีความถี่ในการเกิดต่ำสัญลักษณ์จะไม่มีความยาวเท่ากัน สมมุติให้ความยาวของ รหัสคำคือ k บิต ยาวกว่ารหัสคำที่สั้น รหัสคำที่สั้นกว่าจะไม่เป็น prefix ของรหัสคำที่ยาวกว่า หมายความว่าถ้าเราตกอยู่ที่ k บิตตัวสุดท้ายของรหัสคำที่ยาวกว่ารหัสคำตัวหนึ่งจะมีความชัดเจนขึ้น

เนื่องจาก รหัสคำที่เหมือนกันที่มีอัตราการเกิดต่ำที่สุด ในสัญลักษณีที่เป็นกลุ่มตัวอักขระ ไม่มีรหัสคำตัวใดที่ยาวกว่าตัวนี้ได้ ดังนั้นจึงไม่มีผลกระทบอันใดในการทำให้รหัสคำสั้นลง โดยจะเริ่มที่การเข้ารหัสฟริฟิก ของตัวอื่น เมื่อทำการตกที่ k บิตเราจะให้รหัสใหม่ที่มีความยามเฉลี่ยสั้นกว่าเดิม

2.4.2 กระบวนการการเข้ารหัสแบบฮัฟฟ์แมน

จะมีวิธีการโดยรวมค่าอัตราการเกิดต่ำที่สุด จะมีความแตกต่างที่บิตสุดท้าย จะมีความแตกต่างที่บิตสุดท้ายเท่านั้น นั่นคือ γ ถ้า δ และ คือสัญลักษณ์ 2 ตัวสุดท้ายที่มีอัตราการเกิดต่ำที่สุด ในสัญลักษณ์ของตัวอักขระ และถ้ารหัสคำของ γ คือ m*0 รหัสคำของ δ รหัสของจะเป็น m*1 โดยที่ m คือค่าของ 1s และ0s และ*

ข้อมูลแต่ละตัวจะถูกแปลงไปเป็น ชุดของ ตัวเลข 0 และ1 ซึ่งจะเป็นตัวแสดงถึง เส้นทาง ของมันในแผนภูมิรูปต้นไม้ของ การเข้ารหัสของ การเข้ารหัสข้อมูล แบบฮับฟ์แมน (Huffman encoding) โดยข้อมูลแต่ละตัวจะมีความยาวของชุดตัวเลขดังกล่าวต่างกันไปขึ้กับค่าความถี่ของมันที่ปรากฏในข้อมูลทั้งหมด

ข้อมูลที่มีความถี่สูงสุด จะมีความยาวของชุดตัวเลขน้อยสุดคือ 2 ตัว ส่วนข้อมูลที่มีความถี่ต่ำสุดจะมีความยาวของชุดตัวเลข 2n -1 ตัวเมื่อ n คือค่าบิตของการเก็บข้อมูลปกติ

ยกตัวอย่างเช่น ข้อมูล 8 บิตมีข้อมูลที่เป็นไปได้ 256 ค่า (0-255) เมื่อเข้ารหัสจะมีความยาวของชุดตัวเลขเป็นได้ตั้งแต่ 2 ตัวจนถึง 128 ตัว

ในการถอดรหัสข้อมูลดูจากเส้นทางของข้อมูลเริ่มจากส่วนยอดลงไปเรื่อยๆจนครบตัวเลขที่เป็นไปได้สำหรับข้อมูลตัวหนึ่งๆ จากนั้นจึงเริ่มดูจากส่วนยอดลงมาสำหรับข้อมูลตัวถัดไปเป็นเช่นนี้ไปเรื่อยๆ

ถึงแม้การเข้ารหัสแบบฮัฟฟ์แมน มักจะช่วยทำให้ความจุของข้อมูล ลดลงไม่มากนักแต่มันจะช่วยรักษาตัวข้อมูลเดิมไว้ได้ทั้งหมด เมื่อถอดรหัสออกมา

ตัวอย่าง การทำงานการเข้ารหัสแบบฮัฟฟ์แมน โดยการใช้ Table

ตารางที่ 1

Letter	Probability	Codeword
a1	0.2	c(a1)
a2	0.4	c(a2)
a3	0.2	c(a3)
a4	0.1	c(a4)
a5	0.1	c(a5)

ขั้นตอนที่ 1 ทำการเรียงสัญลักษณ์ใหม่โดยเรียงจากอัตราการเกิด จากมากไปน้อย

ตารางที่ 2

Letter	Probability	Codeword
a2	0.4	c(a2)
a1	0.2	c(a1)
a3	0.2	c(a3)
a4	0.1	c(a4)
a5	0.1	c(a5)

ขั้นตอนที่2 สัญลักษณ์ที่มีอัตราการเกิดต่ำที่สุด 2 ตัวสุดท้าย ในที่นี้คือ a4 และ a5 นำค่าอัตราการเกิดรวมกันจะได้ค่า ของอัตราการเกิดใหม่ และกำหนดให้ รหัสคำใหม่ คือ α ดังตารางที่ 3

ตารางที่ 3

Letter	Probability	Codeword
a2	0.4	c(a2)
a1	0.2	c(a1)
a3	0.2	c(a3)
a4,a5	0.2	$\alpha 1$

ทำขั้นตอนที่ 1 และ 2 ไปเรื่อยๆ จนกว่าจะได้ค่าที่เหลือสุดท้าย 2 ค่า ดังตารางที่ 4 และ 5

ตารางที่ 4

Letter	Probability	Codeword
a2	0.4	c(a2)
a3,a4,a5	0.4	$\alpha 2$
a1	0.2	c(a1)

ตารางที่ 5

Letter	Probability	Codeword
a3,a4,a5,a1	0.6	$\alpha 3$
a2	0.4	c(a2)

ขั้นตอนที่ 3 กำหนดค่าโคดเวิร์ดให้กับสองค่าสุดท้ายโดยให้ค่าที่มีอัตราการเกิดสูงสุดมีค่าเท่ากับ 0 และค่าที่มีอัตราการเกิดต่ำกว่ามีค่าเท่ากับ 1

$$\alpha 3 = 0$$
$$c(a2) = 1$$

จาก $\alpha 3$ คือ ค่าของ $\alpha 2 + c(a1)$ ดังนั้นจะได้

$$\alpha 2 = (\alpha 3 * 0) = 00$$
$$c(a1) = (\alpha 3 * 1) = 01$$

จาก $\alpha 3$ คือ ค่าของ $c(a3) + \alpha 1$ ดังนั้นจะได้

$$c(a2) = (\alpha 2 * 0) = 000$$
$$\alpha 1 = (\alpha 2 * 1) = 001$$

จาก $\alpha 3$ คือ ค่าของ $c(a4) + c(a5)$ ดังนั้นจะได้

$$c(a4) = (\alpha 1 * 0) = 0010$$
$$c(a5) = (\alpha 1 * 1) = 0011$$

เราจะได้ผลของการถอดรหัสแบบฮัฟฟ์แมนได้ดังตารางที่ 6

ตารางที่ 6

Letter	Probability	Codeword
a2	0.4	1
a1	0.2	01
a3	0.2	000
a4	0.1	0010
a5	0.1	0011

2.5 อัตราส่วนของการบีบอัดข้อมูล (Compression Ratio:CR)

คือ การเปรียบเทียบค่าระหว่างขนาดของข้อมูลต้นฉบับ (n1) กับข้อมูลที่ถูกบีบอัดข้อมูลแล้ว (n2)

$$CR = \frac{n1}{n2}$$

2.6 การหาความผิดพลาดของการบีบอัดข้อมูล

เราสามารถใช้ค่าของ Signal to Noise Ratio (SNR) และ Peak Signal to Noise Ratio (PSNR) โดยจะเป็นตัววัดคุณภาพของการบีบอัดข้อมูลว่า เมื่อบีบอัดข้อมูลแล้วคุณภาพของภาพที่ได้ใหม่จะมีการสูญเสียมากน้อยเพียงใด

SNR ถูกนิยามในหน่วย (dB) ดังต่อไปนี้

$$SNR_{(dB)} = 10\log_{10}\left\{\frac{\left[\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\left(s(m,n)\right)^2\right]}{\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\left(s(m,n)-\hat{s}(m,n)\right)^2}\right\}$$

โดยที่ $s(m,n)$ และ $\hat{s}(m,n)$ คือ ค่าข้อมูลภาพต้นฉบับและภาพที่ถูกคลายหลังการบีบอัดแล้วตามลำดับ สำหรับ m และ n เป็นค่าดัชนีตามแนวนอนและแนวตั้งของภาพทั้ง2 หรืออาจวัดค่าความผิดพลาดของการบีบอัดข้อมูลในรูปแบบของ PSNR ดังต่อไปนี้

$$PSNR_{(dB)} = 10\log_{10}\left\{\frac{\max|s(m,n)|}{\frac{1}{MN}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\left(s(m,n)-\hat{s}(m,n)\right)^2}\right\}$$

โดยที่ $\max|s(m,n)|$ คือ ค่าสูงสุดของข้อมูลภาพต้นแบบและค่า MN คือ ขนาดของข้อมูลภาพ

