# การสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยการใช้เทคนิคในสพาเชียลโดเมน

# SPATIAL DOMAIN TECHNIQUE FOR DIGITAL WATERMARKING IMAGE

นางสาวกรรณิการ์ ทองวงค์ญาติ รหัส 42360396



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2546

# ใบรับรองโครงงานวิจัย

| | |
|---|---|
| **หัวข้อโครงงาน** | **การสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยการใช้เทคนิคในสพาเชียลโดเมน** |
| ผู้ดำเนินโครงงาน | นางสาวกรรณิการ์ ทองวงค์ญาติ รหัส 42360396 |
| อาจารย์ที่ปรึกษา | อาจารย์สุชาติ แย้มเม่น |
| สาขา | วิศวกรรมคอมพิวเตอร์ |
| ภาควิชา | วิศวกรรมไฟฟ้าและคอมพิวเตอร์ |
| ปีการศึกษา | 2546 |

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการตรวจสอบโครงงาน

.....ประธานกรรมการ

(อาจารย์ สุชาติ แย้มเม่น)

.....กรรมการ

(อาจารย์ ศิริพร เดชะศิลารักษ์)

.....กรรมการ

(อาจารย์ ไพศาล มุณีสว่าง)

| | |
|---|---|
| **หัวข้อโครงงาน** | การสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยการใช้เทคนิคในสพาเชียลโดเมน |
| **ผู้ดำเนินโครงงาน** | นางสาวกรรณิการ์ ทองวงค์ญาติ รหัส 42360396 |
| **อาจารย์ที่ปรึกษา** | อาจารย์สุชาติ แย้มแม่น |
| **สาขา** | วิศวกรรมคอมพิวเตอร์ |
| **ภาควิชา** | วิศวกรรมไฟฟ้าและคอมพิวเตอร์ |
| **ปีการศึกษา** | 2546 |

## บทคัดย่อ


โครงงานนี้เป็นการศึกษาและพัฒนาการสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยใช้เทคนิคในสพาเชียลโดเมน (spatial domain) เพื่อใช้กับไฟล์รูปภาพในการป้องกันการละเมิดลิขสิทธิ์ของไฟล์รูปภาพให้กับเจ้าของภาพ ทั้งภาพพิมพ์ลายน้ำที่มองเห็นได้ (visible watermarked) และภาพพิมพ์ลายน้ำที่มองไม่เห็น (invisible watermarked) ภาพพิมพ์ลายน้ำแบบที่มองเห็นได้จะกระทำโดยการนำภาพพิมพ์ลายน้ำฝังลงไปในภาพหลักในพิกเซลของภาพที่เลือกไว้ และภาพพิมพ์ลายน้ำแบบที่มองไม่เห็นจะกระทำโดยการแปลงเมตริกซ์ของข้อมูลของภาพพิมพ์ลายน้ำให้เป็นเวกเตอร์ แล้วทำการชุ่มพิกเซลของภาพหลักเพื่อนำภาพพิมพ์ลายน้ำไปฝัง การป้องกันการกู้กลับคืนภาพของภาพพิมพ์ลายน้ำแบบที่มองไม่เห็นจะใช้วิธีการสร้างสัญญาณรบกวนใส่ไปในเวกเตอร์ของภาพพิมพ์ลายน้ำและใช้ key เป็นตัวเซ็ตค่าสถานะของสัญญาณรบกวน

ผลที่ได้จากการทำโครงงานนี้แสดงให้เห็นว่าภาพพิมพ์ลายน้ำดิจิตอลแบบมองเห็นได้และมองเห็นไม่ได้สามารถถูกสร้างและกู้กลับคืนได้โดยใช้เทคนิคในสพาเชียลโดเมน สุดท้ายภาพพิมพ์ลายน้ำแบบมองไม่เห็นจะมีความปลอดภัยจากการถูกกู้กลับคืนได้ดีกว่าภาพพิมพ์ลายน้ำแบบที่มองเห็นได้

| | | | |
|---|---|---|---|
| **Project Title** | Spatial Domain Technique for Digital Watermarking Image | | |
| **Name** | Miss Kunnika | Thongvongyat | ID. 42360396 |
| **Project Advisor** | Dr. Suchart | Yammen | |
| **Major** | Computer Engineering | | |
| **Department** | Eleetrical and Computer Engineering | | |
| **Academic Year** | 2003 | | |


## ABSTRACT

This project studies and develops a digital watermarking image by using a spatial domain technique for preventing steal copy images obtained from unauthentic author for visible watermarked images as well as invisible watermarked images. A visible watermarked image is operated by embedding a watermarked image on selected pixels in an original image. An invisible watermarked image is obtained from reshaping a watermarked image from matrix to vector, then random pixels in an original image are embedded by the vector to obtain an invisible watermarked image. To recover an invisible watermarked image, pseudo-random noise for input on a vector of an invisible watermarked image is generated by using key to set status value of the noise.

