เทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย

DATA HIDING TECHNIQUE FOR THAI DOCUMENT IMAGE

นงนุช อาจวาริน

NONGNUCH ARTWARIN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2544

ISBN 974-648-245-6

เทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย

DATA HIDING TECHNIQUE FOR THAI DOCUMENT IMAGE

นงนุช อาจวารินทร์

NONGNUCH ARTWARIN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2544

ISBN 974-648-245-6

DATA HIDING TECHNIQUE FOR THAI DOCUMENT IMAGE

NONGNUCH ARTWARIN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2001

ISBN 974-648-245-6

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ เทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย

DATA HIDING TECHNIQUE FOR THAI DOCUMENT IMAGE

ชื่อนักศึกษา นางสาวนงนุช อาจวารินทร์

รหัสประจำตัว 40067042

ปริญญา วิทยาศาสตรมหาบัณฑิต

สาขาวิชา เทคโนโลยีสารสนเทศ

อาจารย์ผู้ควบคุมวิทยานิพนธ์ ดร.นพพร โชติกกำธร

คณะกรรมการสอบวิทยานิพนธ์		ลายมือชื่อ
ดร.นพพร	โชติกกำธร	
รศ.ดร.วิเชียร	เปรมชัยสวัสดิ์	
ผศ.ดร.บุญธีร์	เครือตราชู	
ดร.จันทร์บูรณ์	สถิตวิริยวงศ์	
ดร.วรพจน์	กรีสุระเดช	

วัน/เดือน/ปี 20 ธันวาคม 2543 **เวลา** 15.00 น. เป็นต้นไป

สถานที่สอบ ห้อง LAB 316 ชั้น 3 อาคารสำนักวิจัยและบริการคอมพิวเตอร์

บัณฑิตวิทยาลัยรับรองแล้ว

(รศ.ดร.บุญวัฒน์ อัตชู)

คณบดีบัณฑิตวิทยาลัย

วันที่ 29 **เดือน** ธันวาคม **พ.ศ.** 2544

หัวข้อวิทยานิพนธ์	เทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย
นักศึกษา	นางสาว นงนุช อาจวารินทร์
รหัสประจำตัว	40067042
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
พ.ศ.	2544
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ดร. นพพร โชติกกำธร

บทคัดย่อ

เนื้อหาของวิทยานิพนธ์เล่มนี้กล่าวถึงการพัฒนาเทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพ ซึ่งวิธีการซ่อนข้อมูลที่มีอยู่ในปัจจุบัน เช่น วิธี Line shift coding และ Word shift coding เป็นวิธีที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ เมื่อนำไปวิธีการเหล่านี้ไปประยุกต์ใช้กับเอกสารที่เป็นภาษาอื่นๆ เช่น ภาษาไทย พบว่ามีข้อจำกัดบางประการซึ่งเกิดจากความแตกต่างทางด้านโครงสร้างภาษา งานวิจัยนี้ได้นำเสนอเทคนิคการซ่อนข้อมูลที่เหมาะสมสำหรับเอกสารรูปภาพภาษาไทย ซึ่งมีความจุในการซ่อนข้อมูลสูงกว่าวิธีการที่มีอยู่ในปัจจุบัน วัตถุประสงค์หลักของการซ่อนข้อมูลในงานวิจัยนี้ก็เพื่อใช้ตรวจสอบความถูกต้องของเอกสาร (Document authentication) โดยจะซ่อนข้อมูลลงในตำแหน่งบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับชั้นที่สองกับสามโดยการเปลี่ยนแปลงขนาดความกว้างของช่องว่างที่ตำแหน่งนั้นเพื่อแทนค่าบิตข้อมูลที่ต้องการซ่อน วิธีการที่นำเสนอนี้ได้นำหลักการเพิ่มบิตข้อมูลซ้ำ (Data redundancy) และการสร้างชุดรหัสข้อมูลแฮมมิง (Hamming code) มาประยุกต์ใช้เพื่อป้องกันและแก้ไขความผิดพลาดของข้อมูลที่อาจเกิดขึ้นได้ ในงานวิจัยนี้ได้ทำการทดลองซ่อนข้อมูลลงในเอกสารภาษาไทยขนาด A4 ซึ่งใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดตัวอักษรแบบชิดขอบคอลัมน์เดียวเพื่อทดสอบประสิทธิภาพการทำงานของวิธีการซ่อนข้อมูลที่นำเสนอนี้โดยจะพิจารณาถึงกรณีที่เอกสารผ่านกระบวนการปรับขนาดและกรณีที่เอกสารผ่านกระบวนการถ่ายเอกสาร พบว่าวิธีการดังกล่าวสามารถดึงข้อมูลออกจากเอกสารอิเลคทรอนิคที่ถูกขยายขนาด 30% และถูกย่อขนาด 15% ได้อย่างถูกต้องทั้งหมด และสามารถดึงข้อมูลออกจากเอกสารที่ผ่านการพิมพ์และการทำสำเนา 3 ครั้งได้ถูกต้อง 48 หน้าจากทั้งหมด 50 หน้า

Thesis Title	Data Hiding Technique for Thai Document Image
Student	Miss. Nongnuch Artwarin
Student ID.	40067042
Degree	Master of science
Programme	Information Technology
Year	2001
Thesis Advisor	Dr. Nopporn Chotikakamthorn

ABSTRACT

In this thesis, a new data hiding technique for Thai document image is proposed. Existing techniques such as line shift coding and word shifting coding methods, designed for English document, has some drawbacks when applied to document written in Thai. The problem is due to difference in language structure. The proposed data hiding technique has been developed for document images written in Thai language, with high data embedding capacity. The objective of this research is for authentication of the document image's modification. The proposed technique embeds data bit in any character block which has a space between second and third level, by changing the width of the space according to the value of the data bit to be embedded. The proposed technique applies data redundancy principle and Hamming code to prevent and correct decoding errors. Experiments have been performed on the A4 size documents with 14 points of AngsanaUPC character font in a single column format. After the document in electronic form was scaled from -15% to 30%, the embedded data can be successfully recovered. From experimental result was performed repeatedly copied for three times, the embedded data in 48 out of 50 documents can be correctly.

กิตติกรรมประกาศ

การจัดทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี เพราะได้รับความเมตตาจากท่านอาจารย์ที่ปรึกษา ดร.นพพร โชติกกำธร ซึ่งได้ให้คำปรึกษาในการแก้ไขปัญหาทุกๆเรื่องและให้คำชี้แนวทางในการทำวิจัยตลอดมา ผู้วิจัยรู้สึกซาบซึ้งในความกรุณาจากท่าน ขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณอาจารย์ประจำคณะเทคโนโลยีสารสนเทศทุกท่านที่ได้สั่งสอนวิชาพื้นฐานและอบรมผู้วิจัยซึ่งเป็นรากฐานมาสู่ความสำเร็จของงานวิจัยนี้ รวมทั้งขอขอบพระคุณคณะเทคโนโลยีสารสนเทศที่สนับสนุนเครื่องมือและอุปกรณ์ที่ใช้ในการทดลอง รวมทั้งเจ้าหน้าที่ที่คอยอำนวยความสะดวกในด้านต่างๆในระหว่างการศึกษาให้เป็นอย่างดี

ขอขอบพระคุณคุณพ่อคุณแม่ที่คอยห่วงใยให้กำลังใจเสมอมาและขอขอบพระคุณเพื่อนร่วมรุ่นที่ได้ให้กำลังใจกันและกันตลอดเวลาที่ศึกษาอยู่

งานวิจัยนี้ที่นำเสนอในวิทยานิพนธ์ฉบับนี้ ส่วนหนึ่งกระทำภายใต้ห้องปฏิบัติการ Multimedia and Virtual Research Lab, สำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ, สจล.

สุดท้ายนี้ขอขอบพระคุณบัณฑิตวิทยาลัยที่ได้ให้ทุนสนับสนุนการทำวิทยานิพนธ์ในครั้งนี้ คุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

นงนุช อาจวารินทร์

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันเทคโนโลยีทางด้านการสื่อสารมีการพัฒนาเพิ่มขึ้น ทำให้การแพร่กระจายเอกสารในรูปแบบอิเลคทรอนิคได้รับความนิยมเพิ่มมากขึ้น ซึ่งมีข้อดีคือสามารถแพร่กระจายเอกสารได้อย่างสะดวกรวดเร็วและไม่สิ้นเปลืองค่าใช้จ่ายมากนัก แต่มีข้อเสียคือเอกสารเหล่านั้นอาจถูกละเมิดลิขสิทธิ์หรืออาจถูกเปลี่ยนแปลงแก้ไขได้ง่าย ดังนั้นจึงได้มีการคิดค้นวิธีการซ่อนข้อมูลบางอย่างลงในเอกสาร (Data hiding techniques) เพื่อป้องกันการละเมิดลิขสิทธิ์ในเอกสารนั้น โดยจะนำข้อมูลที่ซ่อนอยู่ภายในเอกสารมาใช้ตรวจสอบความถูกต้องของเอกสาร (Document authentication) ว่าเอกสารนั้นถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องหรือไม่ หรืออาจใช้ตรวจสอบว่าเอกสารนั้นถูกเปลี่ยนแปลงแก้ไขมาแล้วหรือยัง สาเหตุที่เรานำวิธีการซ่อนข้อมูลมาประยุกต์ใช้ในการตรวจสอบความถูกต้องของเอกสารก็เนื่องจากว่าการซ่อนข้อมูลลงไปที่ตัวเอกสารโดยตรงจะทำให้ข้อมูลที่ซ่อนอยู่ภายในเอกสารนั้นคงอยู่กับเอกสารโดยตลอดไม่ว่าเอกสารนั้นจะอยู่ในรูปแบบใดๆก็ตาม (เอกสารที่ได้จากการพิมพ์หรือเอกสารที่ถูกเปลี่ยนแปลงรูปแบบ) ทำให้เราสามารถตรวจสอบความถูกต้องของเอกสารได้โดยตลอด อีกทั้งการซ่อนข้อมูลในรูปแบบนี้จะไม่ทำให้เอกสารเกิดการเปลี่ยนแปลงไปจากเดิมมากนักจึงไม่ทำให้บุคคลอื่นๆสามารถสังเกตเห็นความผิดปกติของเอกสารที่ซ่อนข้อมูลได้

ในปัจจุบันได้มีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารมาบ้างแล้ว แต่วิธีการที่มีอยู่ในปัจจุบันนี้ถูกออกแบบมาให้ใช้งานกับเอกสารภาษาอังกฤษ (หรือเอกสารที่มีรูปแบบคล้ายภาษาอังกฤษ) เท่านั้น ดังนั้นเมื่อนำวิธีการเหล่านั้นมาประยุกต์ใช้กับเอกสารภาษาไทยพบว่ามีข้อจำกัดบางประการซึ่งข้อจำกัดที่สำคัญก็คือสามารถซ่อนข้อมูลได้น้อย ดังนั้นในงานวิจัยนี้จึงได้ศึกษาหาวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารภาษาไทยโดยมีวัตถุประสงค์ในการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสาร

1.2 วัตถุประสงค์ของการศึกษา

1. เพื่อศึกษาหาวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารรูปภาพภาษาไทยโดยอาศัยคุณสมบัติเฉพาะของรูปแบบการเขียนภาษาไทยมาประยุกต์ใช้ในการซ่อนข้อมูล

2. เพื่อประยุกต์ใช้วิธีการซ่อนข้อมูลที่พัฒนาขึ้นสำหรับการตรวจสอบความถูกต้องของเอกสาร

3. เพื่อศึกษาคุณสมบัติและข้อจำกัดต่างๆของวิธีการซ่อนข้อมูลที่พัฒนาขึ้นในด้านต่างๆ เช่น ข้อจำกัดทางด้านรูปแบบหรือขนาดของตัวอักษรและข้อจำกัดเกี่ยวกับความบิดเบือนของเอกสารที่เกิดจากการการพิมพ์ การถ่ายเอกสาร หรือการสแกน

1.3 สมมติฐานของการศึกษา

เนื่องจากรูปแบบโครงสร้างการเขียนภาษาไทยมีการแบ่งระดับชั้นในการแสดงข้อความออกเป็น 4 ระดับซึ่งก็คือระดับของพยัญชนะ สระ วรรณยุกต์ และตัวอักขระพิเศษ โดยที่ในแต่ละระดับนั้นจะถูกแยกออกจากกันด้วยช่องว่างระหว่างระดับชั้นซึ่งจะมีขนาดที่แตกต่างกัน ในงานวิจัยนี้จะนำเอาช่องว่างที่ตำแหน่งเหล่านี้มาประยุกต์ใช้เพื่อซ่อนข้อมูลโดยการเปลี่ยนแปลงขนาดความกว้างของช่องว่างระหว่างระดับชั้นเพียงเล็กน้อยเพื่อกำหนดข้อมูลไบนารีที่จะซ่อนลงที่ตำแหน่งนั้น ตัวอย่างเช่น การลดขนาดความกว้างของช่องว่างระหว่างระดับชั้นเพื่อแทนข้อมูลไบนารี “0” หรือการเพิ่มขนาดความกว้างของช่องว่างระหว่างระดับชั้นเพื่อแทนข้อมูลไบนารี “1” เป็นต้น ซึ่งช่องว่างที่นำมาใช้ซ่อนข้อมูลนี้จะมีความกว้างมากพอสมควรที่จะสามารถปรับเปลี่ยนขนาดเพื่อซ่อนข้อมูลได้โดยที่ไม่ทำให้เอกสารเกิดความผิดปกติไปจากเดิมมากนัก อีกทั้งในการดึงข้อมูลที่ซ่อนอยู่ภายในช่องว่างเหล่านี้ก็สามารถทำได้ถึงแม้ว่าเอกสารที่ใช้ซ่อนข้อมูลนี้จะผ่านกระบวนการประมวลผลทางด้านเอกสาร (การพิมพ์ การสแกน หรือการถ่ายเอกสาร) มาบ้างแล้วก็ตาม

จากการสำรวจข้อมูลเอกสารภาษาไทยขนาด A4 ที่ใช้ตัวอักษรรูปแบบ (Font) AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดตัวอักษรแบบชิดขอบคอลัมน์เดียวจำนวน 300 บรรทัด พบว่าในแต่ละบรรทัดจะมีบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับ (พิจารณาที่ช่องว่างระหว่างระดับพยัญชนะกับระดับของสระระดับบนเท่านั้น) ที่สามารถซ่อนข้อมูลได้ประมาณ 13 ตำแหน่ง ดังนั้นหากเรานำช่องว่างเหล่านี้มาใช้ซ่อนข้อมูลก็จะทำให้เราสามารถซ่อนข้อมูลได้มากกว่าวิธีการอื่นๆที่มีอยู่ในปัจจุบัน ตัวอย่างเช่น วิธี Line shift coding ต้องใช้จำนวนบรรทัดถึง 2 บรรทัดในการซ่อนข้อมูล 1 ตำแหน่ง ส่วนวิธี Word shift coding นั้นปริมาณข้อมูลที่สามารถซ่อนได้จะขึ้นอยู่กับจำนวนช่องว่างระหว่างคำภายในหน้าเอกสารซึ่งวิธีการนี้ไม่เหมาะสมที่จะนำมาประยุกต์ใช้กับเอกสารภาษาไทยเนื่องจากลักษณะการเขียนของภาษาไทยไม่มีการเว้นช่องว่างระหว่างคำแต่จะมีการเว้นช่องว่างระหว่างประโยคเท่านั้น

1.4 ทฤษฎีหรือแนวความคิดที่เกี่ยวข้องกับงานวิจัย

1.4.1 เอกสารรูปภาพ

เอกสารรูปภาพที่เราจะใช้ซ่อนข้อมูลในงานวิจัยนี้เป็นเอกสารรูปภาพไบนารีที่มีการจัดเก็บแบบบิตแมป โดยจะจัดเก็บข้อมูลอยู่ในรูปแบบลำดับของพิกเซลซึ่งค่าของข้อมูลในแต่ละพิกเซลจะมีค่าเป็น "0" หรือ "1" เท่านั้น (ขาวหรือดำ)

การซ่อนข้อมูลลงในเอกสารรูปภาพนั้นเราจะต้องกำหนดตำแหน่งที่จะซ่อนข้อมูลให้ได้เสียก่อน ซึ่งในงานวิจัยนี้จะนำหลักการสร้างฟังก์ชันฮิสโตแกรมของหน้าเอกสาร (Histogram function) มาประยุกต์ใช้ในการกำหนดตำแหน่งบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้โดยการสร้างฟังก์ชันฮิสโตแกรมในแนวนอนของหน้าเอกสาร (Horizontal histogram function) เพื่อกำหนดขอบเขตของแต่ละบรรทัดภายในหน้าเอกสาร จากนั้นจะสร้างฟังก์ชันฮิสโตแกรมในแนวตั้ง (Vertical histogram function) ของแต่ละบรรทัดเพื่อกำหนดขอบเขตของบล็อกตัวอักษรแต่ละตัว โดยบล็อกที่สามารถซ่อนข้อมูลได้นั้นจะต้องเป็นบล็อกที่มีช่องว่างระหว่างระดับ ซึ่งในงานวิจัยนี้เราจะพิจารณาบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับพยัญชนะกับระดับของสระระดับบนเท่านั้น

1.4.2 ลายเซ็นดิจิตอล

วัตถุประสงค์หลักของการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทยในงานวิจัยนี้คือ เพื่อใช้ตรวจสอบความถูกต้องของเอกสารซึ่งจะนำหลักการของลายเซ็นดิจิตอล (Digital signature) มาประยุกต์ใช้ โดยกำหนดให้ข้อมูลที่จะถูกซ่อนลงในเอกสารคือลายเซ็นดิจิตอลของเอกสารแต่ละฉบับ (หมายเลขประจำเอกสารที่ผ่านการเข้ารหัสข้อมูลแล้ว) การตรวจสอบความถูกต้องของเอกสารที่ซ่อนข้อมูลลายเซ็นดิจิตอลนี้สามารถทำได้โดยดึงข้อมูลที่ซ่อนอยู่ภายในเอกสารออกมาเพื่อถอดรหัสข้อมูล จากนั้นจะนำไปเปรียบเทียบกับหมายเลขประจำเอกสารต้นฉบับเพื่อตรวจสอบว่าเอกสารนี้ถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องหรือไม่หรือเอกสารนี้ถูกเปลี่ยนแปลงแก้ไขมาแล้วหรือยัง (การสร้างลายเซ็นดิจิตอลในงานวิจัยนี้จะใช้หลักการเข้ารหัสแบบ RSA Public-Key encryption)

1.4.3 การตรวจสอบและแก้ไขความผิดพลาดของข้อมูล

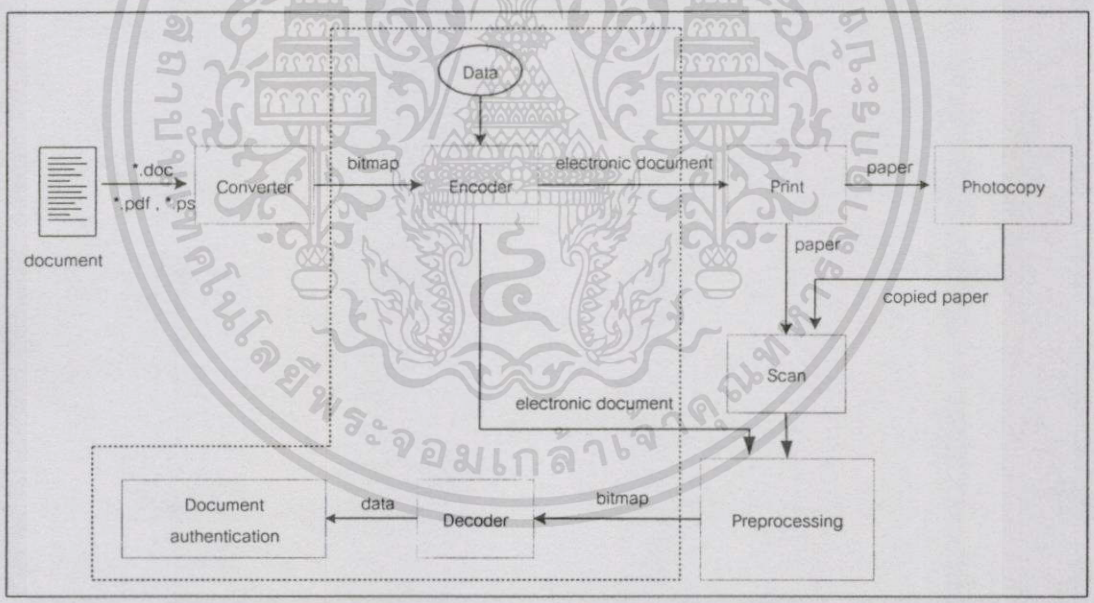
วิธีการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทยในงานวิจัยนี้จะนำวิธีการตรวจสอบและแก้ไขความผิดพลาดของข้อมูล (Error correction) มาประยุกต์ใช้ร่วมด้วยเพื่อทำให้ผลการดึงข้อมูลออกจากเอกสารมีความถูกต้องมากขึ้น เนื่องจากเอกสารที่ถูกซ่อนข้อมูลแล้วอาจถูกนำไปผ่านการประมวลผลทางด้านเอกสาร (Document processing) เช่น การพิมพ์ การถ่ายเอกสาร หรือการสแกน ซึ่งอาจส่งผลให้เอกสารนั้นเปลี่ยนแปลงไปจากเดิมได้ (เช่น มีสิ่งรบกวนเกิดขึ้นภายในเอกสารหรือ

เอกสารถูกทำให้เอียงไปจากเดิม) ทำให้ข้อมูลที่ซ่อนอยู่ภายในเอกสารนั้นเกิดความผิดพลาดหรือสูญหายได้

วิธีการแก้ไขความผิดพลาดของข้อมูลที่นำมาใช้ในงานวิจัยนี้คือ การเข้ารหัสแฮมมิง (Hamming code) โดยจะสร้างชุดข้อมูลรหัสแฮมมิงที่สามารถตรวจสอบและแก้ไขความผิดพลาดของตนเองได้ แต่วิธีการนี้มีข้อจำกัดคือสามารถตรวจสอบและแก้ไขความผิดพลาดของข้อมูลได้เพียงหนึ่งตำแหน่งภายในหนึ่งชุดข้อมูลเท่านั้น ดังนั้นหากข้อมูลเกิดการผิดพลาดเป็นจำนวนมากวิธีการนี้ก็จะไม่สามารถทำงานได้อย่างมีประสิทธิภาพ ดังนั้นในงานวิจัยนี้จึงได้นำเอาวิธีการซ่อนข้อมูลซ้ำ (Redundancy) เข้ามาใช้ร่วมกับวิธีการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลด้วยเพื่อป้องกันปัญหาในกรณีที่ข้อมูลเกิดความผิดพลาดในปริมาณมาก

1.5 ขอบเขตของงานวิจัย

ขอบเขตของงานวิจัยเพื่อหาวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารรูปภาพภาษาไทยแสดงอยู่ในรูปที่ 1.1 (ภายในเส้นประ) โดยมีรายละเอียดดังนี้



รูปที่ 1.1 แสดงขอบเขตของงานวิจัย

1) เอกสารที่ใช้ซ่อนข้อมูลในงานวิจัยนี้จะพิจารณาเฉพาะกรณีของเอกสารรูปภาพที่มีการจัดเก็บแบบบิตแมป (Bitmap file) หรือเอกสารรูปแบบอื่นๆที่ถูกแปลงให้อยู่ในรูปแบบของบิตแมปแล้ว

2) การดึงข้อมูลที่ซ่อนอยู่ภายในเอกสารนั้นสามารถกระทำได้กับเอกสารที่อยู่ในรูปแบบอิเลคทรอนิค (Electronic document) หรือเอกสารที่ผ่านการประมวลผลทางด้านเอกสารมาแล้ว เช่น เอกสารที่ได้จากการพิมพ์หรือการถ่ายเอกสาร

3) เอกสารที่พร้อมจะถูกดึงข้อมูลได้ผ่านกระบวนการกำจัดสิ่งรบกวนและลดความบิดเบือนของเอกสาร (Noise and distortion reduction)

4) วิธีการซ่อนข้อมูลที่พัฒนาขึ้นมานี้จะนำไปประยุกต์ใช้เพื่อตรวจสอบความถูกต้องของเอกสาร (Document authentication)

1.6 ขั้นตอนของการศึกษา

1. ศึกษาวิธีการซ่อนข้อมูลที่ใช้อยู่ในปัจจุบันเพื่อใช้ในการกำหนดหัวข้อ เป้าหมาย จุดประสงค์ และขอบเขตการทำวิทยานิพนธ์
2. ทำการทดลองนำวิธีการซ่อนข้อมูลลงในเอกสารภาษาอังกฤษมาประยุกต์ใช้กับเอกสารภาษาไทยเพื่อระบุปัญหาที่เกิดขึ้นว่าวิธีการใดสามารถนำมาประยุกต์ใช้ได้หรือไม่ อย่างไร
3. หาวิธีการที่เหมาะสมในการซ่อนข้อมูลลงในเอกสารภาษาไทย
4. ทำการทดลองนำวิธีการที่พัฒนาขึ้นมาใช้กับเอกสารภาษาไทยในลักษณะต่างๆ รวมถึงทำการปรับปรุงแก้ไขวิธีการซ่อนข้อมูลนั้นให้สามารถใช้งานได้ตามวัตถุประสงค์ที่ตั้งไว้
5. จัดทำเอกสารประกอบวิทยานิพนธ์

1.7 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ได้แบ่งเนื้อหาออกเป็น 5 บท ประกอบด้วย

บทที่ 1 กล่าวถึง ความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ และสมมติฐานของการศึกษา รวมทั้งทฤษฎีหรือแนวคิด ขอบเขต และวิธีที่ใช้ในการทำวิจัยนี้

บทที่ 2 อธิบายหลักการทั้งหมดของการซ่อนข้อมูลสำหรับเอกสารภาพ ประกอบด้วย หลักการซ่อนข้อมูลเบื้องต้น รูปแบบของเอกสารรูปภาพ วิธีการซ่อนข้อมูลในเอกสารรูปภาพ รวมถึงรูปแบบโครงสร้างการเขียนของภาษาไทย และการแบ่งขอบเขตของตัวอักษรในเอกสารรูปภาพ ในส่วนท้ายของบทนี้จะอธิบายถึงหลักการแก้ไขความผิดพลาดของข้อมูลและหลักการของลายเซ็นดิจิตอลที่นำมาใช้ในงานวิจัยนี้

บทที่ 3 อธิบายวัตถุประสงค์ของการซ่อนข้อมูลที่ใช้ในงานวิจัยนี้ แนวคิดของวิธีการซ่อนข้อมูล อัลกอริทึมการเตรียมข้อมูลที่จะซ่อนลงในเอกสาร อัลกอริทึมการซ่อนข้อมูลลงในเอกสาร และอัลกอริทึมการดึงข้อมูลออกจากเอกสาร

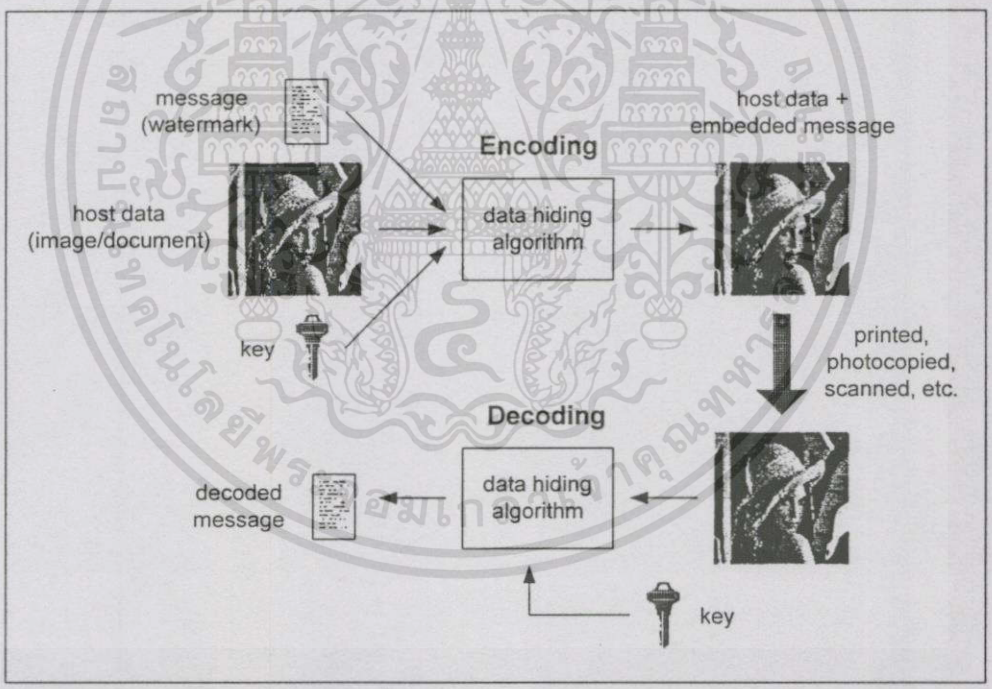
บทที่ 4 กล่าวถึงผลการทดลองซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทยโดยจะแยกเป็น 2 การทดลองคือ การทดลองดึงข้อมูลจากเอกสารที่ผ่านกระบวนการปรับขนาดและการทดลองดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการถ่ายเอกสาร ในส่วนท้ายของบทจะกล่าวถึงข้อสรุปและปัญหาที่พบจากการทดลอง

บทที่ 5 เป็นการสรุปผลการวิจัย ข้อเสนอแนะ และแนวทางการพัฒนาในอนาคต

บทที่ 2
หลักการที่เกี่ยวข้อง

2.1 การซ่อนข้อมูล

การซ่อนข้อมูล (Data hiding) [1, 2] คือ การฝังข้อมูลบางอย่างลงในสื่อโดยที่สื่อนั้นอาจเป็นรูปภาพ วีดีโอ เสียง หรือเอกสารต่างๆ โดยทั่วไปการซ่อนข้อมูลจะแบ่งการทำงานออกเป็น 2 ส่วน (พิจารณารูปที่ 2.1) คือ ส่วนของการซ่อนข้อมูลลงในสื่อ (Encoding) และส่วนของการดึงข้อมูลออกจากสื่อ (Decoding) ซึ่งหลักการทำงานทั้งสองส่วนนี้จะต้องมีความสอดคล้องกันจึงจะให้ผลการทำงานที่ถูกต้อง อนึ่งในกรณีที่ต้องการเพิ่มความปลอดภัยให้กับข้อมูลที่จะซ่อนลงใส่สื่อสามารถกระทำได้โดยการนำข้อมูลเหล่านั้นไปผ่านกระบวนการเข้ารหัสข้อมูล (Encryption) เสียก่อน



รูปที่ 2.1 แสดงรูปแบบการทำงานโดยทั่วไปของการซ่อนข้อมูล

2.1.1 คุณสมบัติของการซ่อนข้อมูล

1) สื่อที่ถูกซ่อนข้อมูลแล้วจะต้องไม่ถูกสังเกตเห็นการเปลี่ยนแปลงได้ง่าย

2) การซ่อนข้อมูลควรจะซ่อนลงไปที่ตัวสื่อโดยตรงแทนที่จะซ่อนในส่วนของเฮดเดอร์ไฟล์ เนื่องจากการซ่อนข้อมูลลงในส่วนของเฮดเดอร์ไฟล์อาจถูกตรวจพบได้ง่ายหรืออาจสูญหายไปได้ในกระบวนการพิมพ์หรือการแปลงรูปแบบเอกสาร

3) ข้อมูลที่ถูกซ่อนภายในสื่อจะต้องมีความคงทนต่อการที่สื่อผ่านกระบวนการต่างๆ เช่น การพิมพ์ (Printing) การสแกน (Scanning) การถ่ายเอกสาร (Photocopying) การแปลงสื่อจากข้อมูลดิจิตอลเป็นข้อมูลอนาล็อก (Digital-to-analog conversion) หรือการแปลงสื่อจากข้อมูลอนาล็อกเป็นข้อมูลดิจิตอล (Analog-to-digital conversion)

2.1.2 วัตถุประสงค์ของการซ่อนข้อมูล

โดยทั่วไปเราสามารถแบ่งวัตถุประสงค์ของการซ่อนข้อมูลได้ 3 ประการดังนี้

1) การซ่อนข้อมูลที่เป็นความลับ (Secret message hiding) มีจุดประสงค์เพื่อที่จะส่งข้อมูลที่เป็นความลับไปพร้อมกับสื่อโดยที่ไม่ทำให้บุคคลอื่นสามารถสังเกตเห็นได้ ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือข้อมูลที่ถูกซ่อนต้องมีความปลอดภัยสูงและควรจะซ่อนข้อมูลได้ปริมาณมาก

2) การซ่อนข้อมูลเพื่อใช้แสดงลิขสิทธิ์ความเป็นเจ้าของในสื่อนั้น (Watermarks) มีจุดประสงค์เพื่อที่จะใช้ในการพิสูจน์หรือตรวจสอบความเป็นเจ้าของในสื่อ ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือข้อมูลที่ถูกซ่อนต้องมีความคงทนสูงต่อการที่สื่อผ่านกระบวนการต่างๆและไม่ถูกทำลายได้ง่าย

3) การซ่อนข้อมูลรายละเอียดของสื่อ (Captions) มีจุดประสงค์เพื่อที่จะซ่อนข้อมูลที่เป็นรายละเอียดของสื่อเพิ่มเติมลงไปในสื่อด้วย ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือต้องสามารถซ่อนข้อมูลได้ในปริมาณมากและสามารถดึงข้อมูลออกมาได้อย่างไม่ยุ่งยากนัก

2.2 รูปแบบของเอกสารรูปภาพ

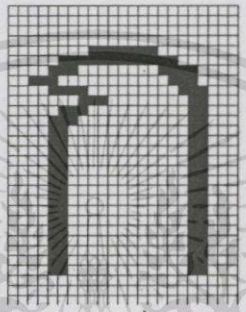
ในปัจจุบันสื่อประเภทข้อความหรือเอกสารต่างๆ เช่น หนังสือพิมพ์หรือวารสารมักจะถูกแปลงให้อยู่ในรูปแบบทางอิเลคทรอนิคที่มีการจัดเก็บข้อมูลแบบดิจิตอล (Electronic documents) เพื่อที่จะสามารถส่งเอกสารข้ามระบบเครือข่ายไปยังผู้รับได้อย่างสะดวกรวดเร็ว โดยเอกสารอิเลคทรอนิคจะมีรูปแบบการจัดเก็บข้อมูลที่แตกต่างกันหลายลักษณะ เช่น เอกสารอิเลคทรอนิคที่อยู่ในรูปแบบของเวิร์ดโปรเซสซิง (Word processing file) เอกสารอิเลคทรอนิคที่อยู่ในรูปแบบของโพสคริปท์ (Postscipt file) หรือเอกสารอิเลคทรอนิคที่อยู่ในรูปแบบรูปภาพ (Document image file) เป็นต้น

เอกสารที่จะนำมาซ่อนข้อมูลในงานวิจัยนี้จะเป็นเอกสารรูปภาพที่มีการจัดเก็บแบบบิตแมป (Bitmap file) ซึ่งมีรูปแบบฟังก์ชันดังนี้ [3]

$$f(x,y) \in \{0,1\}; \qquad x = 0,...,W-1;\ y = 0,..,L-1; \tag{2.1}$$

โดยที่ W และ L คือค่าความกว้างและความยาวของหน้าเอกสารในหน่วยของพิกเซล และ $f(x,y)$ คือค่าความสว่างของแต่ละพิกเซล

เอกสารรูปภาพแบบบิตแมปที่ใช้ในงานวิจัยนี้จะเป็นเอกสารรูปภาพแบบไบนารี (Binary image) ดังนั้นค่าความสว่างของแต่ละพิกเซลจะมีค่าเป็น "0" หรือ "1" เท่านั้น โดยพิกเซลที่มีค่าความสว่างเป็น "0" จะแทนจุดดำมืดทึบ (ส่วนของตัวอักษร) ส่วนพิกเซลที่มีค่าความสว่างเป็น "1" จะแทนจุดสว่างหรือจุดที่เป็นพื้นกระดาษสีขาว (ดูตัวอย่างของเอกสารรูปภาพไบนารีที่มีการจัดเก็บแบบบิตแมปในรูปที่ 2.2)



รูปที่ 2.2 แสดงตัวอย่างของเอกสารรูปภาพไบนารีที่มีการจัดเก็บแบบบิตแมป

2.3 วิธีการซ่อนข้อมูลลงในเอกสารรูปภาพ

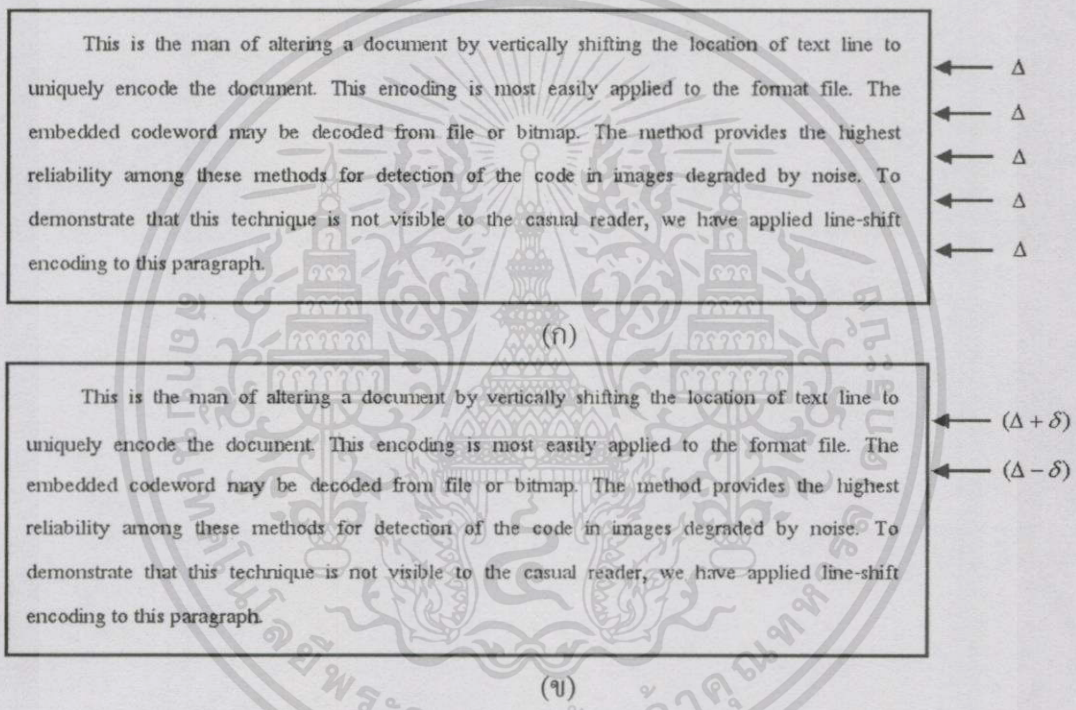
ที่ผ่านมาได้มีงานวิจัยเกี่ยวกับการซ่อนข้อมูลลงในเอกสารรูปภาพหลายวิธี [4, 5, 6, 7 , 8] ตัวอย่างเช่น การซ่อนข้อมูลโดยการเปลี่ยนแปลงรูปแบบของเอกสาร (Text formatting) หรือการซ่อนข้อมูลโดยการเปลี่ยนแปลงลักษณะของตัวอักษรภายในเอกสาร (Textual character) เป็นต้น โดยมีรายละเอียดของวิธีการต่างๆดังนี้

2.3.1 Line shift coding

วิธีการนี้จะซ่อนข้อมูลโดยการเปลี่ยนแปลงตำแหน่งของบรรทัดตามแนวตั้ง (เลื่อนขึ้น-เลื่อนลง) ซึ่งมีหลักการคือ ถ้าบรรทัดใดๆถูกเลื่อนตำแหน่งขึ้นจากตำแหน่งเดิมจะแทนข้อมูลไบนารี "1" แต่ถ้าบรรทัดใดถูกเลื่อนตำแหน่งลงจากตำแหน่งเดิมจะแทนข้อมูลไบนารี "0" โดยบรรทัดที่ถูกเปลี่ยนแปลงตำแหน่งนี้จะต้องอยู่ระหว่างบรรทัดที่ไม่มีการเปลี่ยนแปลงตำแหน่งเท่านั้น ดังนั้นบรรทัดที่สามารถซ่อนข้อมูลได้คือ บรรทัดที่ 2, 4, 6, ... พิจารณารูปที่ 2.3 แสดงตัวอย่างของการซ่อนข้อมูลโดยใช้วิธีการนี้ โดยรูปที่ 2.3 (ก) แสดงรูปเอกสารต้นฉบับที่ยังไม่ถูกซ่อนข้อมูล (ช่องว่างระหว่างบรรทัดจะมีขนาดเท่ากันทั้งหมดในทุกๆตำแหน่ง) ส่วนรูปที่ 2.3 (ข) แสดงรูปเอกสารที่ถูกซ่อนข้อมูลลงในบรรทัดที่สองโดยการเลื่อนตำแหน่งของบรรทัดนี้ลงมาจากตำแหน่งเดิมเล็กน้อย (δ) เพื่อซ่อนข้อมูล "0" ทำให้ช่องว่างระหว่างบรรทัดที่หนึ่งกับสองนั้นจะมีขนาดกว้างขึ้น ($\Delta+\delta$) ส่วนช่องว่างระหว่างบรรทัดที่สองกับสามนั้นจะมีขนาดลดลง ($\Delta-\delta$) สำหรับหลักการ

ที่ใช้ดึงข้อมูลออกจากเอกสารนั้นจะพิจารณาจากขนาดช่องว่างระหว่างบรรทัดที่ถูกเปลี่ยนแปลงไปเพื่อกำหนดค่าบิตข้อมูลที่ถูกซ่อนภายในเอกสาร

ข้อดีของวิธีการนี้คือข้อมูลที่ถูกซ่อนจะมีความทนทานต่อการที่เอกสารผ่านกระบวนการประมวลผลทางด้านเอกสารได้มากกว่าวิธีการอื่นๆ แต่มีข้อจำกัดคือจะซ่อนข้อมูลได้น้อยเนื่องจากต้องใช้จำนวนบรรทัดถึงสองบรรทัดในการซ่อนข้อมูลเพียงหนึ่งบิต อีกทั้งการซ่อนข้อมูลด้วยวิธีการนี้จะทำให้เอกสารถูกสังเกตการเปลี่ยนแปลงได้ง่ายเนื่องจากวิธีการนี้จะเปลี่ยนแปลงตำแหน่งบรรทัดทั้งบรรทัดในการซ่อนข้อมูลทำให้ช่องว่างระหว่างบรรทัดภายในหน้าเอกสารมีขนาดไม่เท่ากันซึ่งโดยทั่วไปแล้วช่องว่างระหว่างบรรทัดภายในหน้าเอกสารควรจะมีขนาดเท่ากัน



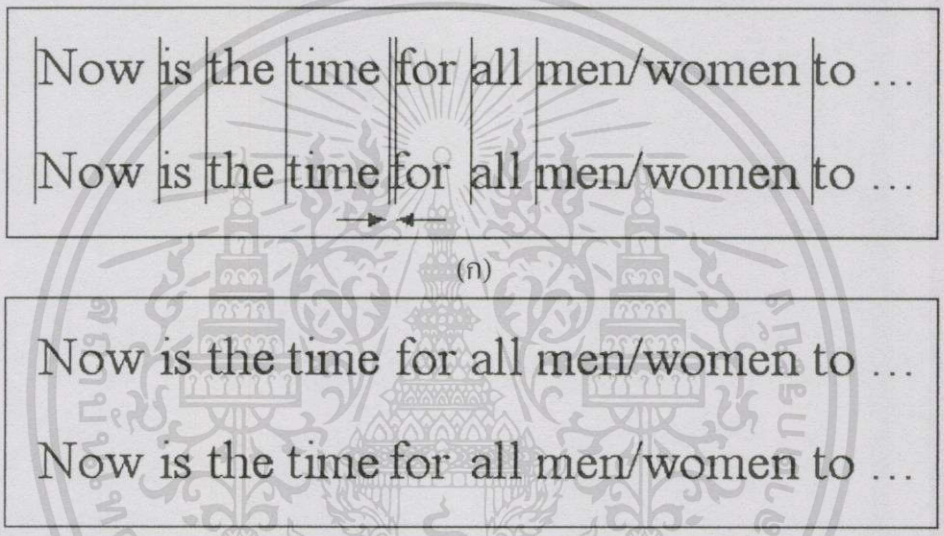
รูปที่ 2.3 แสดงตัวอย่างการซ่อนข้อมูลโดยใช้วิธี Line shift coding (ก) เอกสารต้นฉบับ (ข) เอกสารที่ถูกซ่อนข้อมูล "0" ลงในบรรทัดที่สอง

2.3.2 Word shift coding

วิธีการนี้จะซ่อนข้อมูลโดยการเปลี่ยนแปลงตำแหน่งของคำในแต่ละบรรทัดตามแนวนอน (เลื่อนไปทางซ้าย-ขวา) ซึ่งมีหลักการคือถ้าคำใดถูกเลื่อนจากตำแหน่งเดิมไปทางขวาจะแทนข้อมูลไบนารี "1" แต่ถ้าคำใดถูกเลื่อนจากตำแหน่งเดิมไปทางซ้ายจะแทนข้อมูลไบนารี "0" โดยคำที่ถูกเปลี่ยนแปลงตำแหน่งนี้จะต้องอยู่ระหว่างคำที่ไม่ถูกเปลี่ยนแปลงตำแหน่ง พิจารณารูปที่ 2.4 แสดงตัวอย่างการซ่อนข้อมูลลงในเอกสารโดยใช้วิธีการนี้ ซึ่งข้อมูลถูกซ่อนลงที่ช่องว่างหน้าตำแหน่งของคำว่า "for" โดยการเลื่อนตำแหน่งของคำนี้ไปทางซ้ายเล็กน้อยเพื่อซ่อนข้อมูล "0"

สำหรับหลักการที่ใช้ดึงข้อมูลออกจากเอกสารนั้นจะพิจารณาจากขนาดของช่องว่างระหว่างคำที่ถูกเปลี่ยนแปลงตำแหน่งไปจากเดิมเพื่อกำหนดค่าบิตข้อมูลที่ถูกซ่อน

ข้อดีของวิธีการนี้คือเอกสารจะถูกสังเกตเห็นการเปลี่ยนแปลงได้น้อยกว่าวิธี Line shift coding เนื่องจากช่องว่างระหว่างคำในแต่ละบรรทัดนั้นมีขนาดไม่แน่นอนอยู่แล้ว (สำหรับเอกสารที่มีการจัดแบบชิดขอบ) ดังนั้นถึงแม้ว่าภายในเอกสารจะมีการเปลี่ยนแปลงตำแหน่งของคำก็ไม่สามารถสังเกตเห็นการเปลี่ยนแปลงได้ อีกทั้งวิธีการนี้ยังสามารถซ่อนข้อมูลได้มากกว่าวิธีการแรก แต่จะมีความทนทานต่อการที่เอกสารผ่านกระบวนการประมวลผลเอกสารได้น้อยกว่าและข้อจำกัดอีกอย่างของวิธีการนี้คือสามารถใช้ได้กับเอกสารที่มีช่องว่างระหว่างคำเท่านั้น



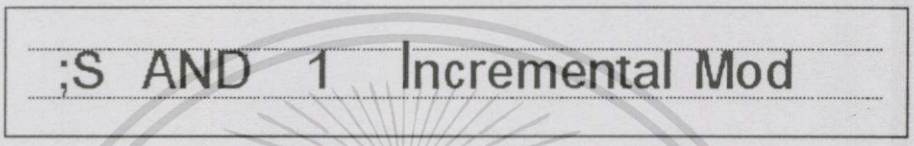
รูปที่ 2.4 แสดงตัวอย่างการซ่อนข้อมูลโดยใช้วิธี Word shift coding (ก) "for" ถูกเลื่อนตำแหน่งไปทางซ้าย ทำให้ช่องว่างทางซ้ายมีขนาดลดลงเพื่อซ่อนข้อมูล "0" (ข) แสดงรูปเดียวกับ (ก) แต่ไม่มีเส้นกำหนดในแนวตั้ง

2.3.3 Feature coding

วิธีการนี้จะซ่อนข้อมูลโดยการเปลี่ยนแปลงลักษณะของตัวอักษรภายในเอกสาร ซึ่งมีหลักการคือตัวอักษรใดที่ถูกเปลี่ยนแปลงจะแทนด้วยข้อมูลไบนารี "1" ส่วนตัวอักษรที่ไม่ถูกเปลี่ยนแปลงจะแทนด้วยข้อมูลไบนารี "0" การเปลี่ยนแปลงตัวอักษรนั้นสามารถทำได้โดยการลดหรือเพิ่มขนาดความยาวของตัวอักษร เช่น การเพิ่มความยาวของตัวอักษร b, d, หรือ h พิจารณารูปที่ 2.5 แสดงตัวอย่างการซ่อนข้อมูลด้วยวิธีการนี้โดยการเพิ่มความยาวของตัวอักษร "l" เพื่อซ่อนข้อมูล "1" และลดความยาวของตัวอักษร "t" เพื่อซ่อนข้อมูล "0" สำหรับหลักการที่ใช้ในการดึงข้อมูลที่ซ่อนออกจากเอกสารนั้นสามารถทำได้โดยการนำเอกสารที่ซ่อนข้อมูลมาเปรียบเทียบกับเอกสารต้นฉบับ