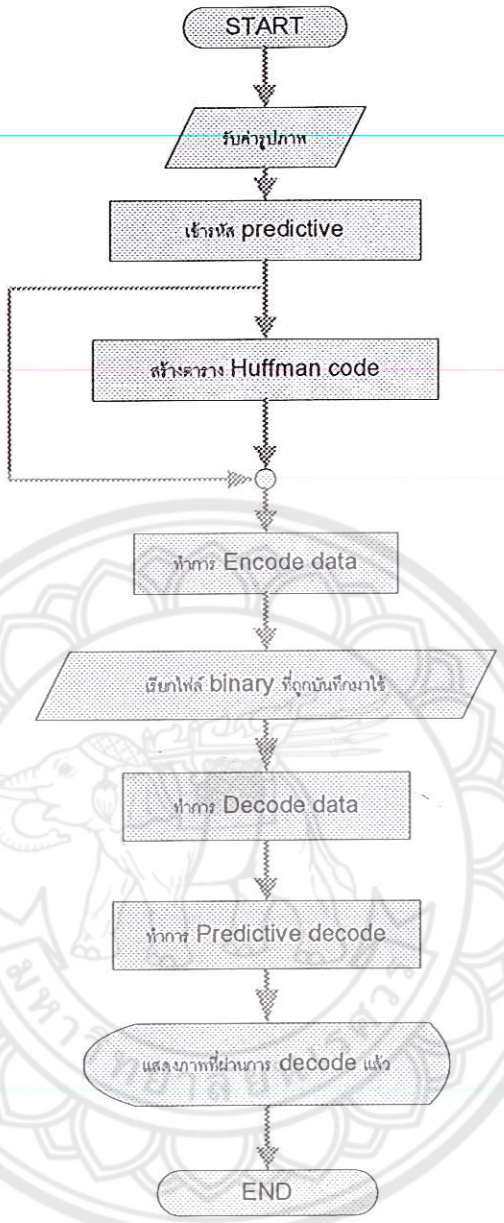
บทที่ 3
ขั้นตอนการดำเนินโครงงาน

ในการบีบอัดข้อมูลภาพโดยวิธีการฮัฟฟ์แมน ได้แบ่งขั้นตอนการโครงงานออกเป็น 4 ขั้นตอน โดยเริ่มจากการศึกษาอัลกอริทึมการเข้ารหัส Predictive และ Huffman และขั้นตอนการออกแบบ เขียนโปรแกรม Predictive coding ก่อนและหลังเข้ารหัส Huffman โปรแกรมสร้างตารางรหัสฮัฟฟ์แมน เขียนโปรแกรม Encode Huffman โปรแกรม Decode Huffman ดังต่อไปนี้

1. ศึกษาอัลกอริทึมการเข้ารหัส predictive และ Huffman และขั้นตอนการออกแบบ
2. เขียนโปรแกรม predictive
3. เขียนโปรแกรม สร้างตาราง Huffman
4. เขียนโปรแกรม เข้ารหัสและถอดรหัส Huffman
5. ทดสอบโปรแกรมและแก้ไข

1.1 ศึกษาอัลกอริทึมการเข้ารหัส Predictive coding และ Huffman coding และขั้นตอนการออกแบบ

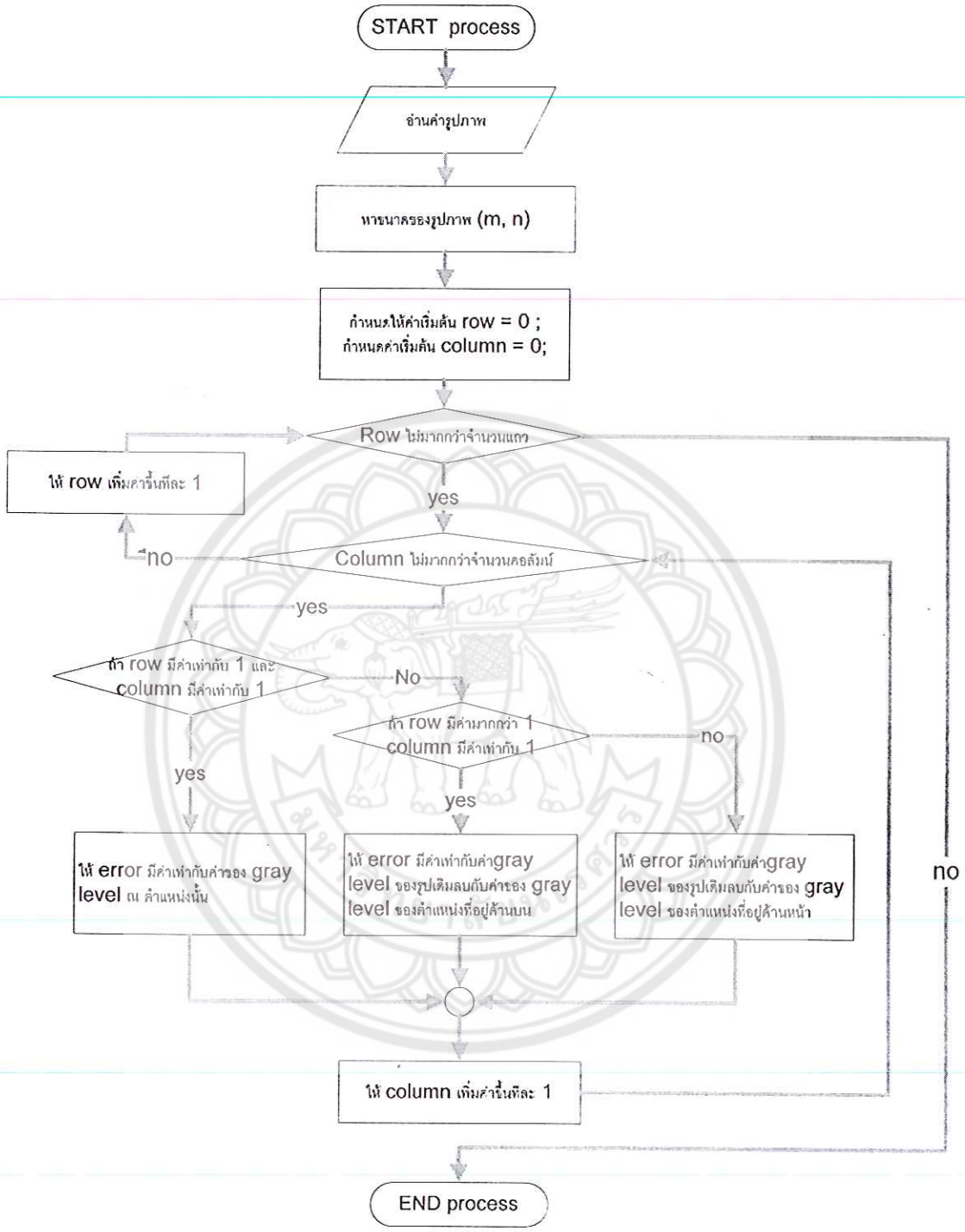
จากบทที่ 2 ศึกษาทฤษฎีและอัลกอลิทึมที่เกี่ยวข้อง เราสามารถนำทฤษฎีและอัลกอลิทึมที่ได้มาออกแบบโปรแกรมการบีบอัดข้อมูลภาพด้วยวิธีการฮัฟฟ์แมนดังแสดงในรูปที่3.1 และนำมาเขียนโปรแกรม Predictive encode, predictive decode, Huffman table, Huffman encode, Huffman decode โปรแกรมสามารถดูได้ที่ภาคผนวก ได้แสดงในหัวข้อต่อไปนี้



รูปที่ 3.1 Flowchart โปรแกรมบีบอัดข้อมูลภาพแบบฮัฟฟ์แมน

1.2 โปรแกรม ส่วนของ encode predictive

ส่วนของโปรแกรมในการ เข้าโคด Predictive แสดงดังรูปที่ 3.2 เป็นการรับค่าเข้ามาแปลงให้ค่ามีความเหมือนกันมากขึ้น โดยใช้ สมาการ $e = f - a$ และ $e = f - b$ จากทฤษฎี predictive ในบทที่ 2 ทำให้ได้ค่าใหม่มีค่าจำนวนแตกต่างกันน้อยลง และเก็บไว้ในตัวแปล ซึ่งจะนำไปใช้ต่อในการ เข้ารหัส Huffman ในหัวข้อต่อไป



รูปที่ 3.2 Flowchart ของโปรแกรมในส่วนของ encode predictive

<u>ตัวอย่าง</u> ค่าที่นำมาเข้าโคด ขนาด 6 X 6 ทางด้าน ซ้ายจะเป็นก่อนเข้า predictive ส่วนทางด้านขวาจะเป็นค่าหลังเข้า predictive ดังตารางที่ 3.1

10	20	30	40	50	60
20	30	40	50	60	50
30	40	50	60	50	40
40	50	60	50	40	30
50	60	50	40	30	20
60	50	40	30	20	10

10	10	10	10	10	10
10	10	10	10	10	-10
10	10	10	10	-10	-10
10	10	10	-10	-10	-10
10	10	-10	-10	-10	-10
10	-10	-10	-10	-10	-10

ตาราง 3.1 แสดงค่าก่อนและหลังเข้ารหัส Predictive

1.3 โปรแกรม ส่วนของ การสร้างตาราง Huffman

การสร้างตาราง Huffman จะนำค่าจากตัวแปลที่เก็บไว้ ซึ่งได้จากส่วนของ encode predictive มาหาค่า probability ของค่าทั้งหมด แล้วทำการจัด sort ค่า probability จากค่ามากไปหาค่าน้อย เก็บไว้ในตัวแปร อาเรย์ตัวใหม่ แล้วนำค่าที่ได้มาทำตามทฤษฎี Huffman โดยจะกำหนดค่าตารางออกมาเป็นอาเรย์ ขนาดเท่ากับ (จำนวนของsymbol X 9) ซึ่ง จะแบ่งออกเป็น 2 ส่วน คือ