The results obtained from this project show that visible digital watermarking images and invisible digital watermarking images are generated and recovered by using a spatial domain technique. Finally an invisiblc watermarked image has a better security than a visible watermarked image in term of recovering the image.

# กิตติกรรมประกาศ

ขอขอบพระคุณทุกๆท่านที่ทำให้โครงงานชิ้นนี้สำเร็จลงด้วยดี โดยเฉพาะอาจารย์สุชาติ แย้มเม่น ที่ให้คำแนะนำเกี่ยวกับปัญหาที่พบและช่วยหาทางแก้ไขให้ ตลอดจนสละเวลาให้คำปรึกษา เอาใจใส่ให้ความช่วยเหลืออย่างดีตลอดเวลาในการทำโครงงานนี้ รวมทั้งขอขอบพระคุณอาจารย์ในภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ทุกท่านที่คอยดูแลให้ความช่วยเหลือและให้ความสะดวกในการทำโครงงานนี้

นางสาวกรรณิการ์ ทองวงค์ญาติ

# สารบัญ

หน้า

**บทคัดย่อภาษาไทย**.....ก

**บทคัดย่อภาษาอังกฤษ**.....ข

**กิตติกรรมประกาศ**.....ค

สารบัญ.....ง

สารบัญตาราง.....ช

สารบัญรูป.....ซ

บทที่ 1 บทนำ.....1

1.1 ที่มาและความสำคัญของโครงการ.....1

1.2 วัตถุประสงค์.....1

1.3 ขอบข่ายงาน.....1

1.4 กิจกรรมการดำเนินงาน.....2

1.5 ผลที่คาดว่าจะได้รับ.....2

1.6 งบประมาณ.....2

บทที่ 2 ความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้อง

2.1 ความรู้พื้นฐานเกี่ยวกับภาพพิมพ์ลายน้ำดิจิตอล.....3

2.1.1 ความเป็นมา.....3

2.1.2 ประเภทของภาพพิมพ์ลายน้ำดิจิตอล.....4

2.1.3 เทคนิคของภาพพิมพ์ลายน้ำประเภทต่าง ๆ.....5

2.1.4 คุณสมบัติที่ภาพพิมพ์ลายน้ำควรมี.....6

2.1.5 การใช้ประโยชน์จากภาพพิมพ์ลายน้ำ.....6

2.2 ความรู้พื้นฐานเกี่ยวกับโหมดสี.....7

2.3 ไฟล์ฟอร์แม็ตของรูปภาพดิจิตอล (Digital Image File Format).....8

# สารบัญ (ต่อ)

หน้า


บทที่ 3 การออกแบบและการพัฒนา.....9

3.1 การออกแบบและพัฒนาการสร้างภาพพิมพ์ลายน้ำดิจิตอล-
โดยใช้เทคนิคในสพาเชียลโดเมนแบบที่มองเห็นได้.....10

3.2 การออกแบบและพัฒนาการกู้คืนภาพพิมพ์ลายน้ำดิจิตอล -
โดยเทคนิคในสพาเชียลโดเมนแบบที่มองเห็นได้.....15

3.3 การออกแบบและพัฒนาการสร้างภาพพิมพ์ลายน้ำดิจิตอล-
โดยเทคนิคในสพาเชียลโดเมนแบบที่มองไม่เห็น.....17

3.4 การออกแบบและพัฒนาการกู้คืนภาพลายน้ำดิจิตอล-
โดยเทคนิคในสพาเชียลโดเมนแบบมองไม่เห็น.....21

3.5 ขั้นตอนการออกแบบการทำงานของ –
Graphic User Interfacc (GUI).....24

บทที่ 4 ผลการทดลอง.....25

4.1 การทดสอบโปรแกรม ส่วนติดต่อผู้ใช้ (Graphic User Interface (GUI)).....25

4.2 การทดสอบการสร้างภาพพิมพ์ลายน้ำและ-
การกู้กลับคืนภาพพิมพ์ลายน้ำที่มองเห็นได้.....28

4.2.1 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
ในแบบ ออดินารี โอเวอร์แล็ปปิ้ง (Ordinary Overlapping).....28