ข้อดีของวิธีการนี้คือสามารถซ่อนข้อมูลได้ในปริมาณที่มากและเอกสารจะถูกสังเกตเห็นการเปลี่ยนแปลงได้ยากมากเนื่องจากวิธีการนี้จะเปลี่ยนแปลงลักษณะของตัวอักษรเพียงเล็กน้อยเท่านั้น แต่วิธีการนี้มีข้อจำกัดคือข้อมูลที่ถูกซ่อนจะมีความทนทานต่อการที่เอกสารผ่านกระบวนการประมวลผลทางด้านเอกสารน้อยมาก

(ก)



(ข)

รูปที่ 2.5 ตัวอย่างการซ่อนข้อมูลโดยใช้วิธี Feature coding (ก) ข้อความต้นฉบับ (ข) ข้อความที่ซ่อนข้อมูลโดยการเพิ่มความยาวของ "l" และลดความยาวของ "t" (ในภาพเป็นการแสดงการแก้ไขด้วยขนาดที่เกินจริงเพื่อให้สังเกตเห็นตำแหน่งที่ซ่อนได้ง่าย)

2.4 โครงสร้างการเขียนของภาษาไทย

ตัวอักษรที่ใช้ในภาษาไทยจะถูกแบ่งออกเป็นประเภทต่างๆตามลักษณะหน้าที่ของตัวอักษรแต่ละตัวซึ่งเราสามารถสรุปประเภทของตัวอักษรในภาษาไทยได้ 4 ประเภทคือ ตัวอักษรที่เป็นพยัญชนะ สระ วรรณยุกต์และตัวอักขระพิเศษ (ดูรายละเอียดในตารางที่ 2.1)

ตารางที่ 2.1 แสดงรายละเอียดของการแบ่งประเภทของตัวอักษรภาษาไทย

ประเภทที่	ประเภทของตัวอักษร	รายละเอียด
1.	พยัญชนะ	ก – ฮ
2.	สระ - สระระดับบน - สระระดับกลาง - สระระดับล่าง	 อิ อี อึ อื อะ อา โอ ไอ ใอ อุ อู
3.	วรรณยุกต์	อ่ อ้ อ๊ อ๋
4.	ตัวอักขระพิเศษ	อ์ อ็

โครงสร้างการเขียนของภาษาไทยจะมีการแบ่งระดับในการแสดงข้อความออกเป็น 4 ระดับ โดยจะแบ่งตามลักษณะหน้าที่ของตัวอักษรแต่ละประเภทซึ่งมีรายละเอียดดังนี้ (พิจารณาตัวอย่างในรูปที่ 2.6)

- ระดับที่ 1 เป็นระดับบนสุดเป็นระดับของวรรณยุกต์และตัวการันต์ (กรณีที่มีสระระดับบนอยู่ในระดับที่ 2)
- ระดับที่ 2 เป็นระดับกลางบนเป็นระดับของสระระดับบน วรรณยุกต์ การันต์ และไม้ไต่คู้
- ระดับที่ 3 เป็นระดับกลางล่างเป็นระดับของพยัญชนะและสระระดับกลาง
- ระดับที่ 4 เป็นระดับล่างสุดเป็นระดับของสระระดับล่าง

รูปที่ 2.6 แสดงตัวอย่างลักษณะการเขียนภาษาไทย

จะเห็นว่าตัวอักษรที่อยู่ในแต่ละระดับนั้นจะถูกแยกออกจากกันด้วยช่องว่างที่มีขนาดแตกต่างกัน ซึ่งเมื่อเปรียบเทียบขนาดของช่องว่างระหว่างระดับในทุกระดับ พบว่าช่องว่างระหว่างระดับที่สองกับสามจะมีความกว้างมากกว่าช่องว่างระหว่างระดับอื่นๆ และจากการสำรวจเอกสารขนาด A4 ที่ใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์ ที่มีการจัดตัวอักษรแบบชิดขอบคอลัมน์เดียว จำนวน 300 บรรทัด พบว่าบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับที่สองกับสามนั้นมีจำนวนมากกว่าบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับชั้นอื่นๆ โดยในหนึ่งบรรทัดนั้นจะมีจำนวนบล็อกตัวอักษรที่มีช่องว่างระหว่างระดับที่สองกับสามประมาณ 13 ตำแหน่ง ดังนั้นในงานวิจัยนี้จึงมุ่งเน้นที่จะใช้ประโยชน์จากช่องว่างระหว่างระดับที่สองกับสามในการซ่อนข้อมูล

2.5 การแบ่งขอบเขตตัวอักษรภายในเอกสารรูปภาพ

วิธีการแบ่งขอบเขตของตัวอักษรภายในเอกสารรูปภาพในงานวิจัยนี้จะอาศัยหลักการสร้างฟังก์ชันฮิสโตแกรม (Histogram function) ของหน้าเอกสาร [9] ซึ่งมี 2 รูปแบบคือ ฟังก์ชันฮิสโตแกรมในแนวนอน (Horizontal histogram function) และฟังก์ชันฮิสโตแกรมในแนวตั้ง (Vertical histogram function) โดยมีรายละเอียดดังนี้

2.5.1 ฟังก์ชันฮิสโตแกรมในแนวนอน

เป็นฟังก์ชันที่แสดงผลรวมของจำนวนพิกเซลที่มีค่าความสว่างเป็นสีดำ (ON-Value pixel) ในแนวแกน X ซึ่งมีรูปแบบฟังก์ชันดังนี้

$$h(y) = \sum_{x=0}^{W-1} f(x,y), \qquad y \in [0, L-1] \tag{2.2}$$

2.5.2 ฟังก์ชันฮิสโตแกรมในแนวตั้ง

เป็นฟังก์ชันที่แสดงผลรวมของจำนวนพิกเซลที่มีค่าความสว่างเป็นสีดำ (ON-Value pixel) ในแนวแกน Y ซึ่งมีรูปแบบฟังก์ชันดังนี้

$$v(x) = \sum_{y=0}^{L-1} f(x,y), \qquad x \in [0, W-1] \tag{2.3}$$

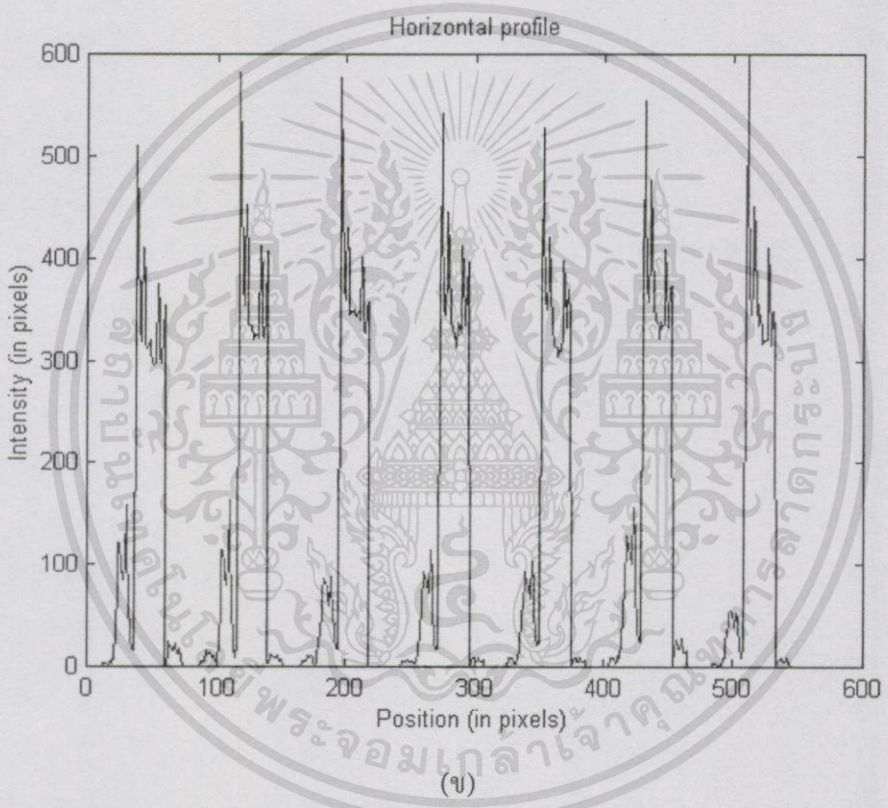
2.5.3 ตัวอย่างการแบ่งขอบเขตตัวอักษรโดยใช้ฟังก์ชันฮิสโตแกรม

ในส่วนนี้จะแสดงตัวอย่างฮิสโตแกรมในแนวนอนและแนวตั้งของเอกสารรูปภาพภาษาไทย โดยในรูปที่ 2.7 แสดงตัวอย่างฮิสโตแกรมในแนวนอนของหน้าเอกสารเพื่อกำหนดขอบเขตของแต่ละบรรทัด โดยมีข้อกำหนดคือระหว่างที่ทำการหาฮิสโตแกรมในแนวนอนถ้าพบว่ามีค่าฮิสโตแกรมที่จุดใดที่เปลี่ยนจากค่า "0" เป็นค่าอื่นที่มากกว่าให้สันนิษฐานว่าจุดนั้นเป็นจุดเริ่มต้นของเส้นบรรทัดบน (รูปที่ 2.7 (ก) แสดงตัวอย่างบางส่วนของเอกสารที่นำมาหาฮิสโตแกรมและรูปที่ 2.7 (ข) แสดงฮิสโตแกรมในแนวนอนของเอกสารในรูปที่ 2.7 (ก))

พิจารณารูปที่ 2.8 แสดงตัวอย่างฮิสโตแกรมในแนวตั้งของข้อความหนึ่งบรรทัดเพื่อกำหนดขอบเขตของตัวอักษรแต่ละตัวภายในบรรทัดนั้น โดยมีข้อกำหนดคือระหว่างที่ทำการหาฮิสโตแกรมในแนวตั้งถ้าพบว่ามีค่าฮิสโตแกรมที่จุดใดที่เปลี่ยนจากค่า "0" เป็นค่าอื่นที่มากกว่าให้สันนิษฐานว่าจุดนั้นเป็นจุดเริ่มต้นของเส้นขอบเขตของบล็อกตัวอักษรตัวใหม่ (รูปที่ 2.8 (ก) แสดงบางส่วนของข้อความในหนึ่งบรรทัดที่นำมาหาฮิสโตแกรมและรูปที่ 2.8 (ข) แสดงฮิสโตแกรมในแนวตั้งของรูปที่ 2.8 (ก)

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาใ ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่าง ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึ เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มาก ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเอี เอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในการค้นหาเอก

(ก)

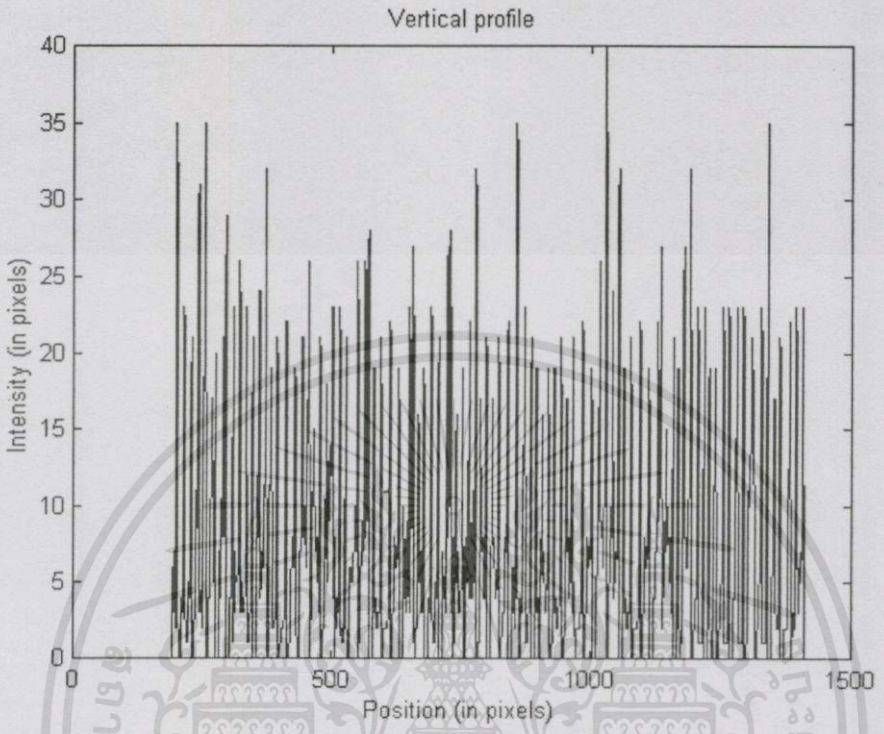


(ข)

รูปที่ 2.7 แสดงตัวอย่างการหาฮิสโตแกรมในแนวนอนของเอกสาร (ก) ตัวอย่างเอกสาร (ข) ฮิสโตแกรมในแนวนอนของเอกสารในรูป (ก)

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ

(ก)



(ข)

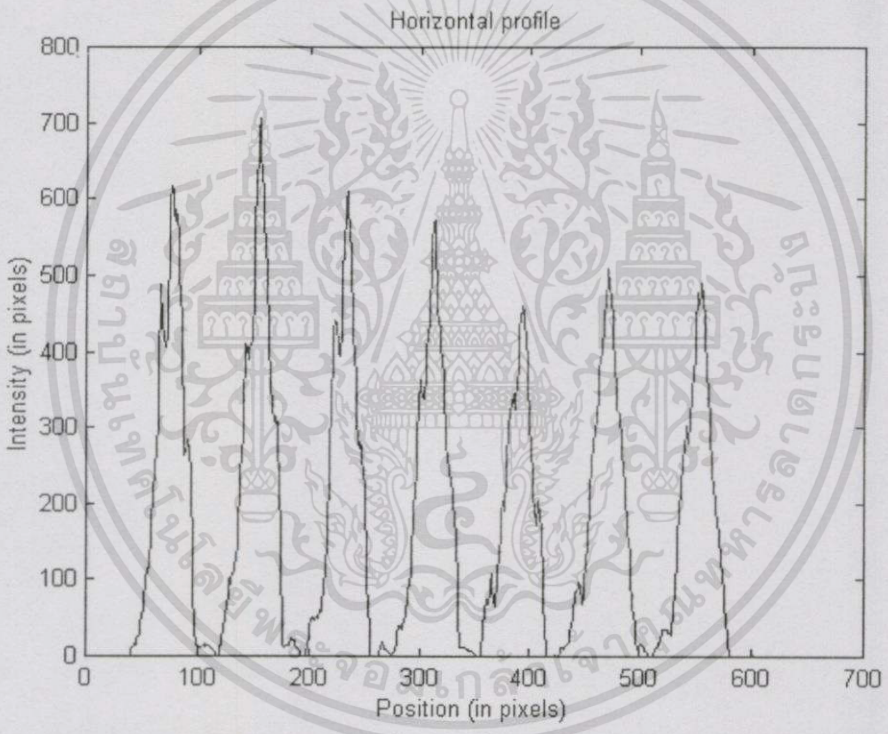
รูปที่ 2.8 แสดงตัวอย่างการหาฮิสโตแกรมในแนวตั้งของข้อความ 1 บรรทัด (ก) ตัวอย่างข้อความ 1 บรรทัด (ข) ฮิสโตแกรมในแนวตั้งของข้อความที่อยู่ในรูป (ก)

2.5.4 ข้อจำกัดของการแบ่งขอบเขตตัวอักษรโดยการใช้ฟังก์ชันฮิสโตแกรม

การใช้ฟังก์ชันฮิสโตแกรมในการแบ่งขอบเขตของตัวอักษรภายในหน้าเอกสารนั้นมีข้อจำกัดคือ ถ้าเอกสารที่จะนำมาแบ่งขอบเขตของตัวอักษรมีสิ่งรบกวนเกิดขึ้นเป็นจำนวนมากหรือเอกสารนั้นถูกทำให้เอียงไปจากเดิมมากอาจทำให้การแบ่งขอบเขตของตัวอักษรเกิดความผิดพลาดได้หรืออาจจะไม่สามารถแบ่งขอบเขตของตัวอักษรได้เลย พิจารณารูปที่ 2.9 แสดงตัวอย่างของเอกสารที่เกิดการบิดเบือนไปจากเดิมค่อนข้างมากเนื่องจากเอกสารผ่านกระบวนการถ่ายเอกสารและการสแกนที่ไม่มีประสิทธิภาพมากนัก (เอกสารสำเนาที่ 4) โดยรูปที่ 2.10 แสดงฮิสโตแกรมในแนวนอนของเอกสารในรูปที่ 2.9 ซึ่งจะเห็นว่าไม่สามารถใช้ฮิสโตแกรมในแนวนอนของเอกสารในการแบ่งขอบเขตของบรรทัดได้ เนื่องจากเอกสารเอียงไปจากเดิมมากจนทำให้ไม่มีช่องว่างระหว่างบรรทัด

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาใ ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่า ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึ เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มาก ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเอี เอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในการค้นหาเอก

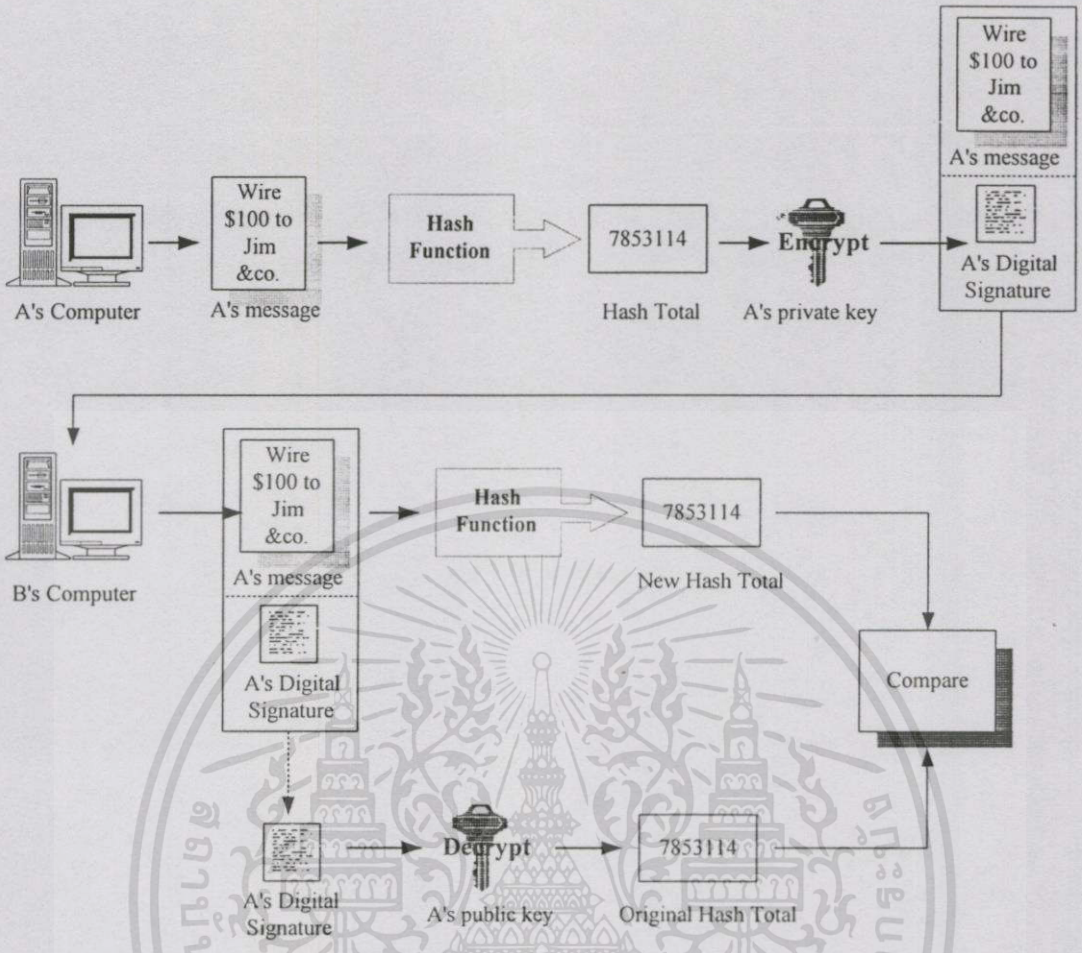
รูปที่ 2.9 แสดงตัวอย่างเอกสารที่เกิดการบิดเบือนไปจากเดิมมาก (เอกสารสำเนาครั้งที่ 5)



รูปที่ 2.10 แสดงฮิสโตแกรมในแนวนอนของเอกสารรูปที่ 2.9

2.6 ลายเซ็นดิจิตอล

ในงานวิจัยนี้ได้นำหลักการของลายเซ็นดิจิตอล [10] มาประยุกต์ใช้ในการตรวจสอบความถูกต้องของเอกสารว่าเอกสารถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องหรือไม่หรือเอกสารถูกเปลี่ยนแปลงแก้ไขมาหรือยัง (Document authentication) โดยการซ่อนข้อมูลที่เป็นลายเซ็นดิจิตอลของเอกสารแต่ละฉบับลงในเอกสาร



รูปที่ 2.11 แสดงหลักการทำงานโดยทั่วไปของลายเซ็นดิจิตอล

การสร้างลายเซ็นดิจิตอลนั้นจะใช้การเข้ารหัสข้อมูลแบบพับลิคคีย์ (Public-key encryption) ซึ่งจะใช้คู่คีย์ที่สอดคล้องกัน (Key–pair) ในการเข้ารหัสข้อมูลคือพับลิคคีย์ (Public key) และไพรเวตคีย์ (Private key) โดยที่พับลิคคีย์จะถูกแพร่กระจายให้บุคคลอื่นๆที่มีส่วนเกี่ยวข้องกับการรับ-ส่งข้อมูลสามารถรับรู้ได้ ส่วนไพรเวตคีย์นั้นจะถูกเก็บไว้เป็นความลับของแต่ละบุคคล โดยวิธีการเข้ารหัสที่นำมาใช้ในการสร้างลายเซ็นดิจิตอลในงานวิจัยนี้คือ การเข้ารหัสแบบ RSA Public-key encryption ซึ่งจะกล่าวถึงรายละเอียดในบทที่ 3 (หัวข้อ 3.3.2)

จากรูปที่ 2.11 แสดงหลักการทำงานของลายเซ็นดิจิตอลเพื่อใช้ตรวจสอบความถูกต้องของข้อความที่ส่งระหว่างผู้รับและผู้ส่ง โดยจะทำการตรวจสอบว่าข้อความที่ผู้รับได้รับนั้นเป็นข้อความที่ส่งมาจากเจ้าของเอกสารจริงๆหรือไม่และข้อความนั้นถูกเปลี่ยนแปลงแก้ไขมาแล้วหรือยัง ซึ่งจากตัวอย่างในรูปนี้กำหนดให้ A เป็นผู้ส่งข้อความและ B คือผู้รับข้อความ โดยมีรายละเอียดของการตรวจสอบความถูกต้องของข้อความดังนี้

1) A จะนำข้อความที่ต้องการส่งไปยัง B มาผ่านกระบวนการแฮชฟังก์ชัน (Hash function) เพื่อสร้างข้อมูลที่มีความเกี่ยวข้องกับเนื้อหาของข้อความโดยจะเรียกข้อมูลนี้ว่าข้อมูลแฮชโทเทล (Hash total)

2) จากนั้น A จะนำแฮชโทเทลที่ได้จากขั้นตอนที่แล้วมาเข้ารหัสด้วยไพรเวตคีย์ของ A เพื่อที่จะสร้างลายเซ็นดิจิตอลของข้อความนั้น และจะส่งข้อความและลายเซ็นดิจิตอลที่ได้นี้ไปยัง B

3) เมื่อ B ได้รับข้อความจาก A ก็จะนำข้อความนั้นมาผ่านกระบวนการแฮชฟังก์ชัน (ซึ่งเป็นกระบวนการเดียวกับที่ A ใช้) เพื่อสร้างข้อมูลแฮชโทเทลอันใหม่อีกหนึ่งอัน (New hash total)

4) จากนั้น B ก็ทำการถอดรหัสข้อความที่ได้รับจาก A ด้วยพับลิคคีย์ของตนเองซึ่งจะได้ข้อมูลแฮชโทเทลต้นฉบับ (Original hash total) นั่นเอง ผลลัพธ์ที่ได้จากการทำงานในขั้นตอนนี้จะทำให้ทราบว่าข้อความที่ B ได้รับนั้นเป็นข้อความที่ส่งมาจาก A จริงๆอย่างแน่นอน เนื่องจากจะมี A เพียงคนเดียวเท่านั้นที่สามารถเข้ารหัสข้อความนั้นด้วยรหัสลับของตนเองได้

5) จากนั้น B จะทำการเปรียบเทียบข้อมูลแฮชโทเทลต้นฉบับที่ได้มาจากการถอดรหัสลายเซ็นดิจิตอลกับข้อมูลแฮชโทเทลอันใหม่ที่คำนวณจากข้อความที่ได้รับ ซึ่งถ้าข้อมูลทั้งสองตัวนี้เหมือนกันก็จะทำให้เราสามารถยืนยันได้ว่าข้อความที่ส่งมานี้เป็นข้อความที่ส่งมาจาก A จริงๆและไม่ถูกเปลี่ยนแปลงแก้ไขมาก่อน

2.7 การตรวจสอบและแก้ไขความผิดพลาดของข้อมูล

ในงานวิจัยนี้ได้นำวิธีการแก้ไขความผิดพลาดของข้อมูลเข้ามาประยุกต์ใช้ร่วมกับการซ่อนข้อมูลด้วย เนื่องจากว่าข้อมูลที่ซ่อนอยู่ภายในเอกสารอาจเกิดความผิดพลาดได้เมื่อเอกสารนั้นผ่านการประมวลผลทางด้านเอกสารบางอย่าง เช่น การพิมพ์ การถ่ายเอกสาร การสแกน หรือการปรับขนาด ในส่วนนี้จะกล่าวถึงลักษณะของความผิดพลาดที่เกิดขึ้น [11] ประเภทของการตรวจหาและแก้ไขความผิดพลาด [12] และในส่วนท้ายจะกล่าวถึงรายละเอียดของวิธีการแก้ไขความผิดพลาดที่นำมาประยุกต์ใช้ในงานวิจัยนี้

2.7.1 ลักษณะของความผิดพลาด

1) ความผิดพลาดตำแหน่งเดียว (Single bit error) เป็นความผิดพลาดของข้อมูลที่เกิดขึ้นเพียง 1 ตำแหน่งภายในหนึ่งบล็อกข้อมูลเท่านั้น ตัวอย่างเช่น กำหนดให้ข้อมูลที่ส่งไปยังผู้รับคือ “11010100” แต่ข้อมูลที่ผู้รับได้รับคือ “11010110” จะเห็นว่ามีข้อมูลเพียงบิตเดียวที่เกิดความผิดพลาด (ตำแหน่งที่ 7 นับจากซ้ายมือ) โดยค่าของข้อมูลเปลี่ยนจาก “0” เป็น “1”

2) ความผิดสองตำแหน่งติดกัน (Double bit error) เป็นความผิดพลาดของข้อมูลที่เกิดขึ้นกับข้อมูลสองตำแหน่งที่อยู่ติดกันภายในหนึ่งบล็อกข้อมูล ตัวอย่างเช่น กำหนดให้ข้อมูลที่ส่งไปยังผู้รับคือ “11010100” แต่ข้อมูลที่ผู้รับได้รับคือ “11011000” จะเห็นว่าข้อมูลในตำแหน่งที่ 5 และ 6 (นับจากซ้ายมือ) เกิดความผิดพลาดขึ้นพร้อมกัน

3) ความผิดพลาดหลายตำแหน่งติดกัน (Burst error) เป็นความผิดพลาดของข้อมูลที่เกิดขึ้นติดต่อกันหลายตำแหน่งภายในหนึ่งบล็อกข้อมูล ตัวอย่างเช่น กำหนดให้ข้อมูลที่ส่งไปยังผู้รับคือ "1101010011110000" แต่ข้อมูลที่ผู้รับได้รับคือ "1101**10110000**1000" จะเห็นว่ามีข้อมูลที่ผิดพลาดติดต่อกันถึง 9 ตำแหน่ง (ตำแหน่งที่ขีดเส้นใต้)

2.7.2 ประเภทของการตรวจสอบและแก้ไขความผิดพลาด

วิธีการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลสามารถแบ่งได้เป็น 2 ลักษณะคือ

2.7.1.1 การตรวจจับความผิดพลาด

วิธีการตรวจจับความผิดพลาดของข้อมูล (Error detection) เป็นวิธีที่ทำให้ทราบว่าข้อมูลที่ได้รับมานั้นมีความผิดพลาดเกิดขึ้นหรือไม่ โดยมีหลักการคือจะเพิ่มข้อมูลบางอย่างร่วมไปกับข้อมูลที่ต้องการส่งซึ่งเรียกว่ารีดันแดนซี (Redundancy) ตัวอย่างของวิธีการตรวจจับความผิดพลาด เช่น การใช้พาริตี้บิตแบบต่างๆ (Parity checking) หรือการใช้วัฏจักรข่าวสาร (Cyclic redundancy check) ซึ่งเป็นวิธีที่นิยมนำมาใช้กันมากแต่การตรวจจับความผิดพลาดของข้อมูลนี้ไม่สามารถระบุตำแหน่งข้อมูลที่เกิดความผิดพลาดได้

2.7.1.2 การแก้ไขความผิดพลาด

วิธีการแก้ไขความผิดพลาดของข้อมูล (Error correction) เป็นวิธีที่สามารถตรวจจับความผิดพลาดของข้อมูลที่เกิดขึ้นและระบุตำแหน่งข้อมูลที่เกิดความผิดพลาดได้ทำให้สามารถแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ ซึ่งเราจะแบ่งลักษณะการแก้ไขความผิดพลาดของข้อมูลได้ 2 ลักษณะคือ

1) การส่งซ้ำ (Retransmission) เป็นหลักการที่ง่ายและให้ประสิทธิภาพสูงสุดในด้านความถูกต้องโดยที่ต้นทางและปลายทางจะต้องกำหนดให้มีรูปแบบการสื่อสารหรือโปรโตคอล (Protocol) แบบเดียวกันโดยกำหนดให้ปลายทางเป็นฝ่ายบอกกับต้นทางให้ส่งข้อมูลกลับมาใหม่โดยอัตโนมัติ (Automatic repeat request) เมื่อตรวจพบว่าข้อมูลเกิดความผิดพลาด

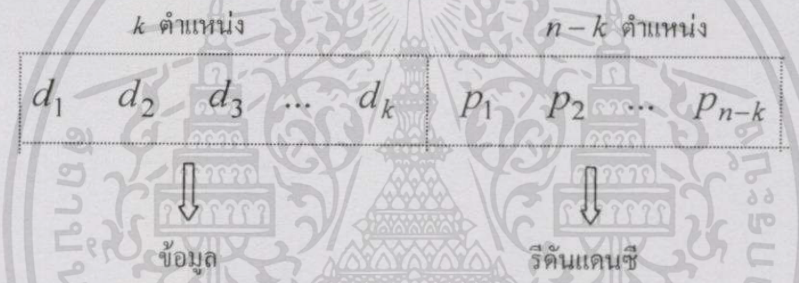
2) การแก้ไขทางตรง (Forward error correction) เป็นหลักการที่นำเอาวิธีการเพิ่มรีดันแดนซี่เข้ามาใช้ตรวจสอบความผิดพลาดของข้อมูลรวมทั้งแก้ไขความผิดพลาดที่เกิดขึ้นด้วย โดยข้อมูลทั้งหมดจะอยู่ในรูปแบบของการเข้ารหัส (Coding) ซึ่งมีอยู่หลายวิธี เช่น การเข้ารหัสแฮมมิง (Hamming code), รหัสฮาเกลบาร์เกอร์ (Hargelbarger code) และรหัสคอนโวลูชัน (Convolution) เป็นต้น เมื่อรหัสข้อมูลนี้ถูกส่งไปยังผู้รับหรือปลายทางก็จะถูกนำมาถอดรหัสด้วยวิธีที่สอดคล้องกับการเข้ารหัสเพื่อทำการตรวจสอบและแก้ไขความผิดพลาดที่อาจเกิดขึ้น

ขอบเขตลักษณะการเกิดความผิดพลาดของข้อมูลในงานวิจัยนี้ (ภายหลังจากที่ข้อมูลได้ผ่านกระบวนการป้องกันการเกิดความผิดพลาดแล้ว) จะเป็นความผิดพลาดเป็นแบบตำแหน่งเดียว

(Single bit error) เป็นส่วนใหญ่ ดังนั้นวิธีการแก้ไขความผิดพลาดในงานวิจัยนี้จึงเลือกที่จะใช้วิธีการแก้ไขแบบทางตรงคือสามารถตรวจจับความผิดพลาดของข้อมูลที่เกิดขึ้นและแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ทันทีโดยจะเลือกใช้วิธีการเข้ารหัสแฮมมิง (Hamming code) [13] ซึ่งเป็นวิธีการเข้ารหัสที่ได้รับการพัฒนาเพื่อใช้ตรวจสอบพาริตี้ชนิดหนึ่งที่กระทำการเข้ารหัสข้อมูลเป็นบล็อกหรือเป็นชุดๆ ได้

2.7.3 การเข้ารหัสแฮมมิง

การเข้ารหัสแฮมมิงในงานวิจัยนี้จะเป็นการเข้ารหัสแบบลิเนียร์บล็อก (Linear block code) มีหลักการคือจะจัดชุดข้อมูลให้มีขนาดเท่ากับ k บิตต่อบล็อกเพื่อนำมาเข้ารหัสซึ่งจะได้ส่วนต่อท้ายที่เรียกว่ารีดันแดนซีขนาด $n-k$ บิต โดยที่ n คือจำนวนบิตข้อมูลทั้งหมดของชุดรหัส (ดูโครงสร้างชุดข้อมูลแฮมมิงในรูปที่ 3.3.5)



รูปที่ 2.12 แสดงโครงสร้างชุดข้อมูลรหัสแฮมมิง *(n,k)*

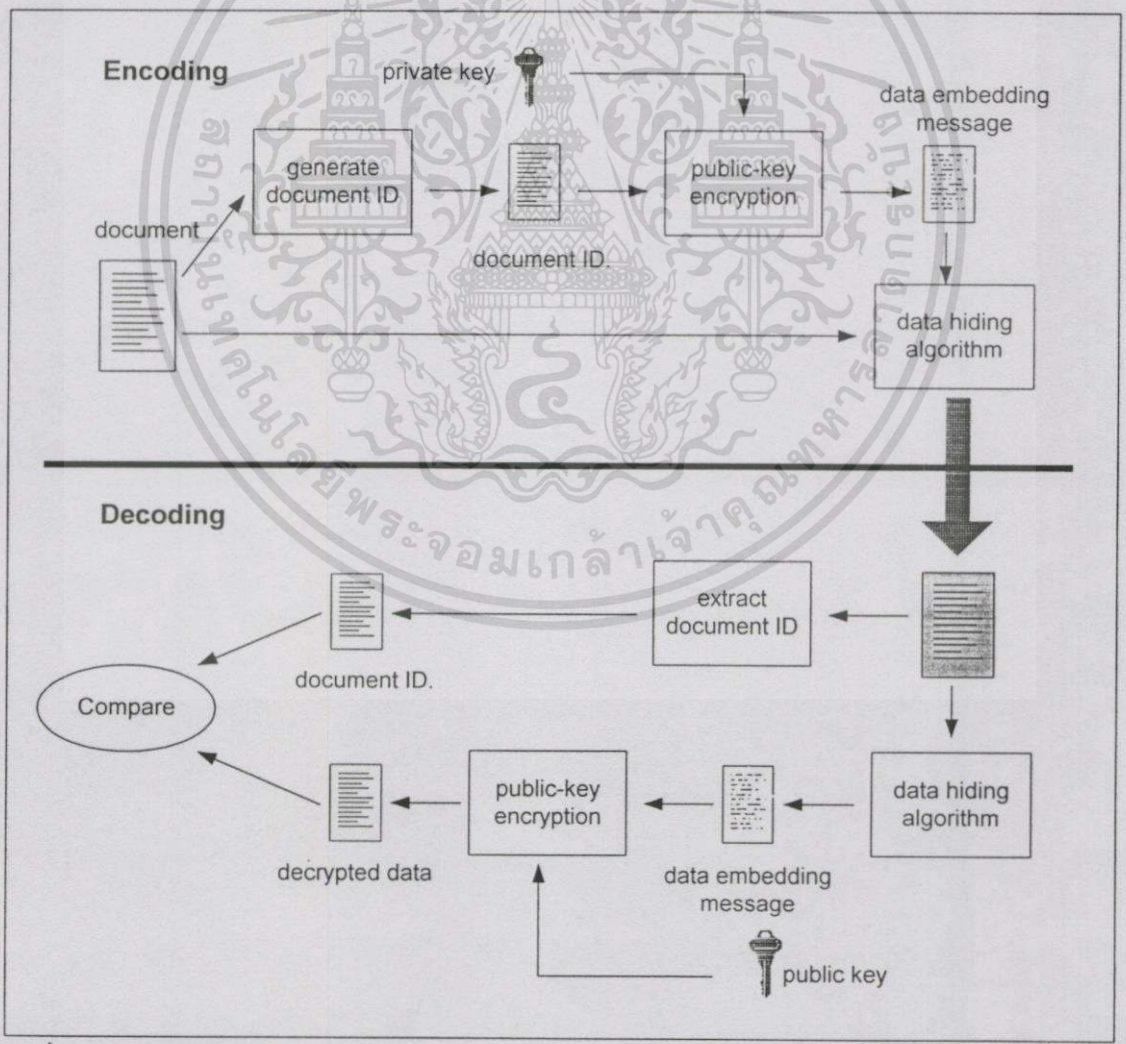
วิธีการเข้ารหัสแฮมมิงที่นำมาใช้ในงานวิจัยนี้คือการเข้ารหัสแฮมมิงแบบ (7,4) โดยที่ค่า n (จำนวนข้อมูลทั้งหมดในหนึ่งชุดข้อมูล) จะมีค่าเท่ากับ 7 และค่า k (จำนวนบิตข้อมูล) มีค่าเท่ากับ 4 ดังนั้นข้อมูลที่เป็นส่วนของรีดันแดนซีจะมีขนาดเท่ากับ $n-k$ ซึ่งก็คือ 3 ตำแหน่งนั่นเอง (รายละเอียดของการสร้างชุดข้อมูลรหัสแฮมมิง (7,4) จะแสดงในบทที่ 3 หัวข้อ 3.3.5.1)

บทที่ 3

วิธีการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย

3.1 การซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสาร

งานวิจัยนี้มีวัตถุประสงค์ที่จะนำวิธีการซ่อนข้อมูลที่พัฒนาขึ้นมาประยุกต์ใช้กับการตรวจสอบความถูกต้องของเอกสาร (Document authentication) [14, 15] โดยพิจารณาว่าเอกสารนั้นถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องหรือไม่หรือเอกสารนั้นถูกเปลี่ยนแปลงแก้ไขไปจากเดิมหรือยัง ซึ่งหลักการทำงานโดยทั่วไปของการซ่อนข้อมูลเพื่อวัตถุประสงค์นี้จะแบ่งออกเป็น 2 กระบวนการคือ กระบวนการซ่อนข้อมูล (Encoding) และกระบวนการดึงข้อมูล (Decoding) โดยมีรายละเอียดดังนี้ (ดูรูปที่ 3.1)



รูปที่ 3.1 แสดงหลักการทำงานทั่วไปของการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสาร

3.1.1 กระบวนการซ่อนข้อมูล มีขั้นตอนการทำงานดังนี้

1) สร้างหมายเลขประจำเอกสารแต่ละฉบับ (Document identification)

2) นำหมายเลขเอกสารที่ได้จากขั้นตอนแรกมาเข้ารหัสลับ (Public-key encryption) ซึ่งข้อมูลที่ได้นี้จะถูกนำมาซ่อนลงในเอกสาร (Data embedding message)

3) นำข้อมูลที่ได้จากขั้นตอนที่สองมาซ่อนลงในเอกสาร (Data hiding algorithm)

3.1.2 กระบวนการดึงข้อมูล มีขั้นตอนการทำงานดังนี้

1) ทำการดึงข้อมูลออกจากเอกสารที่ต้องการตรวจสอบความถูกต้อง (วิธีที่ใช้ดึงข้อมูลนี้จะต้องสอดคล้องกับวิธีการซ่อนข้อมูล)

2) นำข้อมูลที่ได้จากขั้นตอนแรกมาถอดรหัสลับโดยใช้วิธีการที่สอดคล้องกับการเข้ารหัสลับ (Public-key encryption)

3) จากนั้นนำข้อมูลที่ได้จากการถอดรหัสมาเปรียบเทียบกับหมายเลขประจำเอกสารต้นฉบับนั้นโดยมีกฎดังนี้

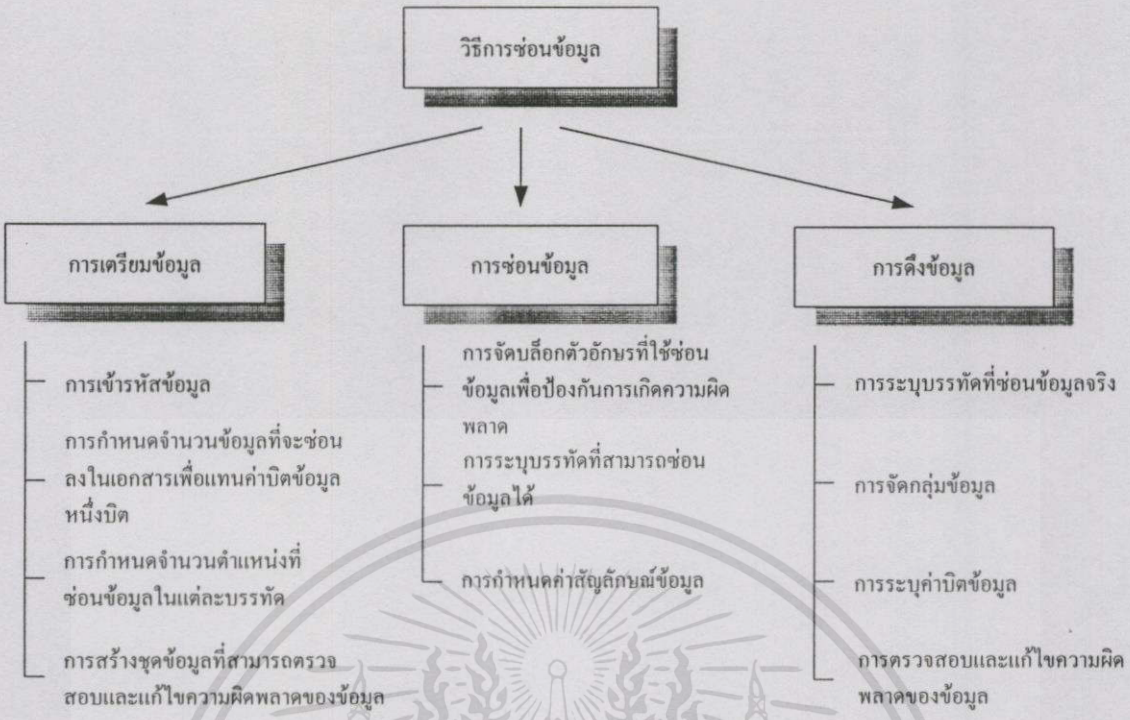
- ถ้าหมายเลขทั้งสองเหมือนกันแสดงว่าเอกสารฉบับนี้ถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องและไม่ถูกแก้ไขหรือเปลี่ยนแปลงแต่อย่างใด
- ถ้าหมายเลขทั้งสองไม่เหมือนกันแสดงว่าเอกสารฉบับนี้อาจถูกเปลี่ยนแปลงแก้ไขมาแล้ว

3.2 แนวคิดของวิธีการซ่อนข้อมูล

การซ่อนข้อมูลลงเอกสารภาพภาษาไทยในงานวิจัยนี้จะแบ่งการทำงานออกเป็น 3 ส่วนหลัก คือ ส่วนของการเตรียมข้อมูลที่จะซ่อนลงในเอกสาร ส่วนของการซ่อนข้อมูลลงในเอกสาร และส่วนของการดึงข้อมูลที่ถูกซ่อนออกจากเอกสาร (ดูรูปที่ 3.2)

3.2.1 การเตรียมข้อมูล

ข้อมูลที่จะซ่อนลงในเอกสารนั้นจะเป็นหมายเลขประจำเอกสารของเอกสารแต่ละฉบับ ซึ่งในงานวิจัยนี้จะสร้างหมายเลขประจำเอกสารแต่ละฉบับโดยการสุ่มตัวเลขจำนวนนับ (Random number) ขึ้นมาจำนวนหนึ่ง (หมายเลขประจำเอกสารอาจถูกสร้างมาจากวิธีการอื่นก็ได้) จากนั้นจะนำหมายเลขประจำเอกสารนี้มาเข้ารหัสลับเพื่อสร้างลายเซ็นดิจิตอลให้กับเอกสารแต่ละฉบับโดยใช้วิธี RSA Public-key encryption ซึ่งข้อมูลนี้จะถูกแปลงให้อยู่ในรูปแบบข้อมูลไบนารี (ข้อมูล “0” หรือ “1”) ก่อนที่จะซ่อนลงในเอกสาร

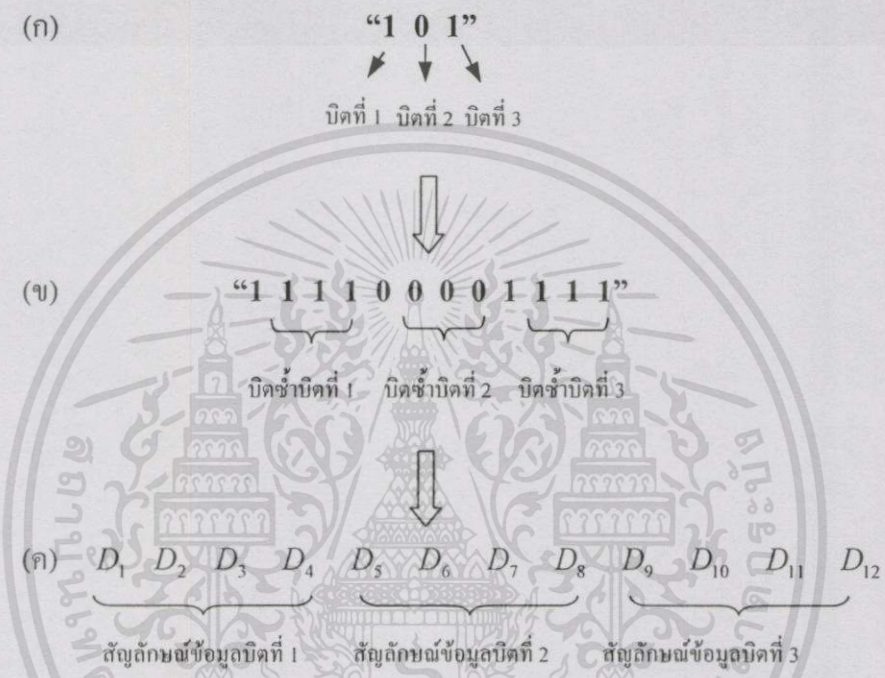


รูปที่ 3.2 แสดงโครงสร้างแนวคิดของการซ่อนข้อมูล

จากที่กล่าวในบทที่แล้วว่าข้อมูลที่ถูกซ่อนภายในเอกสารนั้นอาจเกิดความผิดพลาดหรือสูญหายได้เมื่อเอกสารนั้นผ่านกระบวนการประมวลผลทางด้านเอกสาร ดังนั้นในขั้นตอนของการเตรียมข้อมูลจึงได้นำวิธีการที่สามารถป้องกันหรือช่วยลดความผิดพลาดของข้อมูลที่อาจเกิดขึ้นมาประยุกต์ใช้ร่วมด้วยโดยมีรายละเอียดของวิธีการต่างๆดังนี้ (สองวิธีการแรกจะเป็นการสร้างข้อมูลเพื่อป้องกันการเกิดความผิดพลาดของข้อมูล ส่วนวิธีการหลังจะเป็นการสร้างข้อมูลเพื่อแก้ไขความผิดพลาดของข้อมูล)

1) การเพิ่มจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าข้อมูลหนึ่งบิต (Data bit) โดยกำหนดให้ R_d เป็นจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต ปัจจัยที่ใช้กำหนดค่า R_d นั้นจะพิจารณาจากจำนวนตำแหน่งข้อมูลที่สูญหายไปในหนึ่งบรรทัดเมื่อเทียบกับจำนวนตำแหน่งข้อมูลที่ถูกซ่อนในแต่ละบรรทัด (ได้มาจากการทดลองเบื้องต้น) ซึ่งมีหลักการคือจะเพิ่มข้อมูลซ้ำต่อท้ายข้อมูลในแต่ละบิต (Redundant data bit) โดยจำนวนข้อมูลที่ถูกเพิ่มขึ้นนี้จะขึ้นอยู่กับค่า R_d ที่กำหนดไว้ จากนั้นแปลงบิตข้อมูลทั้งหมดให้เป็นค่าสัญลักษณ์ข้อมูล (Embedded symbols) ที่ใช้กำหนดขนาดช่องว่างระหว่างระดับชั้นในการแทนค่าบิตข้อมูล สำหรับประโยชน์ที่ได้รับจากการเพิ่มจำนวนตำแหน่งที่ใช้ซ่อนข้อมูลคือจะช่วยป้องกันความผิดพลาดของข้อมูลที่อาจเกิดขึ้นได้ เช่น หากมีข้อมูลในบางตำแหน่งสูญหายไปก็ยังสามารถระบุค่าบิตข้อมูลนั้นๆได้โดยคำนวณหาจากค่าเฉลี่ยของข้อมูลที่ซ่อนบิตข้อมูลเดียวกันในตำแหน่งที่เหลือ