1. ส่วนที่ทำการสร้าง Binary Tree
2. ส่วนของการอ่านค่าจาก Binary Tree

จาก 9 Columns ที่สร้างขึ้น จะใช้เก็บค่าดังต่อไปนี้

Column 1 ใช้เก็บค่า numbers

Column 2 ใช้เก็บค่า symbols

Column 3 ใช้เก็บค่า probability

Column 4 ใช้เก็บค่าที่อยู่ต่อจากทางซ้ายของ parent เริ่มต้นทุกค่าจะเท่ากับ 0

Column 5 ใช้เก็บค่าที่อยู่ต่อจากทางขวาของ parent เริ่มต้นทุกค่าจะเท่ากับ 0

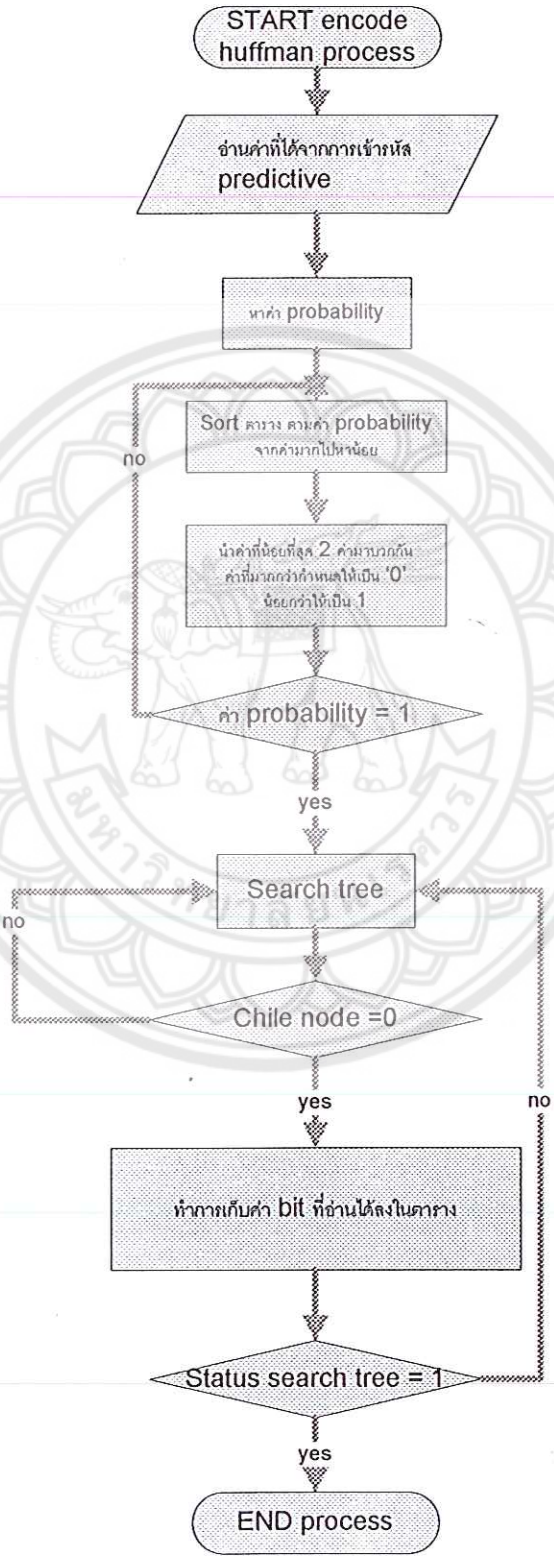
Column 6 ใช้เก็บค่าที่แสดงถึงตัวที่เป็น parent ของ child

Column 7 ใช้เก็บค่าที่จะกำหนดให้ค่ามากให้เป็น '0' น้อยให้เป็น '1' ตรวจจากการบวกกันระหว่าง 2 ค่า ที่น้อยที่สุด

Column 8 ใช้เก็บค่าสถานะในการใช้ไปของ แต่ละ columns เริ่มต้น ทุกค่าเป็น '0' หลังใช้เป็น '1'

Column 9 ใช้เก็บค่าสถานะในการใช้นับค่า bit จาก columns 7 โดย จะมีสถานะเริ่มต้นเป็น '0' หลังใช้ เป็น '1'

ซึ่งจะแสดง Flowchart ของ โปรแกรมในส่วนนี้ ดังรูปที่ 3.3 และจะนำค่าที่ได้เก็บไว้ในตารางใหม่ ซึ่งจะใช้ในส่วนของการ encode Huffman ต่อไป



รูปที่ 3.3 Flowchart ในส่วนของการสร้างตาราง Huffman

ตัวอย่าง ขั้นตอนการสร้างตาราง Huffman dic

ขั้นตอนที่ 1 จัดเรียงลำดับค่าจากค่า Probability จากค่ามากไปหาค่าน้อย พร้อมกับสร้าง ตาราง 9 columns ดังตารางที่ 3.2

No.	symbol	probability	Left node	Right node	parent	Bit=branch	Status create tree	Status search tree
1	A2	0.4	0	0	0		0	0
2	A1	0.2	0	0	0		0	0
3	A3	0.2	0	0	0		0	0
4	A4	0.1	0	0	0		0	0
5	A5	0.1	0	0	0		0	0

ตารางที่ 3.2 ตารางในการสร้าง Huffman table dic

ขั้นตอนที่ 2 ตรวจสอบ status create tree ใน column 8 แล้วนำค่าน้อยที่สุด 2 ค่าที่ยังไม่ได้ใช้มาบวกกันไปเรื่อยๆ พร้อมเปลี่ยน status create tree ใน Column 8 ให้เป็น '1' หลังจากนำค่ามาบวกกัน ทำการกำหนดค่า bit ให้กับ 2 ค่า โดยค่าน้อยกว่าเป็น '1' มากกว่าเป็น '0' กำหนด parent ของตัวมันเองโดยที่มีเงื่อนไขคือ แถวที่นำค่าทั้ง 2 มาบวกกันจะเป็นถือว่าเป็น parent ของ 2 แถวที่มาบวกกัน ดังตารางที่ 3.3

No.	symbol	probability	Left node	Right node	parent	Bit= branch	Status create tree	Status search tree
1	A2	0.4	0	0	0	-	0	0
2	A1	0.2	0	0	0	-	0	0
3	A3	0.2	0	0	0	-	0	0
4	A4	0.1	0	0	6	0	1	0
5	A5	0.1	0	0	6	1	1	0
6	0	0.2	4	5	0	-	0	0

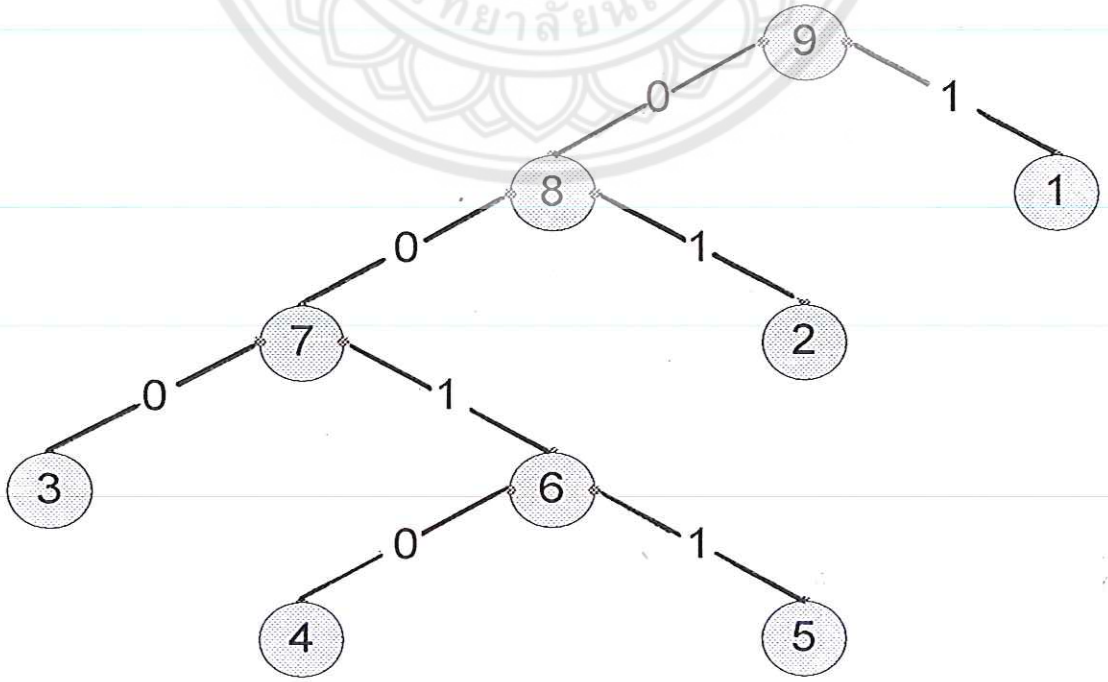
ตารางที่ 3.3 การสร้าง Huffman tree

<u>ขั้นตอนที่ 3</u> ทำตามขั้นตอนที่ 2 ไปเรื่อยๆ จนค่า probability ใน column 3 มีเท่ากับ '1.00' เมื่อมีการตรวจสอบ ซึ่งผลจะได้มาดังตัวอย่างในตารางที่ 3.4

No.	symbol	probability	Left node	Right node	parent	Bit=branch	Status create tree	Status search tree
1	A2	0.4	0	0	9	1	1	0
2	A1	0.2	0	0	8	1	1	0
3	A3	0.2	0	0	7	0	1	0
4	A4	0.1	0	0	6	0	1	0
5	A5	0.1	0	0	6	1	1	0
6	0	0.2	4	5	7	1	1	0
7	0	0.4	3	6	8	0	1	0
8	0	0.6	7	2	9	0	1	0
9	0	1.0	8	1	0	-	0	0

ตาราง 3.4 การสร้าง Huffman tree

เมื่อได้ตารางที่ 3.4 แล้วเราจะได้ Huffman tree ดังรูปที่ 3.5



รูปที่ 3.4 Huffman tree ที่ได้จากตารางที่3.4

ขั้นตอนที่ 4 เมื่อได้ Huffman tree จะทำการอ่านค่า bit หรือ branch ของ Huffman tree ซึ่งจะอยู่ใน column ที่ 7 โดยตัวโปรแกรมจะอ่านลงมาทางซ้ายของ tree ก่อน ซึ่งจะได้ค่า bit ของแต่ละ symbol ดังตารางที่ 3.5