4.2.2 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
แบบ โอเวอร์แล็ปปิ้ง (Overlapping).....30

4.2.3 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
แบบ ไลท์โอเวอร์แล็ปปิ้ง (Light Overlapping).....32

4.2.4 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
แบบเอดโอนรี (Edge Only).....33

**4.2.5 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
แบบ ฮาร์โมนี (Harmony).....34**

# สารบัญ (ต่อ)

หน้า

4.2.6 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ-
แบบ ทรีไดเมนชัน (Three Dimension).....36
4.3 การทดสอบการสร้างภาพพิมพ์ลายน้ำและ-
การกู้กลับคืนภาพพิมพ์ลายน้ำแบบที่มองไม่เห็น.....38

บทที่ 5 บทสรุป.....40
5.1 สรุปผล.....40
5.2 ปัญหาและแนวทางแก้ไข.....40

เอกสารอ้างอิง.....41

ภาคผนวก.....42
ภาคผนวก ก.....43
Index Source Code.....75

ประวัติผู้จัดทำโครงงาน.....76

# สารบัญตาราง


**ตารางที่** **หน้า**

1.1 กิจกรรมการดำเนินงาน.....2

4.1 ผลการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบมองเห็นได้ -
ที่สามารถกู้กลับคืนได้.....28

4.2 ผลการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็น -
ที่สามารถกู้กลับคืนได้.....28

# สารบัญรูป


**รูปที่** **หน้า**

**3.1 ขั้นตอนการสร้างภาพพิมพ์ลายน้ำแบบมองเห็นได้.....10**

**3.2 ขั้นตอนการกู้กลับคืนภาพลายน้ำแบบมองเห็นได้.....15**

**3.3 ขั้นตอนการสร้างภาพลายน้ำแบบมองไม่เห็น.....17**

3.4 ขั้นตอนการกู้กลับคืนภาพลายน้ำแบบมองไม่เห็น.....21

3.5 ขั้นตอนการทำงานของ GUI.....24

4.1 หน้าต่างของส่วนติดต่อผู้ใช้.....25

4.2 แสดงการเลือกการทำภาพพิมพ์ลายน้ำใน Picture1.....26

4.3 แสดงผลของการเลือกการทำภาพพิมพ์ลายน้ำใน Picture1.....26

4.4 แสดงการเลือกการทำการกู้กลับคืนภาพใน Picture1.....27

4.5 แสดงผลการเลือกการทำกู้กลับคืนภาพใน Picture1.....27

4.6 ภาพพิมพ์ลายน้ำออดินารี โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่กู้กลับคืนได้.....29

4.7 ภาพพิมพ์ลายน้ำออดินารี โอเวอร์แล็ปปิ้ง ที่กู้กลับคืนไม่ได้.....30

4.8 ภาพพิมพ์ลายน้ำโอเวอร์แล็ปปิ้งที่เข้มที่สุดที่กู้กลับคืนได้.....31

4.9 ภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้คืนได้.....31

4.10 ภาพพิมพ์ลายน้ำไลท์โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่กู้กลับคืนได้.....32

4.11 ภาพพิมพ์ลายน้ำไลท์โอเวอร์แล็ปปิ้ง ที่กู้กลับคืนไม่ได้.....33

4.12 ภาพพิมพ์ลายน้ำเอดโอนรี ที่เข้มที่สุดที่กู้กลับคืนได้.....34

4.13 ภาพพิมพ์ลายน้ำ เอดโอนรี ที่ไม่สามารถกู้กลับคืนได้.....34

4.14 ภาพพิมพ์ลายน้ำ ฮาร์โมนีที่เข้มที่สุดที่กู้กลับคืนได้.....35

4.15 ภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่กู้กลับคืนไม่ได้.....36

4.16 ภาพพิมพ์ลายน้ำทรีไดเมนชัน ที่เข้มที่สุดที่กู้กลับคืนได้.....37

4.17 ภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่ไม่สามารถกู้กลับคืนได้.....37