พิจารณารูปที่ 3.3 แสดงตัวอย่างการกำหนดจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูล (กำหนดให้จำนวนข้อมูลที่ใช้แทนค่าข้อมูลหนึ่งบิตคือ 4 ตำแหน่ง) โดยรูป 3.3 (ก) แสดงค่าบิตข้อมูลที่ต้องการซ่อนลงในเอกสาร ส่วนรูป 3.3 (ข) คือบิตข้อมูลทั้งหมดหลังจากที่ผ่านการเพิ่มบิตข้อมูลซ้ำแล้วและในรูปที่ 3.3 (ค) แสดงค่าสัญลักษณ์ข้อมูล (D) ที่จะซ่อนลงในเอกสาร (รายละเอียดของการกำหนดจำนวนข้อมูลจะกล่าวถึงในหัวข้อที่ 3.3.3)



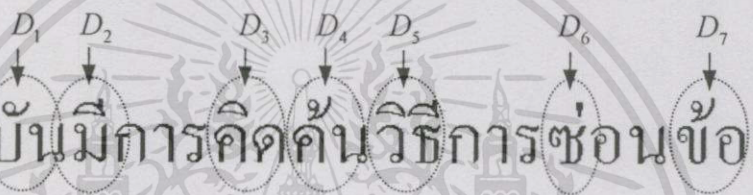
รูปที่ 3.3 แสดงตัวอย่างการกำหนดสัญลักษณ์ข้อมูลเพื่อแทนค่าบิตข้อมูล

2) การกำหนดจำนวนตำแหน่งที่จะซ่อนข้อมูลในแต่ละบรรทัด (L_d) โดยทั่วไปแล้วข้อความในแต่ละบรรทัดจะมีจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้แตกต่างกันขึ้นอยู่กับลักษณะของข้อความในบรรทัดนั้นๆ แต่วิธีการที่นำเสนอในงานวิจัยนี้จะซ่อนข้อมูลลงในแต่ละบรรทัดด้วยจำนวนที่เท่ากัน ดังนั้นจึงต้องมีการเลือกค่า L_d ที่เหมาะสมเพื่อที่จะทำให้วิธีการซ่อนข้อมูลนี้สามารถใช้ประโยชน์จากบรรทัดทุกบรรทัดภายในหน้าเอกสารได้อย่างเต็มที่ โดยปัจจัยที่ใช้กำหนดค่า L_d ที่เหมาะสมคือค่านี้ควรจะมีค่าน้อยกว่าหรือเท่ากับค่าเฉลี่ยของจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดภายในหน้าเอกสารทั้งหมด (M_d) และควรจะจำกัดบริเวณของการเกิดความผิดพลาดของข้อมูลได้ด้วย (ดูรายละเอียดในหัวข้อที่ 3.3.4)

3) ข้อมูลที่ถูกซ่อนลงในเอกสารนี้ควรจะตรวจสอบความถูกต้องของตนเองได้เพื่อแก้ไขความผิดพลาดของข้อมูลที่เกิดขึ้น ในงานวิจัยนี้เลือกใช้วิธีเข้ารหัสแฮมมิง (7,4) มาสร้างชุดข้อมูลที่สามารถตรวจสอบและแก้ไขความผิดพลาดของข้อมูล

3.2.2 การซ่อนข้อมูล

วิธีการซ่อนข้อมูลในงานวิจัยนี้จะซ่อนข้อมูลโดยการปรับเปลี่ยนขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสาม ซึ่งสาเหตุที่เลือกใช้ช่องว่างในตำแหน่งนี้ก็เนื่องจากว่ามีความกว้างมากกว่าช่องว่างในช่วงอื่นๆ (ช่องว่างระหว่างระดับหนึ่งกับสองและระดับสามกับสี่) ทำให้สามารถปรับเปลี่ยนขนาดความกว้างของช่องว่างได้อย่างง่ายและไม่เป็นที่ผิดสังเกตมากนักอีกทั้งยังมีตำแหน่งที่สามารถซ่อนข้อมูลได้มาก พิจารณารูปที่ 3.4 แสดงตัวอย่างของตำแหน่งบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ (บล็อกตัวอักษรภายในเส้นประ) โดยในแต่ละตำแหน่งจะซ่อนข้อมูลโดยการเปลี่ยนแปลงตำแหน่งของตัวอักษรที่อยู่ระดับบน (ระดับสอง) ตามแนวตั้ง (เลื่อนขึ้น-เลื่อนลง) ให้มีขนาดเท่ากับค่าสัญลักษณ์ข้อมูล (D) ที่ใช้แทนค่าบิตข้อมูลที่ต้องการซ่อน



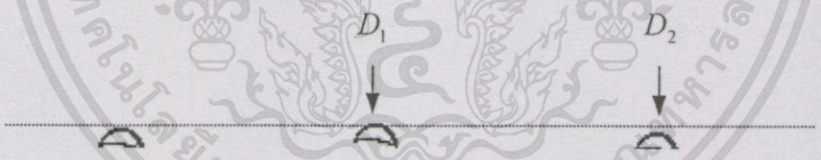
รูปที่ 3.4 แสดงตัวอย่างบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้

วิธีการซ่อนข้อมูลที่กล่าวถึงในงานวิจัยนี้สามารถป้องกันการเกิดความผิดพลาดของข้อมูลได้โดยการจัดบล็อกตัวอักษรให้อยู่ในอยู่ในตำแหน่งที่สามารถลดโอกาสของการเกิดความผิดพลาด เนื่องจากความผิดพลาดของข้อมูลที่เกิดขึ้นส่วนหนึ่งมาจากการที่มีข้อมูลในบางตำแหน่งสูญหายไป (ไม่สามารถตรวจพบช่องว่างที่ใช้ซ่อนข้อมูลได้) เพราะเอกสารผ่านการประมวลผลทางด้านเอกสารบางอย่างนั่นเอง วิธีการจัดตำแหน่งบล็อกตัวอักษรที่เหมาะสมจะพิจารณาจากขนาดของช่องว่างระหว่างบล็อกตัวอักษรที่อยู่ใกล้เคียงกันซึ่งถ้าพบว่าช่องว่างที่ตำแหน่งนั้นมีขนาดเล็กมากก็จะรวมบล็อกตัวอักษรทั้งสองให้เป็นบล็อกเดียวกัน แต่ถ้าช่องว่างที่ตำแหน่งนั้นมีโอกาสที่จะมาชิดติดกันได้ภายหลังจากที่เอกสารผ่านการประมวลผลทางด้านเอกสารก็จะทำการปรับขนาดช่องว่างระหว่างบล็อกตัวอักษรให้แยกออกจากกันอย่างชัดเจนมากยิ่งขึ้น แต่ถ้าช่องว่างที่ตำแหน่งนั้นมีขนาดความกว้างมากพอที่จะไม่เสี่ยงต่อการสูญหายก็จะไม่มีการเปลี่ยนแปลงตำแหน่งบล็อกตัวอักษรเหล่านั้น (ดูรายละเอียดในหัวข้อที่ 3.4.2)

หลักการที่สำคัญอีกอย่างหนึ่งของขั้นตอนการซ่อนข้อมูลคือการเลือกบรรทัดที่จะนำมาซ่อนข้อมูลโดยจะพิจารณาจากจำนวนตำแหน่งบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดซึ่งจะต้องมีค่ามากกว่าหรือเท่ากับ L_d ตำแหน่ง (ค่า L_d จะถูกกำหนดมาตั้งแต่ขั้นตอนของการเตรียมข้อมูล) สำหรับบรรทัดที่ไม่มีคุณสมบัติที่จะนำซ่อนข้อมูลได้นั้น (บรรทัดที่มีจำนวน

บล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้น้อยกว่า L_d ตำแหน่ง) จะถูกซ่อนข้อมูลที่ใช้เป็นค่าเธรดโชลด์ (δ_d) ของการระบุค่าบิตข้อมูลในขั้นตอนของการดึงข้อมูล

จากที่กล่าวมาว่าการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นสามารถทำได้โดยการปรับเปลี่ยนขนาดความกว้างของช่องว่างระหว่างระดับเพื่อแทนค่าข้อมูลที่ต้องการซ่อน (ข้อมูล "0" หรือ "1") ดังนั้นจึงต้องมีการกำหนดค่าสัญลักษณ์ข้อมูล (D) เพื่อกำหนดขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามในหน่วยของพิกเซลเพื่อระบุค่าบิตข้อมูลที่ต้องการซ่อน หลักการที่ใช้กำหนดค่าสัญลักษณ์ข้อมูลที่เหมาะสมสำหรับบิตข้อมูล "0" และ "1" คือค่าสัญลักษณ์ข้อมูลสำหรับบิตข้อมูลทั้งสองนี้ไม่ควรมีค่าแตกต่างกันมากหรือน้อยเกินไป เนื่องจากหากมีค่าแตกต่างกันมากเกินไปจะทำให้เอกสารที่ซ่อนข้อมูลถูกสังเกตเห็นความผิดปกติได้ง่ายแต่ถ้ามีค่าแตกต่างกันน้อยเกินไปอาจทำให้เกิดปัญหาในขั้นตอนการดึงข้อมูลได้ (ดูรายละเอียดในหัวข้อที่ 3.4.3) จากรูปที่ 3.5 แสดงตัวอย่างการซ่อนข้อมูลลงในบล็อกตัวอักษร "ว" โดยรูป 3.5 (ก) แสดงบล็อกตัวอักษรต้นฉบับที่ยังไม่ถูกซ่อนข้อมูล (ไม่มีการเปลี่ยนแปลงขนาดความกว้างของช่องว่างระหว่างระดับ) ส่วนรูป 3.5 (ข) แสดงบล็อกตัวอักษรที่ถูกซ่อนข้อมูล "1" (ค่าสัญลักษณ์ข้อมูลที่ใช้แทนค่าบิตข้อมูล "1" คือ D_1) และรูป 3.5 (ค) แสดงบล็อกตัวอักษรที่ถูกซ่อนข้อมูล "0" (ค่าสัญลักษณ์ข้อมูลที่ใช้แทนค่าบิตข้อมูล "0" คือ D_2) จะเห็นว่าการซ่อนข้อมูลด้วยวิธีการที่นำเสนอนี้จะซ่อนข้อมูลโดยการเปลี่ยนแปลงลักษณะของข้อความภายในเอกสารเพียงเล็กน้อยเท่านั้นทำให้สังเกตเห็นการเปลี่ยนแปลงของเอกสารที่ถูกซ่อนข้อมูลแล้วได้ยาก



ข้อมูล (Data bit) ซึ่งหลักการที่ใช้ระบุบรรทัดที่เป็นข้อมูลจริงจะพิจารณาจากค่าความสัมพันธ์ของค่าสัญลักษณ์ข้อมูลในแต่ละแถว โดยค่าสัญลักษณ์ข้อมูลที่อยู่ใกล้เคียงกันของแถวที่ถูกใช้ซ่อนข้อมูลจะมีค่าใกล้เคียงกัน (ในขั้นตอนของการซ่อนข้อมูลจะใช้ค่าสัญลักษณ์ข้อมูลเดียวกันหลายตำแหน่งเพื่อแทนค่าบิตข้อมูลหนึ่งบิต) แต่ค่าสัญลักษณ์ข้อมูลที่อยู่ใกล้เคียงกันของแถวที่ไม่ได้ถูกใช้ซ่อนข้อมูลจะมีค่าแตกต่างกันมาก (ค่าสัญลักษณ์ข้อมูลในบรรทัดนี้จะเป็นค่าของบิตข้อมูล "0" และ "1" สลับกัน)

หลังจากการระบุตำแหน่งบรรทัดที่ถูกใช้ซ่อนข้อมูลได้แล้วก็จะนำค่าสัญลักษณ์ข้อมูลเหล่านั้นมาจัดกลุ่มเพื่อให้อยู่ในกลุ่มที่ใช้แทนบิตเดียวกันที่ถูกต้องและทำการระบุค่าบิตข้อมูลในแต่ละกลุ่มโดยการนำค่าเฉลี่ยของค่าสัญลักษณ์ข้อมูลในแต่ละกลุ่มมาเปรียบเทียบค่าเธรดโชลด์ (δ_d) ที่คำนวณมาจากค่าเฉลี่ยของค่าสัญลักษณ์ข้อมูลทั้งหมดของบรรทัดที่ไม่ถูกใช้ซ่อนข้อมูล จากนั้นจะนำค่าบิตข้อมูลที่ได้ทั้งหมดไปผ่านกระบวนการตรวจสอบและแก้ไขความผิดพลาดของข้อมูล (ดูรายละเอียดของการดึงข้อมูลในหัวข้อ 3.5)

3.3 อัลกอริทึมของการเตรียมข้อมูล

ในหัวข้อนี้จะกล่าวถึงรายละเอียดของอัลกอริทึมที่ใช้เตรียมข้อมูลที่จะซ่อนลงในเอกสาร

3.3.1 ขั้นตอนการเตรียมข้อมูล

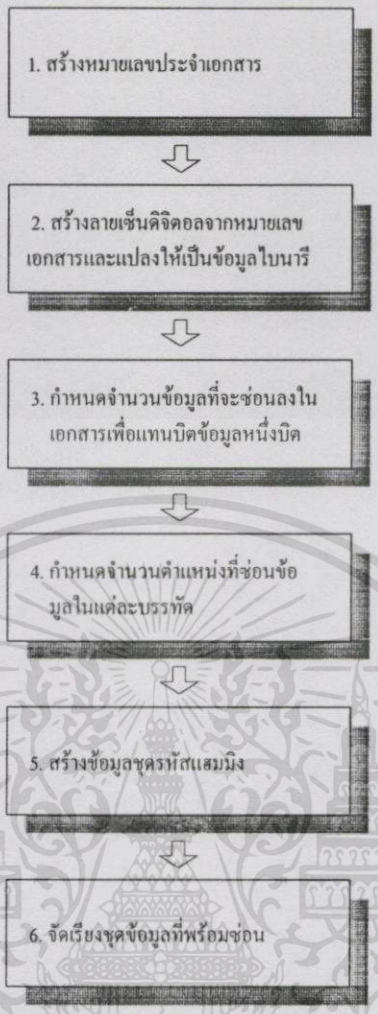
จากรูปที่ 3.6 แสดงขั้นตอนการเตรียมข้อมูลที่จะซ่อนลงในเอกสารซึ่งมีรายละเอียดดังนี้

1) สร้างหมายเลขประจำเอกสารโดยการสุ่มตัวเลขจำนวนนับขึ้นมาจำนวนหนึ่ง

2) นำหมายเลขประจำเอกสารที่ได้มาผ่านกระบวนการเข้ารหัสลับเพื่อสร้างลายเซ็นดิจิตอลโดยใช้วิธีการเข้ารหัสแบบ RSA Public-key encryption (ดูรายละเอียดในหัวข้อ 3.3.2) จากนั้นแปลงตัวเลขที่ได้จากการเข้ารหัสให้อยู่ในรูปแบบข้อมูลไบนารี

3) กำหนดจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต (R_d) จากนั้นทำการเพิ่มข้อมูลซ้ำต่อท้ายบิตข้อมูลแต่ละบิตเป็นจำนวน $R_d - 1$ ตำแหน่ง (ดูรายละเอียดในหัวข้อที่ 3.3.3)

4) กำหนดจำนวนตำแหน่งที่จะซ่อนข้อมูลในแต่ละบรรทัด (L_d) เพื่อจัดข้อมูลให้มีแถวละ L_d ตำแหน่ง หลังจากการจัดแถวข้อมูลทั้งหมดแล้วหากมีข้อมูลในแถวใดที่มีจำนวนไม่ครบ L_d ตำแหน่งก็จะเพิ่มบิตข้อมูล "0" ต่อท้ายข้อมูลในแถวนั้นให้ครบ L_d ตำแหน่ง (ดูรายละเอียดการกำหนดค่า L_d ในหัวข้อที่ 3.3.4)



รูปที่ 3.6 แสดงขั้นตอนการเตรียมข้อมูล

5) นำข้อมูลที่ได้จากขั้นตอนที่แล้วมาสร้างชุดข้อมูลรหัสแฮมมิง (7,4) โดยจะนำข้อมูลมาครั้งละ 4 แถวเพื่อสร้างข้อมูลในส่วนของรีดันแดนซีที่เป็นพาริตี้บิต (วิธีการสร้างชุดรหัสแฮมมิงนี้จะสร้างชุดรหัสในแนวตั้งโดยจะแบ่งเป็นส่วนของข้อมูล 4 ตำแหน่งและส่วนของรีดันแดนซีที่เป็นพาริตี้บิตอีก 3 ตำแหน่งซึ่งจะแสดงรายละเอียดในหัวข้อที่ 3.3.5) จนกระทั่งหมดชุดข้อมูล (สำหรับชุดข้อมูลที่มีไม่ครบ 4 แถวก็ให้เพิ่มแถวข้อมูลที่เป็น "0" เข้าไปจนกระทั่งครบ) ซึ่งชุดข้อมูลรหัสแฮมมิงที่ได้แต่ละชุดจะมีทั้งหมด 7 แถวโดยสี่แถวบนจะเป็นส่วนของข้อมูลและสามแถวล่างจะเป็นส่วนของรีดันแดนซี

6) จัดเรียงชุดรหัสแฮมมิงในแต่ละชุดเข้าด้วยกันเพื่อที่จะนำไปซ่อนลงในเอกสาร

3.3.2 การสร้างลายเซ็นดิจิตอล

3.3.2.1 การเข้ารหัสแบบ RSA Public-key encryption

หลักการทำงานโดยทั่วไปของเข้ารหัสแบบ RSA Public-key encryption [10] นั้นจะเป็นการเข้ารหัสข้อมูลเพื่อรักษาความปลอดภัยให้กับข้อมูลที่จะส่งไปยังปลายทาง (Secrecy) โดยผู้

ส่งจะเข้ารหัสข้อความที่จะส่งด้วยด้วยพับลิคคีย์ (Public key) ของปลายทางและเมื่อปลายทางได้รับข้อความนั้นก็จะถอดรหัสข้อมูลนั้นด้วยไพรเวตคีย์ (Private key) ของตนเอง รูปแบบสมการทั่วไปของการเข้ารหัสและถอดรหัสข้อมูลด้วยวิธีการนี้แสดงอยู่ในสมการที่ 3.1 และ 3.2

$$c = m^e \pmod{n} \tag{3.1}$$

$$m = c^d \pmod{n} \tag{3.2}$$

โดยที่ m คือข้อความต้นฉบับ, c คือข้อความที่ได้จากการเข้ารหัส, e คือพับลิคคีย์ของฝั่งปลายทาง, d คือไพรเวตคีย์ของฝั่งปลายทางและ n คือตัวเลขที่นำมาใช้ในการมอดูโล (Modulo)

สำหรับรายละเอียดของการกำหนดค่าพารามิเตอร์ต่างๆที่ใช้ในการเข้ารหัสและถอดรหัส (e, d และ n) สามารถดูได้จาก [10] โดยแต่ละค่าจะคำนวณมาจากเลขจำนวนเฉพาะ 2 จำนวน (p และ q) ที่มีค่ามาก สาเหตุที่เลขจำนวนเฉพาะสองค่านี้ควรจะเป็นตัวเลขที่มีค่ามากก็เนื่องจากจะสามารถช่วยป้องกันการปลอมแปลงคีย์ที่ใช้ในการเข้ารหัสและถอดรหัสได้เพราะจะมีความเป็นไปได้น้อยมากที่จะสามารถตรวจพบเลขจำนวนเฉพาะที่มีค่ามากได้ตรงกันทั้งสองจำนวน ข้อเสียของการใช้เลขจำนวนเฉพาะที่มีค่ามากคือจะทำให้มีการคำนวนที่ซับซ้อน ดังนั้นเพื่อความสะดวกในการคำนวณหาค่าคีย์ต่างๆในงานวิจัยนี้จึงกำหนดให้ค่า p และ q เป็นเลขจำนวนเฉพาะที่มีค่าน้อยๆ (กำหนดให้ $p = 29$ และ $q = 31$ ส่วนค่าคีย์อื่นๆที่คำนวณได้คือ $e = 517$, $n = 899$ และ $d = 13$) เงื่อนไขที่สำคัญอย่างหนึ่งการเข้ารหัสด้วยวิธีการนี้คือ ค่า m ที่จะนำมาใช้ในการเข้ารหัสนั้นจะต้องมีค่าน้อยกว่าค่า n เสมอ

3.3.2.2 การสร้างลายเซ็นดิจิตอลโดยใช้ RSA Public-key encryption

การนำวิธีเข้ารหัสแบบ RSA Public-key encryption มาประยุกต์ใช้ในการสร้างลายเซ็นดิจิตอลนั้นจะต้องมีการปรับเปลี่ยนรูปแบบของการเข้ารหัสและถอดรหัสเสียก่อน โดยในขั้นตอนของการเข้ารหัสนั้นทางฝั่งต้นทางจะเข้ารหัสข้อความที่จะส่งด้วยไพรเวตคีย์ของตนเองและเมื่อปลายทางได้รับข้อความนั้นก็จะถอดรหัสข้อความนั้นด้วยพับลิคคีย์ของฝั่งต้นทางซึ่งมีรูปแบบสมการของการเข้ารหัสและถอดรหัสข้อมูลสำหรับการสร้างลายเซ็นดิจิตอลดังนี้

$$c = m^d \pmod{n} \tag{3.3}$$

$$m = c^e \pmod{n} \tag{3.4}$$

โดยที่ m คือข้อความต้นฉบับ, c คือข้อความที่ได้จากการเข้ารหัส, e คือพับลิคคีย์ของฝั่งต้นทาง, d คือไพรเวตคีย์ของฝั่งต้นทางและ n คือตัวเลขที่นำมาใช้ในการมอดูโล (Modulo)

3.3.2.3 ตัวอย่างการสร้างลายเซ็นดิจิตอล

กำหนดข้อมูลที่ต้องการสร้างลายเซ็นดิจิตอล (m) คือ 962461 โดยมีค่าพารามิเตอร์ต่างๆที่ใช้ในการสร้างลายเซ็นดิจิตอลดังนี้ $p = 29,\ q = 31,\ e = 517,\ n = 899,\ d = 13$

1) การเข้ารหัสข้อมูล

จากที่กล่าวมาแล้วว่าค่า m ที่จะนำมาเข้ารหัสข้อมูลนั้นจะต้องมีค่าน้อยกว่าค่า n เสมอ ดังนั้นหาก m มีค่ามากกว่าก็จะถูกแบ่งออกเป็นบล็อกๆ โดยที่แต่ละบล็อกจะต้องมีค่าน้อยกว่า n ซึ่งในตัวอย่างนี้ค่า m คือ 962461 (มีค่ามากกว่า 899) จะถูกแบ่งออกเป็น 3 บล็อก (โดยที่แต่ละบล็อกจะเป็นตัวเลขจำนวนนับ 2 ตำแหน่ง) คือ $m_1 = 96,\ m_2 = 24,\ m_3 = 61$ จากนั้นนำค่า m แต่ละบล็อกไปเข้ารหัสตามสมการที่ 3.3 จะได้ผลลัพธ์ดังนี้

- กรณี $m_1 = 96$

  จะได้ $c = 96^{13} \pmod{899} = 303$

- กรณี $m_2 = 24$

  จะได้ $c = 24^{13} \pmod{899} = 632$

- กรณี $m_3 = 61$

  จะได้ $c = 61^{13} \pmod{899} = 309$

โดยค่า c ที่ได้จากการเข้ารหัสแต่ละตัวนั้นจะมีจำนวนตำแหน่งเท่ากับจำนวนตำแหน่งของค่า n ดังนั้นผลของการเข้ารหัสข้อมูล 962461 คือ 303632309

2) การถอดรหัสข้อมูล

จากที่กล่าวมาแล้วว่าจำนวนตำแหน่งข้อมูลที่เข้ารหัสแล้วจะมีค่าเท่ากับจำนวนตำแหน่งของค่า n (3 ตำแหน่ง) ดังนั้นในขั้นตอนของการถอดรหัสจึงต้องแบ่งค่า $c = 303632309$ ออกเป็นบล็อกๆให้มีขนาดเท่ากับ 3 ตำแหน่ง ซึ่งจะได้ข้อมูลในแต่ละบล็อกดังนี้ $c_1 = 303,\ c_2 = 632,\ c_3 = 309$ จากนั้นนำไปถอดรหัสตามสมการที่ 3.4 ซึ่งได้ผลลัพธ์ดังนี้

- กรณี $c_1 = 303$

  จะได้ $m = 303^{517} \pmod{899} = 96$

- กรณี $c_2 = 632$

  จะได้ $m = 632^{517} \pmod{899} = 24$

- กรณี $c_3 = 309$

  จะได้ $m = 309^{517} \pmod{899} = 61$

ดังนั้นผลของการถอดรหัสข้อมูล 303632309 คือ 962461

3.3.3 การกำหนดจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูล

ในส่วนนี้จะกล่าวถึงการกำหนดจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต (R_d) ซึ่งปัจจัยที่ใช้กำหนดค่า R_d นี้จะพิจารณาจากจำนวนตำแหน่งข้อมูลที่เกิดความผิดพลาด (ภายในหนึ่งบรรทัด) และลักษณะของความผิดพลาดที่เกิดขึ้นว่ามีรูปแบบอย่างไร เมื่อกำหนดค่า R_d ได้แล้วก็จะเพิ่มข้อมูลซ้ำต่อท้ายบิตข้อมูลในแต่ละบิต (Redundant data bit) โดยจำนวนบิตข้อมูลที่ถูกเพิ่มขึ้นนี้จะมีค่าเท่ากับ $R_d - 1$ ตำแหน่ง โดยมีรูปแบบสมการที่ใช้กำหนดค่าบิตข้อมูล (หลังการเพิ่มข้อมูลซ้ำ) ที่จะนำมาซ่อนลงในเอกสารคือ

$$I'_k = I_i \tag{3.5}$$

โดยที่ I = บิตข้อมูลที่ต้องการซ่อน

I' = บิตข้อมูลที่จะใช้ทำการซ่อนจริง (หลังการเพิ่มข้อมูลซ้ำ)

$k = R_d(i-1) + j$

i = ลำดับที่ของบิตข้อมูลที่ต้องการซ่อน $(1,2,3,...,N)$

$j = 1,2,3,...,R_d$

ในงานวิจัยนี้ได้ทำการทดลองเบื้องต้นเกี่ยวกับการหาตรวจหาความผิดพลาดของข้อมูลที่ซ่อนอยู่ภายในเอกสารโดยได้ทดลองซ่อนข้อมูลลงในเอกสารภาษาไทยขนาด A4 ซึ่งใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดตัวอักษรแบบชิดขอบคอลัมน์เดียวจำนวน 5 หน้าโดยแต่ละหน้าจะมีจำนวนบรรทัดที่ใช้ในการซ่อนข้อมูลประมาณ 36 บรรทัดและในแต่ละบรรทัดจะมีจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ประมาณ 13 ตำแหน่ง ในการทดลองนี้จะนำเอาผลการดึงข้อมูลออกจากเอกสารที่ผ่านการถ่ายเอกสาร (เอกสารสำเนาที่ 1, เอกสารสำเนาที่ 2 และเอกสารสำเนาที่ 3) มาใช้เป็นข้อมูลในการกำหนดค่าจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต (ดูผลการทดลองในตารางที่ 3.1 และ 3.2)

ตารางที่ 3.1 แสดงผลความผิดพลาดของข้อมูลที่เกิดขึ้นจากการดึงข้อมูล

หน้าที่	จำนวนบิตทั้งหมด	สำเนาที่ 1		สำเนาที่ 2		สำเนาที่ 3	
		จำนวนบิตที่ผิดพลาด	จำนวนบิตที่สูญหาย	จำนวนบิตที่ผิดพลาด	จำนวนบิตที่สูญหาย	จำนวนบิตที่ผิดพลาด	จำนวนบิตที่สูญหาย
1	471	3	6	6	4	7	5
2	517	2	2	7	2	14	2
3	542	1	6	2	10	15	10
4	465	1	4	2	4	7	7
5	473	5	6	6	7	7	9
รวม	2,468	12	24	23	27	50	33
เปอร์เซนต์		0.49	0.97	0.93	1.09	2.03	1.34

ตารางที่ 3.2 แสดงค่าเฉลี่ยของความผิดพลาดที่เกิดขึ้นในแต่ละบรรทัด

ลักษณะความผิดพลาดที่เกิดขึ้นในแต่ละบรรทัด	เปอร์เซ็นต์ของจำนวนตำแหน่งข้อมูลที่สูญหายในแต่ละบรรทัด (%)		
	สำเนาที่ 1	สำเนาที่ 2	สำเนาที่ 3
ไม่มีตำแหน่งข้อมูลที่สูญหาย	89.53	54.65	32.56
มีตำแหน่งข้อมูลที่สูญหาย 1 ตำแหน่ง	12.79	15.12	19.77
มีตำแหน่งข้อมูลที่สูญหาย 2 ตำแหน่ง	0	2.32	10.47
มีตำแหน่งข้อมูลที่สูญหายมากกว่า 2 ตำแหน่ง	0	0	0

จากตารางที่ 3.1 จะเห็นว่าปริมาณความผิดพลาดของข้อมูลที่สามารถตรวจพบได้จากการดึงข้อมูลออกจากเอกสารนั้นจะมีค่ามากขึ้นตามจำนวนครั้งของการทำสำเนาเอกสาร เนื่องจากการทำสำเนาเอกสารนั้นจะทำให้เอกสารบิดเบือนไปจากเดิมซึ่งเป็นผลให้ข้อมูลที่ซ่อนอยู่ภายในเอกสารเกิดความผิดพลาดได้ จากผลการทดลองนี้เราสามารถแบ่งความผิดพลาดของข้อมูลที่เกิดขึ้นได้ 2 ลักษณะคือ ความผิดพลาดที่เกิดจากการระบุค่าบิตข้อมูลผิดพลาด (เช่น บิตข้อมูลเปลี่ยนจาก "0" เป็น "1" หรือเปลี่ยนจาก "1" เป็น "0") และความผิดพลาดที่ไม่สามารถระบุค่าบิตข้อมูลที่ตำแหน่งนั้นได้ (บิตข้อมูลสูญหาย) ซึ่งความผิดพลาดในลักษณะแรกนั้นเป็นความผิดพลาดที่สามารถแก้ไขได้โดยการใช้วิธีการตรวจสอบและแก้ไขความผิดพลาดของข้อมูล (Error correction) (จะกล่าวถึงรายละเอียดในหัวข้อที่ 3.3.5) แต่สำหรับความผิดพลาดในลักษณะที่สองนี้จะเป็นความผิดพลาดที่ส่งผลกระทบต่อบิตข้อมูลอื่นๆที่อยู่ใกล้เคียงกับบิตข้อมูลที่สูญหายทำให้เกิดความผิดพลาดแบบต่อเนื่อง ซึ่งวิธีการแก้ปัญหาอย่างหนึ่งที่สามารถลดความผิดพลาดของข้อมูลในลักษณะนี้ได้คือการเพิ่มจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต เนื่องจากจะช่วย

ให้การระบุค่าบิตข้อมูลมีความถูกต้องได้ถึงแม้ว่าจะมีค่าสัญลักษณ์ข้อมูลบางตำแหน่งสูญหายไปก็ตามเพราะการระบุค่าบิตข้อมูลด้วยวิธีการนี้จะคำนวณจากค่าเฉลี่ยของค่าสัญลักษณ์ข้อมูลทั้งหมดที่แทนบิตข้อมูลเดียวกัน

สำหรับการกำหนดจำนวนข้อมูลที่เหมาะสมที่จะใช้แทนค่าบิตข้อมูลหนึ่งบิตนั้นจะพิจารณาจากข้อมูลในตารางที่ 3.2 จะเห็นว่าส่วนใหญ่แล้วค่าสัญลักษณ์ข้อมูลในแต่ละบรรทัดนั้นจะสูญหายอยู่ในช่วง 1-2 ตำแหน่ง (ในงานวิจัยนี้จำกัดขอบเขตอยู่ที่เอกสารที่ผ่านการถ่ายเอกสารไม่เกินสำเนาที่ 3) ดังนั้นในงานวิจัยนี้จึงกำหนดให้ซ่อนข้อมูลลงในเอกสารเป็นจำนวน 4 ตำแหน่งเพื่อแทนค่าบิตข้อมูลหนึ่งบิต ($R_d = 4$) ซึ่งค่านี้จะช่วยป้องกันการเกิดความผิดพลาดแบบต่อเนื่องได้และช่วยให้การระบุค่าบิตข้อมูลมีความถูกต้องมากขึ้นถึงแม้ว่าจะมีค่าสัญลักษณ์ข้อมูลที่แทนบิตข้อมูลเดียวกันสูญหายไปถึง 2 ตำแหน่งก็ตาม

<u>หมายเหตุ</u> เราอาจจะกำหนดให้ค่า R_d มีค่ามากกว่าหรือน้อยกว่า 4 ก็ได้ (ควรกำหนดให้ค่า R_d มากกว่า 2 เนื่องจากผลการทดลองพบว่าในแต่ละบรรทัดจะมีข้อมูลสูญหายเกินกว่า 1 ตำแหน่ง) แต่ถ้าเรากำหนดให้ค่า R_d มีค่าน้อยไปก็จะทำให้การระบุค่าข้อมูลอาจเกิดความคลาดเคลื่อนได้ เช่น กำหนดให้ $R_d = 3$ ซึ่งก็หมายความว่าจะใช้ข้อมูล 3 ตำแหน่งในการแทนค่าบิตข้อมูล 1 บิต ดังนั้นถ้าหากมีค่าสัญลักษณ์ข้อมูลที่ใช้แทนค่าบิตข้อมูลเดียวกันสูญหายไป 2 ตำแหน่งก็จะเหลือค่าสัญลักษณ์ข้อมูลเพียงตำแหน่งเดียวเท่านั้นที่จะใช้ระบุค่าบิตข้อมูลได้ ซึ่งถ้าค่านี้เกิดความผิดพลาดไปจากเดิมก็จะทำให้การระบุค่าบิตข้อมูลที่ตำแหน่งนั้นเกิดความผิดพลาดไปด้วย แต่ถ้าเรากำหนดให้ R_d มีค่ามากเกินไปก็จะเป็นการสิ้นเปลืองตำแหน่งที่ใช้ซ่อนข้อมูลมากเกินไปทำให้สามารถซ่อนข้อมูลได้น้อยลง

ตัวอย่างของการเพิ่มข้อมูลบิตซ้ำเพื่อนำไปใช้กำหนดค่าสัญลักษณ์ข้อมูลที่จะซ่อนลงในเอกสารคือ กำหนดให้บิตข้อมูลที่ต้องการซ่อน (I) คือ "1010" และกำหนดให้ R_d มีค่าเท่ากับ 4 (จำนวนข้อมูลที่ต้องเพิ่มต่อท้ายบิตข้อมูลเดิมคือ 3 ตำแหน่ง) จากสมการที่ 3.5 จะได้ชุดข้อมูลใหม่ที่จะนำไปใช้กำหนดค่าสัญลักษณ์ข้อมูลที่จะซ่อนลงในเอกสารดังสมการที่ 3.6

$$I' = \text{"}\underbrace{1111}_{i=1}\ \underbrace{0000}_{i=2}\ \underbrace{1111}_{i=3}\ \underbrace{0000}_{i=4} \tag{3.6}$$

3.3.4 การกำหนดจำนวนตำแหน่งที่ซ่อนข้อมูลในแต่ละบรรทัด

โดยทั่วไปแล้วข้อความในแต่ละบรรทัดจะมีจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้แตกต่างกันขึ้นอยู่กับลักษณะของข้อความและรูปแบบของตัวอักษรภายในแต่ละบรรทัด แต่วิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้จะซ่อนข้อมูลลงในแต่ละบรรทัดด้วยจำนวนที่เท่ากันคือ L_d ตำแหน่ง ดังนั้นจึงต้องมีการเลือกค่า L_d ที่เหมาะสมที่จะใช้ประโยชน์จากบรรทัดทุกบรรทัดภายในหน้า

เอกสารได้อย่างเต็มที่ โดยปัจจัยหลักที่ใช้กำหนดค่า L_d นั้นจะพิจารณาจากค่าเฉลี่ยของจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดภายในหน้าเอกสารทั้งหมด (M_d) โดยที่ค่า L_d ที่เลือกนี้จะต้องมีค่าน้อยกว่าหรือเท่ากับค่า M_d ($L_d \leq M_d$) เพื่อให้สามารถซ่อนข้อมูลได้ลงในแต่ละบรรทัดได้ และปัจจัยอีกอย่างที่จะนำมาใช้ในการกำหนดค่า L_d ร่วมด้วยก็คือค่า L_d ที่เลือกมานี้ควรจะสามารถจำกัดบริเวณของการเกิดความผิดพลาดของข้อมูลมิให้เป็นความผิดพลาดแบบต่อเนื่อง นั่นคือค่า L_d นี้ควรจะมีค่าเป็นจำนวนเท่าของจำนวนข้อมูลที่ใช้แทน ค่าบิตข้อมูลหนึ่งบิต (R_d) เพื่อจำกัดความผิดพลาดของข้อมูลให้อยู่ภายในแต่ละบรรทัดเท่านั้น ดังนั้นสมการที่ใช้กำหนดค่า L_d คือ

$$L_d \leq nR_d \leq M_d \tag{3.7}$$

โดยที่ L_d = จำนวนตำแหน่งที่จะซ่อนข้อมูลลงในแต่ละบรรทัด

M_d = ค่าเฉลี่ยของจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดภายในหน้าเอกสารทั้งหมด

R_d = จำนวนข้อมูลจะถูกซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต

n = 1, 2, 3, ...

จำนวนตำแหน่งที่จะซ่อนข้อมูลลงในแต่ละบรรทัด (L_d) ภายในเอกสารนั้นจะมีค่าเป็นเท่าใดก็ได้ตามแต่ลักษณะข้อความภายในเอกสารแต่ละฉบับ สำหรับเอกสารภาษาไทยที่ใช้ซ่อนข้อมูลในงานวิจัยนี้ (เอกสารขนาด A4 ซึ่งใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดแบบชิดขอบคอลัมน์เดียว) จะกำหนดให้ซ่อนข้อมูลเพียงบรรทัดละ 8 ตำแหน่งเท่านั้น ซึ่งจากการสำรวจพบว่าโดยเฉลี่ยแล้วในแต่ละบรรทัดของเอกสารนี้จะสามารถซ่อนข้อมูลได้ประมาณ 13 ตำแหน่ง ($M_d = 13$) และกำหนดให้จำนวนข้อมูลที่จะใช้แทนค่าบิตข้อมูลหนึ่งบิตคือ 4 ตำแหน่ง ($R_d = 4$) (สามารถดูรายละเอียดการกำหนดค่า R_d ได้จากหัวข้อ 3.3.3) จากนั้นนำค่า M_d และ R_d ไปแทนในสมการ 3.7 ซึ่งจะได้ค่า L_d ที่เป็นไปได้ทั้งหมดคือ 4, 8 และ 12 ตำแหน่ง แต่สาเหตุที่งานวิจัยนี้เลือกใช้ค่า $L_d = 8$ ก็เนื่องจากว่าหากเราเลือกที่จะซ่อนข้อมูลบรรทัดละ 4 ตำแหน่ง ($L_d = 4$) จะทำให้บิตข้อมูลที่ถูกซ่อนในบรรทัดหนึ่งๆจะมีเพียงแค่ 1 บิตเท่านั้น (เนื่องจากการซ่อนข้อมูลนี้ต้องใช้ 4 ตำแหน่งในการซ่อนข้อมูล 1 บิต) ซึ่งอาจทำให้มีตำแหน่งที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดเหลือโดยที่ไม่ถูกใช้ประโยชน์ (บางบรรทัดมีตำแหน่งที่สามารถซ่อนข้อมูลได้มากกว่า 4 ตำแหน่ง) แต่ถ้าเราเลือกที่จะซ่อนข้อมูลบรรทัดละ 12 ตำแหน่ง ($L_d = 12$) ก็อาจทำให้มีบางบรรทัดไม่สามารถซ่อนข้อมูลได้ (มีจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้น้อยกว่า 12 ตำแหน่ง) ซึ่งจะทำให้สูญเสียตำแหน่งที่สามารถซ่อนข้อมูลได้ในบรรทัดนั้นๆไปโดยเปล่า

ประโยชน์ทั้งๆที่ในบรรทัดนั้นอาจมีจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้น้อยกว่า 12 บิตเพียง 1 หรือ 2 บิตเท่านั้น ดังนั้นเพื่อที่จะใช้ประโยชน์จากบรรทัดทุกบรรทัดภายในเอกสารให้มีประสิทธิภาพมากที่สุดจึงเลือกที่จะซ่อนข้อมูลลงบรรทัดละ 8 ตำแหน่ง

3.3.5 ชุดข้อมูลรหัสแฮมมิง (7,4)

วิธีการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลที่นำมาใช้ในงานวิจัยนี้คือวิธีการเข้ารหัสแฮมมิง (7,4) [13] ซึ่งมีหลักการดังนี้

3.3.5.1 การสร้างชุดรหัสแฮมมิง (7,4)

สมการทั่วไปของการสร้างชุดรหัสแฮมมิง (n,k) คือ

$$t = d \cdot G \tag{3.8}$$

โดยที่ t คือชุดรหัสแฮมมิง, d คือข้อมูล และ G คือเมตริกซ์ของตัวกระทำ (Generator matrix) ที่ถูกกำหนดมาสำหรับสร้างชุดรหัสในเงื่อนไขต่างๆกัน (ข้อกำหนดของสมการนี้คือจำนวนคอลัมน์ใน d จะต้องเท่ากับจำนวนแถวของเมตริกซ์ G จึงจะสามารถทำการคูณเมตริกซ์กันได้)

ในการสร้างชุดรหัสแฮมมิง (7,4) จะกำหนดให้ G มีค่าดังสมการที่ 3.9 (รายละเอียดของการกำหนดค่า G สามารถดูได้จากเอกสารอ้างอิงเลขที่ [13]) ดังนั้นจะได้สมการที่ใช้สร้างชุดรหัสแฮมมิงดังสมการที่ 3.10

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.9}$$

$$t = d. \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.10}$$

3.3.5.2 การตรวจสอบความผิดพลาดของชุดรหัสแฮมมิง (7,4)

การตรวจสอบความผิดพลาดของชุดข้อมูลรหัสแฮมมิงสามารถกระทำได้โดยกำหนดให้ค่า H เป็นเมตริกซ์ที่ใช้ตรวจสอบพาริตี้ (Parity check matrix) มีค่าเท่ากับสมการที่ 3.11 (รายละเอียดของการกำหนดค่า H ดูในเอกสารอ้างอิงเลขที่ [13])

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{3.11}$$

จากนั้นนำชุดรหัสข้อมูลที่ได้รับมา (r) มาคูณกับเมตริกซ์ทรานสโพสของ H ซึ่งถ้าชุดรหัสข้อมูลที่ได้รับมามีความถูกต้องแล้ว ($r = t$) จะได้ผลลัพธ์เป็นศูนย์ดังสมการที่ 3.12

$$r \cdot H^T = 0 \tag{3.12}$$

แต่ถ้ามีความผิดพลาดเกิดขึ้นในชุดรหัสที่ได้รับมา ($r = t + e$; e คือความผิดพลาด) ผลที่ได้จะเป็นกรณีอื่นซึ่งจะใช้เป็นข้อมูลบอกตำแหน่งของความผิดพลาดที่เกิดขึ้น วิธีการตรวจสอบความผิดพลาดวิธีการนี้เรียกว่าการหาค่าความผิดพลาดซินโดรม (Syndrome : s) ซึ่งมีรูปแบบสมการดังนี้

$$s = r \cdot H^T \tag{3.13}$$

โดยผลของ s ที่ได้เมื่อนำมาเปรียบเทียบว่าตรงกับอันดับแถวที่เท่าใดของเมตริกซ์ H^T ก็สามารถบอกได้ว่าชุดรหัสข้อมูลที่ได้รับมามีบิตที่ผิดพลาดตรงกับอันดับแถวนั้น แต่ทั้งนี้การเข้ารหัสแฮมมิงยังมีข้อจำกัดที่ไม่สามารถตรวจสอบหรือบอกตำแหน่งข้อมูลที่เกิดความผิดพลาดได้ทุกกรณีถ้ามีความผิดพลาดเกิดขึ้นเป็นจำนวนมาก

3.3.5.3 ตัวอย่างการใช้งานชุดข้อมูลรหัสแฮมมิง (7,4)

กำหนดให้ข้อมูลที่จะนำมาสร้างชุดรหัสแฮมมิง (d) คือ [0 1 1 0] และชุดข้อมูลที่ผู้รับได้รับจากผู้ส่ง (r) คือ [0 1 0 0 1 1 0]

1) ตัวอย่างการสร้างชุดข้อมูลรหัสแฮมมิง (7,4) สำหรับข้อมูล $d = [0\ 1\ 1\ 0]$ จากสมการ 3.10 จะได้

$$t = [0 \quad 1 \quad 1 \quad 0] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$= [0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0] \qquad (3.14)$$

ดังนั้นชุดรหัสข้อมูลแฮมมิงที่จะส่งไปยังผู้รับคือ [0 1 1 0 1 1 0]

2) การตรวจสอบความผิดพลาดของข้อมูล $r = [0\ 1\ 0\ 0\ 1\ 1\ 0]$

จากสมการ 3.13 จะได้

$$s = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0] \cdot \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= [0 \quad 1 \quad 1] \qquad (3.15)$$

เมื่อนำค่าความผิดพลาดของซินโดรมที่ได้จากสมการที่ 3.15 ไปเปรียบกับเมตริกซ์ H' พบว่ามีค่าตรงกับแถวที่ 3 ของ H' ทำให้เราทราบว่าค่า r ในตำแหน่งที่ 3 เกิดความผิดพลาด ดังนั้นจึงสามารถแก้ไขข้อมูลที่ตำแหน่งนี้ได้

3.3.5.4 การประยุกต์ใช้ชุดข้อมูลรหัสแฮมมิง (7,4)

จากผลการทดลองเบื้องต้นของงานวิจัยนี้ (การทดลองเดียวกับหัวข้อที่ 3.3.3) แสดงให้เห็นว่าลักษณะการเกิดความผิดพลาดของข้อมูลภายในเอกสารนั้นมีรูปแบบที่ไม่แน่นอน (ข้อมูลเกิดความผิดพลาดกระจายอยู่ทั่วเอกสาร) ขึ้นอยู่กับลักษณะของข้อความในแต่ละหน้าของเอกสาร อีกทั้งยังอาจจะขึ้นอยู่กับเครื่องมือที่ใช้ประมวลผลเอกสารด้วย (เครื่องพิมพ์ เครื่องถ่ายเอกสาร หรือเครื่องสแกนเนอร์) ว่ามีประสิทธิภาพมากน้อยเพียงไร ในการทดลองนี้ได้ทำการเปรียบเทียบความผิดพลาดที่เกิดขึ้นในทิศทางของแนวนอนและแนวตั้งเพื่อหาความน่าจะเป็นของแนวโน้มของลักษณะการเกิดความผิดพลาดซึ่งกระทำโดยการแบ่งข้อมูลที่ซ่อนในเอกสารออกเป็นบล็อกๆทั้งในแนวนอน (ข้อมูลที่ซ่อนอยู่ภายในบรรทัดเดียวกัน) และแนวตั้ง (ข้อมูลที่ซ่อนอยู่ในตำแหน่งที่ตรง

กันในแนวตั้งของแต่ละบรรทัด) ด้วยจำนวนที่เท่ากัน (กำหนดให้ข้อมูลในแต่ละบล็อกมีจำนวนเท่ากับ 7 ตำแหน่งซึ่งมีค่าเท่ากับจำนวนข้อมูลในหนึ่งชุดข้อมูลรหัสแฮมมิง) ซึ่งจะได้ผลการเปรียบเทียบในตารางที่ 3.3 โดยหลักการที่ใช้เปรียบเทียบความผิดพลาดทั้งสองทิศทางนี้คือจะเปรียบเทียบว่าข้อมูลที่เกิดความผิดพลาดมากกว่าหนึ่งตำแหน่งในทิศทางใดมีจำนวนมากกว่ากันเพื่อใช้เป็นข้อมูลในการกำหนดทิศทางของการซ่อนชุดข้อมูลรหัสแฮมมิงลงในเอกสาร เนื่องจากชุดรหัสแฮมมิงนี้สามารถตรวจสอบและแก้ไขความผิดพลาดของข้อมูลได้เพียงหนึ่งตำแหน่งเท่านั้น ดังนั้นถ้าเราซ่อนข้อมูลในทิศทางที่มีโอกาสที่จะเกิดความผิดพลาดไม่มากกว่าหนึ่งตำแหน่งก็จะช่วยให้ชุดข้อมูลรหัสแฮมมิงนั้นสามารถตรวจสอบความถูกต้องของตัวเองได้อย่างมีประสิทธิภาพ

ตารางที่ 3.3 แสดงการเปรียบเทียบปริมาณการเกิดความผิดพลาดในทิศทางแนวนอนและแนวตั้งภายในเอกสาร

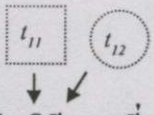
ประเภทเอกสาร	เปอร์เซ็นต์การเกิดความผิดพลาดของข้อมูล มากกว่า 1 ตำแหน่งภายใน 1 บล็อกข้อมูล (%)	
	บล็อกข้อมูลในแนวนอน	บล็อกข้อมูลในแนวตั้ง
สำเนาที่ 1	8.52	1.11
สำเนาที่ 2	11.62	8.89
สำเนาที่ 3	13.17	12.22