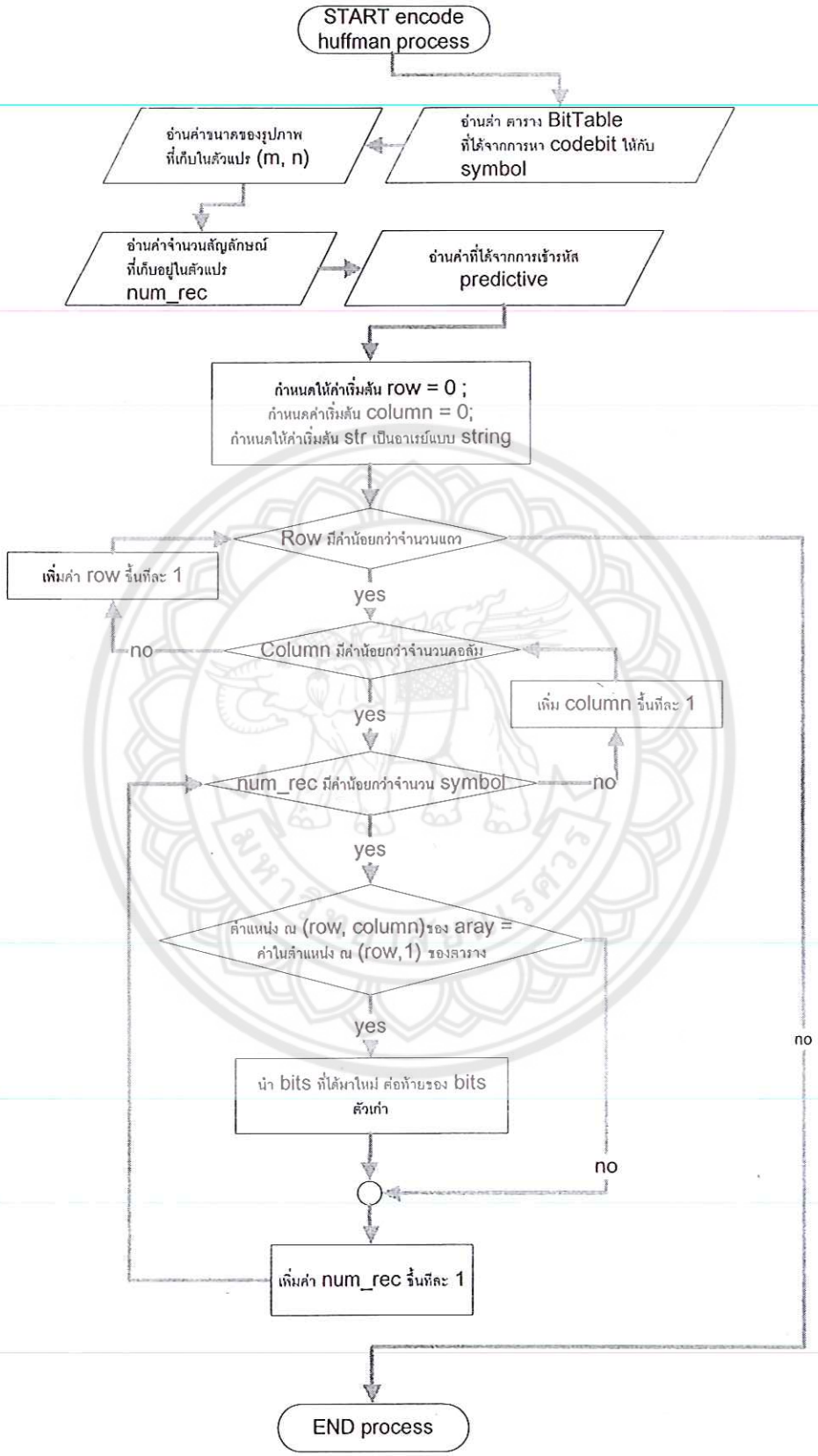
No.	symbol	probability	Code bit
1	A2	0.4	1
2	A1	0.2	01
3	A3	0.2	000
4	A4	0.1	0010
5	A5	0.1	0011
6	0	0.2	001
7	0	0.4	00
8	0	0.6	0
9	0	1.0	-

ตารางที่ 3.5 ตาราง Huffman code ที่ได้จากการ search tree

จากตารางที่ 3.5 เมื่อเรานำ Code bit ไปเทียบกับตารางที่ 6 ในบทที่ 2 ที่เป็นส่วนของ code word เราจะได้ค่า code ของแต่ละ symbol เหมือนกัน คือ A2 = 1, A1 = 01, A3 = 000, A4 = 0010, A5 = 0011

1.4 โปรแกรม ส่วนของ encode Huffman

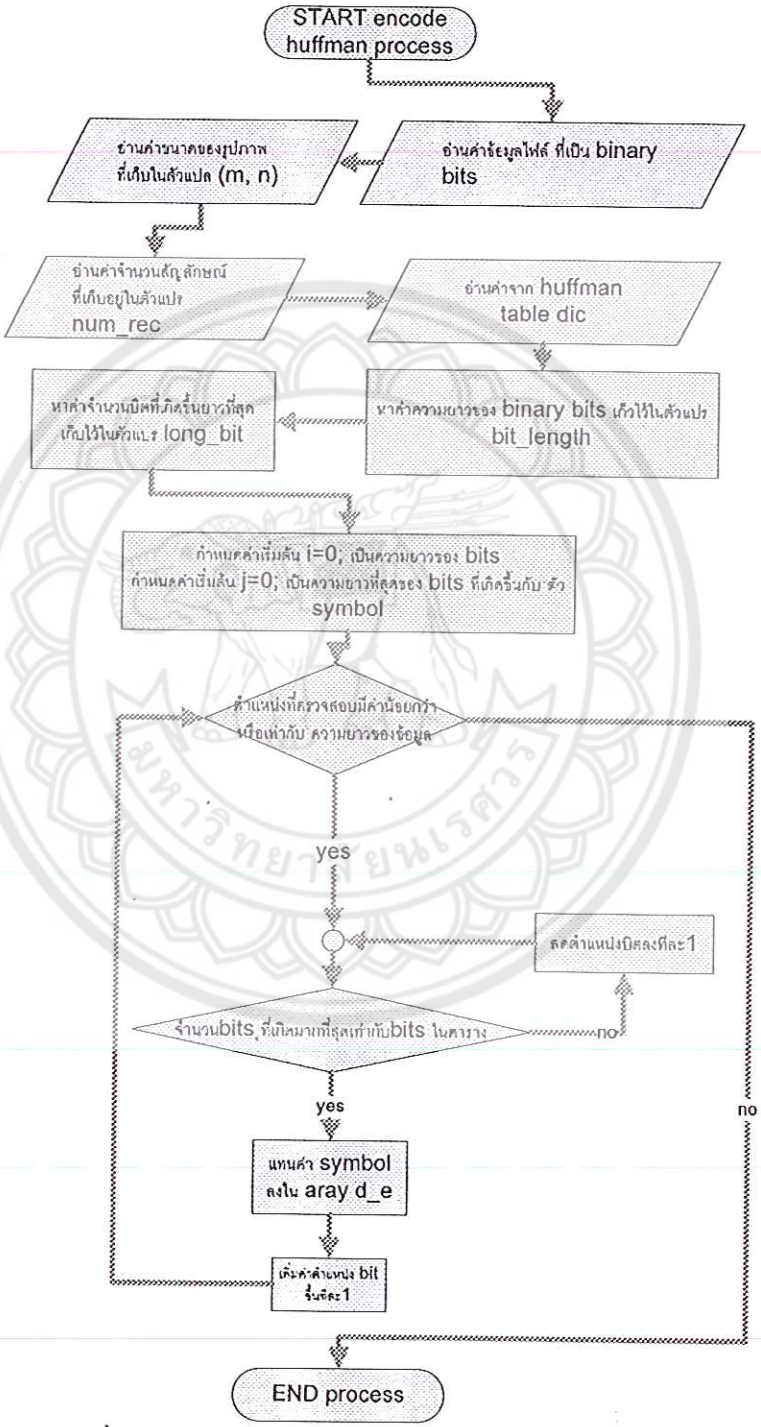
โปรแกรมในส่วนนี้จะใช้ในการเข้ารหัส Huffman ทำการเขียนข้อมูลที่ได้จากการ encode Huffman มาเชื่อมต่อกันไปเรื่อยๆ จนจบการ encode Huffman แล้วจะเก็บไว้เป็น array แบบ string ขั้นตอนการทำงานของฟังก์ชันในส่วนการเข้ารหัสฮัฟฟ์แมน จะแสดงดังรูปที่ 3.5



รูปที่ 3.5 Flowchart ในส่วนของการ encode Huffman

1.5 โปรแกรม ส่วนของ decode Huffman

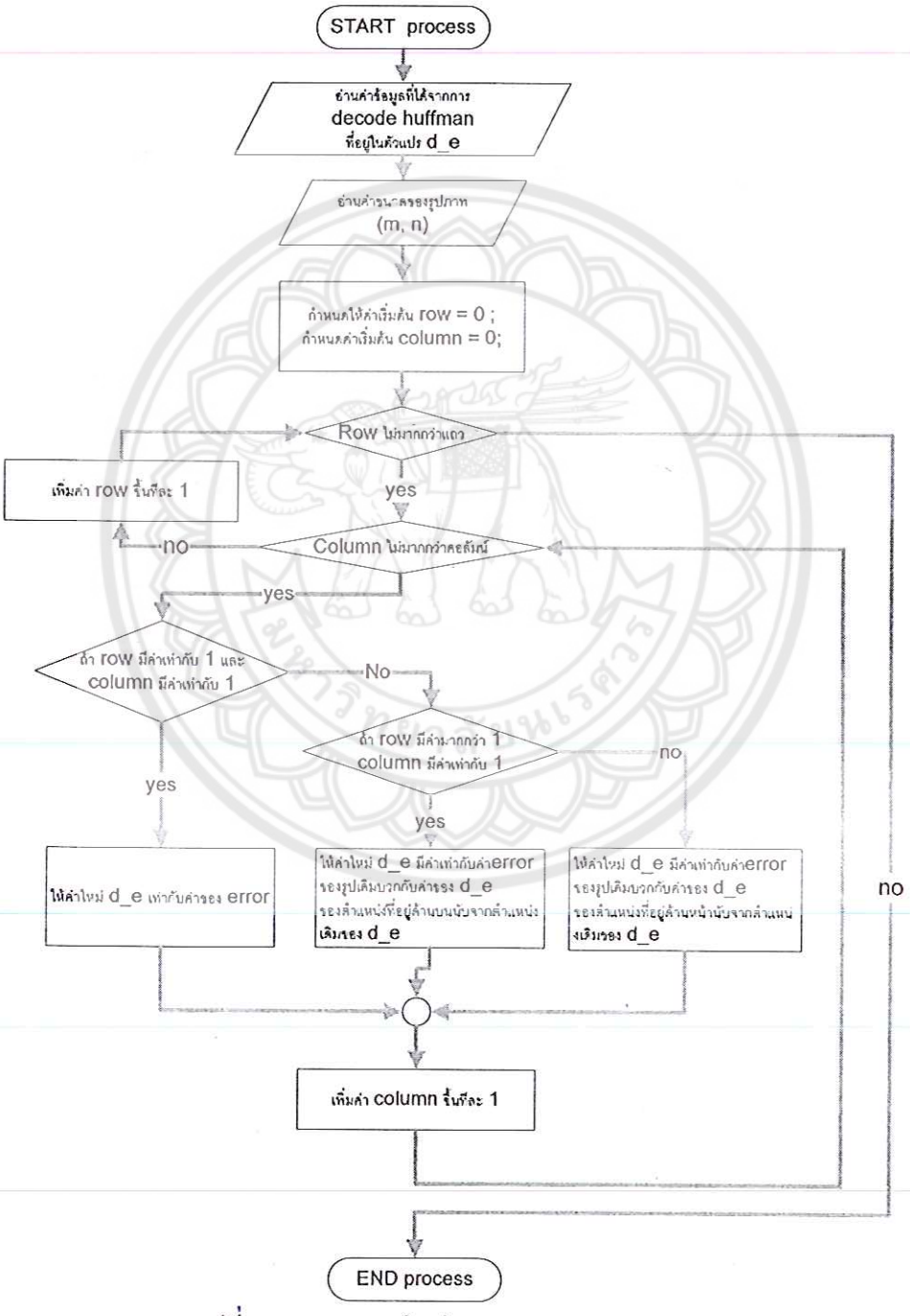
ส่วนของการ decode Huffman จะรับค่าที่ได้จากการ encode Huffman แล้วนำมา decode Huffman ให้กลับไปอยู่ในรูปของ อาเรย์ตามขนาดของรูป ดังจะแสดงขั้นตอนในการทำงานดังรูปที่ 3.6