4.18 ภาพพิมพ์ลายน้ำ k=1.....38

**4.19 ภาพพิมพ์ลายน้ำ k=20.....39**

**4.20 ภาพพิมพ์ลายน้ำที่กู้คืนกลับไม่ได้.....39**

# บทที่ 1
# บทนำ

## 1.1 ที่มาและความสำคัญของโครงการ

ปัจจุบันนี้มีการใช้งานอินเตอร์เน็ตในการทำธุรกิจต่างๆหลายรูปแบบ หรือมีการนำเสนอผลงานต่างๆ รวมทั้งให้ความรู้แก่ผู้ใช้งานอินเตอร์เน็ตซึ่งก็ได้มีการนำรูปภาพต่างๆมาแสดงมากมายซึ่งภาพที่นำมาแสดงเหล่านี้สามารถที่จะถูกนำไปใช้ได้โดยบุคคลอื่นได้ง่ายซึ่งบุคคลเหล่านี้อาจนำภาพไปเป็นของตนเองหรือนำไปใช้ในทางธุรกิจโดยไม่มีการลงทุนอะไรเลย ทำให้เจ้าของภาพได้รับความเสียหาย

ในโครงงานนี้ผู้จัดทำมีความสนใจที่จะพัฒนาและสร้างภาพพิมพ์ลายน้ำเพื่อนำไปใส่ในไฟล์รูปภาพเพื่อใช้ในการป้องกันการขโมยลิขสิทธิ์รูปภาพซึ่งภาพพิมพ์ลายน้ำดิจิตอลของโครงงานนี้จะเป็นแบบ การใส่ภาพพิมพ์ลายน้ำลงไปในสพาเชียลโดเมน (Spatial Domain)

## 1.2 วัตถุประสงค์

1. เพื่อพัฒนาและสร้างภาพพิมพ์ลายน้ำ (Watermarked) โดยใช้เทคนิคการใส่ภาพลายน้ำใน Spatial Domain
2. เพื่อใช้สำหรับการอ้างสิทธิ์ของการเป็นเจ้าของไฟล์รูปภาพ
3. เพื่อลดปัญหาการละเมิดลิขสิทธิ์ของไฟล์รูปภาพ

## 1.3 ขอบข่ายงาน

โครงงานนี้เป็นการจัดทำภาพพิมพ์ลายน้ำดิจิตอลโดยเทคนิคการใส่ภาพพิมพ์ลายน้ำใน Spatial Domain

1. ทำภาพพิมพ์ลายน้ำแบบมองเห็นได้ (Visible Watermarked) และแบบมองไม่เห็น (Invisble Watermarked)
2. ทำการกู้กลับคืนภาพโดยดึงภาพพิมพ์ลายน้ำออกจากภาพหลักทั้งภาพพิมพ์ลายน้ำแบบมองเห็นได้และแบบมองไม่เห็น
3. จัดทำในส่วนของ Graphic User Interface สำหรับดำเนินการข้อ 1. และข้อ 2.

## 1.4 กิจกรรมการดำเนินงาน

ตารางที่ 1.1 กิจกรรมการดำเนินงาน

| กิจกรรม | พ.ย. | ธ.ค. | ม.ค. | ก.พ. | มี.ค. | เม.ย. | พ.ค. |
|---|---|---|---|---|---|---|---|
| 1. เขียนโครงร่างการทำงาน | ←→ | | | | | | |
| 2. ศึกษาและรวบรวมข้อมูล | ← | — | — | — | → | | |
| 3. ดำเนินงานพัฒนาและสร้างภาพพิมพ์ลายน้ำ | | | ← | — | — | → | |
| 4. ทดสอบและแก้ไข | | | | ← | — | → | |
| 5. จัดทำรูปเล่มรายงาน | | | | | ← | — | → |
| 6. จัดทำรายงานฉบับสมบูรณ์ | | | | | | | ←→ |

## 1.5 ผลที่คาดว่าจะได้รับ

1. สามารถสร้างภาพพิมพ์ลายน้ำดิจิตอลที่นำไปใช้งานได้จริง
2. เป็นแนวทางในการพัฒนาโปรแกรมการสร้างภาพพิมพ์ลายน้ำ ต่อไป
3. สนับสนุนให้ผู้ใช้มีความกระตือรือร้นในการร่วมกันต่อต้านการละเมิดลิขสิทธิ์ในทรัพย์สินทางปัญญาให้มากขึ้น

## 1.6 งบประมาณ

| | | |
|---|---|---|
| ค่าจัดทำรูปเล่ม | 500 | บาท |
| ค่าเอกสาร | 400 | บาท |
| ค่า CD โปรแกรม | 100 | บาท |
| รวมค่าใช้จ่าย | 1,000 | บาท |

(หนึ่งพันบาทถ้วน)