จากตารางที่ 3.3 จะเห็นว่าการซ่อนชุดรหัสข้อมูลแฮมมิงในแนวนอนนั้นจะเกิดความผิดพลาดมากกว่าการซ่อนชุดข้อมูลรหัสแฮมมิงในแนวตั้งของเอกสารทุกสำเนา ดังนั้นในงานวิจัยนี้จึงได้สร้างชุดข้อมูลรหัสแฮมมิงให้อยู่ในแนวตั้งเพื่อที่จะนำชุดข้อมูลเหล่านั้นมาซ่อนลงในเอกสาร โดยมีเงื่อนไขในการซ่อนข้อมูลคือจะทำการซ่อนชุดข้อมูลรหัสแฮมมิงลงในตำแหน่งที่ตรงกันในแนวตั้งของแต่ละบรรทัด (ดูตัวอย่างในรูปที่ 3.7) โดยข้อมูลที่ถูกซ่อนในตัวอย่างนี้แสดงอยู่ในสมการที่ 3.16 ซึ่งประกอบด้วยชุดข้อมูลรหัสแฮมมิง 2 ชุด (ในสมการที่ 3.17 และ 3.18)

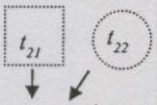
$$t = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \\ t_{31} & t_{32} \\ t_{41} & t_{42} \\ t_{51} & t_{52} \\ t_{61} & t_{62} \\ t_{71} & t_{72} \end{bmatrix} \qquad (3.16)$$

$$t_1 = \begin{bmatrix} t_{11} \\ t_{21} \\ t_{31} \\ t_{41} \\ t_{51} \\ t_{61} \\ t_{71} \end{bmatrix} \tag{3.17}$$

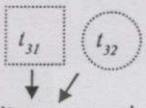
$$t_2 = \begin{bmatrix} t_{12} \\ t_{22} \\ t_{32} \\ t_{42} \\ t_{52} \\ t_{62} \\ t_{72} \end{bmatrix} \tag{3.18}$$



เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการ

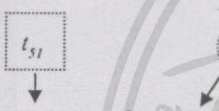


ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทย

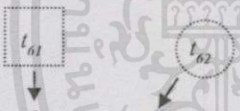


ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความ

เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ใน



ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่



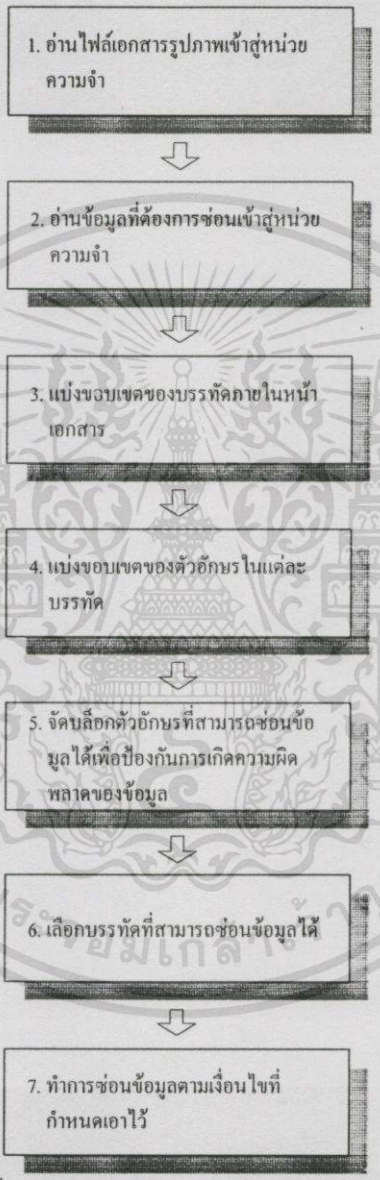
เอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในก

โดยในบทความนี้จะมุ่งเน้นที่การซ่อนข้อมูลลงในสื่อประเภทข้อความเนื่องจา

รูปที่ 3.7 ตัวอย่างการซ่อนชุดข้อมูลรหัสแฮมมิงในแนวตั้ง (ข้อมูลในสี่เหลี่ยมเป็นข้อมูลรหัสแฮมมิงชุดที่ 1 และข้อมูลในวงกลมคือข้อมูลรหัสแฮมมิงชุดที่ 2)

3.4 อัลกอริทึมของการซ่อนข้อมูล

ในส่วนนี้จะกล่าวถึงรายละเอียดของอัลกอริทึมในการซ่อนข้อมูล (ดูรูปที่ 3.8) โดยกำหนดให้เอกสารที่จะนำมาซ่อนข้อมูลเป็นเอกสารรูปภาพแบบบิตแมปหรือเอกสารอื่นที่ถูกแปลงให้อยู่ในรูปแบบของบิตแมปแล้ว



รูปที่ 3.8 แสดงรายละเอียดของขั้นตอนการซ่อนข้อมูล

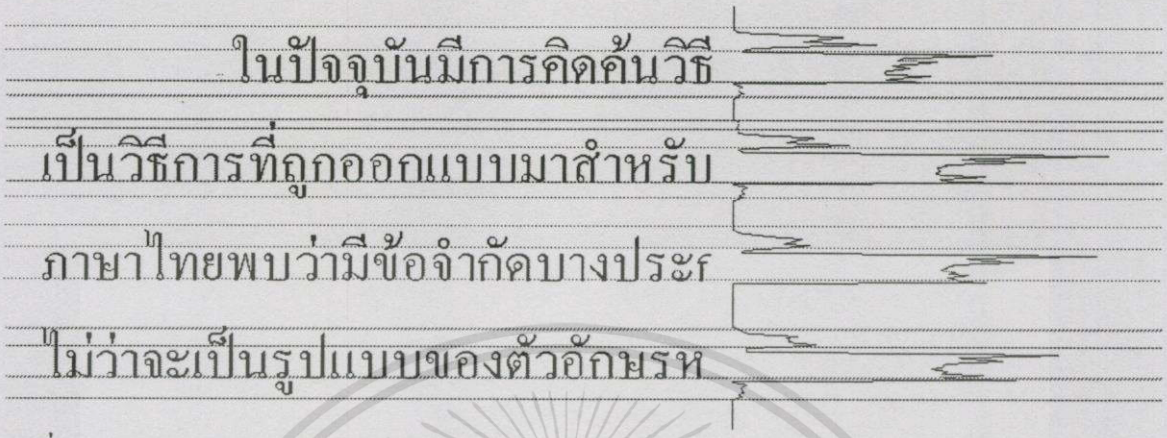
3.4.1 ขั้นตอนการซ่อนข้อมูล

จากรูปที่ 3.8 มีรายละเอียดของขั้นตอนการซ่อนข้อมูลดังนี้

1) อ่านไฟล์เอกสารรูปภาพที่ต้องการซ่อนข้อมูลเข้ามาในหน่วยความจำ

2) อ่านข้อมูลที่ต้องการซ่อนเข้าสู่หน่วยความจำ

3) แบ่งขอบเขตของบรรทัดภายในหน้าเอกสารโดยการสร้างฮิสโตแกรมในแนวนอนของหน้าเอกสารนั้น (ดูตัวอย่างในรูปที่ 3.9)



รูปที่ 3.9 แสดงตัวอย่างฮิสโตแกรมในแนวนอนของข้อความจำนวน 4 บรรทัด

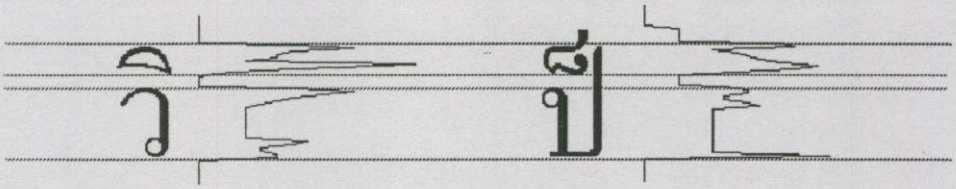
4) แบ่งขอบเขตของตัวอักษรในแต่ละบรรทัดออกเป็นบล็อกๆโดยมีขั้นตอนการแบ่งดังนี้

4.1) สร้างฮิสโตแกรมในแนวตั้งของแต่ละบรรทัดที่ได้จากขั้นตอนที่ 3 (ดังตัวอย่างในรูปที่ 3.10)

รูปที่ 3.10 แสดงตัวอย่างฮิสโตแกรมในแนวตั้งของข้อความบางส่วนในบรรทัดที่สองของรูปที่ 3.9

4.2) จากนั้นสร้างฮิสโตแกรมในแนวนอนของแต่ละบล็อกตัวอักษรเพื่อตรวจหาตำแหน่งที่สามารถซ่อนข้อมูลได้โดยพิจารณาจากบล็อกที่มีช่องว่างระหว่างระดับชั้นที่สองกับสามเท่านั้น โดยบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้จะถูกแบ่งออกเป็น 2 ส่วนคือส่วนของตัว

อักษรที่อยู่ในระดับบนและส่วนของตัวอักษรที่อยู่ในระดับล่าง พิจารณารูปที่ 3.11 แสดงตัวอย่างของบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้และบล็อกตัวอักษรที่ไม่สามารถซ่อนข้อมูลได้



(ก) (ข)

รูปที่ 3.11 แสดงตัวอย่างฮิสโตแกรมในแนวนอนของบล็อกตัวอักษร (ก) บล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ (ข) บล็อกตัวอักษรที่ไม่สามารถซ่อนข้อมูลได้

4.3) สร้างฮิสโตแกรมในแนวตั้งของบล็อกตัวอักษรที่อยู่ระดับบนเพื่อตรวจหาว่าภายในบล็อกนั้นจะมีตำแหน่งที่สามารถซ่อนข้อมูลได้มากกว่า 1 ตำแหน่งหรือไม่

5) นำบล็อกตัวอักษรที่ได้จากขั้นตอนที่แล้วมาผ่านกระบวนการป้องกันการเกิดความผิดพลาดของข้อมูล (Error prevention) ซึ่งจะทำการตรวจสอบบล็อกตัวอักษรต่างๆว่าอยู่ในตำแหน่งที่มีโอกาสจะเกิดความผิดพลาดหรือไม่ โดยถ้าพบว่าบล็อกใดมีโอกาสที่จะเกิดการความผิดพลาดได้ง่ายก็จะทำการเปลี่ยนแปลงตำแหน่งบล็อกตัวอักษรนั้นให้อยู่ในตำแหน่งเหมาะสม ซึ่งจะแสดงรายละเอียดการทำงานของกระบวนการนี้ในหัวข้อ 3.4.2

6) จากนั้นเลือกบรรทัดที่จะนำมาซ่อนข้อมูลโดยพิจารณาจากจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ในแต่ละบรรทัดซึ่งจะต้องมากกว่าหรือเท่ากับ L_d ตำแหน่ง

7) ทำการซ่อนข้อมูลลงในแต่ละบรรทัดภายในเอกสารโดยมีข้อกำหนดในการซ่อนข้อมูลดังนี้

7.1) ซ่อนข้อมูล "0" สลับกับข้อมูล "1" ลงใน 2 บรรทัดแรกที่สามารถซ่อนข้อมูลได้ในทุกตำแหน่งเพื่อใช้เป็นข้อมูลในการกำหนดค่าเธรดโชลด์ (δ_d) ที่ใช้ระบุค่าบิตข้อมูล

7.2) จากนั้นซ่อนข้อมูลที่ได้มาจากขั้นตอนที่สองลงในบรรทัดถัดๆไปที่สามารถซ่อนข้อมูลได้ โดยถ้าบรรทัดใดมีตำแหน่งที่สามารถซ่อนข้อมูลได้มากกว่า L_d ตำแหน่งก็จะซ่อนข้อมูลตามตำแหน่งสุดท้ายลงในตำแหน่งที่เหลือ

7.3) สำหรับบรรทัดที่ไม่สามารถซ่อนข้อมูล (มีตำแหน่งที่สามารถซ่อนข้อมูลได้น้อยกว่า L_d ตำแหน่ง) หรือบรรทัดที่เหลือจากการซ่อนข้อมูลนั้นจะถูกซ่อนข้อมูล "0" สลับกับข้อมูล "1" ทั้งบรรทัดเพื่อใช้เป็นข้อมูลในการกำหนดค่าเธรดโชลด์ L_d ที่ใช้ระบุค่าบิตข้อมูล

7.4) เงื่อนไขที่ใช้กำหนดค่าสัญลักษณ์ข้อมูล (ขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสาม) เพื่อใช้แทนค่าข้อมูลที่จะซ่อนลงในเอกสารโดยจะแสดงรายละเอียดในหัวข้อที่ 3.4.3

3.4.2 การจัดตำแหน่งบล็อกตัวอักษรเพื่อป้องกันการเกิดความผิดพลาด

การจัดบล็อกตัวอักษรให้อยู่ในตำแหน่งที่เหมาะสมนั้นสามารถช่วยป้องกันการเกิดความผิดพลาดของข้อมูลได้ วิธีการนี้จะกระทำโดยการพิจารณาจากขนาดของช่องว่างระหว่างบล็อกตัวอักษรที่อยู่ใกล้เคียงกัน (S) ว่ามีความกว้างมากน้อยเพียงไรเพื่อที่จะจัดการกับบล็อกตัวอักษรเหล่านั้นซึ่งสามารถแยกพิจารณาได้เป็น 3 กรณีคือ

1) ถ้าขนาดของช่องว่างระหว่างบล็อกตัวอักษรมีขนาดน้อยกว่าค่าต่ำสุดของช่องว่างระหว่างบล็อกตัวอักษรที่กำหนดเอาไว้ ($S \le S_{min}$) ก็จะเลื่อนตำแหน่งบล็อกตัวอักษรทั้งสองบล็อกให้มาติดกันเพื่อรวมให้เป็นบล็อกเดียวกัน (ช่องว่างมีขนาดเล็กมากจึงควรจะรวมเป็นบล็อกเดียวกัน)

2) ถ้าขนาดของช่องว่างระหว่างบล็อกตัวอักษรมีขนาดมากกว่าค่า S_{min} แต่น้อยกว่า S_{max} ($S_{min} < S < S_{max}$) แสดงว่าบล็อกตัวอักษรทั้งสองบล็อกนี้มีโอกาสที่จะชิดติดกันได้เมื่อเอกสารผ่านการประมวลผลทางด้านเอกสาร ดังนั้นเพื่อเป็นการป้องกันมิให้บล็อกตัวอักษรทั้งสองบล็อกนี้มีโอกาสชิดติดกันก็จะเลื่อนตำแหน่งบล็อกตัวอักษรทั้งสองบล็อกนี้ให้ห่างกันเท่ากับ S_{max} เพื่อทำให้บล็อกตัวอักษรทั้งสองบล็อกแยกออกจากกันอย่างชัดเจน (ช่องว่างมีขนาดพอสมควรจึงควรจะแยกบล็อกตัวอักษรออกจากกันมากกว่าที่จะทำให้มันชิดติดกันเพราะอาจทำให้เอกสารเกิดความผิดปกติได้)

3) ถ้าขนาดของช่องว่างระหว่างบล็อกตัวอักษรมีขนาดมากกว่าหรือเท่ากับ S_{max} ที่กำหนดเอาไว้ แสดงว่าบล็อกตัวอักษรทั้งสองบล็อกนี้มีความกว้างมากพอสมควรที่จะไม่มีโอกาสมาชิดติดกันได้ดังนั้นจึงไม่ต้องทำการเปลี่ยนแปลงตำแหน่งบล็อกตัวอักษรใดๆเลย

<u>หมายเหตุ</u> ค่า S_{min} และ S_{max} คือตัวแปรที่ใช้กำหนดตำแหน่งบล็อกตัวอักษรซึ่งถูกกำหนดขึ้นมาจากการพิจารณาลักษณะของเอกสารที่จะซ่อนข้อมูล (เช่น รูปแบบของตัวอักษรที่ใช้ในเอกสารหรือค่าความละเอียดของเอกสาร) ว่ามีลักษณะอย่างไรและเมื่อเอกสารผ่านการประมวลผลทางด้านเอกสารจะมีการเปลี่ยนแปลงไปอย่างไร

ในงานวิจัยนี้ได้ทำการทดลองหาค่า S_{min} และ S_{max} ที่เหมาะสมกับเอกสารที่จะใช้ซ่อนข้อมูลโดยจะพิจารณาจากเอกสารที่ผ่านการประมวลผลทางด้านเอกสารว่าเกิดการเปลี่ยนแปลงอย่างไรบ้าง (เอกสารที่ใช้ในการทดลองคือเอกสารขนาด A4 ที่ใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์ซึ่งมีการจัดเก็บแบบชิดขอบคอลัมน์เดียวโดยมีค่าความละเอียดของเอกสารเท่ากับ

300 จุดต่อนิ้ว) หลังจากที่ซ่อนข้อมูลลงในเอกสารเรียบร้อยแล้วก็จะพิมพ์เอกสารเหล่านั้นด้วยเครื่องพิมพ์ซึ่งกำหนดค่าความละเอียดของการพิมพ์เท่ากับ 600 จุดต่อนิ้ว และนำเอกสารนั้นไปทำสำเนาโดยการถ่ายเอกสารซ้ำ 3 ครั้งเพื่อให้ได้เอกสารสำเนาที่ 3 ซึ่งจะเป็นเอกสารที่เราจะใช้พิจารณาเพื่อหาค่า S_{min} และ S_{max} (คุณสมบัติของเครื่องพิมพ์และเครื่องถ่ายเอกสารที่ใช้ในการทดลองนี้สามารถดูรายละเอียดได้จากบทที่ 4 ในหัวข้อ 4.1.1)

ตารางที่ 3.3 แสดงความน่าจะเป็นที่บล็อกตัวอักษรที่อยู่ใกล้เคียงกันจะติดกัน (สำหรับเอกสารสำเนาที่ 3 ซึ่งมีค่าความละเอียดเท่ากับ 300 จุดต่อนิ้ว)

ขนาดของช่องว่างระหว่างบล็อกตัวอักษร (พิกเซล)	ความน่าจะเป็นที่บล็อกตัวอักษรที่อยู่ใกล้เคียงกันจะติดกัน (%)
1	98.89
2	67.65
3	32.35
4	1.32
มากกว่า 4	0

พิจารณาตารางที่ 3.3 ซึ่งแสดงผลความน่าจะเป็นที่บล็อกตัวอักษรที่อยู่ใกล้เคียงกันจะชิดติดกันเมื่อมีระยะห่างระหว่างบล็อกตัวอักษรที่แตกต่างกัน จะเห็นว่าช่องว่างระหว่างบล็อกตัวอักษรที่มีขนาดเท่ากับ 1 พิกเซลนั้นมีโอกาสสูงมาก (98.89%) ที่บล็อกตัวอักษรทั้ง 2 บล็อกจะชิดติดกัน เนื่องจากบล็อกตัวอักษรทั้งสองบล็อกนี้อยู่ใกล้กันมากเกินไป สำหรับบล็อกตัวอักษรที่มีระยะห่างระหว่างกันเท่ากับ 2 ถึง 3 พิกเซลนั้นก็มีโอกาสที่จะชิดติดกันได้เช่นกันแต่น้อยกว่ากรณีแรก แต่สำหรับบล็อกตัวอักษรที่มีระยะห่างระหว่างกันเท่ากับ 4 พิกเซลหรือมากกว่าจะมีโอกาสน้อยมากที่บล็อกตัวอักษรเหล่านั้นจะชิดติดกัน ดังนั้นในงานวิจัยนี้จึงกำหนดให้ S_{min} มีค่าเท่ากับ 1 พิกเซล และ S_{max} มีค่าเท่ากับ 4 พิกเซล

สำหรับตัวอย่างของการจัดตำแหน่งบล็อกตัวอักษรที่อยู่ใกล้เคียงกันในงานวิจัยนี้จะแบ่งลักษณะของบล็อกตัวอักษรที่อยู่ใกล้เคียงกันออกเป็น 2 ลักษณะ โดยลักษณะแรกคือบล็อกตัวอักษรทั้งสองบล็อกเป็นบล็อกที่สามารถซ่อนข้อมูลได้ทั้งคู่ และลักษณะที่สองคือมีบล็อกตัวอักษรบล็อกใดบล็อกหนึ่งที่ไม่สามารถซ่อนข้อมูลได้ โดยมีรายละเอียดในการจัดตำแหน่งบล็อกตัวอักษรทั้งสองลักษณะดังนี้

1) บล็อกตัวอักษรที่อยู่ใกล้เคียงกันทั้งสองบล็อกเป็นบล็อกที่สามารถซ่อนข้อมูลได้ (บล็อกตัวอักษรที่มีช่องว่างระหว่างระดับชั้นสองกับสาม) มีข้อกำหนดในการจัดตำแหน่งบล็อกตัวอักษรดังนี้

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันไม่มากกว่า 1 พิกเซล ($S \leq 1$) ให้เลื่อนตำแหน่งของบล็อกตัวอักษรที่อยู่ทางด้านขวาเข้ามาชิดกับบล็อกตัวอักษรที่อยู่ทางด้านซ้ายเพื่อให้บล็อกตัวอักษรทั้งสองบล็อกรวมเป็นบล็อกเดียวกัน (ดูตัวอย่างในรูปที่ 3.12)

(ก) (ข)

รูปที่ 3.12 แสดงการจัดการบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ทั้งสองบล็อกและระยะห่างกันไม่มากกว่า 1 พิกเซล (ก) บล็อกตัวอักษรต้นฉบับซึ่งมีระยะห่างระหว่างบล็อกข้อมูลเท่ากับ 1 พิกเซล (ข) บล็อกตัวอักษรทางขวา "ติ" ถูกเลื่อนเข้ามาชิดบล็อกทางซ้าย "บั"

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันมากกว่า 1 พิกเซลแต่ไม่มากกว่า 4 พิกเซล ($1 < S < 4$) ให้เลื่อนตำแหน่งของบล็อกตัวอักษรที่อยู่ทางด้านขวาให้ห่างจากบล็อกที่อยู่ทางด้านซ้ายเท่ากับ 4 พิกเซลเพื่อให้บล็อกตัวอักษรทั้งสองบล็อกนี้ถูกแยกออกจากกันอย่างชัดเจน (ดูตัวอย่างในรูปที่ 3.13) เพื่อป้องกันมิให้บล็อกทั้งสองบล็อกนี้ชิดติดกันเมื่อเอกสารนี้ผ่านกระบวนการถ่ายเอกสารเพราะอาจมีผลทำให้ข้อมูลที่ซ่อนสูญหายได้

(ก) (ข)

รูปที่ 3.13 แสดงการจัดการบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้ทั้งสองบล็อกและมีระยะห่างกันมากกว่า 1 พิกเซลแต่ไม่มากกว่า 4 พิกเซล (ก) บล็อกตัวอักษรต้นฉบับซึ่งมีระยะห่างระหว่างบล็อกตัวอักษรเท่ากับ 2 พิกเซล (ข) บล็อกตัวอักษรทางขวา "ธี" ถูกเลื่อนออกจากบล็อกทางซ้าย "วิ" ทำให้มีช่องว่างระหว่างบล็อกเท่ากับ 4 พิกเซล

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันมากกว่าหรือเท่ากับ 4 พิกเซล ($S \geq 4$) ก็จะไม่เปลี่ยนแปลงตำแหน่งของบล็อกตัวอักษรใดๆทั้งสิ้น

2) บล็อกตัวอักษรที่อยู่ใกล้เคียงกันมีบล็อกใดบล็อกหนึ่งที่ไม่สามารถซ่อนข้อมูลได้ (บล็อกที่ฮิสโตแกรมในแนวนอนเป็นแบบไม่มีช่องว่างและฮิสโตแกรมในแนวตั้งสูงกว่าตัวอักษร

ปกติ เช่น บล็อกตัวอักษร “ไ” หรือ “ป” เป็นต้น) มีข้อกำหนดในการจัดตำแหน่งบล็อกตัวอักษรดังนี้

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันไม่มากกว่า 1 พิกเซล ($S \leq 1$) ให้เลื่อนตำแหน่งบล็อกตัวอักษรที่อยู่ทางด้านขวาเข้ามาชิดกับบล็อกตัวอักษรที่อยู่ทางด้านซ้ายเพื่อให้บล็อกตัวอักษรทั้งสองบล็อกรวมเป็นบล็อกเดียวกันซึ่งจะมีผลทำให้บล็อกตัวอักษรนี้ไม่สามารถใช้ซ่อนข้อมูลได้ (ดูตัวอย่างในรูปที่ 3.14)

(ก) (ข)

รูปที่ 3.14 แสดงการจัดการบล็อกตัวอักษรที่อยู่ใกล้เคียงกับบล็อกที่ไม่สามารถซ่อนข้อมูลได้โดยมีระยะห่างกันไม่มากกว่า 1 พิกเซล (ก) บล็อกตัวอักษรต้นฉบับซึ่งมีระยะห่างระหว่างบล็อกตัวอักษรเท่ากับ 1 พิกเซล (ข) บล็อกตัวอักษรทางขวา “ไ” ถูกเลื่อนเข้ามาชิดบล็อกทางซ้าย “ค”

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันมากกว่า 1 พิกเซลแต่ไม่มากกว่า 4 พิกเซล ให้เลื่อนตำแหน่งบล็อกตัวอักษรที่อยู่ทางด้านขวาให้ห่างจากบล็อกตัวอักษรที่อยู่ทางด้านซ้าย 4 พิกเซล ($1 < S < 4$) เพื่อทำให้บล็อกตัวอักษรทั้งสองถูกแยกออกจากกันอย่างชัดเจน (ดูตัวอย่างในรูปที่ 3.15) เพื่อป้องกันมิให้บล็อกทั้งสองบล็อกนี้ชิดติดกันเมื่อเอกสารผ่านกระบวนการถ่ายเอกสารซึ่งอาจมีผลทำให้ข้อมูลที่ซ่อนเกิดการสูญหายได้

(ก) (ข)

รูปที่ 3.15 แสดงการจัดการบล็อกตัวอักษรที่อยู่ใกล้เคียงกับบล็อกตัวอักษรที่ไม่สามารถซ่อนข้อมูลได้โดยมีระยะห่างกันมากกว่า 1 พิกเซลแต่ไม่มากกว่า 4 พิกเซล (ก) บล็อกตัวอักษรต้นฉบับซึ่งมีระยะห่างระหว่างบล็อกตัวอักษรเท่ากับ 2 พิกเซล (ข) บล็อกตัวอักษรทางขวา “ไ” ถูกเลื่อนออกจากบล็อกทางซ้าย “ช้” ให้มีช่องว่างระหว่างบล็อกเท่ากับ 4 พิกเซล

- ถ้าบล็อกตัวอักษรทั้งสองห่างกันมากกว่า 4 พิกเซล ($S \geq 4$) ก็จะไม่เปลี่ยนแปลงตำแหน่งของบล็อกตัวอักษรใดๆทั้งสิ้น

3.4.3 การกำหนดค่าสัญลักษณ์ข้อมูลที่จะซ่อนลงในเอกสาร

จากที่กล่าวมาแล้วว่าวิธีการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นของเอกสารรูปภาพภาษาไทยสามารถทำได้โดยการปรับเปลี่ยนขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามในหน่วยของพิกเซลเพื่อแทนค่าบิตข้อมูลที่ถูกซ่อน โดยมีรูปแบบสมการของการกำหนดค่าสัญลักษณ์ข้อมูลที่ซ่อนลงในเอกสารดังนี้

$$D = \Delta \pm \delta \tag{3.19}$$

โดยที่ D = ค่าสัญลักษณ์ข้อมูล

Δ = ค่าเฉลี่ยความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามภายในเอกสารทั้งหมด

δ = ปริมาณการปรับเปลี่ยนขนาดความกว้างของช่องว่าง (ซึ่งสามารถหาได้จากสมการที่ 3.22)

ค่าสัญลักษณ์ข้อมูลที่ซ่อนลงในเอกสารจะมีค่าเป็นเท่าใดนั้นขึ้นอยู่กับค่าบิตข้อมูลที่ต้องการซ่อนซึ่งมีข้อกำหนดดังนี้

- ถ้าข้อมูลที่ต้องการซ่อนคือ “1” จะกำหนดให้ค่าสัญลักษณ์ข้อมูลคือ

$$D = \Delta + \delta \tag{3.20}$$

- ถ้าข้อมูลที่ต้องการซ่อนคือ “0” จะกำหนดให้ค่าสัญลักษณ์ข้อมูลคือ

$$D = \Delta - \delta \tag{3.21}$$

<u>หมายเหตุ</u> ค่าสัญลักษณ์ข้อมูลจะเป็นจำนวนเต็มที่มีค่าใกล้เคียงกับค่าที่คำนวณได้ที่สุด

สำหรับสมการที่ใช้กำหนดค่าปริมาณการปรับเปลี่ยนขนาดความกว้างของช่องว่าง (δ) คือ

$$\delta = \alpha\Delta \tag{3.22}$$

โดยที่ α คือค่าพารามิเตอร์ที่ใช้ปรับขนาดความกว้างของช่องว่างระหว่างระดับเพื่อกำหนดค่าบิตข้อมูล "0" หรือ "1" (ขึ้นอยู่กับเอกสารแต่ละฉบับ)

คุณสมบัติที่สำคัญของค่า α ที่เหมาะสมคือจะต้องสามารถสร้างค่าสัญลักษณ์ข้อมูลสำหรับบิตข้อมูล "1" และ "0" ที่มีความแตกต่างกันมากที่สุด แต่จะต้องไม่ทำให้เอกสารที่ซ่อนข้อมูลเปลี่ยนแปลงไปจากเดิมมากนัก นั่นคือค่า α ไม่ควรมีค่ามากเกินไปหรือน้อยเกินไป เนื่องจากหาก α มีค่ามากเกินไปจะทำให้ขนาดของช่องว่างที่ใช้แทนบิตข้อมูล "0" และ "1" จะมีความแตกต่างกันมากจนทำให้สามารถสังเกตเห็นความผิดปกติของเอกสารที่ซ่อนข้อมูลได้ แต่หาก α มีค่าน้อยเกินไปก็จะทำให้ขนาดของช่องว่างที่ใช้แทนค่าข้อมูล "0" และ "1" มีความแตกต่างกันน้อยซึ่งอาจเป็นสาเหตุให้เกิดความผิดพลาดในการระบุค่าบิตข้อมูลในขั้นตอนของการดึงข้อมูลได้

3.5 อัลกอริทึมของการดึงข้อมูล

ในงานวิจัยนี้กำหนดให้เอกสารที่จะถูกดึงข้อมูลเป็นเอกสารรูปภาพแบบบิตแมปที่ผ่านกระบวนการกำจัดสิ่งรบกวนและลดความบิดเบือนของเอกสารแล้ว โดยอัลกอริทึมของการดึงข้อมูลแสดงอยู่ในรูปที่ 3.16

3.5.1 ขั้นตอนการดึงข้อมูล

รายละเอียดของขั้นตอนการดึงข้อมูลมีดังนี้ (ดูรูปที่ 3.16 ประกอบ)

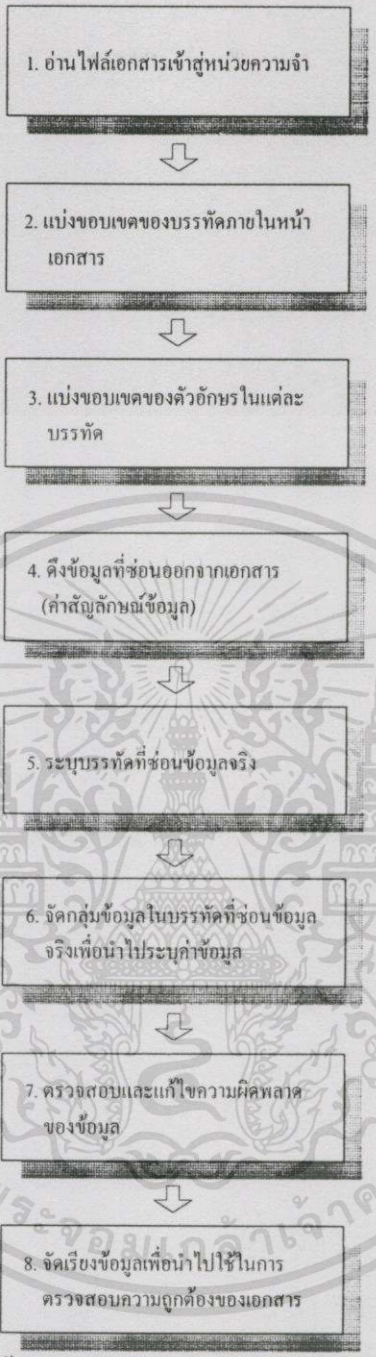
1) อ่านไฟล์เอกสารที่ต้องการดึงข้อมูลเข้ามาสู่หน่วยความจำ

2) แบ่งขอบเขตของแต่ละบรรทัดภายในหน้าเอกสารโดยการสร้างฮิสโตแกรมในแนวนอนของหน้าเอกสารนั้น (ดูตัวอย่างจากรูปที่ 3.9 ในหัวข้อ 3.4)

3) แบ่งขอบเขตของตัวอักษรในแต่ละบรรทัดออกเป็นบล็อกๆซึ่งมีขั้นตอนดังนี้

3.1) สร้างฮิสโตแกรมในแนวตั้งของแต่ละบรรทัดที่ได้จากขั้นตอนที่ 2 (ดูตัวอย่างจากรูปที่ 3.11 ในหัวข้อ 3.4)

3.2) สร้างฮิสโตแกรมในแนวนอนของแต่ละบล็อกตัวอักษรเพื่อตรวจหาตำแหน่งบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้โดยพิจารณาจากบล็อกที่มีช่องว่างระหว่างระดับชั้นที่สองกับสามเท่านั้น โดยที่บล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้จะถูกแบ่งออกเป็น 2 ส่วนคือ



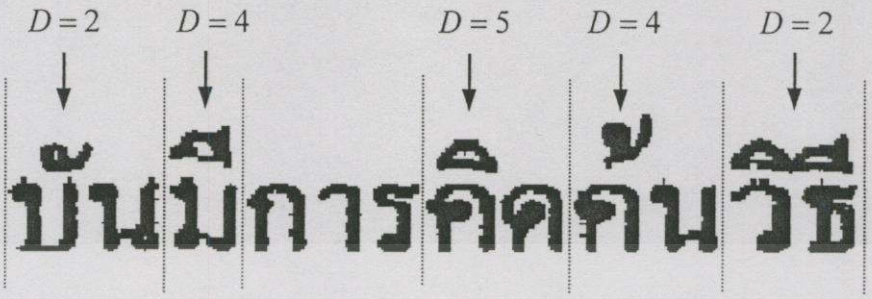
รูปที่ 3.16 แสดงรายละเอียดของขั้นตอนการดึงข้อมูล

คือส่วนของตัวอักษรที่อยู่ในระดับบนและส่วนของตัวอักษรที่อยู่ระดับล่าง (ดูตัวอย่างจากรูปที่ 3.12 ในหัวข้อ 3.4)

3.3) สร้างฮิสโตแกรมในแนวตั้งของบล็อกตัวอักษรที่อยู่ระดับบนเพื่อตรวจหาว่าภายในบล็อกนั้นจะมีตำแหน่งที่สามารถซ่อนข้อมูลได้มากกว่า 1 ตำแหน่งหรือไม่

4) ดึงข้อมูลที่ซ่อนออกจากเอกสารในทุกๆตำแหน่งที่ได้จากขั้นตอนที่ 3 โดยการหาค่าความกว้างระหว่างระดับชั้นของแต่ละตำแหน่ง (ค่าสัญลักษณ์ข้อมูล) จากนั้นทำการจัดเก็บค่า

เหล่านั้นลงตัวแปรอาร์เรย์ที่มีจำนวนแถวเท่ากับจำนวนบรรทัดภายในเอกสาร (ดูตัวอย่างในรูปที่ 3.17 จะได้ค่าชุดค่าสัญลักษณ์ข้อมูลจากบางส่วนของข้อความในบรรทัดนี้คือ [2 4 5 4 2])



รูปที่ 3.17 แสดงตัวอย่างการดึงค่าสัญลักษณ์ข้อมูลออกจากเอกสาร (ช่องว่างระหว่างระดับชั้นในหน่วยของพิกเซล) ออกจากบางส่วนของข้อความในเอกสารสำเนาที่ 1

5) ระบุตำแหน่งแถวที่คาดว่าจะเป็นแถวที่ซ่อนข้อมูล เนื่องจากในขั้นตอนของการซ่อนข้อมูลนั้นจะมีบางบรรทัดที่สามารถซ่อนข้อมูลได้และก็มีบางบรรทัดที่ไม่สามารถซ่อนข้อมูลได้ ดังนั้นจึงจำเป็นต้องระบุตำแหน่งบรรทัดที่ถูกซ่อนข้อมูลให้ได้เสียก่อน (ดูรายละเอียดในหัวข้อที่ 3.5.2) ซึ่งผลที่ได้จากขั้นตอนนี้คือชุดข้อมูลของแถวที่ถูกซ่อนข้อมูลและแถวที่ไม่ได้ซ่อนข้อมูล สำหรับชุดข้อมูลของแถวที่ซ่อนข้อมูลนั้นจะถูกจัดเก็บข้อมูลเพียง L_d ตำแหน่งเท่านั้น (เนื่องจากในขั้นตอนของการซ่อนข้อมูลจะซ่อนข้อมูลเพียงบรรทัดละ L_d ตำแหน่งเท่านั้น) ส่วนชุดข้อมูลของแถวที่ไม่ถูกซ่อนข้อมูลนั้นจะนำมาคำนวณหาค่าเธรดโชลด์ (δ_d) ที่จะใช้ในการระบุค่าข้อมูล

6) นำชุดข้อมูลที่ได้จากขั้นตอนที่ 5 (เฉพาะชุดข้อมูลของแถวถูกซ่อนข้อมูล) มาผ่านกระบวนการจัดกลุ่มข้อมูลเพื่อให้ข้อมูลในแต่ละตำแหน่งอยู่ในกลุ่มข้อมูลที่ถูกต้อง (เนื่องจากการซ่อนบิตข้อมูล 1 บิตลงในเอกสารนั้นจะใช้ตำแหน่งที่ซ่อนข้อมูลหลายตำแหน่ง) จากนั้นหาค่าเฉลี่ยของข้อมูลในแต่ละกลุ่มเพื่อนำไปมาระบุค่าบิตข้อมูลที่ถูกซ่อนโดยนำมาเปรียบเทียบกับค่าเธรดโชลด์ (δ_d) ที่ได้จากขั้นตอนที่แล้ว (ดูรายละเอียดในหัวข้อที่ 3.5.3) ซึ่งมีกฎในการระบุค่าข้อมูลดังนี้

- ถ้าค่าเฉลี่ยของข้อมูลในแต่ละกลุ่มมีค่ามากกว่าหรือเท่ากับ δ_d แสดงว่าบิตข้อมูลที่ถูกซ่อนคือ "1"
- ถ้าค่าเฉลี่ยข้อมูลในแต่ละกลุ่มมีค่าน้อยกว่า δ_d แสดงว่าบิตข้อมูลที่ถูกซ่อนคือ "0"

7) นำชุดข้อมูลที่ได้จากการระบุค่าในขั้นตอนที่แล้วมาผ่านกระบวนการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลโดยจะนำมาตรวจสอบและแก้ไขความผิดพลาดครั้งละ 7 แถวตามลำดับ (ดูตัวอย่างการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลได้จากหัวข้อ 3.3.5)

8) จัดเรียงข้อมูลที่จะนำไปใช้ในการตรวจสอบความถูกต้องเอกสารโดยการนำข้อมูล 4 แถวบนของแต่ละชุดรหัสแฮมมิงที่ได้จากขั้นตอนที่แล้วมาเรียงต่อกันก็จะได้ชุดข้อมูลที่ถูกซ่อนลงในเอกสาร (หมายเลขประจำเอกสาร) เพื่อนำไปใช้ในการตรวจสอบว่าเอกสารนั้นถูกส่งมาจากเจ้าของเอกสารจริงๆหรือไม่

3.5.2 การระบุบรรทัดที่ถูกใช้ซ่อนข้อมูล

หลักการที่จะกล่าวถึงในหัวข้อนี้เป็นหลักการที่ใช้เลือกตำแหน่งบรรทัดที่ถูกใช้ซ่อนข้อมูล ซึ่งไม่สามารถเลือกได้จากจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลในแต่ละบรรทัดได้ (ในขั้นตอนการซ่อนข้อมูลจะพิจารณาบรรทัดที่สามารถซ่อนข้อมูลได้จากจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้) เนื่องจากอาจมีบล็อกตัวอักษรที่ถูกใช้ซ่อนข้อมูลบางบล็อกอาจสูญหายไป (ไม่สามารถตรวจพบช่องว่างระหว่างระดับ) เมื่อเอกสารผ่านกระบวนการประมวลผลทางด้านเอกสาร ดังนั้นวิธีที่จะเลือกตำแหน่งบรรทัดที่ถูกใช้ซ่อนข้อมูลในงานวิจัยนี้จะพิจารณาจากความสัมพันธ์ของค่าสัญลักษณ์ข้อมูลในตำแหน่งที่ใกล้เคียงกันของแต่ละบรรทัด โดยค่าสัญลักษณ์ข้อมูลที่อยู่ในตำแหน่งที่ติดกันของบรรทัดที่ถูกใช้ซ่อนข้อมูลจะมีค่าใกล้เคียงกัน (เนื่องจากจะใช้ค่าสัญลักษณ์ข้อมูลหลายตำแหน่งเพื่อแทนค่าข้อมูลหนึ่งบิต) ส่วนบรรทัดที่ไม่ได้ถูกซ่อนข้อมูลจะมีค่าสัญลักษณ์ข้อมูลของตำแหน่งที่อยู่ติดกันแตกต่างกัน (ค่าสัญลักษณ์ข้อมูลของบิตข้อมูล "0" และ "1" สลับกัน) นั่นคือจะใช้ค่าเฉลี่ยของผลต่างของข้อมูลที่อยู่ในตำแหน่งที่ติดกันในการระบุว่าบรรทัดใดเป็นบรรทัดที่ถูกใช้ซ่อนข้อมูล โดยรายละเอียดของขั้นตอนการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลมีดังนี้

1) นำค่าสัญลักษณ์ข้อมูลทั้งหมดที่ได้จากการขั้นตอนการดึงข้อมูล (ขั้นตอนที่ 4) มาหาค่าเฉลี่ยของค่าเฉลี่ยผลต่างของข้อมูลในตำแหน่งที่อยู่ติดกันของทุกบรรทัด (μ)

2) หาค่าเฉลี่ยของผลต่างของค่าสัญลักษณ์ข้อมูลที่อยู่ในตำแหน่งที่ติดกันในแต่ละบรรทัด (M_i)

3) นำค่า M_i ของแต่ละบรรทัดที่ได้จากขั้นตอนที่ 2 มาเปรียบเทียบกับค่า μ เพื่อระบุว่าบรรทัดใดเป็นบรรทัดที่ถูกใช้ซ่อนข้อมูลซึ่งมีหลักเกณฑ์ดังนี้

- ถ้าค่า M_i ของบรรทัดใดๆมีค่ามากกว่าหรือเท่ากับ μ แสดงว่าบรรทัดนั้นไม่ได้ถูกใช้ซ่อนข้อมูล

- ถ้าค่า M_i ของบรรทัดใดๆมีค่าน้อยกว่าค่า μ แสดงว่าบรรทัดนั้นเป็นบรรทัดที่ถูกใช้ซ่อนข้อมูล

4) จัดเรียงข้อมูลเฉพาะบรรทัดที่ถูกระบุว่าถูกใช้ซ่อนข้อมูลเพื่อที่นำไปจัดกลุ่มข้อมูลในขั้นตอนต่อไป

5) จากนั้นนำข้อมูลที่เหลือการระบุบรรทัดที่ใช้ซ่อนข้อมูลทั้งหมดมาหาคำนวนหาค่าเฉลี่ยเพื่อกำหนดให้เป็นค่าเธรดโชลด์ที่จะใช้ระบุค่าบิตข้อมูล (δ_d)

6) ตัวอย่างการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูล

กำหนดให้ D คือชุดข้อมูลที่จะนำมระบุบรรทัดที่ถูกใช้ซ่อนข้อมูล

$$D=\begin{bmatrix} 5 & 6 & 5 & 5 & 3 & 2 & 2 & 3 & 3 & 2 & 2 & 3 \\ 3 & 6 & 2 & 5 & 3 & 6 & 2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.23}$$

และกำหนดให้ค่าเฉลี่ยของผลต่างของข้อมูลที่อยู่ติดกันของข้อมูลทั้งหมดที่ได้จากขั้นตอนการดึงข้อมูลออกจากเอกสาร (μ) เท่ากับ 1.5 (ข้อมูลนี้ได้มาจากการทดลอง) จะได้ว่า

- จากข้อมูลในแถวที่ 1 จะได้ค่าเฉลี่ยของผลต่างของข้อมูลที่อยู่ในตำแหน่งที่ติดกันดังนี้

$$M_1 = \frac{1+1+0+2+1+0+1+0+1+0+1}{11} = 0.7273 \tag{3.24}$$

จะเห็นว่า M_1 มีค่าน้อยกว่า 1.5 ดังนั้นข้อมูลในบรรทัดนี้เป็นบรรทัดที่ถูกใช้ซ่อนข้อมูล

- จากข้อมูลแถวที่ 2 จะได้ค่าค่าเฉลี่ยของผลต่างของข้อมูลที่อยู่ในตำแหน่งที่ติดกันดังนี้

$$M_2 = \frac{3+4+3+2+3+4}{6} = 3.1667 \tag{3.25}$$

จะเห็นว่า M_2 มีค่ามากกว่า 1.5 ดังนั้นข้อมูลในบรรทัดนี้ไม่ได้ถูกใช้ซ่อนข้อมูล

3.5.3 การจัดกลุ่มข้อมูล

โดยทั่วไปแล้วเมื่อเอกสารผ่านการประมวลผลทางด้านเอกสารอาจทำให้เอกสารเกิดความบิดเบือนไปจากเดิมซึ่งเป็นสาเหตุให้ข้อมูลที่ซ่อนอยู่ภายในเอกสารบางตำแหน่งสูญหายได้ โดยข้อมูลที่สูญหายนี้จะส่งผลกระทบต่อข้อมูลในตำแหน่งถัดๆไปทำให้เกิดเป็นความผิดพลาดแบบต่อเนื่อง ในงานวิจัยนี้ได้นำเสนอวิธีการแก้ไขปัญหานี้โดยใช้วิธีการซ่อนข้อมูลลงในเอกสารหลายตำแหน่งเพื่อแทนค่าบิตข้อมูลหนึ่งบิต (เช่น ซ่อนข้อมูล 4 ตำแหน่งเพื่อแทนค่าบิตข้อมูลหนึ่งบิต) ดังนั้นหากมีข้อมูลบางตำแหน่งสูญหายไปก็ยังสามารถหาค่าบิตข้อมูลที่ตำแหน่งนั้นได้จากข้อมูลที่เหลือโดยที่จะไม่ส่งผลกระทบไปยังข้อมูลในตำแหน่งถัดๆไป ดังนั้นก่อนที่จะระบุค่าบิตข้อมูลที่ซ่อนในขั้นตอนของการดึงข้อมูลจึงต้องมีการจัดกลุ่มค่าสัญลักษณ์ข้อมูลที่แทนค่าบิตข้อมูลเดียว

กันให้อยู่ในกลุ่มที่ถูกต้องเสียก่อน ซึ่งหลักการที่ใช้จัดกลุ่มค่าสัญลักษณ์ข้อมูลนี้จะพิจารณาจากค่าความแตกต่างของค่าสัญลักษณ์ข้อมูลของตำแหน่งที่อยู่ใกล้เคียงกัน ซึ่งถ้ามีความแตกต่างกันน้อยแสดงว่าข้อมูลเหล่านี้ควรจะอยู่ในกลุ่มเดียวกันแต่ถ้ามีความแตกต่างกันมากแสดงว่าข้อมูลเหล่านั้นไม่ควรที่จะอยู่ในกลุ่มเดียวกัน หลักการจัดกลุ่มข้อมูลที่จะนำเสนอนี้เป็นหลักการสำหรับการจัดกลุ่มข้อมูล 4 ตำแหน่งที่แทนค่าบิตข้อมูลเดียวกัน ดังนั้นข้อมูลที่อยู่ในแต่ละกลุ่มนี้จะมีค่าไม่เกิน 4 ตำแหน่ง โดยมีข้อกำหนดในการจัดกลุ่มข้อมูลคือจะไม่มีค่าสัญลักษณ์ข้อมูลสูญหายติดต่อกันมากกว่า 2 ตำแหน่งภายในค่าสัญลักษณ์ข้อมูลที่ถูกซ่อนต่อเนื่องกัน 8 ตำแหน่ง ซึ่งมีรายละเอียดในการจัดกลุ่มข้อมูลสำหรับแต่ละบรรทัดดังนี้ (ดูรูป 3.18 ประกอบ)

1. กำหนดให้ g_i คือเซ็ตของค่าสัญลักษณ์ข้อมูลที่แทนค่าบิตข้อมูลเดียวกันซึ่งมีจำนวนสมาชิก (N_{g_i}) เท่ากับ 4 ตำแหน่งคือ $\{b_{(i-1)*4+1}, b_{(i-1)*4+2}, b_{(i-1)*4+3}, b_{(i-1)*4+4}$ โดยที่ b_j คือค่าสัญลักษณ์ข้อมูล ณ ตำแหน่งที่ j^{th} ของการซ่อนข้อมูลในแต่ละแถวในเอกสารต้นฉบับ และกำหนดให้ b'_j คือข้อมูลในตำแหน่งการซ่อนที่ j^{th} ในเอกสารที่ต้องการตรวจสอบ