รูปที่ 3.6 Flowchart ในส่วนของการ decode Huffman

1.6 โปรแกรม ส่วนของ decode predictive

ส่วนของโปรแกรมในการ decode Predictive แสดงดังรูปที่ 3.7 เป็นการรับค่าหลังจากการ decode Huffman แล้ว มาทำการแปลงค่าให้กลับไปอยู่ในรูปดังเดิม โดยใช้ สมการ $f = e + a$ และ $f = e + b$ จากทฤษฎี predictive ในบทที่ 2 ทำให้ได้ค่าเท่ากับค่าเริ่มต้น



รูปที่ 3.7 Flowchart ในส่วนของการ decode predictive

10	10	10	10	10	10
10	10	10	10	10	-10
10	10	10	10	-10	-10
10	10	10	-10	-10	-10
10	10	-10	-10	-10	-10
10	-10	-10	-10	-10	-10

10	20	30	40	50	60
20	30	40	50	60	50
30	40	50	60	50	40
40	50	60	50	40	30
50	60	50	40	30	20
60	50	40	30	20	10

ตาราง 3.6 แสดงค่าก่อนและหลังถอดรหัส Predictive

บทที่ 4
การทดลอง

4.1 วัตถุประสงค์การทดลอง

1. ทดลองการบีบอัดข้อมูลภาพโดยวิธี predictive coding เพื่อลดจำนวนบิตของข้อมูลภาพ
2. ทำการหาค่าอัตราส่วนในการลดของการลดข้อมูล

4.2 การเรียกใช้โปรแกรม

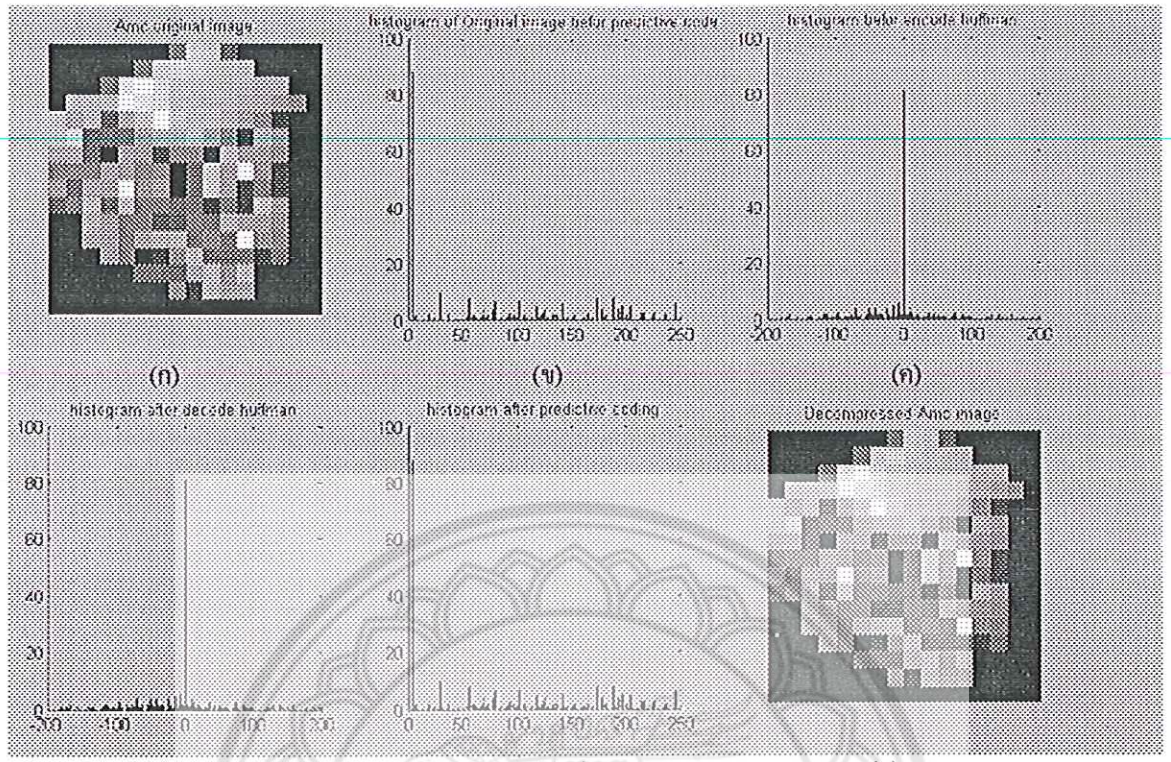
โปรแกรมการบีบอัดข้อมูลภาพโดยวิธี predictive coding มีขั้นตอนการทำงานดังรูปที่ 3.2 (โปรแกรมสามารถดูได้ที่ภาคผนวก) โดยเรียกใช้โปรแกรมจาก predictive.m

4.3 การทดลองเข้ารหัสและการถอดรหัส Predictive coding

การทดลองจะทำการเข้ารหัส predictive แล้ว ทำการ encode Huffman และทำการ decode Huffman แล้วถอดรหัส predictive โดยใช้โปรแกรม MATLAB เปรียบเทียบค่าระหว่าง ข้อมูลรูปเดิมกับข้อมูลที่ผ่าน Huffman โดยจะวัดจากอัตราการบีบอัดข้อมูลภาพ โดยไฟล์ตัวอย่างที่ใช้ในการทดลองเป็นไฟล์ที่เป็นไฟล์ภาพ ดังนี้

4.3.1 ผลการทดลองการบีบอัดข้อมูลภาพ

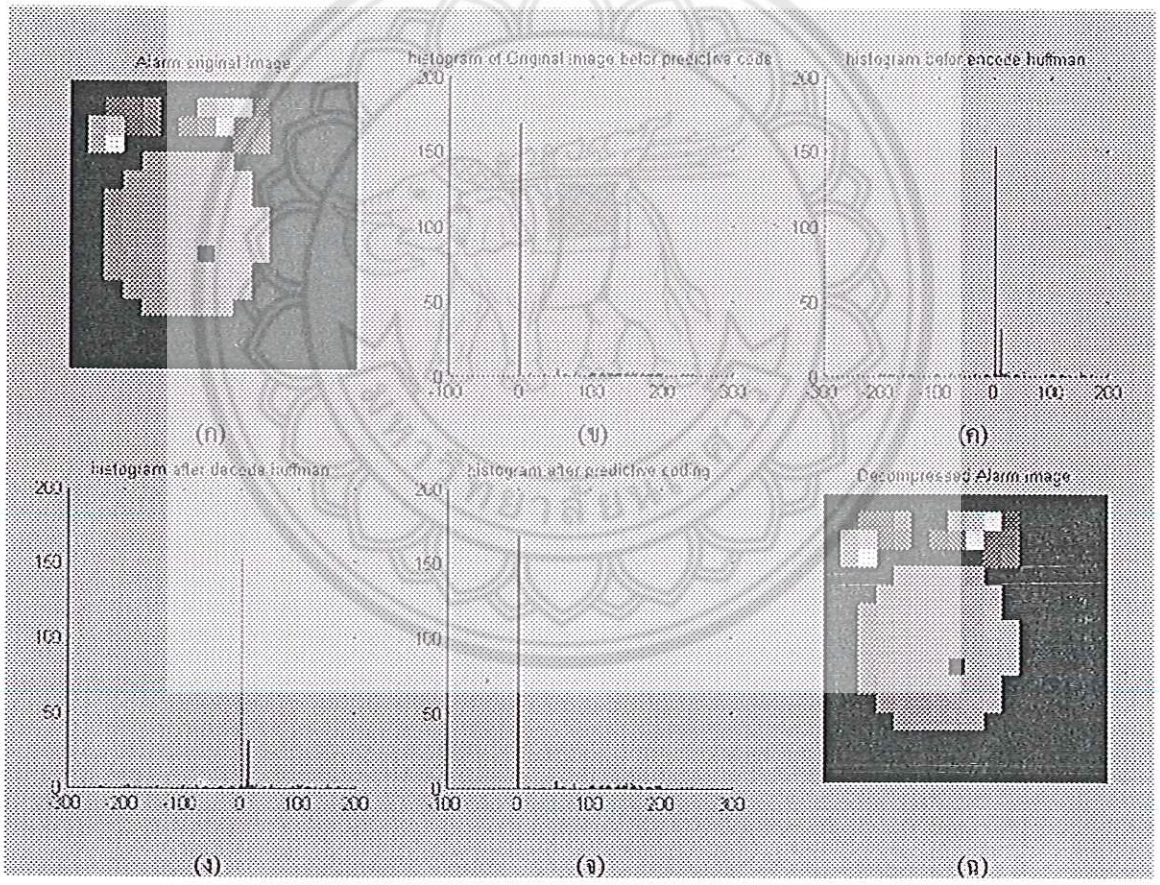
รูปที่ 4.1 (ก) เป็นรูปขาวดำของภาพ Amc ซึ่งเป็นรูป icon ที่มีขนาด 16X16 ก่อนเข้ารหัส ซึ่งมีขนาดไฟล์ที่จัดเก็บ 1,078 และรูปที่ 4.1(ข) แสดง histogram ของภาพ Amc ก่อนการเข้ารหัส predictive bytes และรูปที่ 4.1 (ค) แสดง histogram หลังการเข้ารหัส predictive รูปที่ 4.1 (ง) เป็นรูปที่แสดง histogram ของการ decode Huffman ซึ่งจะมีรูป histogram ที่เหมือนกับรปูที่ 4.1 (ค) และรูปที่ 4.1 (จ) แสดง histogram หลังการเข้ารหัส predictive ซึ่งจะมีรูปที่เหมือนกับรูป 4.1(ข) ส่วนรูปที่ 4.1 (ฉ) เป็นรูปที่ได้หลังจากการแปลงกลับของรหัส predictive ซึ่งสามารถ เปรียบเทียบได้กับรูปที่ 4.1(ก) ส่วนขนาดข้อมูลหลังการเข้ารหัส Huffman จัดเก็บ 174 bytes และนำค่าไปคำนวณค่าอัตราส่วนการบีบอัดข้อมูลภาพเท่ากับ 6.2 และทำการเปรียบเทียบภาพทั้ง 2 ได้ค่า SNR และ PSNR เท่ากับ infinity แสดงว่าให้เห็นว่าการบีบอัดและการคลายการบีบอัดข้อมูลภาพ Amc มีคุณลักษณะเหมือนกันทุกประการ



รูปที่ 4.1 แสดง Histogram ภาพ Amc ก่อน encode predictive และ หลัง encode predictive

(CR = 6.2, SNR = ∞ , PSNR = ∞)

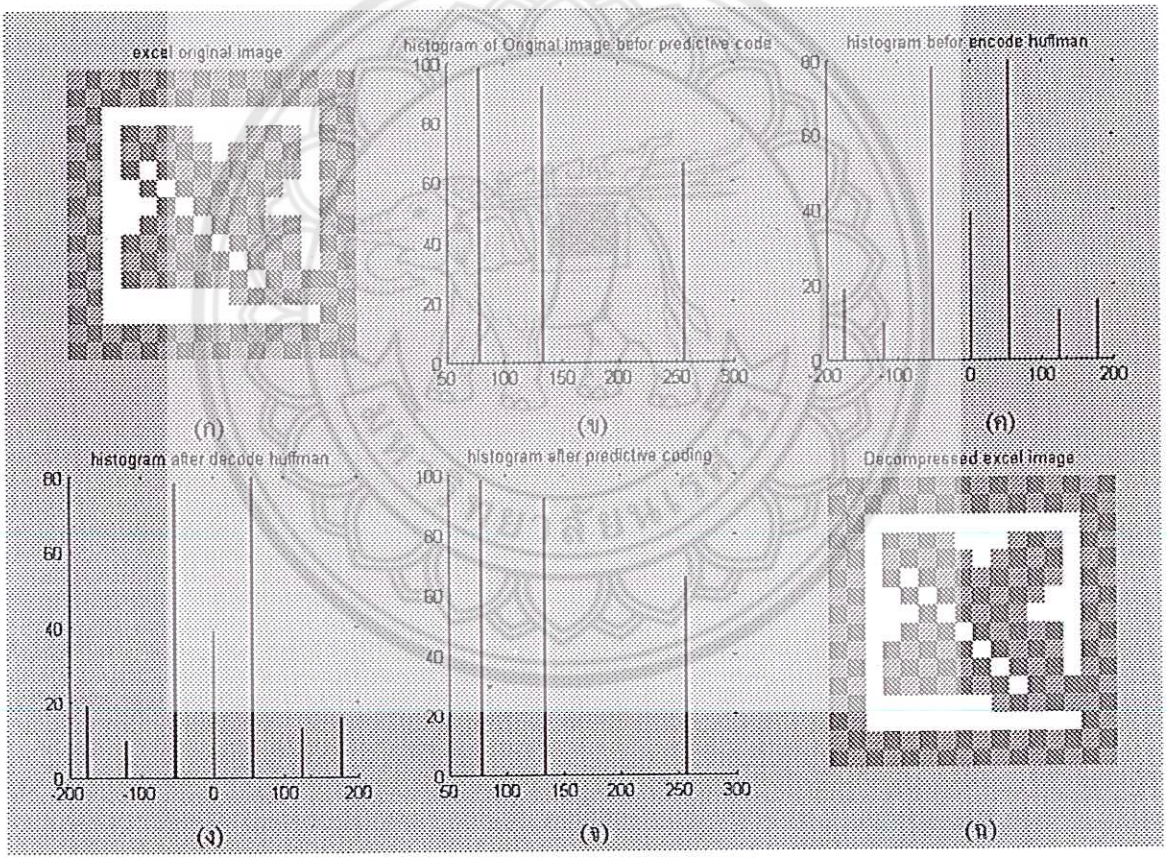
รูปที่ 4.2 (ก) เป็นรูปขาวดำของภาพ Alarm ซึ่งเป็นรูป icon ที่มีขนาด 16X16 ก่อนเข้ารหัส ซึ่งมีขนาดไฟล์ที่จัดเก็บ 1,078 และรูปที่ 4.2(ข) แสดง histogram ของภาพ Alarm ก่อนการเข้ารหัส predictive bytes และรูปที่ 4.2 (ค) แสดง histogram หลังการเข้ารหัส predictive รูปที่ 4.2 (ง) เป็นรูปที่แสดง histogram ของการ decode Huffman ซึ่งจะมีรูป histogram ที่เหมือนกับรปูที่ 4.2 (ค) และรูปที่ 4.2 (จ) แสดง histogram หลังการเข้ารหัส predictive ซึ่งจะมีรูปที่เหมือนกับรูป 4.2(ข) ส่วนรูปที่ 4.2 (ฉ) เป็นรูปที่ได้หลังจากการแปลงกลับของรหัส predictive ซึ่งสามารถ เปรียบเทียบได้กับรูปที่ 4.2(ก) ส่วนขนาดข้อมูลหลังการเข้ารหัส Huffman จัดเก็บ 89 bytes และนำค่าไปคำนวณค่าอัตราส่วนการบีบอัดข้อมูลภาพเท่ากับ 6.2 และทำการเปรียบเทียบภาพทั้ง 2 ได้ค่า SNR และ PSNR เท่ากับ infinity แสดงว่าให้เห็นว่าการบีบอัดและการคลายการบีบอัดข้อมูลภาพ Alarm มีคุณลักษณะเหมือนกันทุกประการ



รูปที่ 4.2 แสดง Histogram ภาพ Alarm ก่อน encode predictive และ หลัง encode predictive

(CR = 12.11, SNR = ∞ , PSNR = ∞)

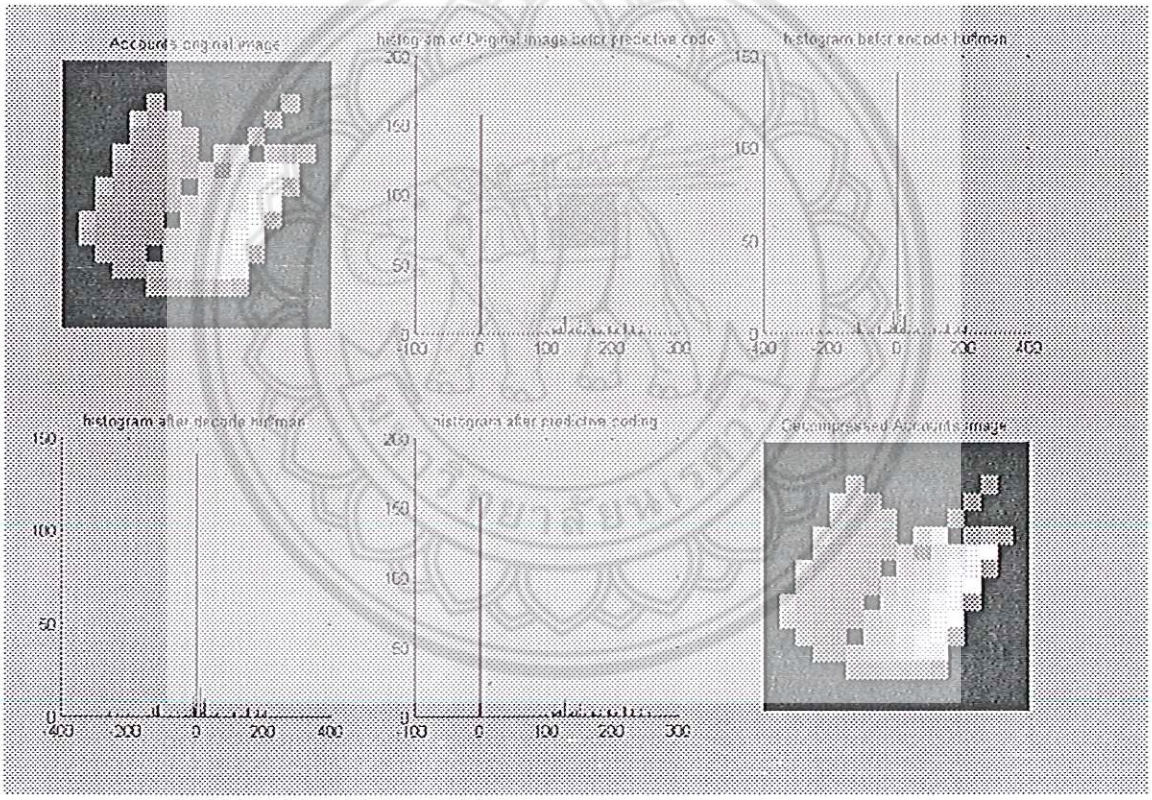
รูปที่ 4.3 (ก) เป็นรูปขาวดำของภาพ excel ซึ่งเป็นรูป icon ที่มีขนาด 16X16 ก่อนเข้ารหัส ซึ่งมีขนาดไฟล์ที่จัดเก็บ 1,078 และรูปที่ 4.3(ข) แสดง histogram ของภาพ excel ก่อนการเข้ารหัส predictive bytes และรูปที่ 4.3 (ค) แสดง histogram หลังการเข้ารหัส predictive รูปที่ 4.3 (ง) เป็นรูปที่แสดง histogram ของการ decode Huffman ซึ่งจะมีรูป histogram ที่เหมือนกับรปูที่ 4.3 (ค) และรูปที่ 4.3 (จ) แสดง histogram หลังการเข้ารหัส predictive ซึ่งจะมีรูปที่เหมือนกับรูป 4.3(ข) ส่วนรูปที่ 4.3 (ฉ) เป็นรูปที่ได้หลังจากการแปลงกลับของรหัส predictive ซึ่งสามารถ เปรียบเทียบได้กับรูปที่ 4.3(ก) ส่วนขนาดข้อมูลหลังการเข้ารหัส Huffman จัดเก็บ 81 bytes และนำค่าไปคำนวณค่าอัตราส่วนการบีบอัดข้อมูลภาพเท่ากับ 6.2 และทำการเปรียบเทียบภาพทั้ง 2 ได้ค่า SNR และ PSNR เท่ากับ infinity แสดงว่าให้เห็นว่าการบีบอัดและการคลายการบีบอัดข้อมูลภาพ excel มีคุณลักษณะเหมือนกันทุกประการ