2. เริ่มการจัดกลุ่มข้อมูลโดยกำหนดให้ $i=1, j=1$

3. กำหนดให้ g'_i คือเซ็ตของค่าสัญลักษณ์ข้อมูลที่แทนค่าบิตข้อมูลเดียวกันในลำดับที่ i^{th} (ซึ่งอาจจะมีจำนวนสมาชิกน้อยกว่าหรือเท่ากับ 4 ตำแหน่ง อันเนื่องมาจากการที่มีค่าสัญลักษณ์ข้อมูลในบางตำแหน่งสูญหายไป) โดยเริ่มต้นจะกำหนดให้ $\{b'_j, b'_{j+1}\} \subseteq g'_i$ และ $\{b'_{j+4}, b'_{j+5}\} \subseteq g'_{i+1}$ ซึ่งแต่ละเซตมีค่าเฉลี่ยคือ $\overline{m}_{g'_i} = \frac{b'_j + b'_{j+1}}{2}$ และ $\overline{m}_{g'_{i+1}} = \frac{b'_{j+4} + b'_{j+5}}{2}$

4. จัดกลุ่มข้อมูล b'_{j+2} และ b'_{j+3} ว่าควรจะอยู่ในเซ็ตข้อมูล g'_i หรือ g'_{i+1} โดยการเปรียบเทียบค่าข้อมูล b'_{j+2} และ b'_{j+3} กับค่าเฉลี่ยของเซตข้อมูลทั้งสองเซ็ต ซึ่งสามารถแยกเป็นกรณีต่างๆ ได้ดังนี้

4.1 if $\left|b'_{j+2} - \overline{m}_{g'_i}\right| < \left|b'_{j+2} - \overline{m}_{g'_{i+1}}\right|$ & $\left|b'_{j+3} - \overline{m}_{g'_i}\right| < \left|b'_{j+3} - \overline{m}_{g'_{i+1}}\right|$ then

$\{b'_{j+2}, b'_{j+3}\} \subseteq g'_i$

4.2 if $\left|b'_{j+2} - \overline{m}_{g'_i}\right| > \left|b'_{j+2} - \overline{m}_{g'_{i+1}}\right|$ & $\left|b'_{j+3} - \overline{m}_{g'_i}\right| > \left|b'_{j+3} - \overline{m}_{g'_{i+1}}\right|$ then

$\{b'_{j+2}, b'_{j+3}\} \subseteq g'_{i+1}$

4.3 if $\left|b'_{j+2} - \overline{m}_{g'_i}\right| < \left|b'_{j+2} - \overline{m}_{g'_{i+1}}\right|$ & $\left|b'_{j+3} - \overline{m}_{g'_i}\right| > \left|b'_{j+3} - \overline{m}_{g'_{i+1}}\right|$ then

$\{b'_{j+2}\} \subseteq g'_i$,

$\{b'_{j+3}\} \subseteq g'_{i+1}$

4.4 if $\left|b'_{j+2} - \overline{m}_{g'_{i+1}}\right| < \left|b'_{j+2} - \overline{m}_{g'_i}\right|$ & $\left|b'_{j+3} - \overline{m}_{g'_i}\right| < \left|b'_{j+3} - \overline{m}_{g'_{i+1}}\right|$ then

4.4.1 if $\left|b'_{j+2} - \overline{m}_{g'_{i+1}}\right| < \left|b'_{j+3} - \overline{m}_{g'_i}\right|$ then

$g'_{i+1} = \{b'_{j+2}, b'_{j+3}$

4.4.2 if $\left|b'_{j+2}-\overline{m}_{g_{i+1}}\right| > \left|b'_{j+3}-\overline{m}_{g_i}\right|$ then

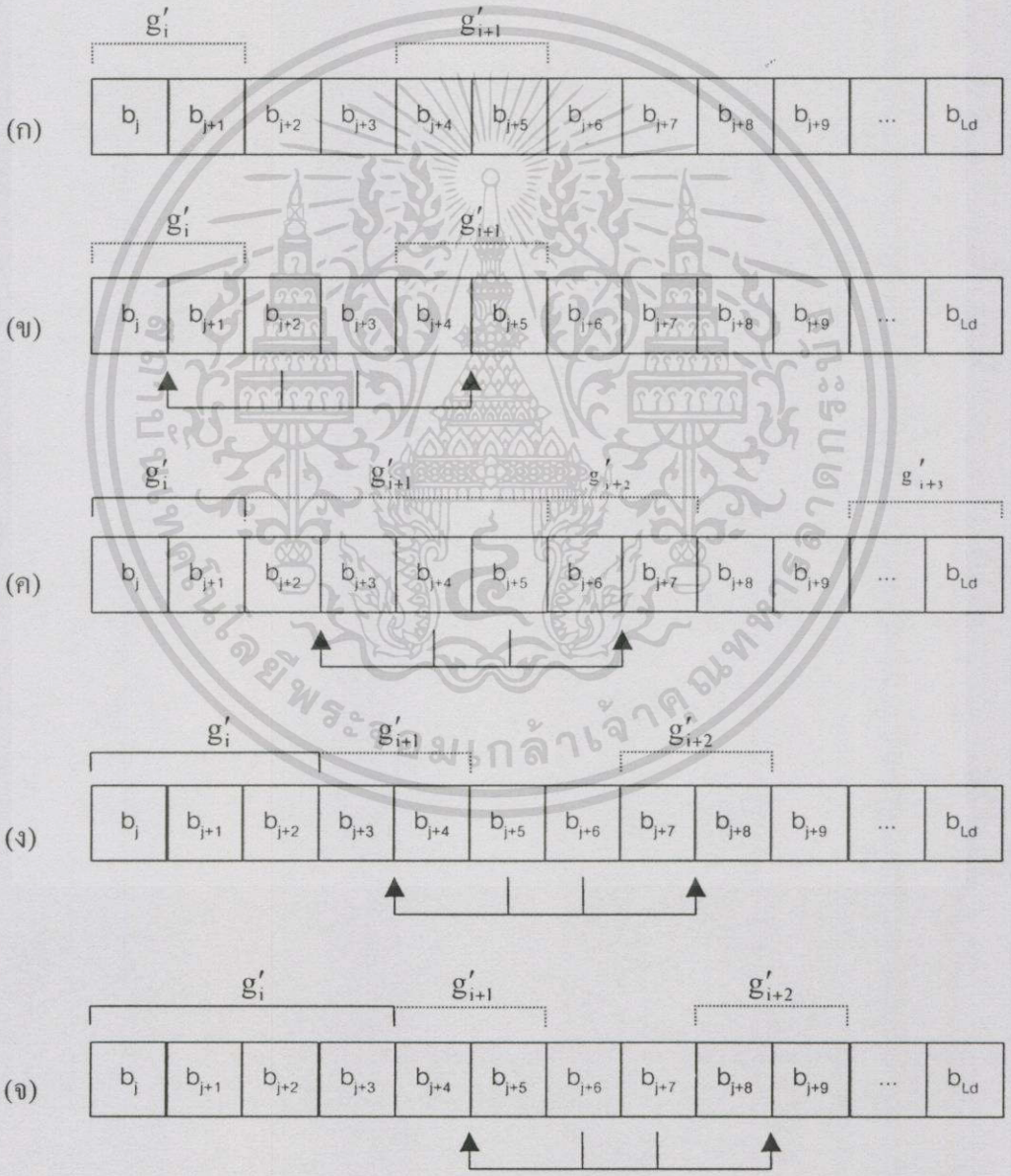
$g'_i = \{b'_{j+2}, b'_{j+3}$

5. เพิ่มค่า i และ j เพื่อการจัดกลุ่มข้อมูลในตำแหน่งถัดไป โดยการเพิ่มค่าตัวแปรทั้งสองตัวนี้จะพิจารณาจากจำนวนสมาชิกข้อมูลที่อยู่ใน g'_i ซึ่งสามารถแบ่งเป็นกรณีต่างๆได้ดังนี้

5.1 if $N_{g'_i}=2$ then $i=i+2, j=j+6$

5.2 if $N_{g'_i}>2$ then $i=i+1, j=j+N_{g_i}$

6. จัดกลุ่มข้อมูลในตำแหน่งถัดไปโดยทำตามขั้นตอนที่ 3–5 จนกระทั่งหมดชุดข้อมูล



รูปที่ 3.18 แสดงหลักการจัดกลุ่มข้อมูล (เส้นประ หมายถึงกลุ่มข้อมูลเริ่มต้น และเส้นทึบหมายถึงกลุ่มข้อมูลที่ได้จากการจัดกลุ่ม)

3.5.4 ตัวอย่างการตรวจสอบและแก้ไขความผิดพลาดของข้อมูล

วิธีการตรวจสอบและแก้ไขความผิดพลาดวิธีการนี้จะทำการตรวจหาค่าความผิดพลาดของซินโดรมของชุดข้อมูล (s) เพื่อที่จะหาตำแหน่งที่เกิดความผิดพลาดและทำการแก้ไขข้อมูลที่ตำแหน่งนั้นให้ถูกต้อง (ดูรายละเอียดของการตรวจหาค่าความผิดพลาดของซินโดรมของชุดข้อมูลได้จากหัวข้อ 3.3.5) พิจารณาตัวอย่างของการตรวจสอบและแก้ไขความผิดพลาดของชุดข้อมูลดังนี้

ตัวอย่างชุดข้อมูลที่จะนำมาตรวจสอบและแก้ไขความผิดพลาดของข้อมูลคือชุดข้อมูล r (สมการที่ 3.26) ซึ่งประกอบด้วยข้อมูล 2 ชุด คือ r_1 และ r_2 (สมการที่ 3.27 และ 3.28)

$$r = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.26}$$

$$r_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.27}$$

$$r_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.28}$$

- นำชุดข้อมูล r_1 มาตรวจสอบและแก้ไขความผิดพลาดโดยจัดชุดข้อมูลนี้ให้อยู่ในรูปแบบของเมตริกซ์ทรานสโพส (Transposed matrix) เพื่อที่จะนำไปแทนค่าในสมการที่ 3.13 ซึ่งจะได้ค่าความผิดพลาดของซินโดรมดังนี้

$$s = [0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0] \cdot \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= [0 \quad 0 \quad 0] \tag{3.29}$$

จะเห็นว่าค่าความผิดพลาดของซินโดรมของชุดข้อมูล r_1 คือศูนย์แสดงว่าข้อมูลรหัสแฮมมิงชุดนี้ไม่มีความผิดพลาดนั่นเอง

- นำชุดข้อมูล r_2 มาตรวจสอบและแก้ไขความผิดพลาดโดยจัดชุดข้อมูลนี้ให้อยู่ในรูปแบบของเมตริกซ์ทรานสโพส (Transposed matrix) เพื่อที่จะนำไปแทนค่าในสมการที่ 3.13 ซึ่งจะได้ค่าความผิดพลาดของซินโดรมดังนี้

$$s = [1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1] \cdot \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= [1 \quad 0 \quad 1] \tag{3.30}$$

จะเห็นว่าค่าความผิดพลาดของซินโดรมของข้อมูล r_2 คือ $[1 \quad 0 \quad 1]$ แสดงว่าข้อมูลรหัสแฮมมิงชุดนี้เกิดความผิดพลาดและเมื่อนำค่าความผิดพลาดของซินโดรมนี้ไปเปรียบเทียบกับเมตริกซ์ H' (เมตริกซ์ทรานสโพสที่ใช้ตรวจสอบพาริตี้) พบว่าตำแหน่งความผิดพลาดของข้อมูลในชุดข้อมูลนี้คือตำแหน่งที่ 2 (นับจากทางขวามือ) ดังนั้นเมื่อทราบตำแหน่งข้อมูลที่เกิดความผิดพลาดแล้วก็สามารถทำการแก้ไขข้อมูลให้ถูกต้องได้

3.6 การประยุกต์ใช้วิธีการซ่อนข้อมูลเพื่อวัตถุประสงค์ต่างๆ

จากที่กล่าวมาแล้วว่าวัตถุประสงค์หลักของวิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้ก็เพื่อใช้ตรวจสอบความถูกต้องของเอกสารว่าเอกสารนั้นถูกส่งมาจากเจ้าของเอกสารที่ถูกต้องหรือไม่ หรือเอกสารนั้นถูกเปลี่ยนแปลงแก้ไขมาแล้วหรือยัง ซึ่งในทางปฏิบัตินั้นเราสามารถนำวิธีการซ่อนข้อ

มูลที่นำเสนอนี้ไปประยุกต์ใช้งานเพื่อวัตถุประสงค์อย่างอื่นก็ได้ เช่น การซ่อนข้อมูลที่เป็นรายละเอียดเพิ่มเติมของเอกสารลงไปที่ตัวเอกสารด้วยเพื่อใช้ในการค้นหาเอกสารหรือข้อมูลอื่นๆที่เกี่ยวข้องได้

3.6.1 การซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสาร

วิธีการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสารในงานวิจัยนี้สามารถนำไปประยุกต์ใช้ร่วมกับวิธีการตรวจสอบความถูกต้องของเอกสารแบบอื่นๆได้เพื่อเพิ่มประสิทธิภาพการทำงานของการตรวจสอบเอกสารให้มีความถูกต้องมากยิ่งขึ้น เช่น การประทับตราเวลาในการรับ-ส่งเอกสารลงที่ตัวเอกสาร (Time-stamping) หรืออีกวิธีการหนึ่งคือการนำเอาข้อมูลภายในเอกสารมา สร้างเป็นหมายเลขประจำเอกสารเพื่อที่จะซ่อนลงในเอกสาร ซึ่งทั้งสองวิธีมีรายละเอียดดังนี้

3.6.1.1 การประทับตราเวลารับ-ส่งเอกสาร

วิธีการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสารโดยการใช้วิธีการประทับตราเวลาการรับ-ส่งเอกสาร (Time-stamping) จะช่วยให้เอกสารได้รับความปลอดภัยจากการถูกปลอมแปลงมากขึ้นเนื่องจากทางผู้รับและผู้ส่งจะมีการกำหนดช่วงระยะเวลาในการรับ-ส่งเอกสารร่วมกันเพื่อใช้ในการพิจารณาความถูกต้องของเอกสารนอกเหนือจากการเปรียบเทียบข้อมูลที่ซ่อนอยู่ภายในเอกสารกับหมายเลขประจำเอกสาร ดังนั้นหากมีเอกสารที่ส่งมาถึงผู้รับหลังช่วงระยะเวลาที่กำหนดเอาไว้ก็จะถือว่าเอกสารนั้นเป็นเอกสารที่ไม่ถูกต้องโดยทางผู้รับก็จะร้องขอกลับไปยังทางผู้ส่งเพื่อให้ส่งเอกสารฉบับนั้นกลับมาใหม่อีกครั้ง สำหรับการทำงานของวิธีการซ่อนข้อมูลในลักษณะนี้แสดงอยู่ในรูปที่ 3.19 ซึ่งมีรายละเอียดดังนี้

1) ผู้ส่งเอกสารจะส่งเอกสารที่ได้ซ่อนหมายเลขประจำเอกสารที่ผ่านการเข้ารหัสด้วยไพรเวตคีย์ของตนเองไปยัง B โดยที่เอกสารนั้นได้ประทับเวลาขณะที่ทำการส่งลงไปในเอกสารพร้อมทั้งหมายเลขประจำเอกสาร (ที่ยังไม่ได้เข้ารหัส) ลงไปด้วย

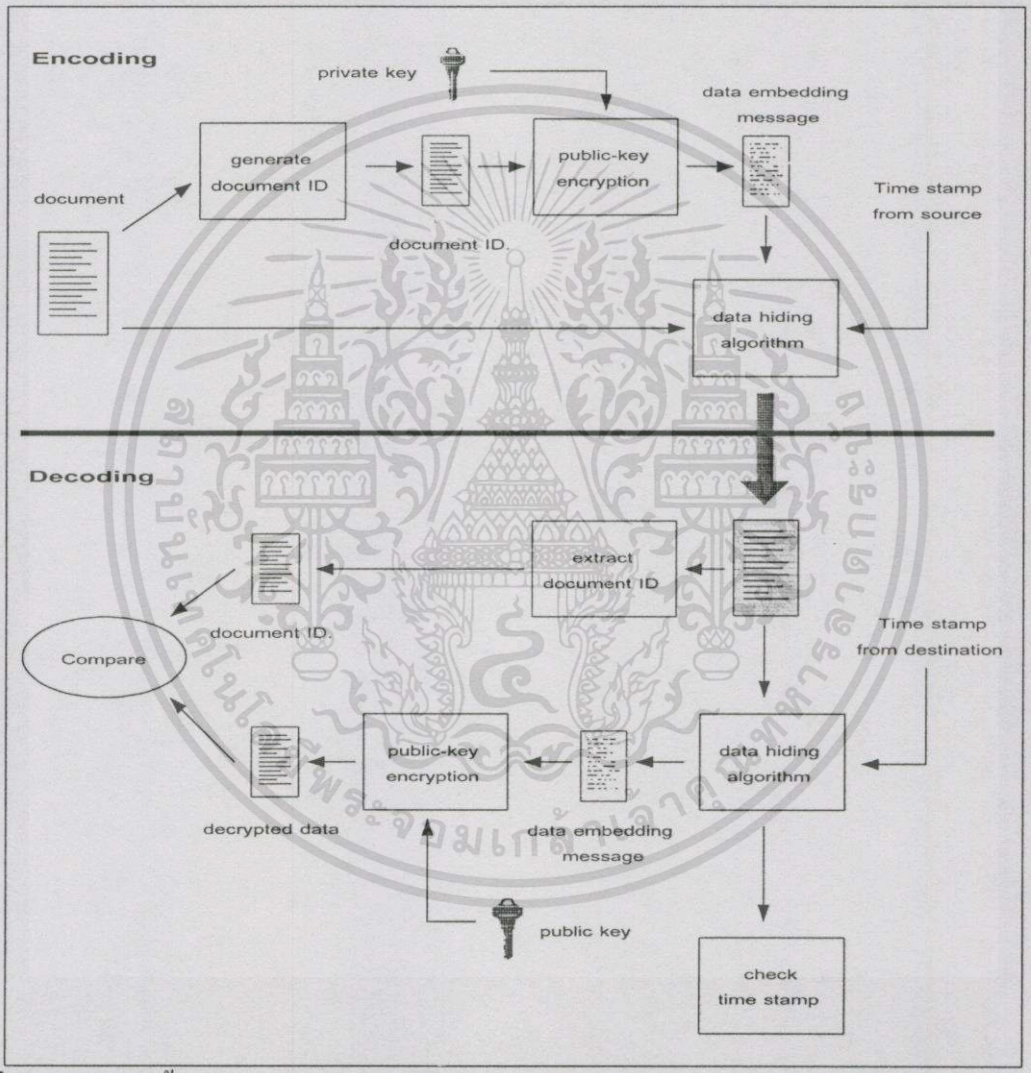
2) เมื่อทางผู้รับได้รับเอกสารนั้นก็จะบันทึกเวลาที่ได้รับเอกสารนั้นเอาไว้ และทำการดึงข้อมูลที่ซ่อนอยู่ภายในเอกสารออกมาเพื่อถอดรหัสข้อมูลนั้นด้วยพับลิคคีย์ของตนเอง จากนั้นทำการเปรียบเทียบเวลาส่งที่ได้ประทับมากับตัวเอกสารกับเวลาที่ได้รับเอกสารว่ามีความแตกต่างกันเกินช่วงระยะเวลาที่กำหนดไว้หรือไม่ โดยสามารถแยกพิจารณาได้ 2 กรณีคือ

2.1) ถ้าช่วงระยะเวลาของการรับ-ส่งเอกสารมีระยะเวลามากกว่าช่วงเวลาที่กำหนดเอาไว้ก็จะถือว่าเอกสารนั้นเป็นเอกสารที่ไม่ถูกต้อง ให้ทางผู้รับร้องขอกลับไปยังทางผู้ส่งเพื่อให้ส่งเอกสารฉบับนั้นกลับมาอีกครั้ง

2.2) ถ้าช่วงระยะเวลาของการรับ-ส่งเอกสารมีระยะเวลาน้อยกว่าหรือเท่ากับช่วงเวลาที่กำหนดเอาไว้แสดงว่าเอกสารฉบับนั้นมีแนวโน้มที่จะเป็นเอกสารที่ถูกต้องและเพื่อเป็นการยืนยันความถูกต้องของเอกสารอีกครั้งก็จะทำการเปรียบเทียบข้อมูลที่ดึงออกมาจากเอกสารกับ

หมายเลขเอกสารว่ามีความเหมือนหรือแตกต่างกัน ซึ่งถ้าหมายเลขทั้งสองเหมือนกันแสดงว่าเอกสารนี้มีความถูกต้องอย่างแน่นอน แต่ถ้าหมายเลขทั้งสองเกิดความแตกต่างกันมากเกินขอบเขตที่กำหนดเอาไว้ก็ให้ถือว่าเอกสารฉบับนั้นไม่ถูกต้อง

สำหรับค่าความถูกต้องของการตรวจสอบความถูกต้องของเอกสารนั้นจะขึ้นอยู่กับการกำหนดช่วงระยะเวลาของการรับ-ส่งเอกสารซึ่งควรจะกำหนดให้มีค่าน้อยที่สุดเท่าที่จะเป็นไปได้ เนื่องจากจะทำให้โอกาสที่เอกสารจะถูกปลอมแปลงได้นั้นมีน้อยมากนั่นเอง

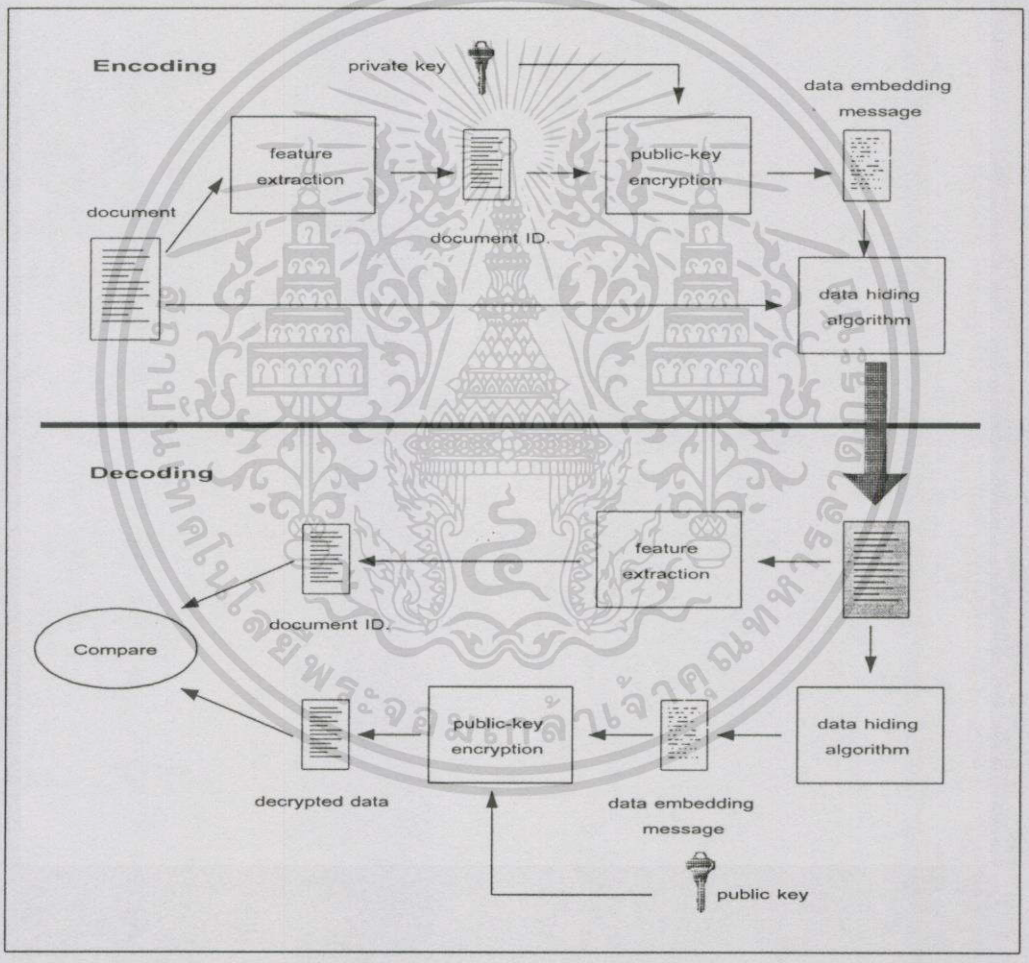


รูปที่ 3.19 แสดงขั้นตอนการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสารโดยนำมาประยุกต์ใช้ร่วมกับวิธีการประทับตราเวลาในการรับ-ส่งเอกสาร

3.6.1.2 การสร้างหมายเลขเอกสารจากข้อมูลภายในเอกสาร

วิธีการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสารโดยการนำเอาข้อมูลภายในเอกสาร เช่น ตัวเลขต่างๆที่อยู่ภายในเอกสารหรือชื่อเจ้าของเอกสาร มาผ่านกระบวนการบาง

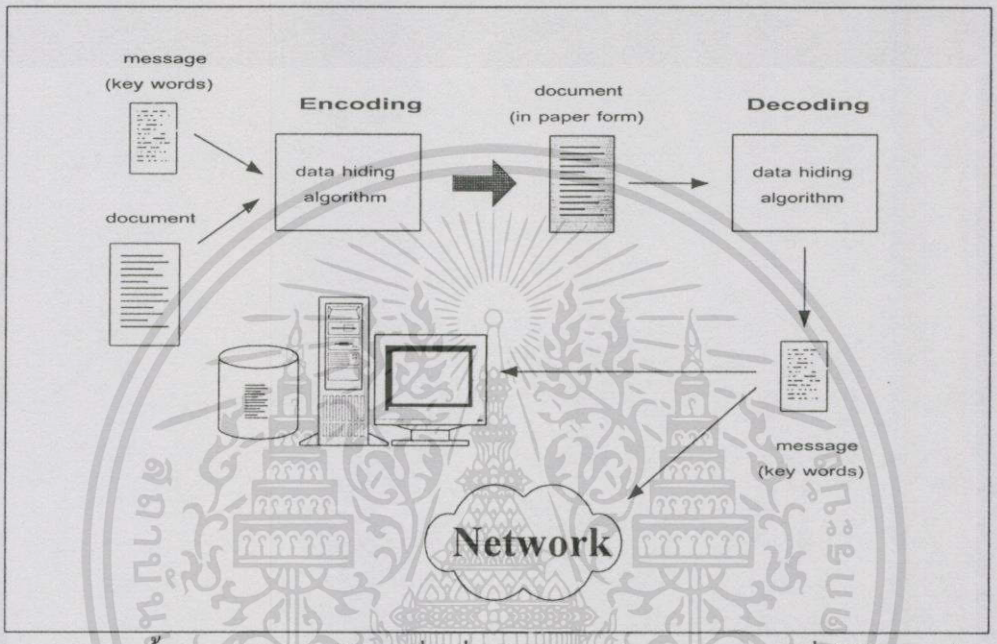
อย่างเพื่อสร้างเป็นหมายเลขประจำเอกสาร (เช่น กระบวนการเข้ารหัสแบบแฮชฟังก์ชัน [10]) ซึ่งวิธีการนี้สามารถช่วยให้เอกสารมีความปลอดภัยจากการถูกปลอมแปลงมากขึ้น เนื่องจากหมายเลขประจำเอกสารที่ถูกสร้างขึ้นมานี้จะมีความสัมพันธ์โดยตรงกับเนื้อหาบางส่วนของเอกสารนั้น ดังนั้นหากข้อมูลบางส่วนภายในเอกสารถูกทำให้เปลี่ยนแปลงไปก็จะสามารถตรวจสอบความถูกต้องได้ แต่วิธีการนี้ยังคงมีข้อจำกัดในด้านของเวลาคือ หากผู้ที่ต้องการปลอมแปลงเอกสารนั้นมีระยะเวลามากพอสมควร (เนื่องจากผู้รับและผู้ส่งได้ไม่ได้จำกัดเวลาของการรับ-ส่งเอกสาร) ในการตรวจสอบลักษณะความผิดปกติของเอกสารที่เกิดจากการซ่อนข้อมูลก็สามารถทำเอกสารเลียนแบบขึ้นมาได้ ดังนั้นวิธีการนี้จึงเหมาะสำหรับเอกสารที่ไม่มีความสำคัญมากนัก



รูปที่ 3.20 แสดงขั้นตอนการซ่อนข้อมูลเพื่อใช้ตรวจสอบความถูกต้องของเอกสาร โดยนำมาประยุกต์ใช้ร่วมกับการสร้างหมายเลขเอกสารจากข้อมูลภายในเอกสาร

3.6.2 การซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสาร

วิธีการซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสารลงไปในเอกสารนั้น (เช่น การซ่อนคำสำคัญของเอกสารหรือหมายเลขเอกสารที่จะนำมาใช้ในการอ้างอิง) มีจุดประสงค์เพื่อที่จะนำข้อมูลที่ซ่อนอยู่ไปใช้ในการค้นหาเอกสารหรือข้อมูลอื่นๆที่เกี่ยวข้องกับเอกสาร (เมื่อเอกสารนั้นอยู่ในรูปแบบกระดาษ) โดยแสดงรูปแบบการทำงานอยู่ในรูปที่ 3.21



รูปที่ 3.21 แสดงขั้นตอนการซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสารเพื่อใช้ในการค้นหาข้อมูลหรือเอกสารอื่นๆที่เกี่ยวข้อง

บทที่ 4
ผลการทดลองและปัญหาที่พบ

ในบทนี้จะกล่าวถึงการทดลองซ่อนข้อมูลลงในเอกสารภาษาไทยเพื่อศึกษาข้อจำกัดของเอกสารที่ถูกปรับขนาดและเอกสารที่ถูกสำเนาโดยการถ่ายเอกสาร โดยจะแบ่งการทดลองออกเป็น 2 การทดลองคือ การทดลองดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการปรับขนาดและการทดลองดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการถ่ายเอกสาร

4.1 ข้อกำหนดในการทดลอง

ในส่วนนี้จะกล่าวถึงข้อกำหนดต่างๆของการเตรียมข้อมูลและเอกสารที่จะใช้ในการทดลอง รวมถึงแสดงรายละเอียดของอุปกรณ์ต่างๆที่ใช้ในการทดลองด้วย และในส่วนท้ายของหัวข้อนี้จะแสดงตัวอย่างการเตรียมข้อมูล การซ่อนข้อมูล และการดึงข้อมูลออกจากเอกสารรวมทั้งอธิบายหลักการที่ใช้ในการตรวจสอบความถูกต้องของเอกสารด้วย

4.1.1 การเตรียมเอกสาร

เอกสารที่ใช้ทำการทดลองในงานวิจัยนี้เป็นเอกสารภาษาไทยขนาด A4 ที่ใช้ตัวอักษรรูปแบบ (Font) AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดแบบชิดขอบคอลัมน์เดียว โดยรายละเอียดของการเตรียมเอกสารมีดังนี้

1) สร้างเอกสารจากโปรแกรมเวิร์ดโปรเซสซิง (Word processing) จำนวน 5 หน้า โดยแต่ละหน้าจะมีจำนวนบรรทัดประมาณ 36 บรรทัด

2) แปลงเอกสารจากขั้นตอนที่ 1 ให้เป็นโพสคริปท์ไฟล์ (Postcript file)

3) จากนั้นแปลงเอกสารที่อยู่ในรูปแบบโพสคริปท์ไฟล์ให้เป็นเอกสารรูปภาพไบนารีที่มีการจัดเก็บแบบบิตแมปโดยกำหนดให้ค่าความละเอียดของเอกสารมีค่าเท่ากับ 300 จุดต่อนิ้ว

4) สำหรับเอกสารที่ใช้ในการทดลองที่สอง (การทดลองดึงข้อมูลจากเอกสารที่ผ่านกระบวนถ่ายเอกสาร) มีข้อกำหนดเพิ่มเติมดังนี้

- การพิมพ์เอกสารจะใช้เครื่องพิมพ์เลเซอร์ยี่ห้อ Fujitsu รุ่น FJPP 14V (มีอายุการใช้งานประมาณ 2 ปี) โดยกำหนดค่าความละเอียดในการพิมพ์เท่ากับ 600 จุดต่อนิ้ว
- การสร้างสำเนาเอกสารจะใช้เครื่องถ่ายเอกสารยี่ห้อ CANON รุ่น NP6650 (มีอายุการใช้งานประมาณ 1 ปี) โดยกำหนดค่าความสว่างของการถ่ายเอกสารให้อยู่ในระดับที่ 3 (มีทั้งหมด 6 ระดับ)

- การแปลงเอกสาร (รูปแบบกระดาษ) กลับให้อยู่ในรูปแบบอิเลคทรอนิคนั้น กระทำโดยการสแกนเอกสารเข้าเครื่องคอมพิวเตอร์ด้วยเครื่องสแกนเนอร์ยี่ห้อ Hewlett Packard ScanJet 4C (อายุการใช้งานประมาณ 4 ปี) โดยกำหนดค่าความละเอียดในการสแกนเอกสารเท่ากับ 300 จุดต่อนิ้ว, ระดับค่าความสว่างเท่ากับ 125 (จากทั้งหมด 250 ระดับ) และอัตราส่วนการปรับขนาดเท่ากับ 100 เปอร์เซ็นต์

- เอกสารที่จะนำมาดึงข้อมูลนั้นต้องผ่านกระบวนการกำจัดสิ่งรบกวนและปรับความเอียงของเอกสาร (Noise and distort reduction) มาแล้ว โดยวิธีการที่ใช้กำจัดสิ่งรบกวนภายในเอกสารนั้นจะกระทำโดยการลบส่วนที่เป็นจุดดำเล็กๆที่คาดว่าจะไม่ใช่ส่วนประกอบของตัวอักษรออกจากเอกสารและสำหรับวิธีการปรับความเอียงของเอกสารจะกระทำโดยการตรวจดูว่าข้อความที่อยู่ภายในเอกสารนั้นทำมุมเอียงไปจากเดิมเท่าใดเพื่อที่จะปรับค่ามุมเอียงของเอกสารนั้นให้กลับไปเหมือนเดิม (โดยในที่นี้ใช้วิธีการสร้างเส้นตรงแนวนอนไว้ที่ส่วนล่างของเอกสารเพื่อเป็นเครื่องมือในการปรับค่ามุมเอียงของเอกสาร)

4.1.2 การเตรียมข้อมูล

ข้อมูลที่จะซ่อนลงในเอกสารก็คือหมายเลขประจำเอกสารแต่ละฉบับซึ่งมีข้อกำหนดในการเตรียมข้อมูลดังนี้ (รายละเอียดของข้อมูลทั้งหมดแสดงไว้ในส่วนของภาคผนวก ก.)

- หมายเลขประจำเอกสารแต่ละฉบับได้มาจากตัวเลขสุ่มจำนวนนับ 6 หลัก
- การซ่อนข้อมูลหนึ่งบิตลงในเอกสารนั้นจะใช้จำนวนสัญลักษณ์ข้อมูลทั้งหมด 4 ตำแหน่ง (1 data bit = 4 embedded symbols : $R_d = 4$)
- ในแต่ละบรรทัดจะซ่อนข้อมูลเพียง 8 ตำแหน่งเท่านั้น ($L_d = 8$)

4.1.3 การวิเคราะห์ความถูกต้องของข้อมูลที่ซ่อนภายในเอกสาร

การวิเคราะห์ความถูกต้องของข้อมูลที่ดึงออกมาจากเอกสารในงานวิจัยนี้จะแยกพิจารณาได้เป็น 3 ลักษณะคือ

1) ความผิดพลาดของข้อมูลที่เกิดขึ้นในแต่ละเฟรม (Frame error rate) โดยขอบเขตของคำว่าเฟรมอาจจะครอบคลุมถึงข้อความในเอกสารทั้งหน้า (Page) หรืออาจจะครอบคลุมถึงข้อความในแต่ละคอลัมน์ (Column) ของเอกสาร (ถ้าเอกสารมีมากกว่าหนึ่งคอลัมน์) หรืออาจจะครอบคลุมถึงข้อความในแต่ละย่อหน้า (Paragraph) ของเอกสารก็ได้ แต่ขอบเขตของเฟรมที่กำหนดให้ใช้ในงานวิจัยนี้จะครอบคลุมถึงข้อความในเอกสารทั้งหน้า ดังนั้นอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละเฟรมจึงหมายถึงจำนวนหน้าเอกสารที่เกิดความผิดพลาดต่อจำนวนหน้าเอกสารทั้งหมด

2) ความผิดพลาดของข้อมูลที่เกิดขึ้นในแต่ละบล็อก (Block error rate) โดยกำหนดให้ขอบเขตของหนึ่งบล็อกข้อมูลคือหนึ่งชุดข้อมูลรหัสแฮมมิง (ประกอบด้วยข้อมูล 7 ตำแหน่งในแนว

ตั้ง) ดังนั้นอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละบล็อกข้อมูลจึงหมายถึงจำนวนชุดข้อมูลรหัสแฮมมิงที่เกิดความผิดพลาดต่อจำนวนชุดข้อมูลรหัสแฮมมิงทั้งหมด

3) ความผิดพลาดของข้อมูลที่เกิดขึ้นแต่ละบิต (Bit error rate) หมายถึงจำนวนบิตข้อมูลที่เกิดความผิดพลาดต่อจำนวนบิตข้อมูลทั้งหมด (พิจารณาจากบิตข้อมูลทั้งหมดที่อยู่ในชุดข้อมูลรหัสแฮมมิง)

4.1.4 ตัวอย่างการเตรียมข้อมูล

1) สุ่มเลขจำนวนนับขึ้นมา 6 หลักเพื่อกำหนดให้เป็นหมายเลขประจำเอกสาร โดยเลขสุ่มที่ได้จากตัวอย่างนี้คือ "962461"

2) นำเลขสุ่มที่ได้จากขั้นตอนที่แล้วมาเข้ารหัสข้อมูลโดยใช้วิธี RSA Public-key encryption เพื่อสร้างลายเซ็นดิจิตอลซึ่งให้ผลลัพธ์คือ "303632309" (ดูรายละเอียดการเข้ารหัสจากหัวข้อ 3.3.2 ในบทที่ 3)

3) แปลงข้อมูลลายเซ็นดิจิตอลให้อยู่ในรูปแบบข้อมูลไบนารี โดยมีหลักการคือจะแบ่งข้อมูลที่ได้จากขั้นตอนที่แล้วออกเป็นบล็อกๆ โดยกำหนดให้แต่ละบล็อกเป็นเลขจำนวนนับ 3 หลัก และนำข้อมูลที่ได้มาแปลงให้เป็นข้อมูลไบนารีดังนี้

303 ข้อมูลไบนารีคือ 0100101111

632 ข้อมูลไบนารีคือ 1001111000

309 ข้อมูลไบนารีคือ 0100110101

จากนั้นนำชุดข้อมูลไบนารีทั้งหมดมาเรียงต่อกัน (ทั้งหมด 30 บิต) ได้ดังนี้ "010010111110011110000100110101"

4) เพิ่มข้อมูลซ้ำต่อท้ายข้อมูลในแต่ละบิตเพื่อเพิ่มจำนวนสัญลักษณ์ข้อมูลที่ใช้แทนค่าบิตข้อมูลแต่ละบิต ซึ่งในตัวอย่างนี้จะใช้ค่าสัญลักษณ์ข้อมูล 4 ตำแหน่งในการแทนค่าข้อมูลหนึ่งบิต ดังนั้นจึงต้องเพิ่มข้อมูลซ้ำต่อท้ายบิตข้อมูลเดิมเข้าไปอีก 3 ตำแหน่ง (ในทุกๆบิตข้อมูล) ทำให้ได้ชุดข้อมูลดังนี้ "**0000**1111**0000**0000**1111**0000**1111**1111**1111**1111**1111**0000**0000**1111**1111**1111**1111**0000**0000**0000**0000**0000**1111**0000**0000**1111**1111**0000**1111**0000**1111**" (มีทั้งหมด 120 บิต)

หมายเหตุ ตัวเลขที่เป็นตัวหนาคือบิตข้อมูลที่ต้องการซ่อน ส่วนตัวเลขที่ถูกขีดเส้นใต้คือบิตข้อมูลที่ถูกเพิ่มเข้ามา

5) นำข้อมูลที่ได้จากขั้นตอนที่แล้วไปสร้างชุดข้อมูลรหัสแฮมมิงโดยมีรายละเอียดของการจัดชุดข้อมูลดังนี้

5.1) แบ่งชุดข้อมูลที่ได้จากขั้นตอนที่แล้วออกเป็นแถวๆ โดยกำหนดให้มีข้อมูลแถวละ 8 ตำแหน่ง ($L_d = 8$) สำหรับแถวที่มีจำนวนข้อมูลไม่ถึง 8 ตำแหน่ง (หมดชุดข้อมูล) ก็จะเพิ่มข้อมูล "0" ต่อท้ายข้อมูลในแถวนั้นเพื่อให้มีจำนวนครบ 8 ตำแหน่งโดยมีรายละเอียดของข้อมูลดังนี้

แถวที่ 1 : 0 0 0 0 1 1 1 1

แถวที่ 2 : 0 0 0 0 0 0 0 0

แถวที่ 3 : 1 1 1 1 0 0 0 0

แถวที่ 4 : 1 1 1 1 1 1 1 1

แถวที่ 5 : 1 1 1 1 1 1 1 1

แถวที่ 6 : 1 1 1 1 0 0 0 0

แถวที่ 7 : 0 0 0 0 1 1 1 1

แถวที่ 8 : 1 1 1 1 1 1 1 1

แถวที่ 9 : 1 1 1 1 0 0 0 0

แถวที่ 10 : 0 0 0 0 0 0 0 0

แถวที่ 11 : 0 0 0 0 1 1 1 1

แถวที่ 12 : 0 0 0 0 0 0 0 0

แถวที่ 13 : 1 1 1 1 1 1 1 1

แถวที่ 14 : 0 0 0 0 1 1 1 1

แถวที่ 15 : 0 0 0 0 1 1 1 1

5.2) นำข้อมูลที่ได้จากขั้นตอนที่ 5.1 มาสร้างชุดข้อมูลรหัสแฮมมิงเพื่อสร้างข้อมูลในส่วนของรีดันแดนซีที่เป็นพาริตี้บิต (วิธีการสร้างชุดรหัสแฮมมิงนี้จะทำการสร้างชุดรหัสในแนวตั้งโดยจะแบ่งเป็นส่วนของข้อมูล 4 ตำแหน่งและส่วนของรีดันแดนซีที่เป็นพาริตี้บิตอีก 3 ตำแหน่ง) โดยจะนำมาสร้างชุดข้อมูลครั้งละ 4 แถวจนกระทั่งหมดชุดข้อมูล (สำหรับชุดข้อมูลที่มีไม่ครบ 4 แถวก็ให้เพิ่มแถวข้อมูลที่เป็น "0" เข้าไปจนครบ) โดยมีรายละเอียดของชุดข้อมูลรหัสแฮมมิงดังนี้

- ข้อมูลรหัสแฮมมิงชุดที่ 1 ประกอบด้วย

ข้อมูลแถวที่ 1 : 0 0 0 0 1 1 1 1

ข้อมูลแถวที่ 2 : 0 0 0 0 0 0 0 0

ข้อมูลแถวที่ 3 : 1 1 1 1 0 0 0 0

ข้อมูลแถวที่ 4 : 1 1 1 1 1 1 1 1

} บิตข้อมูล

รหัสแฮมมิง 1 : 1 1 1 1 0 0 0 0

รหัสแฮมมิง 2 : 0 0 0 0 0 0 0 0

รหัสแฮมมิง 3 : 0 0 0 0 1 1 1 1

} รีดันแดนซี

- ข้อมูลรหัสแฮมมิงชุดที่ 2 ประกอบด้วย

ข้อมูลแถวที่ 5	:	1	1	1	1	1	1	1	1	บิตข้อมูล
ข้อมูลแถวที่ 6	:	1	1	1	1	0	0	0	0	
ข้อมูลแถวที่ 7	:	0	0	0	0	1	1	1	1	
ข้อมูลแถวที่ 8	:	1	1	1	1	1	1	1	1	
รหัสแฮมมิง 1	:	1	1	1	1	0	0	0	0	รีดันแดนซี
รหัสแฮมมิง 2	:	0	0	0	0	1	1	1	1	
รหัสแฮมมิง 3	:	0	0	0	0	0	0	0	0	

- ข้อมูลรหัสแฮมมิงชุดที่ 3 ประกอบด้วย

ข้อมูลแถวที่ 9	:	1	1	1	1	0	0	0	0	บิตข้อมูล
ข้อมูลแถวที่ 10	:	0	0	0	0	0	0	0	0	
ข้อมูลแถวที่ 11	:	0	0	0	0	1	1	1	1	
ข้อมูลแถวที่ 12	:	0	0	0	0	0	0	0	0	
รหัสแฮมมิง 1	:	1	1	1	1	0	0	0	0	รีดันแดนซี
รหัสแฮมมิง 2	:	1	1	1	1	1	1	1	1	
รหัสแฮมมิง 3	:	0	0	0	0	1	1	1	1	

- ข้อมูลรหัสแฮมมิงชุดที่ 4 ประกอบด้วย

ข้อมูลแถวที่ 13	:	1	1	1	1	1	1	1	1	บิตข้อมูล
ข้อมูลแถวที่ 14	:	0	0	0	0	1	1	1	1	
ข้อมูลแถวที่ 15	:	0	0	0	0	1	1	1	1	
แถวข้อมูลที่ถูกเพิ่มเข้า ⟶ **ข้อมูลแถวที่ 16**	:	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
รหัสแฮมมิง 1	:	1	1	1	1	0	0	0	0	รีดันแดนซี
รหัสแฮมมิง 2	:	1	1	1	1	0	0	0	0	
รหัสแฮมมิง 3	:	0	0	0	0	0	0	0	0	

6) จัดเรียงชุดข้อมูลที่ได้จากขั้นตอนที่ 5 ในแต่ละชุดเข้าด้วยกันซึ่งจะได้ชุดข้อมูลที่จะซ่อนลงในเอกสารดังนี้

แถวที่ 1	:	0	0	0	0	1	1	1	1
แถวที่ 2	:	0	0	0	0	0	0	0	0
แถวที่ 3	:	1	1	1	1	0	0	0	0
แถวที่ 4	:	1	1	1	1	1	1	1	1

แถวที่ 5 : 1 1 1 1 0 0 0 0
แถวที่ 6 : 0 0 0 0 0 0 0 0
แถวที่ 7 : 0 0 0 0 1 1 1 1
แถวที่ 8 : 1 1 1 1 1 1 1 1
แถวที่ 9 : 1 1 1 1 0 0 0 0
แถวที่ 10 : 0 0 0 0 1 1 1 1
แถวที่ 11 : 1 1 1 1 1 1 1 1
แถวที่ 12 : 1 1 1 1 0 0 0 0
แถวที่ 13 : 0 0 0 0 1 1 1 1
แถวที่ 14 : 0 0 0 0 0 0 0 0
แถวที่ 15 : 1 1 1 1 0 0 0 0
แถวที่ 16 : 0 0 0 0 0 0 0 0
แถวที่ 17 : 0 0 0 0 1 1 1 1
แถวที่ 18 : 0 0 0 0 0 0 0 0
แถวที่ 19 : 1 1 1 1 0 0 0 0
แถวที่ 20 : 1 1 1 1 1 1 1 1
แถวที่ 21 : 0 0 0 0 1 1 1 1
แถวที่ 22 : 1 1 1 1 1 1 1 1
แถวที่ 23 : 0 0 0 0 1 1 1 1
แถวที่ 24 : 0 0 0 0 1 1 1 1
แถวที่ 25 : 0 0 0 0 0 0 0 0
แถวที่ 26 : 1 1 1 1 0 0 0 0
แถวที่ 27 : 1 1 1 1 0 0 0 0
แถวที่ 28 : 0 0 0 0 0 0 0 0

4.1.5 ตัวอย่างการซ่อนข้อมูล

ในส่วนนี้จะแสดงตัวอย่างของการซ่อนข้อมูลลงในเอกสาร โดยข้อมูลที่จะซ่อนลงในเอกสารจะเป็นชุดข้อมูลที่ได้จากตัวอย่างการเตรียมข้อมูลในหัวข้อที่แล้ว โดยกำหนดให้ขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามในหน่วยของพิกเซล (ค่าสัญลักษณ์ข้อมูล) ที่ใช้ระบุค่าบิตข้อมูลที่จะซ่อนลงในเอกสารที่เหมาะสมกับเอกสารที่ใช้ในงานวิจัยนี้มีค่าพารามิเตอร์ของการปรับขนาดความกว้างของช่องว่างเท่ากับ 0.4 (α) ซึ่งค่านี้ได้มาจากการทดลองเบื้องต้นเกี่ยวกับการหาค่าสัญลักษณ์ข้อมูลที่เหมาะสมที่ใช้แทนค่าบิตข้อมูลที่แตกต่างกัน (บิตข้อมูล “1” และ “0”)

โดยไม่ทำให้บิตข้อมูลที่ซ่อนสูญหายได้ง่ายและไม่ทำให้เอกสารที่ซ่อนข้อมูลนี้ถูกสังเกตเห็นการเปลี่ยนแปลงได้ง่ายด้วยเช่นกัน และกำหนดให้ค่าเฉลี่ยความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามภายในเอกสารทั้งหมด (Δ) มีค่าเท่ากับ 4.5 (ค่านี้จะขึ้นอยู่กับเอกสารแต่ละฉบับ) เมื่อนำค่าทั้งสองนี้ไปแทนในสมการที่ 3.22 (ในบทที่ 3) จะได้ค่าปริมาณการปรับขนาดความกว้างของช่องว่างระหว่างระดับชั้นเท่ากับ 1.8 (0.4 x 4.5) และเมื่อนำค่านี้ไปแทนค่าในสมการที่ 3.20 และ 3.21 เพื่อหาค่าสัญลักษณ์ข้อมูลเพื่อแทนค่าบิตข้อมูล "0" และ "1" จะได้ว่าค่าสัญลักษณ์ข้อมูลที่ใช้แทนค่าบิตข้อมูล "1" จะมีค่าเท่ากับ 6 (จำนวนเต็มที่มีค่าใกล้เคียงกับ 4.5+1.8) และค่าสัญลักษณ์ที่ใช้แทนบิตข้อมูล "0" จะมีค่าเท่ากับ 3 (จำนวนเต็มที่มีค่าใกล้เคียงกับ 4.5-1.8)