รูปที่ 4.3 แสดง Histogram ภาพ excel ก่อน encode predictive และ หลัง encode predictive

(CR = 13.31, SNR = ∞ , PSNR = ∞)

รูปที่ 4.4 (ก) เป็นรูปขาวดำของภาพ Accounts ซึ่งเป็นรูป icon ที่มีขนาด 16X16 ก่อนเข้ารหัส ซึ่งมีขนาดไฟล์ที่จัดเก็บ 1,078 และรูปที่ 4.4(ข) แสดง histogram ของภาพ Accounts ก่อนการเข้ารหัส predictive bytes และรูปที่ 4.4 (ค) แสดง histogram หลังการเข้ารหัส predictive รูปที่ 4.4 (ง) เป็นรูปที่แสดง histogram ของการ decode Huffman ซึ่งจะมีรูป histogram ที่เหมือนกับรปูที่ 4.4 (ค) และรูปที่ 4.4 (จ) แสดง histogram หลังการเข้ารหัส predictive ซึ่งจะมีรูปที่เหมือนกับรูป 4.4(ข) ส่วนรูปที่ 4.4 (ฉ) เป็นรูปที่ได้หลังจากการแปลงกลับของรหัส predictive ซึ่งสามารถ เปรียบเทียบได้กับรูปที่ 4.4 (ก) ส่วนขนาดข้อมูลหลังการเข้ารหัส Huffman จัดเก็บ 99 bytes และนำค่าไปคำนวณค่าอัตราส่วนการบีบอัดข้อมูลภาพเท่ากับ 6.2 และทำการเปรียบเทียบภาพทั้ง 2 ได้ค่า SNR และ PSNR เท่ากับ infinity แสดงว่าให้เห็นว่าการบีบอัดและการคลายการบีบอัดข้อมูลภาพ Accounts มีคุณลักษณะเหมือนกันทุกประการ



รูปที่ 4.4 แสดง Histogram ภาพ Accouts ก่อน encode predictive และ หลัง encode predictive

(CR = 10.89, SNR = ∞ , PSNR = ∞)

จากผลการทดลองการเข้ารหัส Predictive และ Huffman สามารถสรุปผลที่ได้จากการบีบอัดข้อมูลภาพโดยวิธี predictive และ Huffman ดังตารางที่ 4.1

ตารางที่ 4.1 สรุปผลที่ได้จากการทดลองข้อมูลภาพ 4 กรณี

ลำดับที่	ชื่อภาพ	CR	SNR(dB)	PSNR(dB)
1	Amc	6.2	∞	∞
2	Alarm	12.11	∞	∞
3	excel	13.31	∞	∞
4	Accounts	10.89	∞	∞

4.4 ผลการวิเคราะห์การทดลอง

จากผลการทดลองค่าในตารางที่ 4.1 สรุปผลที่ได้จากการบีบอัดข้อมูลภาพโดยวิธีการแบบ Predictive coding ทั้ง 4 กรณี ค่าของอัตราการบีบอัดข้อมูลอยู่ที่ประมาณ 6.2-13.31 ในขณะที่ค่า SNR และ PSNR มีค่าเข้าสู่เลขอนันต์ทุกกรณีนั้น แสดงให้เห็นว่าการบีบอัดและการคลายข้อมูลภาพแบบ predictive coding และใช้ Huffman coding ในส่วนของการ encode และ decode มีคุณลักษณะเหมือนกันทุกประการ

ดังนั้นการบีบอัดข้อมูลภาพโดย Predictive coding อัตราการบีบอัดจะอยู่ในช่วย 6.2-13.31 และคุณภาพของข้อมูลที่ได้หลังการ predictive coding เหมือนข้อมูลเดิมทุกประการ

บทที่ 5
บทสรุป

5.1 สรุปผลการทดลอง

จากบทที่ 4 ผลการทดลองที่ได้ทำให้ทราบว่า การบีบอัดข้อมูลภาพแบบ predictive coding และใช้ Huffman coding มาช่วยในการ encode และ decode นั้นอัตราค่าบีบอัด CR ส่วนใหญ่จะอยู่ในช่วง 1.3-1.7 แต่จะมีบางกรณี ที่มีค่าบีบอัดมากกว่าหรือน้อยกว่ามากๆ เนื่องจากภาพมีค่าสีที่มีค่าเหมือนกันมากๆ และในการเข้ารหัสนี้ไม่มีการสูญเสียข้อมูลภาพหลังการคลายการบีบอัด ซึ่งค่า SNR และ PSNR ที่ได้จะมีค่าเข้าสู่เลขอนันต์ทั้ง 2 กรณี

5.2 ปัญหาในการทดลองและแนวทางแก้ไข

ปัญหาที่เกิดขึ้น

ถ้าใช้ไฟล์ภาพขนาดใหญ่จะใช้เวลาในการรันโปรแกรมนาน ในช่วงของการ decode ซึ่งต้องอ่านค่าจาก bits ที่มากที่สุดค่อยๆลดลงทีละ bit จนเจอ แล้วแสดงค่าออกมา ซึ่งทำให้ใช้เวลาในการ decode มาก

ในการ save file ถ้าต้องการนำไปใช้จริง ต้องทำการ save ตาราง Huffman , ตาราง symbol และขนาดของรูปเข้าไปด้วย ซึ่งทำให้ต้องเพิ่มพื้นที่ในส่วนนี้เข้าไป

ปัญหาในการ save file ไม่สามารถอ่าน file ที่มีค่ามากๆได้ เมื่ออ่านแล้วเจอค่ามากๆ โปรแกรมจะอ่านเป็นค่ามากที่สุดที่จะรับได้ แล้วจะซ้ำกันไปเรื่อยๆ

แนวทางแก้ไขปัญหา

เนื่องจากใช้เวลาในการ decode เพราะ Huffman coding จะทำการอ่านทีละตัว ทำให้ใช้เวลานานในการรันโปรแกรม ดังนั้นจึงมีการกำหนดขนาดภาพ ให้เล็กลงเพื่อที่จะใช้เวลาในการรันน้อยลง

เนื่องจากถ้าค่าสีมีความแตกต่างกันมากๆ หรือเหมือนกันมากๆ,ถ้าใช้การบีบอัดแบบ predictive coding จะทำให้ใช้พื้นที่มากกว่าปกติ ดังนั้น predictive coding จึงเหมาะสำหรับใช้กับภาพที่มีสีกระจาย

5.3 แนวทางการพัฒนาในอนาคต

เนื่องจากในโปรแกรมยังไม่สามารถแยกโปรแกรม ส่วนของ encode และ decode ยังใช้ตาราง Huffman ที่สร้างขึ้นด้วยกัน และในการรันโปรแกรมที่มีไฟล์ขนาดใหญ่ยังใช้เวลานาน และนี่เป็นโปรแกรมใช้สำหรับภาพขาวดำ ดังนั้นอาจนำไปเป็นแนวทางในการเขียนโปรแกรมให้ใช้เวลาในการรัน

น้อยลง ทำการบันทึกตาราง Huffman ไปด้วย และเป็นแนวทางในการพัฒนาโปรแกรมสำหรับใช้กับภาพสีในรูปแบบของ RGB ต่อไป

เอกสารอ้างอิง

[1] รศ.ดร.มนัส สังวรศิลป์ . คู่มือการใช้งาน MATLAB ฉบับสมบูรณ์. กรุงเทพมหานคร: อินโฟเพรสม 2543

[2] David A. Huffman. "A Method for the Construction of Minimum-Redundancy Code", roc. IRE., Vol.40,pp.1098-1101, Sept. 1952.

[3] Ming-I Lu. And Chang-Fuu Chean. "An Encoding Procedure and a Decoding Procedure for a New Modifier Huffman Code",IEEE trans.On Acoustivs Speech and Signal Processing. Vol.38,No.1,pp.128-130, Jan. 1990.

[2] Gonzalez. Rafael C.and Richard E. Woods "Digital Image Processing second Edition" ,p. cm. Includes bibliographical references, ISBN 0-201-18075-8, TA 1632. G66 2001

ภาคผนวก

โปรแกรม Predictive coding and Huffman coding

SOURCE PROGRAM

```
clear all
close all

input=imread('Amc','bmp');
input=rgb2gray(input);
I=double(input);
II=I(:);
II=sort(II);
length_Ori=length(II);
min_Ori=II(1);
max_Ori=II(length_Ori);
x=min_Ori:1:max_Ori;
[m n]=size(I);
% ====>>>>predictive code
for i=1:m,
   for j=1:n,
      if j==1&i==1
         e(i,j)=I(i,j);
      elseif i~=1&j==1
         e(i,j)=I(i,j)-I(i-1,j);
      else
         e(i,j)=I(i,j)-I(i,j-1);
      end
   end
end
new_e=e(:);
length_=m*n;
```

```
% sort by to plot histogram
new_e=sort(new_e);
% num of variable
num=1;
for i=1:length_-1,
   if (new_e(i)~=new_e(i+1))
      num=num+1;
   end
end

% fine count num of variable
% Table of num,count,probability
table=zeros(num,3);

table(1,1)=new_e(1);
table(1,2)=1;
k=1;
for i=1:length_-1,
   if(new_e(i)==new_e(i+1))
      table(k,2)=table(k,2)+1;
   else
      k=k+1;
      table(k,1)=new_e(i+1);
      table(k,2)=table(k,2)+1;
   end
end

% finc probability
for i=1:num,
   table(i,3)=table(i,2)/length_;
end
```

```
% fine max and min value
min_pre=new_e(1);
max_pre=new_e(length_);
y=min_pre:1:max_pre;
num_rec=num;

% ********************************************************************
% ************************** Huffman code **************************
% ********************************************************************

for i=1:num,
    table_code(i,1)=table(i,1);
    table_code(i,2)=table(i,3);
end

% k is number of sort
for i=1:num-1,
   for j=1:num-i,
      if(table_code(j,2)<table_code(j+1,2))
         temp1=table_code(j,1);
         temp=table_code(j,2);
         table_code(j,2)=table_code(j+1,2);
         table_code(j,1)=table_code(j+1,1);
         table_code(j+1,2)=temp;
         table_code(j+1,1)=temp1;
      end
   end
end
treeTable(num,9)=0;
count=0;
for i=1:num,
```

```
    for j=1:2,
        treeTable(i,j+1)=table_code(i,j);
        treeTable(i,1)=count+i;
    end
end
% Table Dic
MyTrue=1;
num_s=num;
num_n=num-1;
datestr(now)


% ********** create bit Table ****************************
while MyTrue==1
    if MyTrue==1
        num=num+1;
        treeTable(num,1)=num;
        treeTable(num,3)=treeTable(num_n,3)+treeTable(num_s,3);
        treeTable(num,4)=num_s;
        treeTable(num,5)=num_n;
        treeTable(num_s,6)=num;
        treeTable(num_n,6)=num;
        treeTable(num_s,7)=1;
        treeTable(num_n,7)=1;
        treeTable(num_s,9)=1;
        treeTable(num_n,9)=0;
    end
    j=1;
    tempTree=0;
    for i=1:num
        if treeTable(i,7)~=1
            tempTree(j,1)=treeTable(i,1);
```

```
            tempTree(j,2)=treeTable(i,3);
            j=j+1;
          end
        end
        if (j>2)
          b1=sortrows(tempTree,2);
          num_s=b1(1,1);
          num_n=b1(2,1);
          if j==2
            MyTrue=0;
          end
        else
          MyTrue=0;
        end
      end
      % search tree
      BL=treeTable(num,4);
      BR=treeTable(num,5);
      CL=BL;
      WhileC=1;
      k=0;
      ii=0;
      if i==1
        PointerNow=BL;
      else
        PointerNow=BR;
      end
      str='';TmpBitTable='';
      BitNum=0;
      strnum=0;
      while WhileC==1
        ii=ii+1;
```

```
if treeTable(CL,8)==1    % complete node
    CL=treeTable(CL,6);
    str='';
    ii=0;
elseif(treeTable(CL,4)==0)&(treeTable(CL,5)==0) % last node
    str(ii)=dec2bin(treeTable(CL,9));
    BitNum(k+1,ii)=CL;
    treeTable(CL,8)=1;
    k=k+1;
    for jj=1:ii
       TmpBitTable(k,jj)=str(jj);
    end
    strnum(k)=CL;
    CL=treeTable(CL,6);
    ii=0;
    str='';
    CL=num;
elseif(treeTable(CL,4)~=0)&(treeTable(CL,5)==0) % go left node
    if CL~=num
       BitNum(k+1,ii)=CL;
       str(ii)=dec2bin(treeTable(CL,9));
    else
       ii=ii-1;
    end
    CL=treeTable(CL,4);
elseif(treeTable(CL,4)==0)&(treeTable(CL,5)~=0) % go right node
    if CL~=num
       BitNum(k+1,ii)=CL;
       str(ii)=dec2bin(treeTable(CL,9));
    else
       ii=ii-1;
    end
```

```
        CL=treeTable(CL,5);
    elseif(treeTable(CL,4)~=0)&(treeTable(CL,5)~=0)  %go Left node
      if (treeTable(treeTable(CL,5),8)==1)&(treeTable(treeTable(CL,4),8)==0)  % check tree status
        if CL~=num
          BitNum(k+1,ii)=CL;
          str(ii)=dec2bin(treeTable(CL,9));
        else
          ii=ii-1;
        end
        CL=treeTable(CL,4);
      elseif (treeTable(treeTable(CL,4),8)==1)&(treeTable(treeTable(CL,5),8)==0)
        if CL~=num
          BitNum(k+1,ii)=CL;
          str(ii)=dec2bin(treeTable(CL,9));
        else
          ii=ii-1;
        end
        CL=treeTable(CL,5);
      elseif (treeTable(treeTable(CL,5),8)==0)&(treeTable(treeTable(CL,4),8)==0)
        if CL~=num
          BitNum(k+1,ii)=CL;
          str(ii)=dec2bin(treeTable(CL,9));
        else
          ii=ii-1;
        end
        CL=treeTable(CL,4);
      elseif treeTable(CL,8)==0
        BitNum(k+1,ii)=CL;
        str(ii)=dec2bin(treeTable(CL,9));
        treeTable(CL,8)=1;
        k=k+1;
        for jj=1:ii
```

```
            TmpBitTable(k,jj)=str(jj);
        end
        strnum(k)=CL;
        CL=num;
        ii=0;
      end
    end
    if (treeTable(BL,8)==1)&(treeTable(BR,8)==1)
      WhileC=0;
    end
  end
  [ii,jj]=size(TmpBitTable);
  for i=1:num_rec
    for j=1:num-1
      xx=strnum(j);
      if strnum(j)==i
        for k=1: jj
          BitTable(i,k)=TmpBitTable(j,k);
        end
      end
    end
  end
  [ii,jj]=size(BitTable);
  bit_len=jj;          % max length of sambol bits -->>>>>


  % *************** start encode *************************
  str='';
  for i=1:m
    for j=1:n
      %***search***
```

```
        for k=1:num_rec
          if e(i,j)==table_code(k,1)
            str=strcat(str,BitTable(k,:));
            break;
          end
        end
        %****** end search ****
    end
end
[mm,nn]=size(e);        % size of image aray

% ############################ save file ############################
fid=fopen('test.ee','wb');
fwrite(fid,str,'ubit1');
fclose(fid);
% end saved file

% **********************decode**************************
str2='';
str3='';
mmi=1;
mmj=1;

max_bin=length(str);
i=1;
while i<=max_bin
    %***search***
 j=bit_len-1;
 while (j+1>0)
    if i+j>max_bin
      j=max_bin-i;
    end
```

```
        st=str(i:i+j);
        for k=1:num_rec
          st2=BitTable(k,:);
            if bin2dec(st2)==bin2dec(st)
              tb='';
              for jj=1:bit_len
                tb= strcat(tb,BitTable(k,jj));
              end
              if length(tb)==length(st)
                str2=strcat(str2,',',num2str(table_code(k,1)));
                d_e(mmi,mmj)=table_code(k,1);
                mmj=mmj+1;
                if mmj>nn
                  mmj=1;
                  mmi=mmi+1;
                end
                i=i+j;
                j=0;
                break;
              end
            end
        end
        j=j-1;
        %*** end search ***
      end
      i=i+1;
end

%======>>>>>>>>> pedictive decode
 for i=1:mm,
   for j=1:nn,
     if j==1&i==1
```

```
            f(i,j)=round(d_e(i,j));
        elseif j==1&i~=1
            f(i,j)=round(d_e(i,j)+f(i-1,j));
        else
            f(i,j)=round(d_e(i,j)+f(i,j-1));
        end
    end
end
d_ee=d_e(:);
f2=f(:);

figure
subplot(2,3,1),imshow(input),title('Amc original image')
subplot(2,3,2),hist(II,x),title('histogram of Original image befor predictive code')
subplot(2,3,3),hist(new_e,y),title('histogram befor encode huffman')
subplot(2,3,4),hist(d_ee,y),title('histogram after decode huffman')
subplot(2,3,5),imshow(uint8(f)),title('Decompressed Amc image')
subplot(2,3,6),hist(f2,x),title('histogram after predictive coding')
```

ประวัติผู้เขียนโครงงาน

ชื่อ นาย ชาญวิทย์ ปิ่นคำ

ภูมิลำเนา 187 ม.3 ต.แม่กรณ์ อ.เมือง จ.เชียงราย 57000

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน จุฬาภรณราชวิทยาลัย เชียงราย
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : modern_100@hotmail.com

ชื่อ นายธนภูมิ อึ้งตระกูล

ภูมิลำเนา 50 ม.4 ต. เวียง อ.เชียงคำ จ.พะเยา 56110

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน จุฬาภรณราชวิทยาลัย เชียงราย
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : jiddyza@hotmail.com

ชื่อ นาย เฉลิมเกียรติ ยั่งยืนเจริญสุข

ภูมิลำเนา 90/1 ม.10 ต.ห้วยซ้อ อ.เชียงของ จ.เชียงราย 57140

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน ห้วยซ้อวิทยาคม รัชมังคลาภิเษก
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : dennis_yew@hotmail.com

ประวัติผู้เขียนโครงงาน

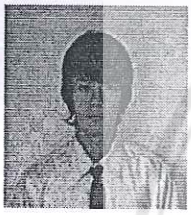
ชื่อ นาย ชาญวิทย์ ปิ่นคำ

ภูมิลำเนา 187 ม.3 ต.แม่กรณ์ อ.เมือง จ.เชียงราย 57000

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน จุฬาภรณราชวิทยาลัย เชียงราย
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : modern_100@hotmail.com



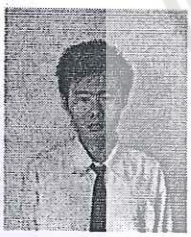
ชื่อ นายธนภูมิ อึ้งตระกูล

ภูมิลำเนา 50 ม.4 ต. เวียง อ.เชียงคำ จ.พะเยา 56110

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน จุฬาภรณราชวิทยาลัย เชียงราย
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : jiddyza@hotmail.com



ชื่อ นาย เฉลิมเกียรติ ยั่งยืนเจริญสุข

ภูมิลำเนา 90/1 ม.10 ต.ห้วยซ้อ อ.เชียงของ จ.เชียงราย 57140

ประวัติการศึกษา

- จบการศึกษาจากโรงเรียน ห้วยซ้อวิทยาคม รัชมังคลาภิเษก
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E – mail : dennis_yew@hotmail.com