พิจารณารูปที่ 4.1 แสดงตัวอย่างของเอกสารต้นฉบับที่ยังไม่ถูกซ่อนข้อมูลซึ่งเป็นเอกสารภาพภาษาไทยขนาด A4 ที่ใช้ตัวอักษรรูปแบบ AngsanaUPC ขนาด 14 พอยท์โดยมีการจัดตัวอักษรแบบชิดขอบคอลัมน์เดียว และรูปที่ 4.2 แสดงตัวอย่างของเอกสารที่ถูกซ่อนข้อมูลแล้ว ซึ่งจากการซ่อนข้อมูลลงในเอกสารนี้พบว่ามีบางบรรทัดที่ไม่สามารถใช้ซ่อนข้อมูลได้คือ บรรทัดที่ 1, 2, 14, 25, 33, 34, 35 และ 36 (บรรทัดที่ 1 และ 2 ไม่ถูกนำมาใช้ซ่อนข้อมูลเนื่องจากเป็นบรรทัดที่ถูกกำหนดไว้สำหรับซ่อนข้อมูลที่จะนำมาคำนวณค่าเธรดโชลด์ที่ใช้ระบุค่าบิตข้อมูล ส่วนบรรทัดที่ 14 และ 25 ไม่สามารถซ่อนข้อมูลได้เนื่องจากมีตำแหน่งที่สามารถซ่อนข้อมูลได้น้อยว่า 8 ตำแหน่ง และสำหรับบรรทัดที่ 33, 34, 35 และ 36 เป็นบรรทัดที่เหลือจากการซ่อนข้อมูลเนื่องจากในตัวอย่างนี้จะซ่อนข้อมูลลงเอกสารทั้งหมด 28 บรรทัดเท่านั้น) โดยบรรทัดเหล่านี้จะถูกซ่อนข้อมูล "0" สลับกับข้อมูล "1" เพื่อที่จะนำมาคำนวณหาค่าเธรดโชลด์ที่ใช้ระบุค่าบิตข้อมูล (δ_d)

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ ซึ่งวิธีการที่มีอยู่นั้นเป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาประยุกต์ใช้กับเอกสารภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่างจากภาษาอังกฤษมาก ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึงวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเอียดของเอกสารลงในเอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในการค้นหาเอกสารที่เกี่ยวข้องได้ง่าย โดยในบทความนี้จะมุ่งเน้นที่การซ่อนข้อมูลลงในสื่อประเภทข้อความเนื่องจากสื่อประเภทข้อความเป็นสื่อที่ถูกใช้กันอย่างแพร่หลายมากไม่ว่าจะเป็นหนังสือพิมพ์หรือวารสารต่างๆ เนื่องจากว่าสื่อประเภทนี้เป็นสื่อที่ทำให้เกิดความเข้าใจได้ง่ายระหว่างผู้รับกับผู้ส่ง ในปัจจุบันเทคโนโลยีทางด้านการสื่อสารมีการพัฒนาเพิ่มมากขึ้นทำให้สื่อประเภทนี้สามารถถูกแพร่กระจายได้ง่ายขึ้นโดยผ่านระบบเครือข่าย ซึ่งเอกสารที่จะถูกส่งผ่านระบบเครือข่ายนี้จะต้องอยู่ในรูปแบบอิเลคทรอนิคเท่านั้นและผลจากการที่เอกสารสามารถถูกแพร่กระจายได้ง่ายนี้เป็นสาเหตุให้เกิดการแพร่กระจายเอกสารแบบผิดกฎหมายได้ง่ายเช่นกัน ดังนั้นจึงได้มีการคิดค้นวิธีการซ่อนข้อมูลบางอย่างลงในเอกสารเพื่อใช้ในการแสดงความเป็นเจ้าของในเอกสารนั้น

สำหรับวิธีการซ่อนข้อมูลที่จะกล่าวถึงในบทความนี้จะเป็นวิธีการที่เหมาะสมกับเอกสารภาษาไทย เนื่องจากว่าวิธีการนี้จะเป็นการซ่อนข้อมูลโดยอาศัยโครงสร้างเฉพาะของภาษาไทยโดยการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นของตัวอักษร ซึ่งวิธีการนี้จะสามารถซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน ซึ่งเป็นวิธีการที่ออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษหรือภาษาอื่นที่มีโครงสร้างใกล้เคียงกับภาษาอังกฤษเท่านั้น และในส่วนท้ายของบทความนี้จะแสดงผลการทดลองการซ่อนข้อมูลลงในเอกสารภาษาไทยโดยใช้วิธีที่กล่าวมาในบทความนี้ การซ่อนข้อมูลลงในเอกสารนั้นจะมีวัตถุประสงค์ในการทำงานได้หลายแบบซึ่งนอกจากจะซ่อนข้อมูลเพื่อใช้ในการแสดงความเป็นเจ้าของแล้วอาจใช้ในการซ่อนข้อมูลความลับหรือข้อมูลที่เป็นรายละเอียดของเอกสารก็ได้ ในบทความนี้จะกล่าวถึงการซ่อนข้อมูลลงในเอกสารเพื่อวัตถุประสงค์ในการเพิ่มเติมรายละเอียดของเอกสารลงไปในเอกสารด้วย เช่น การซ่อนคำสำคัญของเอกสารเพื่อประโยชน์ในการค้นหาข้อมูลเอกสารที่เกี่ยวข้องได้ง่าย โดยเอกสารที่สามารถใช้วิธีการซ่อนข้อมูลที่กล่าวถึงในบทความนี้ได้นั้นจะต้องเป็นเอกสารที่มีการจัดเก็บแบบรูปภาพ

การซ่อนข้อมูลเป็นการฝังข้อมูลบางอย่างลงไปในตัวสื่อเพื่อวัตถุประสงค์ในการทำงานที่แตกต่างกัน ซึ่งเราสามารถแบ่งได้ 3 วัตถุประสงค์ โดยวัตถุประสงค์แรกคือการซ่อนข้อมูลที่เป็นความลับซึ่งมีจุดประสงค์เพื่อที่จะส่งข้อมูลความลับไปพร้อมกับสื่อโดยที่ไม่มีใครสามารถสังเกตเห็นได้ ดังนั้นข้อมูลที่ถูกซ่อนนี้จะต้องมีความปลอดภัยสูงและถูกตรวจพบได้ยาก สำหรับวัตถุประสงค์ที่สองคือการซ่อนข้อมูลเพื่อใช้แสดงความเป็นเจ้าของ มีจุดประสงค์เพื่อป้องกันการละเมิดลิขสิทธิ์หรือใช้ในการพิสูจน์ความเป็นเจ้าของในเอกสารนั้น ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือข้อมูลที่ถูกซ่อนจะต้องมีความคงทนสูงต่อกระบวนการต่างๆที่กระทำต่อตัวเอกสาร เช่นการพิมพ์ การถ่ายเอกสาร หรือการสแกน อีกทั้งข้อมูลที่ซ่อนอยู่จะต้องไม่สามารถถูกลบออกไปได้ สำหรับวัตถุประสงค์สุดท้ายคือ การซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสาร ซึ่งมีจุดประสงค์เพื่อเพิ่มเติมรายละเอียดของเอกสารลงไปด้วย เช่น ชื่อเจ้าของเอกสารหรือคำสำคัญของเอกสาร ข้อมูลที่ถูกซ่อนนี้ควรจะถูกตรวจพบได้ง่ายเนื่องจากไม่ใช่ข้อมูลที่เป็นความลับ ดังนั้นจึงสามารถดึงข้อมูลเหล่านี้ออกจากเอกสารได้ง่ายกว่าการซ่อนข้อมูลประเภทอื่นๆ

รูปที่ 4.1 แสดงตัวอย่างเอกสารภาษาไทยที่นำมาซ่อนข้อมูล

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ ซึ่งวิธีการที่มีอยู่นั้นเป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาประยุกต์ใช้กับเอกสารภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่างจากภาษาอังกฤษมาก ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึงวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเอียดของเอกสารลงในเอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในการค้นหาเอกสารที่เกี่ยวข้องได้ง่าย โดยในบทความนี้จะมุ่งเน้นที่การซ่อนข้อมูลลงในสื่อประเภทข้อความเนื่องจากสื่อประเภทข้อความเป็นสื่อที่ถูกใช้กันอย่างแพร่หลายมากไม่ว่าจะเป็นหนังสือพิมพ์หรือวารสารต่างๆ เนื่องจากว่าสื่อประเภทนี้เป็นสื่อที่ทำให้เกิดความเข้าใจได้ง่ายระหว่างผู้รับกับผู้ส่ง ในปัจจุบันเทคโนโลยีทางด้านการสื่อสารมีการพัฒนาเพิ่มมากขึ้นทำให้สื่อประเภทนี้สามารถถูกแพร่กระจายได้ง่ายขึ้นโดยผ่านระบบเครือข่าย ซึ่งเอกสารที่จะถูกส่งผ่านระบบเครือข่ายนี้จะต้องอยู่ในรูปแบบอิเลคทรอนิคเท่านั้นและผลจากการที่เอกสารสามารถถูกแพร่กระจายได้ง่ายนี้เป็นสาเหตุให้เกิดการแพร่กระจายเอกสารแบบผิดกฎหมายได้ง่ายเช่นกัน ดังนั้นจึงได้มีการคิดค้นวิธีการซ่อนข้อมูลบางอย่างลงในเอกสารเพื่อใช้ในการแสดงความเป็นเจ้าของในเอกสารนั้น

สำหรับวิธีการซ่อนข้อมูลที่จะกล่าวถึงในบทความนี้จะเป็นวิธีการที่เหมาะสมกับเอกสารภาษาไทย เนื่องจากว่าวิธีการนี้จะเป็นการซ่อนข้อมูลโดยอาศัยโครงสร้างเฉพาะของภาษาไทยโดยการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นของตัวอักษร ซึ่งวิธีการนี้จะสามารถซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน ซึ่งเป็นวิธีการที่ออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษหรือภาษาอื่นที่มีโครงสร้างใกล้เคียงกับภาษาอังกฤษเท่านั้น และในส่วนท้ายของบทความนี้จะแสดงผลการทดลองการซ่อนข้อมูลลงในเอกสารภาษาไทยโดยใช้วิธีที่กล่าวมาในบทความนี้ การซ่อนข้อมูลลงในเอกสารนั้นจะมีวัตถุประสงค์ในการทำงานได้หลายแบบซึ่งนอกจากจะซ่อนข้อมูลเพื่อใช้ในการแสดงความเป็นเจ้าของแล้วอาจใช้ในการซ่อนข้อมูลความลับหรือข้อมูลที่เป็นรายละเอียดของเอกสารก็ได้ ในบทความนี้จะกล่าวถึงการซ่อนข้อมูลลงในเอกสารเพื่อวัตถุประสงค์ในการเพิ่มเติมรายละเอียดของเอกสารลงไปในเอกสารด้วย เช่น การซ่อนคำสำคัญของเอกสารเพื่อประโยชน์ในการค้นหาข้อมูลเอกสารที่เกี่ยวข้องได้ง่ายโดยเอกสารที่สามารถใช้วิธีการซ่อนข้อมูลที่กล่าวถึงในบทความนี้ได้นั้นจะต้องเป็นเอกสารที่มีการจัดเก็บแบบรูปภาพ

การซ่อนข้อมูลเป็นการฝังข้อมูลบางอย่างลงไปในตัวสื่อเพื่อวัตถุประสงค์ในการทำงานที่แตกต่างกัน ซึ่งเราสามารถแบ่งได้ 3 วัตถุประสงค์ โดยวัตถุประสงค์แรกคือการซ่อนข้อมูลที่เป็นความลับซึ่งมีจุดประสงค์เพื่อที่จะส่งข้อมูลความลับไปพร้อมกับสื่อโดยที่ไม่มีใครสามารถสังเกตเห็นได้ ดังนั้นข้อมูลที่ถูกซ่อนนี้จะต้องมีความปลอดภัยสูงและถูกตรวจพบได้ยาก สำหรับวัตถุประสงค์ที่สองคือการซ่อนข้อมูลเพื่อใช้แสดงความเป็นเจ้าของ มีจุดประสงค์เพื่อป้องกันการละเมิดลิขสิทธิ์หรือใช้ในการพิสูจน์ความเป็นเจ้าของในเอกสารนั้น ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือข้อมูลที่ถูกซ่อนจะต้องมีความคงทนสูงต่อกระบวนการต่างๆที่กระทำต่อตัวเอกสาร เช่นการพิมพ์ การถ่ายเอกสาร หรือการสแกน อีกทั้งข้อมูลที่ซ่อนอยู่จะต้องไม่สามารถถูกลบออกไปได้ สำหรับวัตถุประสงค์สุดท้ายคือ การซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสาร ซึ่งมีจุดประสงค์เพื่อเพิ่มเติมรายละเอียดของเอกสารลงไปด้วย เช่น ชื่อเจ้าของเอกสารหรือคำสำคัญของเอกสาร ข้อมูลที่ถูกซ่อนนี้ควรจะถูกตรวจพบได้ง่ายเนื่องจากไม่ใช่ข้อมูลที่เป็นความลับ ดังนั้นจึงสามารถดึงข้อมูลเหล่านี้ออกจากเอกสารได้ง่ายกว่าการซ่อนข้อมูลประเภทอื่นๆ

รูปที่ 4.2 แสดงตัวอย่างเอกสารภาษาไทยที่ถูกซ่อนข้อมูลแล้ว

4.1.6 ตัวอย่างการดึงข้อมูล

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ ซึ่งวิธีการที่มีอยู่นั้นเป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาประยุกต์ใช้กับเอกสารภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่างจากภาษาอังกฤษมาก ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึงวิธีการซ่อนข้อมูลที่เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเอียดของเอกสารลงในเอกสาร เช่น การซ่อนคำสำคัญของเอกสารลงในเอกสารเพื่อความสะดวกในการค้นหาเอกสารที่เกี่ยวข้องได้ง่าย โดยในบทความนี้จะมุ่งเน้นที่การซ่อนข้อมูลลงในสื่อประเภทข้อความเนื่องจากสื่อประเภทข้อความเป็นสื่อที่ถูกใช้กันอย่างแพร่หลายมากไม่ว่าจะเป็นหนังสือพิมพ์หรือวารสารต่างๆ เนื่องจากว่าสื่อประเภทนี้เป็นสื่อที่ทำให้เกิดความเข้าใจได้ง่ายระหว่างผู้รับกับผู้ส่ง ในปัจจุบันเทคโนโลยีทางด้านการสื่อสารมีการพัฒนาเพิ่มมากขึ้นทำให้สื่อประเภทนี้สามารถถูกแพร่กระจายได้ง่ายขึ้นโดยผ่านระบบเครือข่าย ซึ่งเอกสารที่จะถูกส่งผ่านระบบเครือข่ายนี้จะต้องอยู่ในรูปแบบอิเลคทรอนิคเท่านั้นและผลจากการที่เอกสารสามารถถูกแพร่กระจายได้ง่ายนี้เป็นสาเหตุให้เกิดการแพร่กระจายเอกสารแบบผิดกฎหมายได้ง่ายเช่นกัน ดังนั้นจึงได้มีการคิดค้นวิธีการซ่อนข้อมูลบางอย่างลงในเอกสารเพื่อใช้ในการแสดงความเป็นเจ้าของในเอกสารนั้น

สำหรับวิธีการซ่อนข้อมูลที่จะกล่าวถึงในบทความนี้จะเป็นวิธีการที่เหมาะสมกับเอกสารภาษาไทย เนื่องจากว่าวิธีการนี้จะเป็นการซ่อนข้อมูลโดยอาศัยโครงสร้างเฉพาะของภาษาไทยโดยการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นของตัวอักษร ซึ่งวิธีการนี้จะสามารถซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการที่ใช้อยู่ในปัจจุบัน ซึ่งเป็นวิธีการที่ออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษหรือภาษาอื่นที่มีโครงสร้างใกล้เคียงกับภาษาอังกฤษเท่านั้น และในส่วนท้ายของบทความนี้จะแสดงผลการทดลองการซ่อนข้อมูลลงในเอกสารภาษาไทยโดยใช้วิธีที่กล่าวมาในบทความนี้ การซ่อนข้อมูลลงในเอกสารนั้นจะมีวัตถุประสงค์ในการทำงานได้หลายแบบซึ่งนอกจากจะซ่อนข้อมูลเพื่อใช้ในการแสดงความเป็นเจ้าของแล้วอาจใช้ในการซ่อนข้อมูลความลับหรือข้อมูลที่เป็นรายละเอียดของเอกสารก็ได้ ในบทความนี้จะกล่าวถึงการซ่อนข้อมูลลงในเอกสารเพื่อวัตถุประสงค์ในการเพิ่มเติมรายละเอียดของเอกสารลงไปในเอกสารด้วย เช่น การซ่อนคำสำคัญของเอกสารเพื่อประโยชน์ในการค้นหาข้อมูลเอกสารที่เกี่ยวข้องได้ง่ายโดยเอกสารที่สามารถใช้วิธีการซ่อนข้อมูลที่กล่าวถึงในบทความนี้ได้นั้นจะต้องเป็นเอกสารที่มีการจัดเก็บแบบรูปภาพ

การซ่อนข้อมูลเป็นการฝังข้อมูลบางอย่างลงไปในตัวสื่อเพื่อวัตถุประสงค์ในการทำงานที่แตกต่างกัน ซึ่งเราสามารถแบ่งได้ 3 วัตถุประสงค์ โดยวัตถุประสงค์แรกคือการซ่อนข้อมูลที่เป็นความลับซึ่งมีจุดประสงค์เพื่อที่จะส่งข้อมูลความลับไปพร้อมกับสื่อโดยที่ไม่มีใครสามารถสังเกตเห็นได้ ดังนั้นข้อมูลที่ถูกซ่อนนี้จะต้องมีความปลอดภัยสูงและถูกตรวจพบได้ยาก สำหรับวัตถุประสงค์ที่สองคือการซ่อนข้อมูลเพื่อใช้แสดงความเป็นเจ้าของ มีจุดประสงค์เพื่อป้องกันการละเมิดลิขสิทธิ์หรือใช้ในการพิสูจน์ความเป็นเจ้าของในเอกสารนั้น ซึ่งคุณสมบัติที่สำคัญของการซ่อนข้อมูลในลักษณะนี้คือข้อมูลที่ถูกซ่อนจะต้องมีความคงทนสูงต่อกระบวนการต่างๆที่กระทำต่อตัวเอกสาร เช่นการพิมพ์ การถ่ายเอกสาร หรือการสแกน อีกทั้งข้อมูลที่ซ่อนอยู่จะต้องไม่สามารถถูกลบออกไปได้ สำหรับวัตถุประสงค์สุดท้ายคือ การซ่อนข้อมูลเพื่อเพิ่มเติมรายละเอียดของเอกสาร ซึ่งมีจุดประสงค์เพื่อเพิ่มเติมรายละเอียดของเอกสารลงไปด้วย เช่น ชื่อเจ้าของเอกสารหรือคำสำคัญของเอกสาร ข้อมูลที่ถูกซ่อนนี้ควรจะถูกตรวจพบได้ง่ายเนื่องจากไม่ใช่ข้อมูลที่เป็นความลับ ดังนั้นจึงสามารถดึงข้อมูลเหล่านี้ออกจากเอกสารได้ง่ายกว่าการซ่อนข้อมูลประเภทอื่นๆ

รูปที่ 4.3 ตัวอย่างเอกสารที่นำมาดึงข้อมูล (เอกสารสำเนาที่ 3)

ในส่วนนี้จะแสดงตัวอย่างการดึงข้อมูลออกจากเอกสาร โดยเอกสารที่นำมาใช้เป็นตัวอย่างนี้เป็นเอกสารสำเนาที่ 3 (ดูรูปที่ 4.3) โดยมีรายละเอียดของการดึงข้อมูลดังนี้

1) ค่าสัญลักษณ์ข้อมูลที่ดึงออกจากเอกสาร (ขนาดของช่องว่างระหว่างระดับชั้นของแต่ละตำแหน่ง) ของแต่ละบรรทัดคือ

แถวที่ 1 : 3 6 3 6 4 6 3 6 3 5 4 7 3 6 3
แถวที่ 2 : 2 6 3 5 3 6 3 6 3 5 2 6 0 0 0
แถวที่ 3 : 2 3 3 2 6 5 6 5 0 0 0 0 0 0 0
แถวที่ 4 : 3 2 3 3 3 3 3 3 0 0 0 0 0 0 0
แถวที่ 5 : 7 7 7 7 4 4 3 3 0 0 0 0 0 0 0
แถวที่ 6 : 7 7 7 6 7 7 7 7 0 0 0 0 0 0 0
แถวที่ 7 : 6 7 6 6 3 3 4 3 0 0 0 0 0 0 0
แถวที่ 8 : 3 3 3 4 3 3 3 3 0 0 0 0 0 0 0
แถวที่ 9 : 3 2 3 3 6 6 6 7 0 0 0 0 0 0 0
แถวที่ 10 : 6 6 5 6 7 6 6 6 0 0 0 0 0 0 0
แถวที่ 11 : 8 6 7 6 4 3 3 4 0 0 0 0 0 0 0
แถวที่ 12 : 3 4 4 4 6 7 7 7 0 0 0 0 0 0 0
แถวที่ 13 : 7 6 6 6 6 6 6 6 0 0 0 0 0 0 0
แถวที่ 14 : 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0
แถวที่ 15 : 6 6 6 4 4 3 3 3 0 0 0 0 0 0 0
แถวที่ 16 : 4 7 4 7 4 6 3 0 0 0 0 0 0 0 0
แถวที่ 17 : 3 3 4 3 7 7 7 7 0 0 0 0 0 0 0
แถวที่ 18 : 4 3 4 4 4 4 3 3 0 0 0 0 0 0 0
แถวที่ 19 : 4 7 4 7 4 0 0 0 0 0 0 0 0 0 0
แถวที่ 20 : 6 6 7 7 3 3 4 4 0 0 0 0 0 0 0
แถวที่ 21 : 4 4 4 3 4 4 3 3 0 0 0 0 0 0 0
แถวที่ 22 : 3 3 3 3 6 7 6 5 0 0 0 0 0 0 0
แถวที่ 23 : 4 4 3 4 5 4 4 0 0 0 0 0 0 0 0
แถวที่ 24 : 6 6 7 7 4 3 4 4 0 0 0 0 0 0 0
แถวที่ 25 : 4 7 4 7 0 0 0 0 0 0 0 0 0 0 0
แถวที่ 26 : 6 7 7 6 7 7 7 4 0 0 0 0 0 0 0
แถวที่ 27 : 3 3 2 6 6 7 6 0 0 0 0 0 0 0 0
แถวที่ 28 : 6 6 6 7 8 7 7 7 0 0 0 0 0 0 0

แถวที่ 29 : 3 3 3 6 7 7 7 4 0 0 0 0 0 0 0

แถวที่ 30 : 3 4 3 4 7 7 7 6 0 0 0 0 0 0 0

แถวที่ 31 : 3 3 3 3 3 3 3 3 0 0 0 0 0 0 0

แถวที่ 32 : 6 7 7 6 4 3 3 4 0 0 0 0 0 0 0

แถวที่ 33 : 6 6 6 5 3 3 3 3 0 0 0 0 0 0 0

แถวที่ 34 : 4 3 3 3 3 3 3 3 0 0 0 0 0 0 0

แถวที่ 35 : 3 7 4 6 3 6 4 6 0 0 0 0 0 0 0

แถวที่ 36 : 3 6 0 0 0 0 0 0 0 0 0 0 0 0 0

2) เลือกแถวข้อมูลที่คาดว่าจะเป็นแถวที่ถูกใช้ซ่อนข้อมูลและจัดเก็บข้อมูลในแถวนั้นเพียงแค่ 8 ตำแหน่งเท่านั้น (ในขั้นตอนของการซ่อนข้อมูลจะซ่อนข้อมูลเพียงบรรทัดละ 8 ตำแหน่งเท่านั้น) สำหรับค่าสัญลักษณ์ข้อมูลในแถวที่ไม่ได้ถูกซ่อนข้อมูลจะถูกนำมาคำนวณค่าเธรดโชลด์ที่ใช้ระบุค่าบิตข้อมูล (δ_d) โดยมีรายละเอียดของข้อมูลทั้งสองชุดดังนี้

2.1) ค่าสัญลักษณ์ข้อมูลของแถวที่ถูกใช้ซ่อนข้อมูล

แถวที่ 3 : 2 3 3 2 6 5 6 5

แถวที่ 4 : 3 2 3 3 3 3 3 3

แถวที่ 5 : 7 7 7 7 4 4 3 3

แถวที่ 6 : 7 7 7 6 7 7 7 7

แถวที่ 7 : 6 7 6 6 3 3 4 3

แถวที่ 8 : 3 3 3 4 3 3 3 3

แถวที่ 9 : 3 2 3 3 6 6 6 7

แถวที่ 10 : 6 6 5 6 7 6 6 6

แถวที่ 11 : 8 6 7 6 4 3 3 4

แถวที่ 12 : 3 4 4 4 6 7 7 7

แถวที่ 13 : 7 6 6 6 6 6 6 6

แถวที่ 15 : 6 6 6 4 4 3 3 3

แถวที่ 17 : 3 3 4 3 7 7 7 7

แถวที่ 18 : 4 3 4 4 4 4 3 3

แถวที่ 20 : 6 6 7 7 3 3 4 4

แถวที่ 21 : 4 4 4 3 4 4 3 3

แถวที่ 22 : 3 3 3 3 6 7 6 5

แถวที่ 23 : 4 4 3 4 5 4 4 0

แถวที่ 24 : 6 6 7 7 4 3 4 4

แถวที่ 26 : 6 7 7 6 7 7 7 4

แถวที่ 27 : 3 3 3 2 6 6 7 6

แถวที่ 28 : 6 6 6 7 8 7 7 7

แถวที่ 29 : 3 3 3 6 7 7 7 4

แถวที่ 30 : 3 4 3 4 7 7 7 6

แถวที่ 31 : 3 3 3 3 3 3 3 3

แถวที่ 32 : 6 7 7 6 4 3 3 4

แถวที่ 33 : 6 6 6 5 3 3 3 3

แถวที่ 34 : 4 3 3 3 3 3 3 3

2.2) ค่าสัญลักษณ์ข้อมูลของแถวที่ไม่ได้ถูกซ่อนข้อมูล

แถวที่ 1 : 3 6 3 6 4 6 3 6 3 5 4 7 3 6 3

แถวที่ 2 : 2 6 3 5 3 6 3 6 3 5 2 6 0 0 0

แถวที่ 14 : 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0

แถวที่ 16 : 4 7 4 7 4 6 3 0 0 0 0 0 0 0 0

แถวที่ 19 : 4 7 4 7 4 0 0 0 0 0 0 0 0 0 0

แถวที่ 25 : 4 7 4 7 0 0 0 0 0 0 0 0 0 0 0

แถวที่ 35 : 3 7 4 6 3 6 4 6 0 0 0 0 0 0 0

แถวที่ 36 : 3 6 0 0 0 0 0 0 0 0 0 0 0 0 0

2.3) ค่าเธรดโชลด์ (δ_d) ที่คำนวณได้ข้อมูลในข้อ 2.2 คือ 4.8519

3) นำข้อมูลทั้งหมดของแถวที่ถูกใช้ซ่อนข้อมูลจริงมาผ่านกระบวนการจัดกลุ่มข้อมูล และทำการระบุค่าบิตข้อมูลของข้อมูลแต่ละกลุ่มโดยใช้ค่าเธรดโชลด์ (δ_d) ที่ได้จากขั้นตอนที่แล้ว โดยมีรายละเอียดของข้อมูลดังนี้

3.1) รายละเอียดของการแบ่งกลุ่มข้อมูล

แถวที่ 1 : (2 3 3 2) (6 5 6 5)

แถวที่ 2 : (3 2) (3 3 3 3)

แถวที่ 3 : (7 7 7 7) (4 4 3 3)

แถวที่ 4 : (7 7 7 6) (7 7 7 7)

แถวที่ 5 : (6 7 6 6) (3 3 4 3)

แถวที่ 6 : (3 3 3 4) (3 3 3 3)

แถวที่ 7 : (3 2 3 3) (6 6 6 7)

แถวที่ 8 : (6 6 5 6) (7 6 6 6)

แถวที่ 9 : (8 6 7 6) (4 3 3 4)

แถวที่ 10 : (3 4 4 4) (6 7 7 7)

แถวที่ 11 : (7 6) (6 6 6 6)

แถวที่ 12 : (6 6 6) (4 4 3 3)

แถวที่ 13 : (3 3 4 3) (7 7 7 7)

แถวที่ 14 : (4 3) (4 4 4 4)

แถวที่ 15 : (6 6 7 7) (3 3 4 4)

แถวที่ 16 : (4 4 4 3) (4 4 3 3)

แถวที่ 17 : (3 3 3 3) (6 7 6 5)

แถวที่ 18 : (4 4 3 4) (5 4 4)

แถวที่ 19 : (6 6 7 7) (4 3 4 4)

แถวที่ 20 : (6 7 7 6) (7 7 7 4)

แถวที่ 21 : (3 3 3 2) (6 6 7 6)

แถวที่ 22 : (6 6 6 7) (8 7 7 7)

แถวที่ 23 : (3 3 3) (6 7 7 7)

แถวที่ 24 : (3 4 3 4) (7 7 7 6)

แถวที่ 25 : (3 3 3 3) (3 3 3 3)

แถวที่ 26 : (6 7 7 6) (4 3 3 4)

แถวที่ 27 : (6 6 6 5) (3 3 3 3)

แถวที่ 28 : (4 3) (3 3 3 3)

3.2) รายละเอียดของค่าบิตข้อมูลในแต่ละกลุ่ม

แถวที่ 1 : 0 1

แถวที่ 2 : 0 0

แถวที่ 3 : 1 0

แถวที่ 4 : 1 1

แถวที่ 5 : 1 0

แถวที่ 6 : 0 0

แถวที่ 7 : 0 1

แถวที่ 8 : 1 1

แถวที่ 9 : 1 0
แถวที่ 10 : 0 1
แถวที่ 11 : 1 1
แถวที่ 12 : 1 0
แถวที่ 13 : 0 1
แถวที่ 14 : 0 0
แถวที่ 15 : 1 0
แถวที่ 16 : 0 0
แถวที่ 17 : 0 1
แถวที่ 18 : 0 0
แถวที่ 19 : 1 0
แถวที่ 20 : 1 1
แถวที่ 21 : 0 1
แถวที่ 22 : 1 1
แถวที่ 23 : 0 1
แถวที่ 24 : 0 1
แถวที่ 25 : 0 0
แถวที่ 26 : 1 0
แถวที่ 27 : 1 0
แถวที่ 28 : 0 0

4) จากนั้นนำข้อมูลที่ได้จากขั้นตอนที่แล้วมาผ่านกระบวนการตรวจสอบและแก้ไขความผิดพลาดของข้อมูลโดยจะนำข้อมูลเข้าไปตรวจสอบทีละ 7 แถว ซึ่งมีรายละเอียดดังนี้

4.1) ข้อมูลชุดที่ 1

แถวที่ 1 : 0 1 ┐
แถวที่ 2 : 0 0 ├ บิตข้อมูล
แถวที่ 3 : 1 0 ├
แถวที่ 4 : 1 1 ┘
แถวที่ 5 : 1 0 ┐
แถวที่ 6 : 0 0 ├ รีดันแดนซี
แถวที่ 7 : 0 1 ┘

4.2) ข้อมูลชุดที่ 2

แถวที่ 8	: 1	1	บิตข้อมูล
แถวที่ 9	: 1	0	
แถวที่ 10	: 0	1	
แถวที่ 11	: 1	1	
แถวที่ 12	: 1	0	รีดันแดนซี
แถวที่ 13	: 0	1	
แถวที่ 14	: 0	0	

4.3) ข้อมูลชุดที่ 3

แถวที่ 15	: 1	0	บิตข้อมูล
แถวที่ 16	: 0	0	
แถวที่ 17	: 0	1	
แถวที่ 18	: 0	0	
แถวที่ 19	: 1	0	รีดันแดนซี
แถวที่ 20	: 1	1	
แถวที่ 21	: 0	1	

4.4) ข้อมูลชุดที่ 4

แถวที่ 22	: 1	1	บิตข้อมูล
แถวที่ 23	: 0	1	
แถวที่ 24	: 0	1	
แถวที่ 25	: 0	0	
แถวที่ 26	: 1	0	รีดันแดนซี
แถวที่ 27	: 1	0	
แถวที่ 28	: 0	0	

5) จัดเรียงข้อมูลที่ได้จากขั้นตอนที่ 4 (เฉพาะข้อมูล 4 แถวบนของแต่ละชุดข้อมูลรหัสแฮมมิง) เพื่อนำไปใช้ตรวจสอบความถูกต้องของเอกสาร โดยจะใช้ข้อมูลเพียงแค่ 30 บิตแรกเท่านั้น ซึ่งจะได้ชุดข้อมูลดังนี้ "01001011111001111000010011010 1"

4.1.7 หลักการตรวจสอบความถูกต้องของเอกสาร

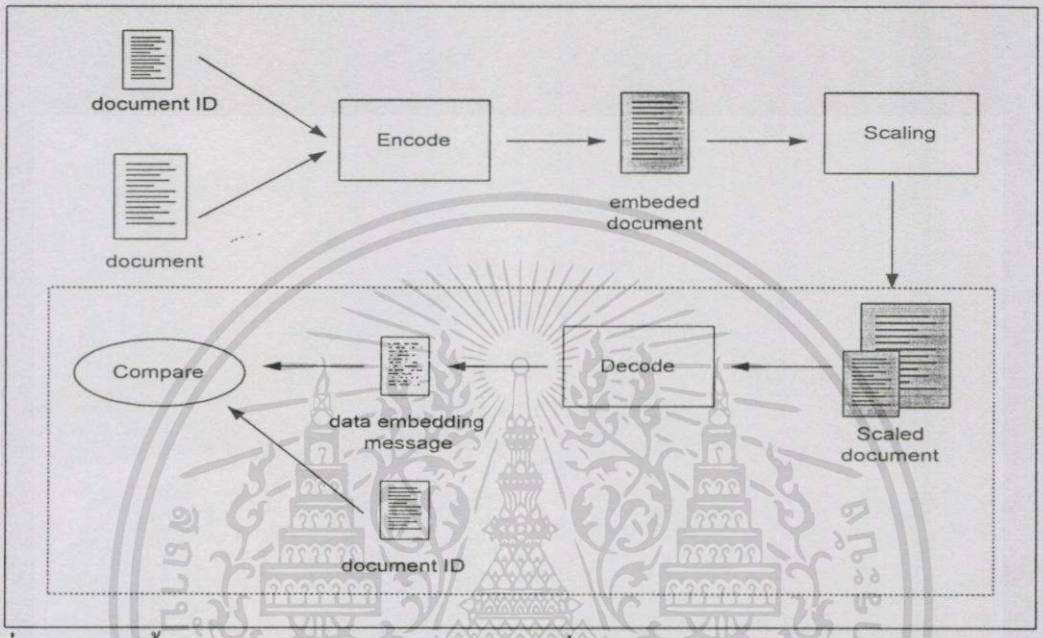
จากที่กล่าวมาแล้วว่าหมายเลขประจำเอกสารที่จะซ่อนลงในเอกสารแต่ละฉบับนั้นเป็นตัวเลขที่ถูกสร้างขึ้นมาจากการสุ่มตัวเลขฐานสิบจำนวน 6 หลัก ดังนั้นเราสามารถที่จะสร้างชุดตัวเลขที่แตกต่างกันได้ถึง 10^6 ชุดเพื่อที่นำมาใช้เป็นหมายเลขประจำเอกสารแต่ละฉบับซึ่งถือว่ามีจำนวนมากเพียงพอสำหรับจำนวนเอกสารที่จะถูกแพร่กระจายไปยังบุคคลอื่นภายในองค์กรหนึ่งๆ ได้

สำหรับในขั้นตอนของการตรวจสอบความถูกต้องของเอกสารนั้นจะนำชุดข้อมูลไบนารีที่ได้จากขั้นตอนการดึงข้อมูลออกจากเอกสารแต่ละฉบับมาทำการเปรียบเทียบกับหมายเลขประจำเอกสารแต่ละฉบับ (ในรูปแบบข้อมูลไบนารี) เพื่อตรวจสอบว่าเอกสารฉบับนั้นเป็นเอกสารที่ถูกต้องหรือไม่ (เป็นเอกสารที่ส่งมาจากเจ้าของที่ถูกต้องและไม่ถูกเปลี่ยนแปลงแก้ไขมาก่อน) สำหรับผลความถูกต้องของเอกสารที่นำมาตรวจสอบนี้จะพิจารณาจากความแตกต่างของบิตข้อมูลที่ได้จากขั้นตอนการดึงข้อมูลกับหมายเลขประจำเอกสารที่ถูกกำหนดไว้สำหรับเอกสารแต่ละฉบับว่ามีมากน้อยเพียงใด ซึ่งผลของความแตกต่างที่จะนำมาใช้ตัดสินว่าเอกสารฉบับใดเป็นเอกสารที่ไม่ถูกต้องนั้นจะขึ้นอยู่กับลักษณะงานแต่ละประเภท ตัวอย่างเช่น เอกสารทางด้านกฎหมายอาจจะไม่ยอมให้ข้อมูลทั้งสองตัวมีความแตกต่างกันเลยก็ได้ แต่สำหรับงานเอกสารที่ต้องผ่านการประมวลผลทางด้านเอกสารก็จะอนุโลมให้ข้อมูลทั้งสองตัวมีความแตกต่างกันได้ในระดับหนึ่ง (ตามข้อตกลงของแต่ละองค์กร) โดยหลังจากที่ทำการตรวจสอบเอกสารแล้วพบว่าเอกสารฉบับนั้นเป็นเอกสารที่ไม่ถูกต้อง (เอกสารนั้นอาจเป็นเอกสารที่ไม่ถูกต้องจริงๆหรือเป็นเอกสารที่เกิดความผิดพลาดจากการประมวลผลทางด้านเอกสาร) ก็จะร้องขอไปทางผู้ส่งเอกสารให้ส่งเอกสารกลับมาใหม่อีกครั้งเพื่อยืนยันความแน่ใจในเอกสารนั้น

ถึงแม้ว่าวิธีการตรวจสอบความถูกต้องของเอกสารโดยการซ่อนข้อมูลที่เป็นหมายเลขประจำเอกสารแต่ละฉบับลงในเอกสารนั้นจะสามารถป้องกันการปลอมแปลงเอกสารได้ในระดับหนึ่งก็ตาม แต่ยังคงมีข้อจำกัดบางประการเนื่องจากหมายเลขประจำเอกสารที่จะถูกซ่อนลงในเอกสารนี้เป็นตัวเลขที่ถูกสร้างขึ้นมาโดยไม่มีความสัมพันธ์ใดๆกับเนื้อความในเอกสารทั้งสิ้น ดังนั้นหากมีผู้ไม่ประสงค์ดีทราบหมายเลขประจำเอกสารเหล่านั้นก็สามารถสร้างเอกสารปลอมขึ้นมาใหม่ได้

4.2 การทดลองดึงข้อมูลจากเอกสารที่ผ่านกระบวนการปรับขนาด

ในการทดลองนี้จะเป็นการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทยโดยจะพิจารณาถึงข้อจำกัดของการดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการปรับขนาด โดยวิธีการปรับขนาดที่นำมาใช้ในวิจัยนี้มี 3 แบบคือ Nearest neighbor, Bilinear และ Bicubic [16]



รูปที่ 4.4 แสดงขั้นตอนการดึงข้อมูลออกจากเอกสารที่ผ่านการปรับขนาด (ภายในเส้นประ)

4.2.1 ขั้นตอนการทดลอง

จากรูปที่ 4.4 แสดงขั้นตอนของการทดลองดังนี้

1) ซ่อนข้อมูลที่เตรียมไว้ลงในเอกสารทั้ง 5 หน้า

2) เมื่อซ่อนข้อมูลเรียบร้อยแล้วให้ทำการปรับขนาดเอกสารทั้งหมดด้วยอัตราส่วนที่แตกต่างกันโดยจะทำการปรับขนาดเอกสารด้วยวิธีการทั้งสามวิธีที่กล่าวมาแล้วข้างต้น สำหรับเงื่อนไขที่ใช้ในการปรับขนาดคือจะปรับขนาดเอกสารทีละ 5% โดยจะกำหนดให้อยู่ในช่วง $\pm 30\%$ ของเอกสารต้นฉบับ (70%, 75%, 80%, ..., 130%)

3) จากนั้นทดลองดึงข้อมูลที่ซ่อนออกจากเอกสารทั้งหมด (ดูรูปที่ 4.4 ในส่วนของเส้นประ)

4.2.2 ตัวอย่างเอกสารที่ผ่านกระบวนการปรับขนาด

ในส่วนนี้จะแสดงตัวอย่างบางส่วนของเอกสารที่ผ่านการปรับขนาดโดยจะพิจารณาเปรียบเทียบกับเอกสารต้นฉบับที่แสดงอยู่ในรูปที่ 4.5 โดยในรูปที่ 4.6 แสดงตัวอย่างบางส่วนของเอกสารที่ถูกขยายขนาดจากเอกสารต้นฉบับเป็น 120% และรูปที่ 4.7 แสดงตัวอย่างบางส่วนของ

เอกสารที่ถูกย่อขนาดจากเอกสารต้นฉบับเป็น 80% (รูปในตัวอย่างใช้วิธีการปรับขนาดแบบ Bilinear)

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมา ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่ ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าว เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่ม ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละ

รูปที่ 4.5 แสดงตัวอย่างบางส่วนของเอกสารต้นฉบับ (มีขนาดเท่ากับ 100%)

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัด เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิ ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภา ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบ เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลไ

รูปที่ 4.6 แสดงตัวอย่างบางส่วนของเอกสารที่ถูกขยายขนาดเป็น 120% จากเอกสารต้นฉบับ

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมา ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่าง ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึ เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มาก ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเ

รูปที่ 4.7 แสดงตัวอย่างบางส่วนของเอกสารที่ถูกย่อขนาดเป็น 80% จากเอกสารต้นฉบับ

จากรูปข้างต้นจะเห็นว่าเมื่อเอกสารถูกปรับขนาด (ย่อ/ขยาย) จะมีผลทำให้ช่องว่างระหว่างระดับที่เราใช้ในการซ่อนข้อมูลนั้นเกิดการเปลี่ยนแปลงตามไปด้วย ซึ่งอาจมีผลกระทบต่อความถูกต้องของค่าสัญลักษณ์ข้อมูลที่ซ่อนอยู่ภายในเอกสารการได้โดยจะแสดงรายละเอียดของผลการทดลองในส่วนถัดไป

4.2.3 ผลการทดลอง

สำหรับผลการดึงข้อมูลออกจากเอกสารที่อยู่ในรูปแบบอิเลคทรอนิคที่ผ่านกระบวนการปรับขนาด (ย่อ/ขยาย) ทั้ง 3 แบบ (Nearest neighbor, Bilinear และ Bicubic) โดยจะพิจารณาถึงอัตราความผิดพลาดของการดึงข้อมูลในแต่ละหน้าเอกสาร (Frame error rate) ซึ่งแสดงอยู่ในตารางที่ 4.1

ตารางที่ 4.1 แสดงผลการดึงข้อมูลออกจากเอกสารอิเลคทรอนิคต้นฉบับและเอกสารที่ผ่านกระบวนการปรับขนาดจำนวน 5 หน้า

ประเภทการปรับขนาด	ขนาดของเอกสาร (%)	จำนวนเอกสารที่เกิดความผิดพลาด / จำนวนเอกสารทั้งหมด		
		Nearest neighbor	Bilinear	Bicubic
ไม่มีการปรับขนาด (เอกสารต้นฉบับ)	100	0/5	0/5	0/5
การขยาย	105	0/5	0/5	0/5
	110	0/5	0/5	0/5
	115	0/5	0/5	0/5
	120	0/5	0/5	0/5
	125	0/5	0/5	0/5
	130	0/5	0/5	0/5
การย่อ	95	0/5	0/5	0/5
	90	0/5	0/5	0/5
	85	0/5	0/5	0/5
	80	1/5	1/5	1/5
	75	1/5	1/5	1/5
	70	2/5	2/5	2/5

จากตารางที่ 4.1 จะเห็นว่าเราสามารถดึงข้อมูลจากเอกสารอิเลคทรอนิคที่ไม่มีการปรับขนาดได้ถูกต้องทั้งหมดเนื่องจากเอกสารเหล่านี้ยังไม่ได้ผ่านการประมวลผลทางด้านเอกสารใดๆ ทั้งสิ้น และสำหรับเอกสารที่ถูกขยายขนาดตั้งแต่ 5% ถึง 30% (เอกสารขนาด 105%-130%) ไม่ว่าจะถูกปรับขนาดด้วยวิธีการใดก็ตามสามารถดึงข้อมูลออกมาได้อย่างถูกต้องทั้งหมดเช่นกัน แต่สำหรับการดึงข้อมูลจากเอกสารอิเลคทรอนิคที่ถูกย่อขนาดลงตั้งแต่ 5% ถึง 30% (เอกสารขนาด 95%-70%) พบว่าเอกสารที่ถูกย่อขนาดด้วยปริมาณ 5% ถึง 15% (เอกสารขนาด 95%-85%) สามารถดึงข้อมูลออกมาได้อย่างถูกต้องทั้งหมดแต่สำหรับเอกสารที่ถูกย่อขนาดลงตั้งแต่ 20% ขึ้นไป (เอกสารที่มีขนาดน้อยกว่าหรือเท่ากับ 80%) พบว่าความถูกต้องของการดึงข้อมูลจะลดลง โดยสามารถดึงข้อมูลออกจากเอกสารได้ถูกต้องเพียง 4 หน้าจากเอกสารทั้งหมด 5 หน้า (ไม่ว่าเอกสารจะถูกย่อขนาดด้วยวิธีการใดก็ตาม)

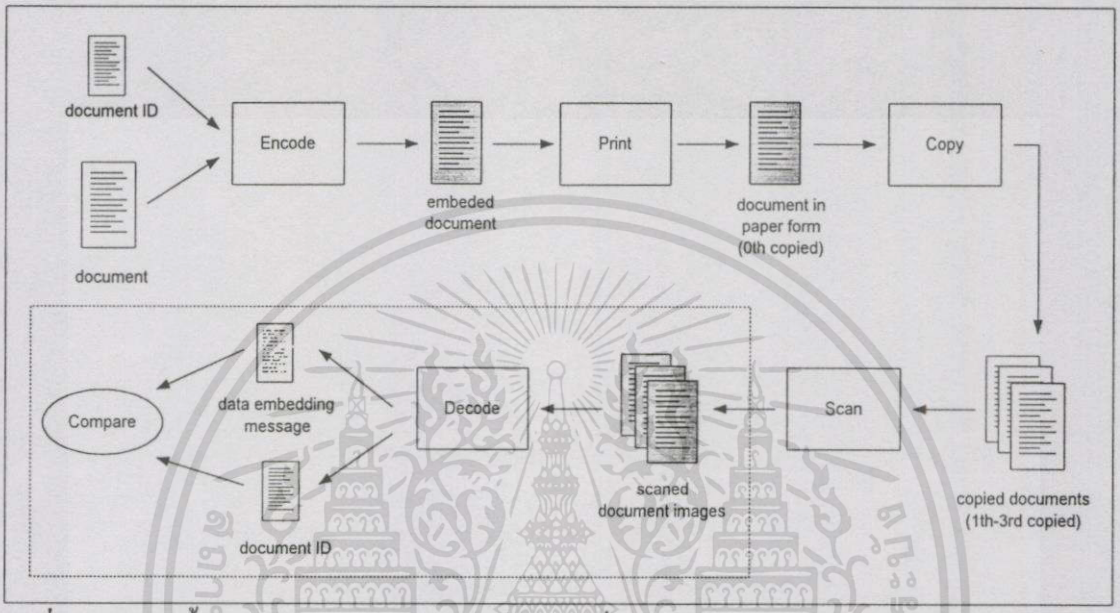
จากการวิเคราะห์ผลการทดลองพบว่าความผิดพลาดของข้อมูลที่เกิดขึ้นทั้งหมดเกิดจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาด (ในขั้นตอนของการซ่อนข้อมูลนั้นจะมีบางบรรทัดที่ถูกใช้ซ่อนข้อมูลและบางบรรทัดที่ไม่ถูกใช้ซ่อนข้อมูล) เช่น การระบุว่าบรรทัดที่ถูกใช้ซ่อนข้อมูลเป็นบรรทัดที่ไม่ได้ถูกซ่อนข้อมูล หรือการระบุว่าบรรทัดที่ไม่ได้ถูกซ่อนข้อมูลเป็นบรรทัดที่ถูกใช้ซ่อนข้อมูล ซึ่งผลที่เกิดขึ้นตามมาคือทำให้การจัดชุดข้อมูลรหัสแฮมมิงเกิดความผิดพลาดตามไปด้วย เช่น อาจจะไม่สามารถนำชุดข้อมูลเหล่านั้นมาทำการจัดชุดข้อมูลรหัสแฮมมิงได้เนื่องจากมีจำนวนบรรทัดที่จะนำมาใช้ในการจัดชุดข้อมูลไม่เพียงพอ (1 ชุดข้อมูลรหัสแฮมมิงประกอบด้วยข้อมูล 7 บรรทัด) เป็นต้น ทำให้ไม่สามารถนำชุดข้อมูลเหล่านั้นมาตรวจสอบความถูกต้องได้

สำหรับสาเหตุของการเลือกบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาดคือเมื่อเอกสารถูกปรับขนาดให้มีขนาดเล็กลงกว่าเอกสารต้นฉบับในปริมาณมากจะทำให้ช่องว่างระหว่างระดับชั้นที่ใช้ซ่อนข้อมูลนี้มีขนาดเล็กลงตามไปด้วย (บางครั้งไม่สามารถตรวจพบช่องว่างได้) ซึ่งเป็นสาเหตุให้ช่องว่างที่ใช้ซ่อนบิตข้อมูลที่แตกต่างกัน (บิตข้อมูล “0” และ “1”) มีความแตกต่างกันน้อยยิ่งขึ้นหรืออาจทำให้ช่องว่างที่ใช้ซ่อนข้อมูลบางตำแหน่งสูญหายไปได้เป็นผลให้ไม่สามารถแยกได้ว่าบรรทัดใดเป็นบรรทัดที่ถูกใช้ซ่อนข้อมูลหรือบรรทัดใดเป็นบรรทัดที่ไม่ถูกใช้ซ่อนข้อมูล

อย่างไรก็ตามวิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้สามารถทำงานได้อย่างมีประสิทธิภาพกับเอกสารที่ผ่านกระบวนการปรับขนาดแบบขยายได้ถึง 30 % และเอกสารที่ถูกย่อขนาดลงได้ถึง 15% (ภายใต้เงื่อนไขที่กำหนดขึ้นในการทดลองนี้)

4.3 การทดลองดึงข้อมูลออกจากเอกสารที่ผ่านการถ่ายเอกสาร

การทดลองนี้เป็นการทดลองซ่อนข้อมูลลงในเอกสารภาพภาษาไทยเพื่อพิจารณาถึงข้อจำกัดของการดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการถ่ายเอกสาร ซึ่งแสดงขั้นตอนการทดลองในรูปที่ 4.8



รูปที่ 4.8 แสดงขั้นตอนการดึงข้อมูลออกจากเอกสารที่ผ่านการถ่ายเอกสาร (ภายในเส้นประ)

4.3.1 ขั้นตอนการทดลอง

1) ทำการซ่อนข้อมูลที่เตรียมไว้ลงในเอกสารทั้ง 5 หน้า

2) พิมพ์เอกสารที่ซ่อนข้อมูลเรียบร้อยแล้วด้วยเครื่องพิมพ์โดยกำหนดค่าพารามิเตอร์ต่างๆของการพิมพ์ตามหัวข้อที่ 4.1.1

3) นำเอกสารที่ได้จากขั้นตอนที่ 2 ไปทำสำเนาเอกสารโดยการถ่ายเอกสารซึ่งในแต่ละหน้าของเอกสารจะถูกทำสำเนาทั้งหมด 4 ครั้ง โดยเอกสารแต่ละสำเนาจะถูกสร้างซ้ำทั้งหมด 10 ครั้งเพื่อทดสอบความถูกต้องของการทดลองโดยมีกำหนดให้

- เอกสารสำเนาที่ 0 แทนเอกสารที่ได้จากการพิมพ์ด้วยเครื่องพิมพ์
- เอกสารสำเนาที่ 1 แทนเอกสารที่ได้จากการทำสำเนาเอกสารสำเนาที่ 0
- เอกสารสำเนาที่ 2 แทนเอกสารที่ได้จากการทำสำเนาเอกสารสำเนาที่ 1
- เอกสารสำเนาที่ 3 แทนเอกสารที่ได้จากการทำสำเนาเอกสารสำเนาที่ 2
- เอกสารสำเนาที่ 4 แทนเอกสารที่ได้จากการทำสำเนาเอกสารสำเนาที่ 3

4) นำเอกสารทุกสำเนามาสแกนกลับให้อยู่ในรูปแบบอิเลคทรอนิคโดยกำหนดค่าพารามิเตอร์ต่างๆของการสแกนตามหัวข้อที่ 4.1.1

5) นำเอกสารที่ได้จากการสแกนมาผ่านกระบวนการกำจัดสิ่งรบกวนและปรับความเอียงของเอกสารเพื่อกำจัดจุดดำเล็กๆที่เกิดขึ้นภายในเอกสารที่คาดว่าจะไม่ใช่ส่วนประกอบของตัวอักษรและในกรณีที่เอกสารถูกทำให้เอียงไปจากเดิมก็จะทำการปรับความเอียงของเอกสารให้มีลักษณะเหมือนเดิม

6) จากนั้นทำการดึงข้อมูลที่ซ่อนอยู่ในเอกสารออกมาทั้งหมด

<u>หมายเหตุ</u> อนึ่งเพื่อเป็นการจำกัดปัจจัยที่มีผลต่อการทดลองเมื่อเอกสารนั้นถูกทำให้เอียงไปจากเดิม ซึ่งกระทำโดยการขีดเส้นตรงแนวนอนที่ส่วนล่างของเอกสารเพื่อใช้เป็นเครื่องมือในการปรับความเอียงของเอกสาร โดยนำวิธีการ Least Mean Square มาประยุกต์ใช้ [17]

4.3.2 ตัวอย่างเอกสารที่ผ่านกระบวนการถ่ายเอกสาร

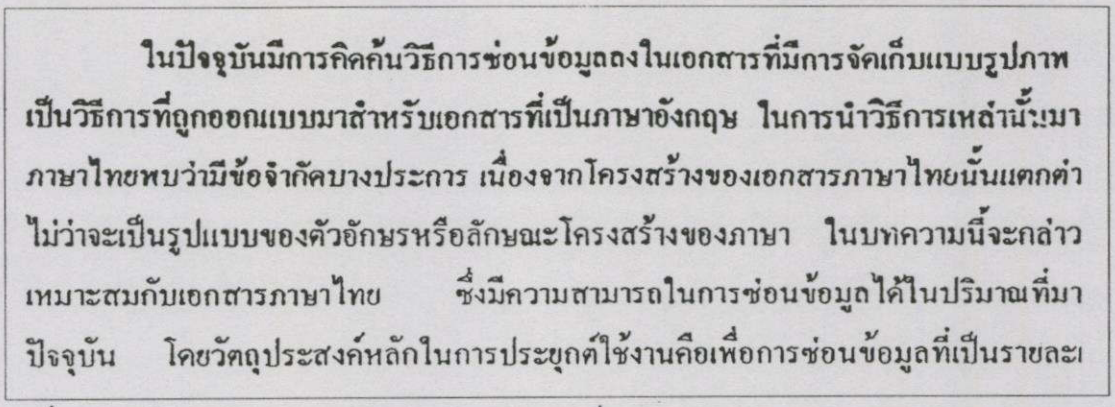
ในส่วนนี้จะแสดงตัวอย่างบางส่วนของเอกสารที่ผ่านกระบวนการถ่ายเอกสารและนำมาใช้ในการทดลองนี้ โดยรูปที่ 4.9–4.12 แสดงตัวอย่างบางส่วนของเอกสารสำเนาที่ 1, 2, 3 และ 4 ตามลำดับ

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาใช้กับภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่างกับภาษาอังกฤษ ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึงวิธีการที่เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการในปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเ

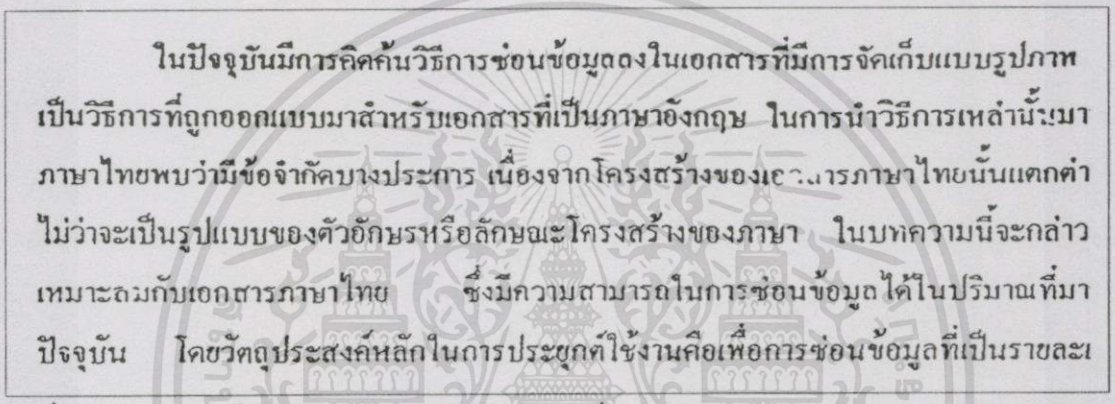
รูปที่ 4.9 แสดงตัวอย่างบางส่วนของเอกสารสำเนาที่ 1

ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมาใช้กับภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่างกับภาษาอังกฤษ ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าวถึงวิธีการที่เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มากกว่าวิธีการในปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละ

รูปที่ 4.10 แสดงตัวอย่างบางส่วนของเอกสารสำเนาที่ 2


ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ
เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมา
ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่า
ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าว
เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มา
ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเ

รูปที่ 4.11 แสดงตัวอย่างบางส่วนของเอกสารสำเนาที่ 3


ในปัจจุบันมีการคิดค้นวิธีการซ่อนข้อมูลลงในเอกสารที่มีการจัดเก็บแบบรูปภาพ
เป็นวิธีการที่ถูกออกแบบมาสำหรับเอกสารที่เป็นภาษาอังกฤษ ในการนำวิธีการเหล่านั้นมา
ภาษาไทยพบว่ามีข้อจำกัดบางประการ เนื่องจากโครงสร้างของเอกสารภาษาไทยนั้นแตกต่า
ไม่ว่าจะเป็นรูปแบบของตัวอักษรหรือลักษณะโครงสร้างของภาษา ในบทความนี้จะกล่าว
เหมาะสมกับเอกสารภาษาไทย ซึ่งมีความสามารถในการซ่อนข้อมูลได้ในปริมาณที่มา
ปัจจุบัน โดยวัตถุประสงค์หลักในการประยุกต์ใช้งานคือเพื่อการซ่อนข้อมูลที่เป็นรายละเ

รูปที่ 4.12 แสดงตัวอย่างบางส่วนของเอกสารสำเนาที่ 4

4.3.3 ผลการทดลอง

ผลการดึงข้อมูลออกจากเอกสารที่ผ่านการทำสำเนาโดยการถ่ายเอกสาร (ตั้งแต่เอกสารสำเนาที่ 1 จนถึงเอกสารสำเนาที่ 4) จะพิจารณาแยกออกเป็น 3 กรณี โดยกรณีแรกจะพิจารณาถึงอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละหน้าเอกสาร (Frame error rate) ซึ่งแสดงผลการทดลองในตารางที่ 4.2 ส่วนในกรณีที่สองจะพิจารณาถึงอัตราการความผิดพลาดของข้อมูลในแต่ละชุดข้อมูลรหัสแฮมมิง (Block error rate) ซึ่งแสดงผลในตารางที่ 4.3 รวมทั้งแสดงจำนวนชุดข้อมูลรหัสแฮมมิงที่สามารถตรวจสอบและแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ในตารางที่ 4.4 และกรณีสุดท้ายที่จะพิจารณาถึงในการทดลองนี้คืออัตราการเกิดความผิดพลาดของข้อมูลในแต่ละบิต (Bit error rate) ที่อยู่ในชุดข้อมูลรหัสแฮมมิงทั้งหมดโดยจะแสดงผลในตารางที่ 4.5

ตารางที่ 4.2 แสดงผลการดึงข้อมูลออกจากเอกสารที่ผ่านการถ่ายเอกสารตั้งแต่เอกสารสำเนาที่ 1 จนถึงเอกสารสำเนาที่ 4 โดยพิจารณาถึงอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละหน้าเอกสาร

หน้าที่	อัตราการเกิดความผิดพลาดของเอกสารในแต่ละฉบับ (จำนวนเอกสารที่เกิดความผิดพลาด / จำนวนเอกสารทั้งหมด)			
	สำเนาที่ 1	สำเนาที่ 2	สำเนาที่ 3	สำเนาที่ 4
1	0/10	0/10	0/10	3/10
2	0/10	0/10	1/10	5/10
3	0/10	0/10	0/10	3/10
4	0/10	0/10	0/10	5/10
5	0/10	1/10	1/10	1/10
รวม	0/50	1/50*	2/50**	17/50***

* เป็นความผิดพลาดที่เกิดจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาดเพียงอย่างเดียว

** เป็นความผิดพลาดที่เกิดจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาด 1 ฉบับและความผิดพลาดที่เกิดขึ้นหลังการตรวจสอบและแก้ไขความผิดพลาดแล้ว 1 ฉบับ

*** เป็นความผิดพลาดที่เกิดจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาด 14 ฉบับและความผิดพลาดที่เกิดขึ้นหลังการตรวจสอบและแก้ไขความผิดพลาดแล้ว 3 ฉบับ

จากตารางที่ 4.2 แสดงผลการดึงข้อมูลออกจากเอกสารที่ผ่านการทำสำเนาทั้งหมด 4 ครั้ง โดยพิจารณาถึงอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละหน้าเอกสาร จะเห็นว่าการดึงข้อมูลออกจากเอกสารสำเนาที่ 1 นั้นสามารถดึงข้อมูลออกมาได้ถูกต้องทั้งหมด (50 ฉบับ) สำหรับผลการดึงข้อมูลออกจากเอกสารสำเนาที่ 2 นั้นพบว่ามีเอกสารเพียงฉบับเดียวเท่านั้นที่เกิดความผิดพลาด (1/50) สำหรับเอกสารสำเนาที่ 3 พบว่ามีเอกสารเพียง 2 ฉบับเท่านั้นที่เกิดความผิดพลาด (2/50) และสำหรับเอกสารสำเนาที่ 4 มีเอกสารที่เกิดความผิดพลาดทั้งหมด 17 ฉบับ (17/50) ซึ่งความผิดพลาดที่เกิดขึ้นนี้สามารถแบ่งได้เป็น 2 ลักษณะ โดยลักษณะแรกจะเป็นความผิดพลาดที่เกิดจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาดทำให้การตรวจสอบและแก้ไขความผิดพลาดของข้อมูลเกิดความผิดพลาดตามไปด้วยเนื่องจากชุดข้อมูลที่ได้นี้เป็นชุดข้อมูลที่ไม่ถูกต้อง และความผิดพลาดในลักษณะที่สองคือความผิดพลาดที่เกิดขึ้นหลังจากที่ข้อมูลผ่านการตรวจสอบและแก้ไขข้อมูลมาแล้วเนื่องจากมีตำแหน่งบิตข้อมูลที่เกิดความผิดพลาดมากกว่า 1 ตำแหน่งในหนึ่งชุดข้อมูลรหัสแฮมมิงทำให้ไม่สามารถแก้ไขบิตข้อมูลที่ผิดพลาดให้ถูกต้องได้

ตารางที่ 4.3 แสดงอัตราการเกิดความผิดพลาดในแต่ละชุดข้อมูลรหัสแฮมมิง (ก่อนการแก้ไขข้อมูล) ของเอกสารที่ผ่านการถ่ายเอกสาร (เอกสารสำเนาที่ 1 ถึงเอกสารสำเนาที่ 4)

หน้าที่	อัตราการเกิดความผิดพลาดในแต่ละชุดข้อมูลรหัสแฮมมิง (จำนวนชุดข้อมูลที่เกิดความผิดพลาด / จำนวนชุดข้อมูลทั้งหมด)			
	สำเนาที่ 1	สำเนาที่ 2	สำเนาที่ 3	สำเนาที่ 4
1	15/80	5/80	3/80	25/80
2	2/80	19/80	10/80	52/80
3	9/80	19/80	19/80	27/80
4	13/80	7/80	29/80	41/80
5	3/80	14/80	17/80	33/80
รวม	42/400	64/400	78/400	178/400

ตารางที่ 4.4 แสดงผลการแก้ไขข้อมูลที่เกิดความผิดพลาดในแต่ละชุดข้อมูลโดยใช้วิธีแฮมมิง (7,4)

หน้าที่	อัตราความถูกต้องของการแก้ไขความผิดพลาดของข้อมูลในแต่ละชุด (จำนวนชุดข้อมูลที่สามารถแก้ไขความผิดพลาดได้ / จำนวนชุดข้อมูลที่เกิดความผิดพลาดทั้งหมด)			
	สำเนาที่ 1	สำเนาที่ 2	สำเนาที่ 3	สำเนาที่ 4
1	15/15	5/5	3/3	1/25
2	2/2	19/19	9/10	15/52
3	9/9	19/19	19/19	6/27
4	13/13	7/7	29/29	22/41
5	3/3	6/14	9/17	14/33
รวม	42/42	56/64	69/78	58/178

เมื่อพิจารณาถึงอัตราการความผิดพลาดของข้อมูลที่เกิดขึ้นในแต่ละชุดข้อมูลรหัสแฮมมิง (ดูผลในตารางที่ 4.3) และผลการแก้ไขข้อมูลที่ผิดพลาดโดยใช้วิธีการแฮมมิง (7,4) นั้น (ดูผลในตารางที่ 4.4) พบว่าชุดข้อมูลรหัสแฮมมิงที่เกิดความผิดพลาดโดยส่วนใหญ่ของเอกสารสำเนาที่ 1 ถึงสำเนาที่ 3 นั้นสามารถตรวจสอบและแก้ไขความผิดพลาดของข้อมูลที่เกิดขึ้นได้ในระดับหนึ่ง เนื่องจากในแต่ละชุดข้อมูลนั้นจะมีบิตข้อมูลที่ผิดพลาดไม่มากกว่า 1 ตำแหน่ง แต่สำหรับเอกสารที่ผ่านการทำสำเนามาหลายครั้ง (ตัวอย่างเช่น เอกสารสำเนาที่ 4 เป็นต้น) จะมีความผิดพลาดของข้อมูลเกิดขึ้นเป็นจำนวนมาก ซึ่งสาเหตุใหญ่ที่ทำให้เกิดความผิดพลาดในปริมาณมากคือการระบุบรรทัดที่

ถูกใช้ซ่อนข้อมูลจริงนั้นไม่ถูกต้องทำให้การจัดชุดข้อมูลรหัสแฮมมิงเกิดความผิดพลาดตามไปด้วย จึงไม่สามารถนำชุดข้อมูลเหล่านั้นมาตรวจสอบและแก้ไขความผิดพลาดได้ โดยในตารางที่ 4.5 จะแสดงผลบิตข้อมูลที่ถูกดึงออกจากเอกสาร (ชุดข้อมูลรหัสแฮมมิง) ซึ่งผ่านการตรวจสอบและแก้ไขข้อมูลที่ผิดพลาดเรียบร้อยแล้ว (พิจารณาความผิดพลาดที่เกิดขึ้นกับข้อมูลแต่ละบิต) จะเห็นว่าข้อมูลที่ถูกดึงออกจากเอกสารสำเนาที่ 1 ถึงสำเนาที่ 3 นั้นจะเกิดความผิดพลาดในปริมาณที่ไม่มากนักโดยความผิดพลาดที่เหลืออยู่นี้จะเป็นความผิดพลาดที่เป็นผลมาจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลผิดพลาดนั่นเอง ซึ่งความผิดพลาดของข้อมูลที่เกิดขึ้นในลักษณะนี้จะเกิดขึ้นเป็นจำนวนมากสำหรับเอกสารสำเนาที่ 4 ทำให้ไม่สามารถตรวจสอบแและแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ทั้งหมด

สำหรับการนำข้อมูลที่ถูกดึงออกจากเอกสารมาตรวจสอบความถูกต้องของเอกสารแต่ละฉบับโดยการเปรียบเทียบกับหมายเลขประจำเอกสารแต่ละฉบับที่กำหนดไว้เบื้องต้น ซึ่งมีผลของการตรวจสอบความถูกต้องของเอกสารในแต่ละสำเนามีดังนี้ (ในแต่ละสำเนามีเอกสารทั้งหมด 50 ฉบับ)

- เอกสารสำเนาที่ 1 มีเอกสารที่มีความถูกต้องทั้งหมด 50 ฉบับ
- เอกสารสำเนาที่ 2 มีเอกสารที่มีความถูกต้องทั้งหมด 49 ฉบับ
- เอกสารสำเนาที่ 3 มีเอกสารที่มีความถูกต้องทั้งหมด 48 ฉบับ
- เอกสารสำเนาที่ 4 มีเอกสารที่มีความถูกต้องทั้งหมด 33 ฉบับ

หมายเหตุ เอกสารที่ถูกต้องในที่นี้คือเอกสารที่บิตข้อมูลที่ถูกดึงออกจากเอกสารมีลักษณะเหมือนกับหมายเลขประจำเอกสารทุกประการเมื่อเปรียบเทียบกันในแต่ละบิต

ตารางที่ 4.5 แสดงผลการดึงข้อมูลออกจากเอกสารที่ผ่านการถ่ายเอกสารตั้งแต่เอกสารสำเนาที่ 1 จนถึงเอกสารสำเนาที่ 4 โดยพิจารณาถึงอัตราการเกิดความผิดพลาดของข้อมูลในแต่ละบิต (ข้อมูลหลังการตรวจสอบและแก้ไขข้อมูลที่ผิดพลาดแล้ว)

หน้าที่	อัตราความผิดพลาดของการดึงข้อมูลจากเอกสารที่ผ่านการถ่ายเอกสาร (จำนวนบิตข้อมูลที่เกิดความผิดพลาด / จำนวนบิตข้อมูลทั้งหมด)			
	สำเนาที่ 1	สำเนาที่ 2	สำเนาที่ 3	สำเนาที่ 4
1	0/560	0/560	0/560	168/560
2	0/560	0/560	2/560	138/560
3	0/560	0/560	0/560	127/560
4	0/560	0/560	0/560	121/560
5	0/560	56/560	56/560	120/560
รวม	0/2800	56/2800	58/2800	674/2800

4.4 ปัญหาที่พบ

ปัญหาที่พบจากการทดลองทั้งสองการทดลองนี้เกิดจากการที่เอกสารผ่านการประมวลผลทางด้านเอกสาร เช่น เอกสารถูกปรับขนาดให้มีขนาดใหญ่ขึ้นหรือเล็กลงหรือเอกสารผ่านการทำสำเนา (ถ่ายเอกสาร) มาหลายครั้ง ซึ่งมีผลให้ขนาดของช่องว่างระหว่างระดับชั้นที่ถูกซ่อนข้อมูลเกิดการเปลี่ยนแปลงตามไปด้วย โดยเราสามารถแบ่งลักษณะการเกิดปัญหาของขั้นตอนการดึงข้อมูลได้ 2 ลักษณะคือ

4.4.1 ปัญหาที่เกิดจากการระบุบรรทัดที่ซ่อนข้อมูลผิดพลาด

จากที่กล่าวมาในบทที่แล้วว่าวิธีการซ่อนข้อมูลลงในเอกสารในแต่ละหน้านั้นจะมีจำนวนบรรทัดที่ถูกใช้ซ่อนข้อมูลเท่ากันแต่จะมีตำแหน่งบรรทัดที่ถูกซ่อนข้อมูลไม่เหมือนกันขึ้นอยู่กับลักษณะของเอกสารในแต่ละหน้า (ในงานวิจัยนี้กำหนดให้เอกสารแต่ละหน้าจะถูกซ่อนข้อมูลทั้งหมด 28 บรรทัด ส่วนบรรทัดที่เหลือจะถูกซ่อนข้อมูลที่จะใช้กำหนดค่าเธรดโฮลด์ของการระบุค่าบิตข้อมูล) โดยข้อมูลชุดรหัสแฮมมิงแต่ละชุดที่ถูกสร้างขึ้นในขั้นตอนการเตรียมข้อมูลจะถูกซ่อนเรียงกันไปในแต่ละบรรทัด ดังนั้นหากขั้นตอนของการดึงข้อมูลเกิดความผิดพลาดในการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลก็จะทำให้เกิดผลที่ตามมาดังนี้โดยจะแยกออกเป็นสองกรณีคือ

- กรณีที่ไม่สามารถระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลจริงได้ครบชุดข้อมูล (ไม่ครบ 28 บรรทัด) ซึ่งผลที่เกิดขึ้นตามมาคือจะไม่สามารถนำข้อมูลเหล่านี้ไปตรวจหาและแก้ไขความผิดพลาดของข้อมูลได้ เนื่องจากมีจำนวนข้อมูลไม่เพียงพอที่จะนำมาจัดเรียงเป็นชุดข้อมูลรหัสแฮมมิง
- กรณีที่สามารถระบุจำนวนบรรทัดที่ถูกใช้ซ่อนข้อมูลได้ครบชุดแต่ไม่ใช่ชุดข้อมูลที่ถูกต้อง (เช่น การระบุว่าบรรทัดที่ถูกซ่อนข้อมูลจริงเป็นบรรทัดที่ไม่ถูกซ่อนข้อมูลหรือการระบุว่าบรรทัดที่ไม่ถูกซ่อนข้อมูลจริงเป็นบรรทัดที่ถูกซ่อนข้อมูลจริง) ซึ่งความผิดพลาดในลักษณะนี้จะทำให้การจัดเรียงชุดข้อมูลรหัสแฮมมิงผิดพลาดตามไปด้วย ดังนั้นหากนำชุดข้อมูลเหล่านี้ไปตรวจสอบและแก้ไขความความผิดพลาดก็จะทำให้ได้ข้อมูลที่ไม่ถูกต้อง

สำหรับสาเหตุสำคัญที่ทำให้การระบุบรรทัดที่ซ่อนข้อมูลผิดพลาดนั้นเกิดจากการที่ไม่สามารถแยกความแตกต่างของช่องว่างที่ใช้ซ่อนข้อมูลที่ต่างกันได้ (ข้อมูล “1” และ “0”) เนื่องจากการระบุบรรทัดที่ถูกใช้ซ่อนข้อมูลจะพิจารณาจากค่าความแตกต่างของขนาดช่องว่างระหว่างระดับชั้นที่ถูกซ่อนข้อมูลในแต่ละบรรทัดว่ามีความแตกต่างกันมากน้อยเพียงใด

4.4.2 ปัญหาที่เกิดจากการระบุค่าบิตข้อมูลผิดพลาด

สาเหตุที่ทำให้เกิดการระบุค่าบิตข้อมูลผิดพลาดสามารถแบ่งได้เป็น 2 สาเหตุคือ

1) การจัดกลุ่มสัญลักษณ์ข้อมูลผิดพลาด

จากที่กล่าวถึงในบทที่แล้วว่าวิธีการซ่อนข้อมูลที่ใช้ในงานวิจัยนี้จะซ่อนข้อมูลหลายตำแหน่ง (ค่าสัญลักษณ์ข้อมูล) เพื่อแทนค่าบิตข้อมูล 1 บิต ดังนั้นในขั้นตอนของการดึงข้อมูล

ออกจากเอกสารจึงต้องมีการจัดกลุ่มค่าสัญลักษณ์ข้อมูลให้อยู่ในกลุ่มที่ถูกต้องเสียก่อน ซึ่งถ้าค่าสัญลักษณ์ถูกจัดให้อยู่ในกลุ่มที่ไม่ถูกต้องก็จะทำให้การระบุค่าบิตข้อมูลของกลุ่มนั้นเกิดความผิดพลาดตามไปด้วย

2) การแก้ไขชุดข้อมูลรหัสแฮมมิงเกิดความผิดพลาด

สาเหตุที่วิธีการตรวจสอบและแก้ไขชุดข้อมูลรหัสแฮมมิงเกิดความผิดพลาดในการทำงานก็เนื่องจากว่ามีจำนวนบิตข้อมูลที่เกิดความผิดพลาดมากกว่า 1 ตำแหน่งภายในหนึ่งชุดข้อมูลรหัสแฮมมิง (วิธีการตรวจสอบและแก้ไขชุดข้อมูลรหัสแฮมมิงนี้สามารถทำงานได้อย่างถูกต้องเมื่อในหนึ่งชุดข้อมูลเกิดความผิดพลาดไม่มากกว่า 1 ตำแหน่งเท่านั้น) ซึ่งส่งผลให้การตรวจหาและแก้ไขตำแหน่งบิตข้อมูลที่มีความผิดพลาดนั้นไม่ถูกต้อง โดยอาจจะไปแก้ไขบิตที่ไม่มีความผิดพลาดแทนบิตที่เกิดความผิดพลาดซึ่งจะทำให้เกิดความผิดพลาดมากยิ่งขึ้น

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

งานวิจัยนี้ได้นำเสนอวิธีการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทยโดยมีวัตถุประสงค์เพื่อใช้ตรวจสอบความถูกต้องของเอกสารซึ่งวิธีการที่นำเสนอนี้จะซ่อนข้อมูลโดยอาศัยโครงสร้างเฉพาะของภาษาไทยทำให้สามารถซ่อนข้อมูลได้มากกว่าวิธีการอื่นๆที่มีอยู่ในปัจจุบัน [4, 5, 6, 7, 8] อีกทั้งเอกสารที่ถูกซ่อนข้อมูลด้วยวิธีการนี้จะไม่ถูกสังเกตเห็นความผิดปกติได้ง่ายเนื่องจากวิธีการซ่อนข้อมูลนี้จะกระทำโดยการเปลี่ยนแปลงขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่สองกับสามเพียงเล็กน้อยเพื่อแทนค่าบิตข้อมูลไบนารีที่ต้องการซ่อน จากผลการเปรียบเทียบวิธีการซ่อนข้อมูลลงในเอกสารภาษาไทยขนาด A4 ซึ่งใช้รูปแบบ AngsanaUPC ขนาด 14 พอยท์ที่มีการจัดตัวอักษรแบบชิดขอบ (ในหนึ่งหน้าเอกสารมีจำนวนบรรทัดประมาณ 36 บรรทัด) โดยใช้วิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้กับวิธีการซ่อนข้อมูลที่มีอยู่ในปัจจุบัน (พิจารณาเฉพาะวิธี Line shift coding เนื่องจากวิธีการอื่นๆไม่สามารถนำมาประยุกต์ใช้กับเอกสารภาษาไทยได้) พบว่าวิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้สามารถซ่อนข้อมูลได้มากกว่าวิธี Line shift coding โดยในหนึ่งบรรทัดนั้นสามารถซ่อนข้อมูลได้โดยเฉลี่ยประมาณ 13 ตำแหน่ง แต่สำหรับวิธี Line shift coding นั้นจะต้องใช้จำนวนบรรทัดถึงสองบรรทัดในการซ่อนข้อมูล 1 ตำแหน่ง ดังนั้นจะได้ว่าการซ่อนข้อมูลด้วยวิธีการที่นำเสนอในงานวิจัยนี้สามารถซ่อนข้อมูลได้ประมาณ 468 ตำแหน่งต่อหนึ่งหน้าเอกสาร แต่สำหรับวิธี Line shift coding นั้นจะซ่อนข้อมูลได้ทั้งหมด 17 ตำแหน่งต่อหนึ่งหน้าเอกสาร

ข้อจำกัดอย่างหนึ่งของการซ่อนข้อมูลด้วยวิธีการที่นำเสนอในงานวิจัยนี้คือเมื่อเอกสารผ่านการประมวลผลทางด้านเอกสารบางอย่าง (การพิมพ์ การถ่ายเอกสาร และการสแกน) หลายๆครั้งอาจทำให้ข้อมูลที่ซ่อนอยู่ภายในเอกสารนั้นเกิดความผิดพลาดได้ ดังนั้นในงานวิจัยนี้จึงได้นำวิธีการป้องกันและแก้ไขความผิดพลาดของข้อมูลเข้ามาประยุกต์ใช้ร่วมกับวิธีการซ่อนข้อมูลที่นำเสนอนี้ โดยหลักการที่ใช้ป้องกันความผิดพลาดของข้อมูลอย่างหนึ่งคือการจัดบล็อกตัวอักษรให้อยู่ในตำแหน่งที่เหมาะสมโดยในงานวิจัยนี้กำหนดให้ $S_{\min}$ และ $S_{\max}$ ที่ใช้จัดตำแหน่งบล็อกตัวอักษรมีค่าเท่ากับ 1 และ 4 ตามลำดับ (ค่านี้ได้มาจากการทดลองเพื่อหาขนาดความกว้างระหว่างบล็อกตัวอักษรที่เหมาะสมที่จะไม่เกิดความผิดพลาดได้ง่าย ซึ่งสามารถดูรายละเอียดได้จากหัวข้อ 3.4.2 ในบทที่ 3) วิธีการป้องกันการเกิดความผิดพลาดของข้อมูลอีกวิธีการหนึ่งคือการเพิ่มจำนวนข้อมูลที่จะซ่อนลงในเอกสารเพื่อแทนค่าบิตข้อมูลหนึ่งบิต (R_d) โดยในงานวิจัยนี้กำหนดให้ R_d มีค่าเท่ากับ 4 (ดูรายละเอียดการกำหนดค่านี้ได้จากหัวข้อ 3.3.3 ในบทที่ 3) และอีกวิธีการหนึ่งที่ใช้ป้องกันการเกิด

ความผิดพลาดได้คือการกำหนดจำนวนข้อมูลที่จะซ่อนลงในแต่ละบรรทัดให้สามารถจำกัดบริเวณของการเกิดความผิดพลาดให้อยู่ภายในแต่ละบรรทัดได้ โดยในงานวิจัยนี้กำหนดให้ซ่อนข้อมูลได้เพียงบรรทัดละ 8 ตำแหน่ง (L_d) เท่านั้น (ดูรายละเอียดของการกำหนดค่านี้ได้จากหัวข้อที่ 3.3.4 ในบทที่ 3) สำหรับหลักการที่นำมาใช้แก้ไขความผิดพลาดของข้อมูลคือการสร้างชุดข้อมูลรหัสแฮมมิงจากข้อมูลที่ต้องการซ่อนลงในเอกสาร โดยชุดข้อมูลนี้สามารถตรวจสอบความผิดพลาดของข้อมูลและทำการแก้ไขข้อมูลที่ผิดพลาดให้ถูกต้องได้ จะเห็นว่าวิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้จะมีข้อมูลที่เป็นส่วนของรีดันแดนซีที่ใช้ในการตรวจสอบและแก้ไขความผิดพลาดซ่อนลงไปในเอกสารด้วยทำให้สามารถซ่อนข้อมูลได้น้อยลง อย่างไรก็ดีจำนวนข้อมูลสุทธิที่ซ่อนได้ยังคงสูงกว่าวิธี Line shift coding

จากผลการทดลองดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการปรับขนาด (บทที่ 4) พบว่าสามารถดึงข้อมูลออกจากเอกสารที่ถูกขยายขนาดถึง 30% ได้ถูกต้องทั้งหมด สำหรับเอกสารที่ถูกย่อขนาดสามารถดึงข้อมูลออกมาได้อย่างถูกต้องเมื่อเอกสารนั้นถูกย่อขนาดไม่เกิน 15% และจากการทดลองดึงข้อมูลออกจากเอกสารที่ผ่านกระบวนการถ่ายเอกสาร พบว่าอัตราความผิดพลาดของการดึงข้อมูลนั้นจะขึ้นอยู่กับจำนวนครั้งในการทำสำเนาเอกสาร นั่นคือเมื่อเอกสารถูกทำสำเนาหลายๆครั้งก็จะทำให้เอกสารเกิดความบิดเบือนไปจากเดิมและทำให้การดึงข้อมูลเกิดความผิดพลาดได้

5.2 ข้อเสนอแนะสำหรับการพัฒนาในอนาคต

เนื่องด้วยวิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้ยังมีปัญหาในเรื่องของการระบุบรรทัดที่ซ่อนข้อมูลผิดพลาดทำให้เกิดความผิดพลาดในการจัดชุดข้อมูลรหัสแฮมมิงซึ่งอาจเป็นผลให้ข้อมูลในแต่ละชุดเกิดความผิดพลาดมากกว่า 1 ตำแหน่ง ซึ่งวิธีการแฮมมิงที่นำมาใช้ในงานวิจัยนี้ไม่สามารถแก้ไขข้อมูลที่ผิดพลาดเหล่านั้นให้ถูกต้องได้ แนวทางในการแก้ไขปัญหานี้สามารถทำได้โดยการจำกัดขอบเขตของชุดข้อมูลรหัสแฮมมิงในแต่ละชุดให้อยู่ในขอบเขตเดียวกัน เช่น กำหนดให้ชุดข้อมูลรหัสแฮมมิงในแต่ละชุดอยู่ภายในย่อหน้าเดียวกัน ดังนั้นหากมีชุดข้อมูลรหัสแฮมมิงชุดใดที่เกิดผิดพลาดก็จะถูกจำกัดให้อยู่ภายในชุดข้อมูลนั้นๆ ซึ่งข้อจำกัดของวิธีการนี้คือจะต้องสามารถควบคุมการจัดย่อหน้าของเอกสารก่อนที่จะซ่อนข้อมูลด้วย และแนวทางการแก้ปัญหาอีกวิธีหนึ่งคือการสร้างชุดข้อมูลที่สามารถตรวจสอบและแก้ไขความผิดพลาดของข้อมูลที่เกิดขึ้นมากกว่า 1 ตำแหน่งได้ ซึ่งวิธีการนี้จะมีจำนวนข้อมูลที่เป็นส่วนของรีดันแดนซีที่ใช้ตรวจสอบความถูกต้องของข้อมูลมีมากขึ้นทำให้สามารถซ่อนข้อมูลได้น้อยลง

การกำหนดค่าพารามิเตอร์ที่ใช้สร้างพับลิคคีย์และไพรเวตคีย์ของการเข้ารหัสและถอดรหัสข้อมูลเพื่อสร้างลายเซ็นดิจิตอลในงานวิจัยนี้จะกำหนดให้มีค่าน้อยๆ ($p = 29$, $q = 31$) เพื่อความสะดวกในการคำนวณสำหรับใช้ในการทดลอง แต่ในการใช้งานจริงควรกำหนดให้ค่าพารามิเตอร์ที่

ใช้สร้างคีย์ต่างๆนี้เป็นเลขจำนวนเฉพาะที่มีค่ามากๆเพราะจะทำให้ค่าคีย์ที่ได้มีความปลอดภัยจากการถูกปลอมแปลงมากขึ้น เนื่องจากมีความเป็นไปได้น้อยมากที่จะสามารถตรวจพบเลขจำนวนเฉพาะที่มีค่ามากได้ตรงกันทั้งสองจำนวน

การตรวจสอบความถูกต้องของเอกสารโดยใช้วิธีการซ่อนข้อมูลที่นำเสนอในงานวิจัยนี้สามารถป้องกันการปลอมแปลงเอกสารได้ในระดับหนึ่งเท่านั้น เนื่องจากวิธีการนี้มีข้อจำกัดในเรื่องของการสร้างหมายเลขประจำเอกสารที่จะซ่อนลงในเอกสาร โดยหมายเลขประจำเอกสารจะถูกสร้างขึ้นมาจากการสุ่มตัวเลขซึ่งไม่มีความสัมพันธ์ใดๆกับเนื้อความในเอกสารทั้งสิ้น ดังนั้นหากมีผู้ไม่ประสงค์ดีทราบหมายเลขประจำเอกสารก็สามารถแก้ไขหรือสร้างเอกสารปลอมขึ้นมาใหม่ได้ วิธีการแก้ปัญหาดังกล่าวอาจทำได้โดยใช้การประทับตราเวลาลงในเอกสารที่ผู้รับได้รับ ดังนั้นหากมีเอกสารมากกว่าหนึ่งฉบับที่มีหมายเลขเอกสารเหมือนกันส่งมายังผู้รับก็จะกำหนดให้เอกสารที่ส่งมาถึงเป็นฉบับแรกเป็นเอกสารที่ถูกต้อง

เอกสารอ้างอิง

[1] Jiri Fridrich "Methods for data hiding" Center for intelligent systems & department of systems science and industrial engineering, SUNY Binghamton, Binghamton, NY 13902-6000, 1997.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu "Techniques for data hiding" IBM Systems Journal 35(3/4), 1996, pp. 313-336.

[3] L. O'Gorman and R. Kasturi "Document Image Analysis" In IEEE Computer Society Tutorial Series, IEEE 1994.

[4] J. Brassil, S. Low, N. Maxemchuk, L.O'Gorman "Electronic Marking and Identification Techniques to Discourage Document Copying" Proceedings of IEEE INFOCOM' 94, vol.3, Toronto, June 1994, pp.1278-1287.

[5] S.H. Low, N.F. Maxemchuk, J.T. Brassil, and L. O'Gorman "Document Marking and Identification Using Both Line and Word Shifting" Proceedings of Infocom'95, April 1995.

[6] S.H. Low, N.F. Maxemchuk, A.M. Lapone "Document Identification for Copyright Protection Using Centroid Detection" IEEE Transactions on Communication, to appear.

[7] S.H. Low and N.F. Maxemchuk "Performance Comparison of two Text Marking and Detection Methods" IEEE Journal on Selected Areas in Communication, 1998.

[8] J. Brassil, S. Low, N.F. Maxemchuk, J.T. Brassil, and L. O'Gorman "Hiding Information in Document Images" in Proceedings of 1995 Conference on Information Sciences and Systems, March 1995, pp. 482-489.

[9] Jaekyu Ha, Robert M. Haralick, and Ihsin T. Phillips "Document Page Decomposition by the Bounding-Box Projection Technique" in Proceedings of the Third International Conference on Document Analysis and Recognition, Montreal, August 1995, pp. 1119-1122.

[10] William Stallings. **Data and Computer Communication.** 5th edition Prentice-Hall International, 1989, pp.638-656.

[11] H.S. Baird "Document Image Defect Models" Structured Document Image Analysis, Springer-Verlag, Berlin, 1992, pp. 546-556.

[12] Jerry FitzGerald and Alan Dennis. **Bussiness Data Communication and Network.** 5th edition John Wiley&Sons, 1996, pp.188-195.

[13] Man Young Rhee. **Error Correcting Coding Theory.** McGraw-Hill International editions, 1989, pp. 35-86.

[14] Minerva M. Yeung and Fred Minitzer "An Invisible watermarking technique for image verification" Proceedings International Conference on published 1997, vol. 2, 1997, pp. 680- 683.

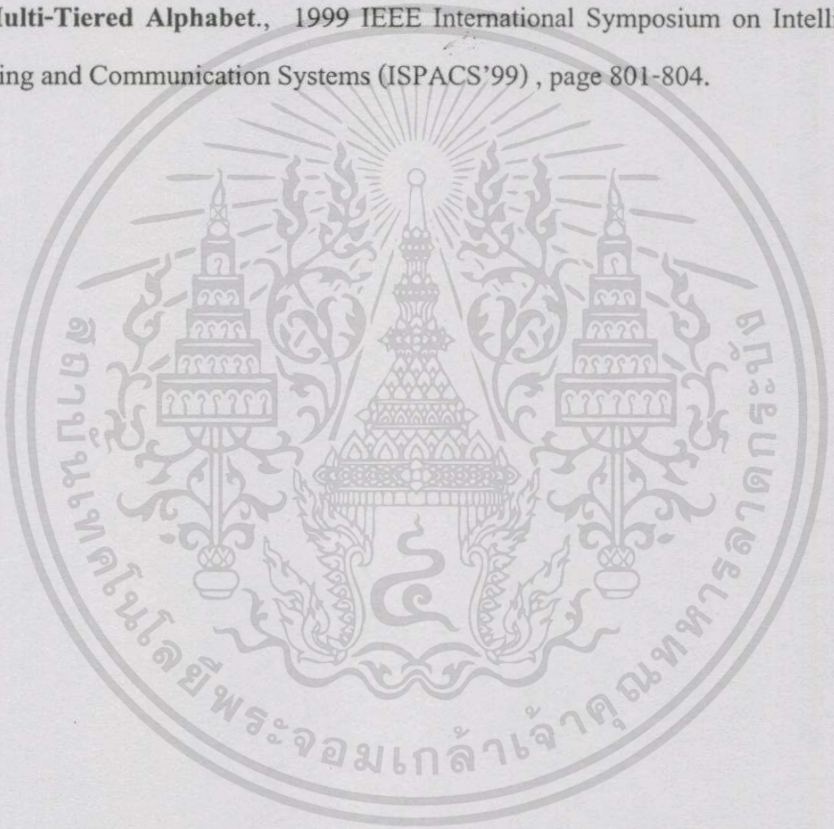
[15] Ping Wah Wong "Public-key Watermark for Image Verification and Authentication" Proceedings of International Conference on ICIP'98 ,vol 1, 1998, pp. 455-459.

[16] William H. Press, Saul A. Teukolsky, William T. Vetterling and Brain P. Flannery. **Numerical recipes in C: The art of scientific computing.** Cambridge University Press, 1992, pp.123-127.

[17] Bruce L. Bowerman and Rechasrd T. O'Connell. **Forcasting and time series : An applied approach.** Duxbury Press, 1993, pp.78-93.

ภาคผนวก ก.
บทความและผลงานวิจัยที่ได้รับการตีพิมพ์

1. นงนุช อาจวารินทร์ และ นพพร โชติกกำธร., **เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย.**, การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 22 (EECON'22), หน้า 385-388.
2. N. Artwarin and N. Chotikakamthorn ., **Data Hiding Technique for Electronic Document with Multi-Tiered Alphabet.**, 1999 IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS'99) , page 801-804.



การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 22
The 22nd Electrical Engineering Conference
(EECON-22)
วันที่ 2 – 3 ธันวาคม 2542
ณ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

ที่ ทม 0407.08/ [illegible]

๑ กันยายน 2542

เรื่อง แจ้งผลการพิจารณาบทความ EECON-22

เรียน คุณนงนุช อาจวารินทร์

ตามที่ท่านได้ส่งบทความเพื่อเสนอในที่ประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 22 (EECON-22) รหัส CP044 เรื่อง เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย

บัดนี้ คณะกรรมการได้ดำเนินการพิจารณาบทความเสร็จแล้ว จึงขอแจ้งผลการพิจารณาดังนี้

☐ ผ่าน โดยไม่ต้องแก้ไขใดๆ ทั้งสิ้น (คณะกรรมการจะตีพิมพ์บทความของท่านโดยใช้บทความฉบับแรก)

☑ ผ่าน โดยมีเงื่อนไขว่า

1. ท่านต้องแก้ไขบทความตามที่กรรมการแนะนำ หากไม่ได้แก้ไขจะถือว่าไม่ผ่านการพิจารณา
2. ส่งบทความที่แก้ไขใหม่แล้ว (ใช้รูปภาพจริง) มายังประธานฯ ภายในวันที่ 30 กันยายน 2542
3. ถ้าหากท่านไม่สามารถส่งบทความที่แก้ไขแล้วได้ภายในวันที่ 30 กันยายน 2542 จะถือว่าท่านสละสิทธิในการส่งบทความในครั้งนี้

☐ ไม่ผ่านการพิจารณา

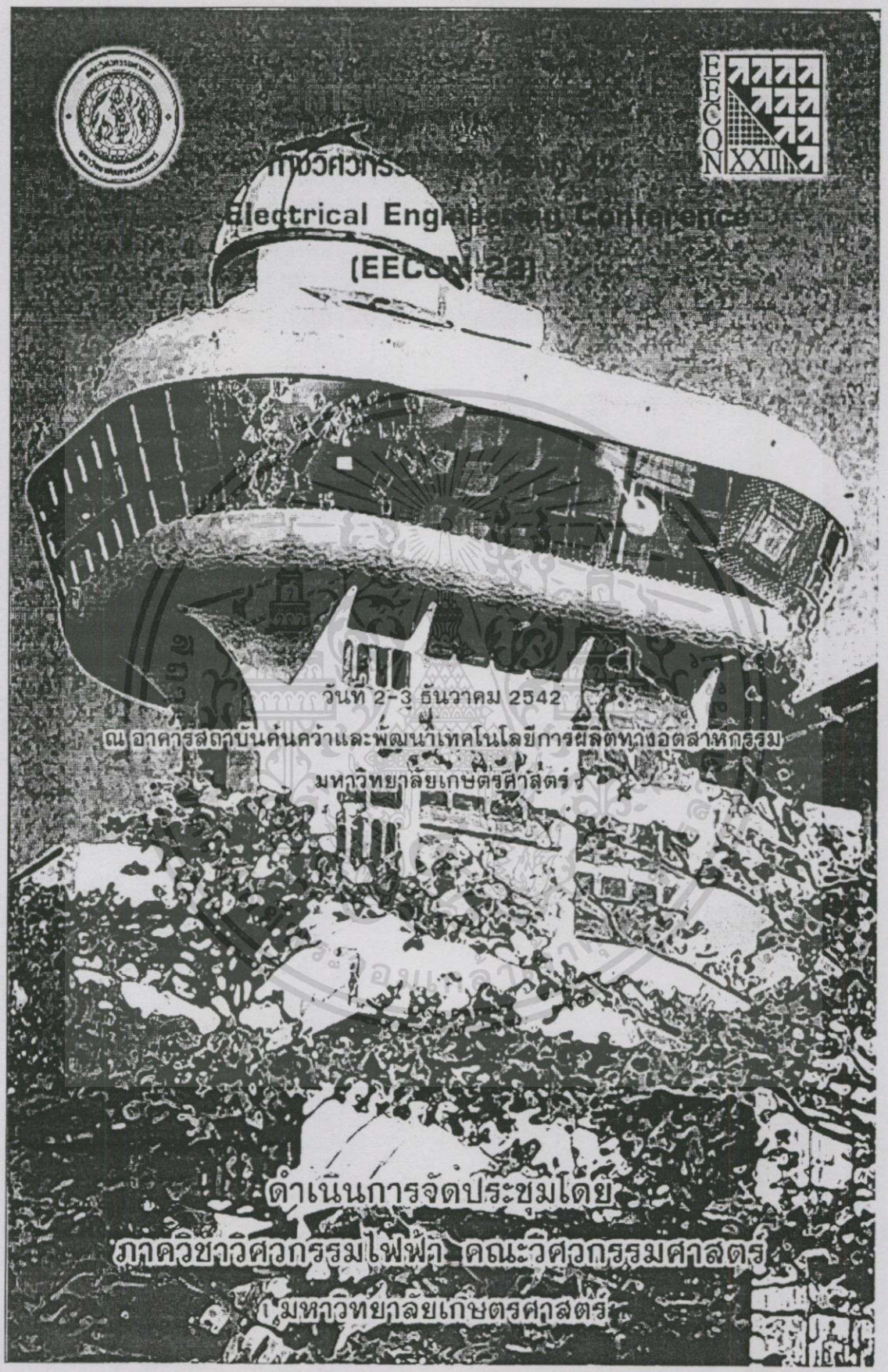
คณะกรรมการขอขอบพระคุณที่ได้ให้ความร่วมมือและสนใจส่งบทความมาเป็นจำนวนมาก และหวังเป็นอย่างยิ่งว่าจะได้รับความร่วมมือในครั้งต่อๆ ไปด้วยดี หากท่านมีข้อสงสัยประการใด กรุณาติดต่อสอบถามได้ที่หมายเลขโทรศัพท์ 9428555 ต่อ 1503-4

อนึ่ง บทความที่ผ่านเพื่อเข้านำเสนอผลงานในงาน EECON-22 ทุกบทความจะต้องมีผู้ลงทะเบียนอย่างน้อย 1 ท่าน เพื่อนำเสนอผลงาน มิฉะนั้นบทความจะไม่ได้รับการพิจารณาให้ตีพิมพ์ เพื่อให้การเตรียมจัดการประชุมเป็นไปด้วยความเรียบร้อย จึงขอความร่วมมือผู้ที่จะลงทะเบียนเข้าประชุมฯ กรุณาลงทะเบียนล่วงหน้า (ภายในวันที่ 15 ตุลาคม 2542) โดยใช้เอกสารลงทะเบียนที่แนบมานี้ จะขอบพระคุณเป็นอย่างสูง

จึงเรียนมาเพื่อโปรดทราบ

ขอแสดงความนับถือ

(ผศ.ดร.ตฤณ แสงสุวรรณ)
ประธานจัดการประชุมวิชาการ ครั้งที่ 22



CP044

เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย

Data Hiding Technique for Electronic Documents in Thai Language

นงนุช อาจวารินทร์ และ นพพร โชคิกกำธร
คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ถนนฉลองกรุง เขตลาดกระบัง กรุงเทพฯ 10520
โทร (02) 737-2551-4 ต่อ 802 โทรสาร (02) 326-9074 E-mail : s0067042@kmitl.ac.th

บทคัดย่อ

บทความนี้กล่าวถึงปัญหาการซ่อนข้อมูลลงในเอกสารรูปภาพที่มีอยู่ในปัจจุบัน วิธีการที่มีอยู่ในปัจจุบันถูกออกแบบมาสำหรับเอกสารภาษาอังกฤษ เมื่อนำวิธีการเหล่านี้มาประยุกต์ใช้กับเอกสารภาษาไทย พบว่ามีข้อจำกัดบางประการเนื่องจากความแตกต่างทางด้านโครงสร้างภาษา บทความนี้นำเสนอเทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพภาษาไทย วิธีการดังกล่าวอาศัยช่องว่างระหว่างระดับชั้นของตัวอักษรในการซ่อนข้อมูลจึงทำให้ (สำหรับเอกสารภาษาไทยโดยทั่วไป) มีความจุในการซ่อนข้อมูลสูง นอกจากนั้นยังได้เสนอเทคนิคในการป้องกันและแก้ไขข้อผิดพลาดในการถอดรหัสข้อมูลสำหรับเอกสารที่เสื่อมคุณภาพเนื่องมาจากเอกสารผ่านกระบวนการถ่ายเอกสารหลายครั้ง จากผลการทดลองซ่อนข้อมูลลงในเอกสารภาษาไทยจำนวน 10 หน้า ซึ่งเป็นเอกสารที่ผ่านกระบวนการพิมพ์และการถ่ายเอกสาร พบว่าวิธีการที่นำเสนอในบทความนี้มีความถูกต้องในการถอดรหัสข้อมูลโดยเฉลี่ยเท่ากับ 98 %

Abstract

The problem of document image data hiding is considered in this paper. The existing word shifting methods, designed for English document, has some drawbacks when applied to document written in Thai. The problem is due to difference in language structure. In this paper we propose a new data hiding technique for document image written in Thai language. The method adjusts the width of space between levels of character components on the same text line. Thus, the method, when applied to typical Thai documents, has high data embedding capacity. In additional, we propose an error correction technique for use with documents degraded by duplication process. Experiment result using 10 pages of copied document is provided. From the experiment, the percentage of bits correctly decoded is about 98 %.

1. คำนำ

ในปัจจุบันเทคโนโลยีทางด้านการสื่อสารมีการพัฒนาเพิ่มขึ้น ทำให้การแพร่กระจายของเอกสารอิเลคทรอนิคได้รับความนิยมเพิ่มขึ้น ซึ่งมีข้อดีคือเอกสารสามารถถูกแพร่กระจายได้อย่างสะดวกรวดเร็วและไม่สิ้นเปลืองค่าใช้จ่ายมากนัก แต่มีข้อเสียคือเอกสารเหล่านั้นอาจถูกละเมิดลิขสิทธิ์ได้ง่าย ดังนั้นจึงได้มีการคิดค้นวิธีการซ่อนข้อมูลบางอย่างลงในเอกสาร (Data hiding in document) เพื่อใช้ในการป้องกันการละเมิดลิขสิทธิ์ในเอกสารนั้น [1] เราสามารถนำวิธีการเหล่านี้ไปประยุกต์ใช้กับการซ่อนข้อมูลที่เป็นรายละเอียดต่างๆของเอกสาร เช่น การซ่อนคำสำคัญ (Keyword) ของเอกสารเพื่อใช้ในการค้นหาข้อมูลหรือเอกสารอื่นๆที่เกี่ยวข้อง (Document searching) หรืออาจนำมาประยุกต์ใช้ในการซ่อนข้อมูลเพื่อตรวจสอบความถูกต้องของเอกสาร (Document authentication)

โดยส่วนใหญ่แล้วเทคนิคการซ่อนข้อมูลลงในสื่อนั้นจะมุ่งเน้นที่สื่อประเภทรูปภาพและวีดีโอเท่านั้น [1] สำหรับเทคนิคการซ่อนข้อมูลลงในสื่อประเภทเอกสารที่นิยมใช้ในปัจจุบันมี 2 วิธีคือ วิธี Line shift coding และ Word shift coding ทั้งสองวิธีการนี้ถูกออกแบบมาสำหรับการซ่อนข้อมูลลงในเอกสารที่เป็นภาษาอังกฤษ [2] โดยเฉพาะวิธี Word shift coding ซึ่งใช้ช่องว่างระหว่างคำในการซ่อนข้อมูล เมื่อนำวิธีการนี้ไปประยุกต์ใช้กับเอกสารที่เป็นภาษาไทยพบว่าความจุในการซ่อนข้อมูลมีปริมาณน้อยลง เนื่องจากเอกสารภาษาไทยไม่มีการเว้นช่องว่างระหว่างคำ แต่มีการเว้นช่องว่างระหว่างประโยคเท่านั้น สำหรับวิธีการ Line shift coding มีข้อเสียคือมีความจุในการซ่อนข้อมูลน้อยดังนั้นวิธีนี้จึงมีข้อจำกัดในการนำไปใช้งาน ในบทความนี้จะนำเสนอเทคนิคใหม่ในการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทย ซึ่งวิธีการนี้จะซ่อนข้อมูลโดยอาศัยโครงสร้างของภาษาไทยโดยเฉพาะ โดยจะทำการซ่อนข้อมูลลงในตำแหน่งที่เป็นช่องว่างระหว่างระดับชั้นในแต่ละบรรทัด ในส่วนถัดไปจะกล่าวถึงรูปแบบของเอกสารรูปภาพและโครงสร้างภาษาไทย หลักการซ่อนข้อมูลลงในเอกสารรูปภาพภาษาไทย รวมถึงแสดงผลการทดลองการซ่อนข้อมูลลงในเอกสารภาษาไทยโดยใช้วิธีการที่เสนอ

2. รูปแบบของเอกสารรูปภาพและโครงสร้างการเขียนของเอกสารภาษาไทย

ในบทความนี้กำหนดให้เอกสารรูปภาพที่จะใช้ซ่อนข้อมูลอยู่ในรูปแบบของบิตแมป (Bitmap file) ซึ่งมีรูปแบบฟังก์ชันดังนี้

CP044

เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย

$$f(x,y) \in \{0,1\}, x = 0,...,W-1,\ y = 0,...,L-1 \qquad (1)$$

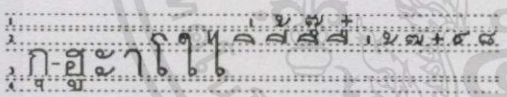
โดยที่ W และ L คือ ความกว้างและความยาวของหน้าเอกสารในหน่วยของพิกเซล และค่าของ $f(x,y)$ คือค่าความสว่างในแต่ละพิกเซล

วิธีการที่จะนำเสนอในบทความนี้ใช้หลักการของโครงร่างเอกสาร (Document profile) เข้ามาช่วยในการในการแบ่งขอบเขตของบรรทัดและตัวอักษรภายในหน้าเอกสาร ซึ่งฟังก์ชันที่ใช้ในการแสดงโครงร่างเอกสารตามแนวนอนและแนวตั้ง (Horizontal and Vertical profile) คือ

$$h(y) = \sum_{x=0}^{W-1} f(x,y),\ 0 \le y \le L-1 \qquad (2)$$

$$v(x) = \sum_{y=0}^{L-1} f(x,y),\ 0 \le x \le W-1 \qquad (3)$$

โครงสร้างการเขียนของภาษาไทยจะแบ่งระดับชั้นในการแสดงข้อความออกเป็น 4 ระดับ โดยระดับที่หนึ่งเป็นระดับของวรรณยุกต์และการันต์ (ในกรณีที่มีสระอยู่ในระดับที่สอง) ส่วนระดับที่สองเป็นระดับของสระระดับบน (เช่น สระอิ สระอี ฯลฯ) และวรรณยุกต์ ระดับที่สามเป็นระดับของพยัญชนะและสระระดับกลาง (เช่น สระอะ สระโอ ฯลฯ) และระดับสี่เป็นระดับของสระระดับล่าง (เช่น สระอุ สระอู) ซึ่งแสดงตัวอย่างดังรูปที่ 1 จะเห็นว่าตัวอักษรที่อยู่ในแต่ละระดับชั้นนั้นถูกแยกออกจากกันด้วยช่องว่างระหว่างระดับชั้น ซึ่งในส่วนถัดไปจะแสดงถึงวิธีการใช้ช่องว่างเหล่านี้ในการซ่อนข้อมูล



รูปที่ 1 ตัวอย่างโครงสร้างการเขียนของภาษาไทย

วิธีการนี้อาจนำไปประยุกต์ใช้กับเอกสารที่ใช้ภาษาอื่นที่มีโครงสร้างคล้ายคลึงกับภาษาไทยได้ เช่น ภาษาฝรั่งเศส เนื่องจากโครงสร้างการเขียนของภาษาฝรั่งเศสนั้นจะมีการแบ่งระดับชั้นในการแสดงข้อความออกเป็น 2 ระดับ โดยระดับบนจะเป็นระดับของสัญลักษณ์ที่ใช้ในการควบคุมเสียง ส่วนระดับล่างจะเป็นระดับของพยัญชนะและสระ

3. วิธีการซ่อนข้อมูลสำหรับเอกสารรูปภาพที่เป็นภาษาไทย

หลักการซ่อนข้อมูลสำหรับเอกสารรูปภาพที่เป็นภาษาไทย [3] คือจะซ่อนข้อมูลโดยการเปลี่ยนแปลงขนาดความกว้างของช่องว่างระหว่างระดับชั้นของตัวอักษรซึ่งสามารถทำได้โดยการเปลี่ยนแปลงตำแหน่งของตัวอักษรที่อยู่ในระดับบนตามแนวตั้ง (เลื่อนขึ้น-เลื่อนลง) ซึ่งวิธีการนี้เลือกที่จะซ่อนข้อมูลลงในช่องว่างที่อยู่ระหว่างระดับชั้นที่สองกับสาม เนื่องจากช่องว่างที่ตำแหน่งนี้จะมีขนาดความกว้างมากกว่าช่องว่างในช่วงอื่นๆ อีกทั้งยังมีจำนวนตำแหน่งที่สามารถซ่อนข้อมูลได้มากกว่าช่องว่างในช่วงอื่นๆ การซ่อนข้อมูลวิธีการนี้แบ่งออกเป็นสองส่วนโดยส่วนแรกจะเป็นส่วนของการซ่อนข้อมูลและส่วนที่สองจะเป็นส่วนของการถอดรหัสข้อมูล โดยมีรายละเอียดดังนี้

3.1 การซ่อนข้อมูล

หลักการซ่อนข้อมูลนี้สามารถที่นำไปประยุกต์ใช้ได้กับทั้งเอกสารที่มีรูปแบบ (Format file) เช่น Postscript file และเอกสารที่เป็นบิตแมป (Bitmap file) ซึ่งในบทความนี้กำหนดให้เอกสารที่จะถูกซ่อนข้อมูลอยู่ในรูปแบบของบิตแมป โดยมีขั้นตอนการซ่อนข้อมูลดังนี้

1. สร้างโครงร่างตามแนวนอนของหน้าเอกสารเพื่อกำหนดขอบเขตของแต่ละบรรทัดภายในหน้าเอกสาร (ดูตัวอย่างรูปที่ 2)
2. สร้างโครงร่างตามแนวตั้งของแต่ละบรรทัดที่ได้จากขั้นตอนที่ 1 เพื่อที่จะแยกตัวอักษรในแต่ละบรรทัดออกเป็นบล็อกๆ (ดูตัวอย่างรูปที่ 3)
3. ทำการสร้างโครงร่างตามแนวนอนของแต่ละบล็อกตัวอักษรที่ได้จากขั้นตอนที่ 2 เพื่อกำหนดตำแหน่งของตัวอักษรที่อยู่ในแต่ละระดับภายในบล็อกนั้น (ดูตัวอย่างรูปที่ 4)
4. เลือกบรรทัดที่จะถูกใช้ซ่อนข้อมูลโดยพิจารณาจากจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้นั้นจะต้องมีค่ามากกว่าหรือเท่ากับค่า L_s (เช่น กำหนดให้ $L_s = 10$) ซึ่งในแต่ละบรรทัดที่ถูกเลือกมาจะซ่อนข้อมูลเพียงแค่ L_s ตำแหน่งเท่านั้น ส่วนตำแหน่งที่เหลือจะซ่อนข้อมูล "0" ลงไปในทุกๆตำแหน่ง และสำหรับบรรทัดที่ไม่ถูกเลือกจะซ่อนข้อมูล "0" ในทุกๆตำแหน่ง
5. จัดเตรียมข้อมูลที่จะใช้ซ่อนลงในเอกสารให้เป็นข้อมูลไบนารี โดยมีข้อกำหนดว่าค่าของข้อมูลที่จะถูกซ่อนในแต่ละบรรทัดนั้นจะต้องไม่มีค่า "0" อยู่ติดกัน ($L_s - \varepsilon - 1)/2$ ตำแหน่ง (เช่น กำหนดให้ ε=1) ซึ่งถ้ามีจะต้องเพิ่มข้อมูล "1" ลงไปที่ตำแหน่งนั้นเพื่อที่จะทำให้บรรทัดที่ถูกเลือกใช้ในการซ่อนข้อมูลนั้นแตกต่างจากบรรทัดที่ไม่ได้ถูกใช้ในการซ่อนข้อมูล
6. จากนั้นจัดข้อมูลที่ได้จากขั้นตอนที่ 5 ให้อยู่ในรูปแบบของบล็อกข้อมูลตามสมการที่ (4) โดยส่วนแรกจะเป็นส่วนของข้อมูลที่จะใช้ซ่อน (Data : D) และส่วนที่สองจะเป็นส่วนของข้อมูลที่ใช้ในการแก้ไขข้อมูลที่ผิดพลาด (Error correction : H) ซึ่งในวิธีการจะใช้วิธีแฮมมิง (Hammimg code (n,k)) [4] ในการสร้างข้อมูลในส่วนที่สอง

$$C_{(n,k)} = \left[D_{m \times (n-k)} \vdots H_{m \times k} \right] \qquad (4)$$

โดยวิธีการนี้จะปรับเปลี่ยนรูปแบบของบล็อกข้อมูลให้อยู่ในแนวตั้ง โดยมีรูปแบบดังสมการที่ (5) เพื่อเป็นการลดโอกาสการเกิดความผิดพลาดมากกว่าหนึ่งตำแหน่งภายในบล็อกข้อมูลเพื่อให้วิธีแฮมมิงทำงานได้อย่างมีประสิทธิภาพ

$$C'_{(7,3)} = C^{T}_{(7,3)} \qquad (5)$$

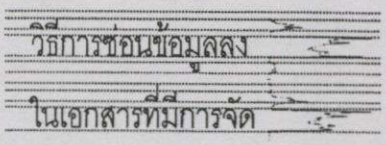
7. จากบรรทัดที่ถูกเลือกใช้ในการซ่อนข้อมูลตามขั้นตอนที่ 4 ทำการซ่อนข้อมูลที่ได้จากขั้นที่ 6 โดยการเลื่อนตำแหน่งของตัวอักษรที่อยู่ในระดับที่ 2 ตามแนวตั้ง เพื่อเปลี่ยนแปลงขนาดความกว้างของช่องว่างระหว่างระดับชั้นที่ 2 และ 3 โดยมีกฎดังนี้
   - ถ้าข้อมูลที่ต้องการจะซ่อนเป็น " 1 " ช่องว่างระหว่างระดับชั้น 2 กับ 3 จะมีขนาดเท่ากับ $\Delta + \delta$
   - ถ้าข้อมูลที่ต้องการจะซ่อนเป็น " 0 " ช่องว่างระหว่างระดับชั้น 2 กับ 3 จะมีขนาดเท่ากับ $\Delta - \delta$

   โดยที่ Δ คือ ค่าเฉลี่ยของความกว้างระหว่างระดับชั้น และ δ คือ ขนาดของช่องว่างที่ถูกเปลี่ยนแปลงตำแหน่งในหน่วยของพิกเซลซึ่งค่าที่เหมาะสมของ δ นั้นได้มาจากการทดลอง ซึ่งจากการทดลองกับเอกสาร CordiaUPC ขนาด 14 พอยท์ ซึ่งมีค่าความละเอียดเท่ากับ 300 จุดต่อนิ้ว พบว่าค่าของ $\Delta + \delta$ ที่เหมาะสมคือ 6 ส่วนค่าของ $\Delta - \delta$ คือ 3

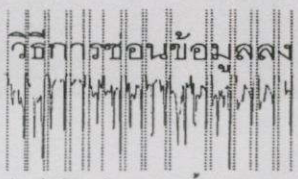
3.2 การถอดรหัสข้อมูล

ในที่นี้กำหนดให้เอกสารที่จะถูกถอดรหัสข้อมูลนั้นจะต้องผ่านกระบวนการปรับปรุงคุณภาพของเอกสารเรียบร้อยแล้ว [5] เช่น การกำจัดสิ่งรบกวนภายในเอกสาร (Salt-and-pepper noise removal) การหมุนเอกสารคืนกลับ (Skewing) ขั้นตอนในการถอดรหัสข้อมูลมีดังนี้

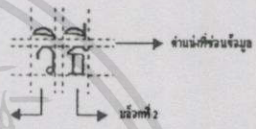
1. สร้างโครงร่างตามแนวนอนของหน้าเอกสารเพื่อกำหนดขอบเขตของแต่ละบรรทัดภายในหน้าเอกสาร
2. สร้างโครงร่างตามแนวตั้งของแต่ละบรรทัดที่ได้จากขั้นตอนที่ 1 เพื่อแบ่งตัวอักษรออกเป็นบล็อกๆ
3. สร้างโครงร่างตามแนวนอนของแต่ละบล็อกของตัวอักษร เพื่อระบุตำแหน่งของตัวอักษรที่อยู่ในแต่ละระดับ
4. ระบุบรรทัดที่ถูกใช้ในการซ่อนข้อมูล โดยพิจารณาจาก
   - จำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้จะต้องมีค่ามากกว่าหรือเท่ากับค่า L_s หรือ
   - ถ้าจำนวนบล็อกตัวอักษรที่สามารถซ่อนข้อมูลมีค่าอยู่ระหว่างค่า $L_s - \varepsilon$ และ L_s แล้วค่าของข้อมูลที่ถูกซ่อนจะต้องมีค่า "1" อยู่ด้วยอย่างน้อย 1 ตัว
5. จากบรรทัดที่ถูกระบุมาตามขั้นตอนที่ 4 ทำการถอดรหัสข้อมูลซึ่งในแต่ละบรรทัดนั้นจะถอดรหัสเพียง L_s ตำแหน่ง โดยการ



รูปที่ 2 ตัวอย่างโครงร่างตามแนวนอนของบรรทัด 2 บรรทัด



รูปที่ 3 ตัวอย่างโครงร่างตามแนวตั้งของตัวอักษรใน 1 บรรทัด



รูปที่ 4 ตัวอย่างบล็อกตัวอักษรที่สามารถซ่อนข้อมูลได้

คำนวณหาความกว้างระหว่างช่องว่างระดับชั้นที่ 2 และ 3 เพื่อกำหนดค่าของข้อมูลที่ถูกซ่อนโดยใช้หลักการดังนี้

- ถ้าช่องว่างระหว่างระดับชั้นที่ 2 กับ 3 มีขนาดมากกว่าหรือเท่ากับ Δ ค่าของข้อมูลที่ถูกซ่อนคือ " 1 "
- ถ้าช่องว่างระหว่างระดับชั้นที่ 2 กับ 3 มีขนาดน้อยกว่า Δ ค่าของข้อมูลที่ถูกซ่อนคือ " 0 "

6. จัดเรียงข้อมูลที่ได้จากขั้นตอนที่ 5 ให้อยู่ในรูปแบบดังสมการที่ (5) จากนั้นทำการตรวจหาตำแหน่งข้อมูลที่ผิดพลาดในแต่ละบล็อกเพื่อที่จะทำการแก้ไข โดยมีหลักการดังนี้
   - ระบุตำแหน่งที่เกิดความผิดพลาดภายในบล็อกข้อมูล ซึ่งตำแหน่งที่ถูกระบุนี้เรายังไม่ทราบว่าเป็นความผิดพลาดแบบใด (ข้อมูลเปลี่ยนไป หรือ ข้อมูลหายไป)
   - ภายใต้สมมติฐานที่ว่าค่าของข้อมูลที่ตำแหน่งนั้นมีเปลี่ยนไปจากเดิม ให้ทำการแก้ไขข้อมูลที่ตำแหน่งนั้นและตรวจนับจำนวนความผิดพลาดที่เกิดขึ้นในบล็อกถัดๆไป
   - ภายใต้สมมติฐานที่ว่าข้อมูลที่ตำแหน่งนั้นหาย ทำการเพิ่มข้อมูลที่ได้จากการแก้ไขข้อมูลแล้วลงที่ตำแหน่งนั้นและตรวจนับจำนวนความผิดพลาดที่เกิดขึ้นในบล็อกถัดๆไป
   - เปรียบเทียบจำนวนความผิดพลาดที่ได้จากทั้งสองสมมติฐาน และเลือกทำตามสมมติฐานที่มีความผิดพลาดน้อย
   - ทำตามขั้นตอนทั้งหมดซ้ำกับบล็อกข้อมูลอื่นๆที่เหลือทั้งหมด

CP044

เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย

ตารางที่ 1 ผลการถอดรหัสข้อมูลจากเอกสารจำนวน 1 หน้า (สำเนาเอกสารครั้งที่ 1 และ 3)

การถอดรหัสข้อมูล	เอกสารสำเนาครั้งที่ 1 (%)	เอกสารสำเนาครั้งที่ 3 (%)
ไม่ได้ใช้วิธีการแก้ไขข้อมูลที่ผิดพลาด	98.49 , 67.46 (บิต) , (ตัวอักษร)	95.66 , 51.42 (บิต) , (ตัวอักษร)
โดยใช้วิธีการแก้ไขข้อมูลที่ผิดพลาด	98.92 , 97.60 (บิต) , (ตัวอักษร)	96.94 , 89.68 (บิต) , (ตัวอักษร)
โดยใช้วิธีการแก้ไขข้อมูลที่ผิดพลาด + โปรแกรมตรวจคำผิด	98.40 (ตัวอักษร)	92.85 (ตัวอักษร)

หมายเหตุ : วิธีการแก้ไขข้อมูลที่ผิดพลาดที่ใช้ในการทดลองนี้คือ วิธีแฮมมิ่ง (7,3)

4. ผลการทดลอง

ในการทดลองนี้ได้ทำการซ่อนคำสำคัญของเอกสารลงในเอกสารภาษาไทยรูปแบบ CordiaUPC ขนาด 14 พอยท์ และมีค่าความละเอียดเท่ากับ 300 จุดต่อนิ้ว จำนวน 10 หน้า โดยทำการพิมพ์เอกสารที่ซ่อนข้อมูลด้วยเครื่องพิมพ์เลเซอร์ด้วยความละเอียด 600 จุดต่อนิ้ว และกำหนดให้เอกสารที่ได้จากการพิมพ์เป็นเอกสารสำเนาครั้งที่ 0 จากนั้นนำเอกสารนี้ไปถ่ายเอกสารและกำหนดให้เป็นเอกสารสำเนาครั้งที่ 1 ทำเช่นนี้ไปจนถึงสำเนาครั้งที่ 5 ตามลำดับ จากนั้นนำเอกสารทั้งหมดไปสแกนด้วยความละเอียด 450 จุดต่อนิ้วเพื่อทำให้เป็นเอกสารรูปภาพ จากนั้นทำการถอดรหัสข้อมูลที่ซ่อนออกจากเอกสารทั้งหมดโดยใช้วิธีการที่กล่าวมาข้างต้น ซึ่งผลที่ได้จากที่ได้จากการถอดรหัสข้อมูลในเอกสารสำเนาครั้งที่ 0 ถึงสำเนาครั้งที่ 3 พบว่าการระบุตำแหน่งบรรทัดที่ใช้ในการซ่อนข้อมูลนั้นสามารถระบุได้ถูกต้องทั้งหมด จะเห็นว่าเมื่อเอกสารผ่านกระบวนการถ่ายเอกสารจะทำให้เกิดสิ่งปลอมแปลง (Noise) ขึ้นในเอกสารทำให้เกิดความผิดพลาดในการถอดรหัสข้อมูล เราสามารถสรุปประเภทของความผิดพลาดในการถอดรหัสข้อมูลได้ 2 ประเภทคือ ความผิดพลาดที่ค่าของข้อมูลเปลี่ยนไปจากค่าเดิมและความผิดพลาดที่ค่าของข้อมูลหายไปเนื่องจากตำแหน่งของตัวอักษรที่อยู่ในระดับชั้นที่ 2 และ 3 นั้นอยู่ติดกัน ซึ่งเปอร์เซนต์ของการเกิดความผิดพลาดทั้งหมดสำหรับเอกสารสำเนาที่ 1 คือ 1.51 และสำหรับเอกสารสำเนาครั้งที่ 3 คือ 4.34

วิธีการที่กล่าวมาข้างต้นได้นำเสนอวิธีการแก้ไขข้อมูลที่ผิดพลาด โดยเราได้ทำการทดลองเพื่อพิสูจน์ว่าวิธีการข้างต้นนั้นสามารถแก้ปัญหาความผิดพลาดที่เกิดจากการถอดรหัสข้อมูลได้ ซึ่งผลที่ได้จากการถอดรหัสข้อมูลในเอกสารสำเนาครั้งที่ 1 สามารถแก้ไขความผิดพลาดได้ถูกต้องตามสมมติฐาน 80 % และสำหรับเอกสารสำเนาครั้งที่ 3 สามารถแก้ไขความผิดพลาดได้ถูกต้องตามสมมติฐาน 77 % อีกทั้งวิธีการนี้สามารถจำกัดขอบเขตของการเกิดความผิดพลาดให้อยู่ภายในบล็อกข้อมูลได้ เนื่องจากข้อมูลที่ได้จากการถอดรหัสนั้นจะถูกจัดรูปแบบให้เป็นบล็อกๆ เพื่อใช้ในการแก้ไขข้อมูลที่ผิดพลาด ดังนั้นความผิดพลาดที่เกิดขึ้นนี้จะมีผลกระทบกับแต่ละบล็อกข้อมูลเท่านั้น จากตารางที่ 1 แสดงผลการถอดรหัสข้อมูลของเอกสารสำเนาที่ 1 และ 3 จะเห็นว่าการนำวิธีแฮมมิ่งและการใช้โปรแกรมตรวจคำผิดเข้ามาช่วยในการถอดรหัสข้อมูลนั้นสามารถช่วยให้การถอดรหัสข้อมูลมีความถูกต้องมากขึ้น

5. บทสรุป

ในบทความนี้ได้เสนอเทคนิคการซ่อนข้อมูลสำหรับเอกสารรูปภาพที่ใช้ภาษาไทย โดยจะทำการซ่อนข้อมูลลงในช่องว่างระหว่างระดับชั้นของตัวอักษรซึ่งทำให้เกิดความเปลี่ยนแปลงของเอกสารเพียงเล็กน้อยเท่านั้น วิธีการนี้จะมีความทนในการซ่อนข้อมูลได้มากกว่าวิธีการอื่นๆ ที่ใช้อยู่ในปัจจุบันเนื่องจากมีตำแหน่งที่สามารถซ่อนข้อมูลได้เป็นจำนวนมาก แต่วิธีการนี้ยังคงมีปัญหาเมื่อนำมาใช้กับเอกสารที่ผ่านกระบวนการถ่ายเอกสารหลายๆครั้ง (เช่น เอกสารสำเนาครั้งที่ 4 และ 5) เนื่องจากเอกสารเกิดการบิดเบือนไปจากเดิมมากซึ่งส่งผลให้การถอดรหัสข้อมูลนั้นเกิดความผิดพลาดมากขึ้นเช่นกัน สำหรับแนวทางในการพัฒนาเทคนิคการซ่อนข้อมูลในอนาคตนั้นจะมุ่งเน้นเกี่ยวกับวิธีการซ่อนข้อมูลมีความคงทนต่อการที่เอกสารผ่านกระบวนการพิมพ์และการถ่ายเอกสารให้มากขึ้น

6. เอกสารอ้างอิง

[1] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, *"Techniques for data hiding,"* IBM Systems Journal 35(3/4), pp. 313-336,1996.

[2] J. Brassil, S. Low, N. Maxemchuk, L.O'Gorman, *"Electronic Marking and Identification Techniques to Discourage Document Copying,"* Proceedings of IEEE INFOCOM' 94, vol.3, Toronto, June 1994, pp.1278-1287.

[3] A. Nongnuch and C. Nopporn, "Data Hiding Techniques for Electronic Document with Multi-Tiered Alphabet," To appear in Proceedings of ISPAC'99, 1999.

[4] Man Young Rhee, "Error Correcting Coding Theory," McGraw-Hill Internation editions, pp. 35-86, 1989.

[5] L. O'Gorman and R. Kasturi, *"Document Image Analysis,"* In IEEE Computer Society Tutorial Series, IEEE 1994.

1999 IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS'99)

Wednesday, December 8 - Friday, December 10, 1999
Phuket Arcadia Hotel & Resort, Phuket, Thailand

ISPACS'99 Symposium Secretariat
Research Center for Communications and Information Technology (ReCCIT)
King Mongkut's Institute of Technology
Ladkrabang, Bangkok 10520, Thailand
Tel.: +662-7372500 Ext.5023, 5024
Fax.: +662-7392375
Email: ispacs99@kmitl.ac.th
http: //www.kmitl.ac.th/~reccit

Signal Processing and Communications Beyond 2000

Receipt No. 99-095

OFFICIAL RECEIPT

Date: 8 December 1999

Received from: Nongnuch Artwarin

Address: Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok 10520, THAILAND

Ref: Cheque / FEE-07

The amount of THB 3,000; In payment of registration/participation in the ISPACS'99, December 8-10, 1999, Phuket, Thailand.

Your registration detail is as:

Tutorial fee	
Conference fee	3,000
Others	
IN TOTAL	3,000 THB

S. Choomchuay

(Assist.Prof.Dr.Somsak Choomchauy)
Finance Chair

PROCEEDINGS

IEEE

ISPACS'99

1999 IEEE International Symposium on Intelligent Signal Processing and Communication Systems

Signal Processing and Communications Beyond 2000

December 8-10, 1999

Pattaya, Thailand

EIC JICA NECTEC NSTDA

ISPACS'99 FEE-07

Data Hiding Technique for Electronic Document with Multi-Tiered Alphabet

Nongnuch Artwarin and Nopporn Chotikakamthorn
Faculty of Information Technology & Research Center for Communication and Information Technology,
King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand
E-mail: Artwarin.Nongnuch@kmitl.ac.th

Abstract

The problem of document image data hiding is considered in this paper. Existing methods such as word shift coding technique, were designed for English document. It has limitation when applied to documents written in some oriental languages such as Thai. The problem is due to difference in language structure. In this paper we propose a data hiding technique for document image with multi-tiered alphabet. The method has high data embedding capacity, when applied to documents written in Thai. Experimental results are provided to demonstrate practical usefulness of the proposed method.

1. Introduction

Currently, advances in communication and media technologies have made electronic document distribution and publication increasingly popular. One of the advantages of electronic document distribution is to make large-scale document dissemination faster and less costly. On the other hand, it is vulnerable to copyright violation. Data embedding is a promising scheme as a solution to the problem [1]. Data hiding techniques can be used for the purpose of ownership verification and authentication, by adding a digital signature into a document image. In additional, data hiding methods can also be used to help searching and retrieving electronic documents, by adding information such as keywords into document. Unfortunately, most of the works on data hiding have been concentrated on media types such as image and video [2]. Moreover, existing data hiding techniques for text such as the line shift coding or word shift coding methods were designed for English document [3],[4],[5]. For a document written in other languages such as Thai, these methods are inefficient. In particular, the word shift coding method is applicable only to documents with space separating between words. Because documents written in Thai have only spaces between sentences, its application to Thai document results in a reduction of data embedding capacity. While the line shift coding method may be applicable in this case, the method's low embedding capacity limits its practical use. In this paper, we propose a new data hiding technique for document image in languages with multi-tiered alphabet. Here, the term 'multi-tiered alphabet' refers to a set of character symbols whose baseline location may be on different vertical levels. An example of such languages with multi-tiered alphabet is Thai (see Figure 1). The main idea of the proposed method is to embed data bits by altering the width of space which separates two character symbols on different vertical levels. In the next section, after providing notations used to describe document image and structure of a multi-tiered alphabet language, detail of the proposed method is described in Section 3. Decoding errors and remedies is provided in Section 4. Experimental result is provided in Section 5. Finally, conclusions is given in Section 6.

2. Document Image with Multi-Tiered Alphabet

In this paper, a document image is assumed to be coded in a bitmap format, and is expressed as

$$f(x,y)\in[0,1],\quad x=0,...,W-1,\ y=0,...,L-1 \qquad (1)$$

where W and L are the width and length of the page in pixels, and $f(x,y)$ is the value of each pixel. Because the proposed method exploits the document profiles, we define the horizontal and vertical profiles, respectively, by

$$h(y)=\sum_{x=0}^{W-1} f(x,y),\ y\in[0,L-1] \qquad (2)$$

$$v(x)=\sum_{y=0}^{L-1} f(x,y),\ x\in[0,W-1] \qquad (3)$$

Next, lets consider an example of languages with multi-tiered alphabet. From Figure 1, we notice that Thai language has spaces separating different vertical levels apart. It contains 3 or 4 levels for consonant, vowel and tonal mark. In the next section, we describe the proposed data hiding method for document image written in languages with the structure as shown in Figure 1.

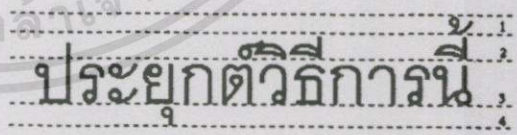


Fig. 1 Thai written language structure

3. Data Embedding Method for Languages with Multi-Tiered Alphabet

In this section, we describe a new document image data hiding technique for languages with multi-tiered alphabet. The technique adjusts the width of space between levels of character components on the same text line. The width of this space is changed by vertically shifting the upper level. In Thai language, we choose to modify space between the second and the third levels because it is wider than spaces between other levels and appears more often in typical. The

proposed method comprising the encoding and decoding parts, are described below.

Encoding part

This encoding procedure can be applied either to formatted files (e.g., postscript) or to a document with an image format (e.g., bitmap). In this paper, the bitmap format is assumed for simplicity. The encoding algorithm is described below.

1) Find a horizontal profile of a document to identify each text line boundary.
2) A vertical profile is calculated for each text line, as identified from Step 1. The profile is used for segmentation of each text line into character blocks.
3) From each block obtained from Step 2, find a horizontal profile again. By using this second horizontal profile, the locations (upper or lower) of character symbols are determined.
4) Check each text line whether it is suitable for data embedding. Here, we use only text lines containing at least L_d (e.g., 10) embeddable character blocks, which contain character symbols on both the 2nd and 3rd levels, well separated by vertical space. Each chosen text line is embedded with only L_d data bits. For any chosen text line of which the number of embeddable character blocks/locations is more than L_d, 0's will be embedded on the remaining embeddable locations. For the text lines which are not chosen (i.e., the ones with less than L_d embeddable locations), 0's are embedded on all available embeddable locations.
5) To embed a data bit, vertically shift the position of symbols on the 2nd level up or down relative to that of the 3rd third level, according to the following rules
   - If the data bit to be embedded is 1, white spacing width between the two groups of symbols (in pixel) is set to $\Delta + \delta$.
   - If the value of the data bit is 0, white spacing width between the two groups of symbols (in pixel) is set to $\Delta - \delta$.

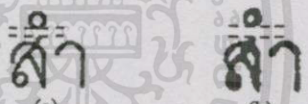
   Here, Δ is the mean value of the spacing width separating the two groups of (character) symbols, and δ is the (small) number of pixels shifted.

Decoding part Here, we assume that the document image to be decoded has already been enhanced by some image preprocessings [7] (e.g., binarization, salt-and-pepper noise removal, deskewing). The decoding steps are described below

1) Find a horizontal profile of a document to identify the boundary of each text line.
2) From each text line as identified from Step 1, find a vertical profile, in order to separate a text line into blocks of characters.
3) Find a horizontal profile of each character block as obtain from Step 2. The locations of symbols on the levels involved (2nd and 3rd) are determined.
4) Identify a text line which contains embedded data, by counting the number of embeddable locations. Any text line is considered containing useful embedded data if
   - the number of embeddable locations is equal or more than L_d, or
   - the number of embeddable locations is between $L_d - \varepsilon$ and L_d, and at least one of the embedded data bits is 1's. Here, ε is a small-value integer ($\varepsilon = 1$ in our study). Detailed discussion on the parameter ε will be given in the next section.
5) Find the width of white space separating two groups of symbols on the two different vertical levels (2nd and 3rd). The value of each embedded data bit is decoded using the following rule
   - If the white spacing width between the two symbol groups is at least $\Delta + \delta$, the embedded data bit is 1.
   - If the white spacing width between the two symbol groups is less than $\Delta - \delta$, the embedded data bit is 0.

4. Decoding Errors and Remedies

Decoding errors are due to noise and distortion occurred during document processing (e.g., photocopying, scanning). Most decoding errors can be categorized into two types. For the first error type, referred to later as type-I error, value of a data bit is changed from its original one. For example, the bit "0" is decoded as "1" in the decoding stage (see Figure 2). For the second error type, later referred to as type-II error, space separating the two symbol groups (on the two vertical levels as previously mentioned) disappears (see Figure 3). Not only the data bit embedded on the embedding location with this type-II error is missed, subsequent embedded data bits on the same text line are also subject to decoding error.



(a) (b)

Fig. 2 Example of type-I error (a) Original embedded document (embedded data is "0") (b) The 3rd copied document (embedded data changed to "1").

(a) (b)

Fig. 3 Example of type-II error (a) Original data embedded document which has space between levels (b) The 3rd copied document which has no space.

To reduce type-I errors, suitable error correction scheme may be applied. The problem due to type-II errors, however, is more difficult to solve. For instance, when any embedding location on a text line is undetected due to type-II error, the remaining embedded data bits on the same text line are vulnerable to decoding error (i.e., a kind of burst-error occurs). Thus, complicated error correction schemes are required to cope with burst errors. Unfortunately, these schemes require more redundancy information to be embedded, and thus reducing effective data embedding capacity. Moreover, if type-II error occurs on the text line

embedded with 0's, the number of embedding locations found during the decoding stage can be less than L_d. Under the decoding rule as described in 5), this text line will be faulty identified as a no-data-embedded line. Thus, all data bits embedded on that line are missed. However, under the assumption that no more than ε (=1 in our experiments) embedding locations are undetected (due to type-II error) in each text line, and provided that on each data-embedded text line there are at least ε+1 1's data bits, this ambiguity can be removed. To ensure that data bits embedded on each text line contain at least ε+1 1's, data bits to be embedded on each text line are observed first. If there are more than $(L_d - \varepsilon - 1)/2$ consecutive 0's in the data bit sequence, one 1's is inserted. By using this 'bit stuffing' method, it above requirement is met. Thus, with only ε embedding locations missed, the data-embedded text line still contains at least one 1's. This property can be used to differentiate data-embedded text lines from the ones without.

Another problem associated with type-II errors is the possibility of having burst errors. Instead of using sophisticated error correction schemes, we propose alternative solution which uses only a simple single-bit error correction method. To avoid burst-error phenomenon, we apply this single-bit error correction along a vertical direction instead. Details on how data bits and redundancy bits are arranged are given below.

After employing 'bit stuffing' scheme as mentioned, the data bit sequence, D, is arranged in a form of a $L_d \times (n-k)$ matrix, where n is the error correction block size, and k is the number of redundancy bits in each block. In this paper, we use Hammimg code (n,k), [6] to generate redundant error correction data (H). The resulting data matrix may be expressed by:

$$C = \left[D_{m \times (n-k)} \vdots H_{m \times k} \right] \tag{4}$$

$$= \begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,(n-k)} & h_{1,1} & h_{1,2} & \cdots & h_{1,k} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,(n-k)} & h_{2,1} & h_{2,2} & \cdots & h_{2,k} \\ d_{3,1} & d_{3,2} & \cdots & d_{3,(n-k)} & h_{3,1} & h_{3,2} & \cdots & h_{3,k} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ d_{m,1} & d_{m,2} & \cdots & d_{m,(n-k)} & h_{m,1} & h_{m,2} & \cdots & h_{m,k} \end{bmatrix} \tag{5}$$

We have performed experiments on some documents, after photocopied for the third time, to collect information on decoding error distribution. We found that errors distributed randomly over an entire page. However, by comparing between the number of errors occurred in each data block along a horizontal direction and those occurred in a block along a vertical direction, percentage of errors occurring more than once for data blocks along the horizontal direction is 26.0 % while that for blocks along the vertical direction is 13.5 %. Hence, instead of providing error correction along a horizontal direction, we propose instead the use of error correction for data block lied along a vertical direction. Example, of this error correction block arrangement using Hamming code (7,3) is shown in Eq. 6.

$$C'_{(7,3)} = C^{T}_{(7,3)} \tag{6}$$

At the decoding steps, the original data bit sequence can be recovered from the decoded data bits by taking the structure of the encoding data matrix (Eq. 6) into account.

With the error correction scheme as stated, one problem remains. The question is, provided that error is detected in any block using the above error correction method, the error found is due to either type-I or type-II error. It is important for any type-II error to be correctly identified, to reduce the chance of multiple-bit errors to occur on subsequent error correction blocks. Here, we propose the algorithm to identify the type of detected error, as follows:

1) With the first block with error found, the error location is identified (by the error correction method). Yet we do not know whether this is due to type-I or type-II error.
2) Correct the value of the data bit at the location identified from step 1 by :
   - Under the assumption that the error is of type-I, change detected value of the error bit. Then, perform error correction on subsequent blocks and count the number of blocks which are found (by error correction method) to contain error.
   - Under the assumption that the error is of type-II, insert new data bit with value as suggested by the error correction algorithm used. Then, perform error correction on subsequent blocks and count the number of blocks which are found (by error correction method) to contain error.
3) Compare the numbers of error blocks obtained as a result of applying the two hypotheses in step 2. Choose the hypothesis which results in less error blocks.
4) Repeat the above steps for the remaining data blocks.

5. Experimental result

We performed some experiments using the proposed data embedding method, applied to Thai document image. Ten document pages using the 14 point CordiaUPC font with 300 dpi resolution were used. Part of the original unmodified document image is shown in Figure 4-a and that of the embedded one is shown in Figure 4-b. As seen from the figure, the modifications made are unnoticeable. The document (in electronic form) was later printed with 600 dpi resolution, and named the 0^{th} copy. The first to fifth generation photocopied document pages, were subsequently scanned back with 450 dpi resolution. Example of the third generation photocopied document is shown in Figure 5. Experiment were performed using those copies of the same document. Hence, each successive experiments used a slightly more degraded version of the document. Table 1 summarizes decoding performance of the method as applied to the 1^{st} and 3^{rd} copies of the document. The first column displays the number of page used. The second column shows the percentage of bits correctly decoded for the 1^{st} and 3^{rd} copies. From the table, detection result is satisfactory for the

1st and 3rd copies. For the case where the embedded data bits represent words, some correction schemes (such as spell checking) may be applied to further improve the decoding performance. We also performed similar experiment with the 5th copy, however, excessive errors were found.

6. Conclusions

New document image data hiding technique for multi-tiered languages has been presented. The technique adjusts the width of white space separating two symbol groups on two different vertical levels, to embed each data bit. This technique can hide data with high embedding capacity, by exploiting particular structure of Thai written language. The technique has some limitation when applied to document which contains excessive noise and distortion (e.g., 4th and 5th photocopied documents in our experiment). This is due to mainly to the fact that the error correction used in our method can only correct a single error bit. Future work includes improvement on the method's robustness, as well as the application of the method for document retrieval and authentication.

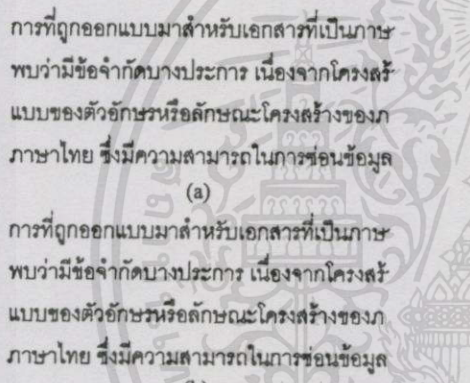


Fig. 4 (a) Original document, (b) document with data embedded

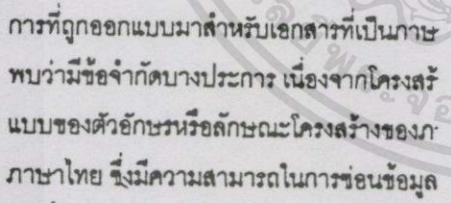


Fig. 5 Example of the third generation photocopy

Acknowledgement

This work was party supported by the Japan International Cooperation Agency.

Table 1 the percentage of bits correctly decoded on each page of the 1st and the 3rd copies

Page No.	The percentage of bits correctly decoded (%)	
	1st copy	3rd copy
1	100	100
2	100	98.6
3	100	98.6
4	100	96.7
5	100	98.6
6	97.1	90.5
7	100	100
8	100	100
9	100	99
10	100	100
Average	99.7	98.2

References

[1] Jiri Fridrich, "Method for data hiding," Center for intelligent system & department of systems science and industril engineering, SUNY Binghamton, 1997.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," IBM Systems Journal 35(3/4), pp. 313-336,1996.

[3] J. Brassil, S. Low, N. Maxemchuk, L.O'Gorman, "Electronic Marking and Identification Techniques to Discourage Document Copying," Proceedings of IEEE INFOCOM' 94, vol.3, Toronto, June 1994, pp.1278-1287.

[4] S. H. Low, N.F. Maxemchuk, J.T. Brassil, and L. O'Gorman, "Document Marking and Identification Using Both Line and Word Shifting," Proceeding of Infocom'95, April 1995.

[5] J. Brassil, S. Low, N.F. Maxemchuk, J.T. Brassil, and L. O'Gorman, "Hiding Information in Document Images," in Proceedings of 1995 Conference on Information Sciences and Systems, pp. 482-489, March 1995.

[6] Man Young Rhee, "Error Correcting Coding Theory," McGraw-Hill International editions, pp. 35-86,1989.

[7] L. O'Gorman and R. Kasturi, "Document Image Analysis," In IEEE Computer Society Tutorial Series, IEEE 1994.

ประวัติผู้เขียน

ชื่อผู้เขียน	นางสาวนงนุช อาจวารินทร์
วันเดือนปีเกิด	วันที่ 17 กรกฎาคม พ.ศ. 2517
สถานที่เกิด	จังหวัดชลบุรี
การศึกษาระดับปริญญาตรี	วิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์) เกียรตินิยมอันดับสอง ภาควิชา วิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิยาลัยกรุงเทพ ปีการศึกษา 2538
ผลงานด้านวิชาการ	1. "เทคนิคการซ่อนข้อมูลลงในเอกสารรูปภาพที่เป็นภาษาไทย" การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 22 (EECON'22) 2. "Data Hiding Technique for Electronic Document with Multi-Tiered Alphabet" 1999 IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS'99)