# บทที่ 2
# ความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้อง

## 2.1 ความรู้พื้นฐานเกี่ยวกับภาพพิมพ์ลายน้ำดิจิตอล

### 2.1.1 ความเป็นมา

ในปัจจุบันเทคโนโลยีต่างๆได้ก้าวหน้าไปเป็นอย่างมากมีน้อยคนที่จะไม่รู้จักและเคยใช้งานอินเตอร์เน็ตไม่ว่าจะเพื่อหาข้อมูล เพื่อการบันเทิงหรือธุรกิจล้วนหาได้จากอินเตอร์เน็ตแทบทั้งสิ้น ซึ่งจากการที่มีผู้ใช้อินเตอร์เน็ตเป็นอย่างมากนี้เองก็ได้ทำให้มีผู้ที่ต้องการนำเสนอผลงานการค้นคว้า หรือความรู้ต่างๆ หรือมีการทำธุรกิจบนอินเตอร์เน็ตซึ่งได้นำผลงานของตนเองมาแสดงทางอินเตอร์ซึ่งอาจจะเพื่อการค้าหรือเสนอผลงานก็แล้วแต่ ซึ่งผลงานที่นำมาเสนอเหล่านี้สามารถที่จะสำเนาเอาไปใช้ได้โดยง่ายซึ่งอาจมีคนนำไปเพื่อค้าขายหรือนำเอาไปเป็นของตนเองโดยที่ไม่ต้องมีการลงทุนอะไรเลยซึ่งก่อให้เกิดความเสียหายแก่ผู้สร้างและนำเสนอผลงานนั้น

การทำภาพพิมพ์ลายน้ำได้มีการค้นพบว่ามีมาตั้งแต่เมื่อประมาณ 700 ปีมาแล้วที่เมือง Fabirano ประเทศอิตาลีสิ่งที่พวกเขาคิดกันตอนนั้นคือทำอย่างไรจึงจะสามารถประทับตราที่เป็นสัญลักษณ์ที่มองไม่เห็นลงไปบนเอกสาร และพวกเขาก็ค้นพบคำตอบคือ ต้องทำให้ข้อความบางส่วนมีความบางของลายเส้นมากกว่าที่ส่วนอื่น เมื่อมองเอกสารนั้นในที่มีแสงสว่างมากๆ จะสามารถเห็นรูปแบบ (pattern) ของส่วนที่เป็นเส้นที่บางกว่าได้วิธีการที่ทำกับเอกสารในลักษณะนี้ถูกเรียกว่า watermarking และสิ่งที่ถูกแทรกเข้าไปในเอกสารถูกเรียกว่า watermark เพราะว่ามันดูเหมือนเป็นน้ำๆ แล้วเห็นอะไรรางๆในเอกสาร ทุกวันนี้การค้นหาวิธีการต่างๆ ในการทำภาพพิมพ์ลายน้ำยังคงมีต่อไปเพียงแต่ว่าข้อมูลข่าวสารในปัจจุบันส่วนใหญ่เป็นข้อมูลดิจิตอลแนวคิดเดิมจึงได้ถูกเปลี่ยนแปลงมาเป็น digital watermarking แทน

ส่วนในประเทศไทยได้มีการศึกษาการสร้างภาพลายน้ำดิจิตอลเพื่อใช้ในการป้องกันการขโมยลิขสิทธิ์ให้กับเทคโนโลยีมัลติมีเดียต่างๆได้มีขึ้นเมื่อประมาณสิบกว่าปีที่ผ่านซึ่งต่อมาก็ได้มีความสำคัญมากขึ้นเรื่อยๆไม่ว่าจะเป็น วีดีโอ วีดีโอซีดี เทป หรือไฟล์รูปภาพ ซึ่งภาพพิมพ์ลายน้ำดิจิตอลก็เป็นอีกวิธีหนึ่งที่ใช้กันมากกับไฟล์รูปภาพในปัจจุบันทำให้ผู้จัดทำมีความสนใจในโครงงานนี้ เหตุผลในการเลือกทำโครงงานภาพพิมพ์ลายน้ำดิจิตอลที่ใช้กับไฟล์รูปภาพนั้นก็เนื่องมาจากการใช้ไฟล์รูปภาพจะช่วยให้ง่ายในการนำเสนอผลงานง่ายต่อการศึกษาและทดลองและง่ายต่อการทำความเข้าใจด้วย

**2.1.2 ประเภทของภาพพิมพ์ลายน้ำดิจิตอล**

ประเภทหลักๆของภาพพิมพ์ลายน้ำดิจิตอลมีด้วยกันสองประเภทคือ แบบมองเห็นได้ (visible watermarked) และแบบมองไม่เห็น (invisible watermarked) ซึ่งทั้ง 2 ประเภทก็มีประโยชน์และเทคนิคในการทำที่แตกต่างกันออกไปซึ่งจะได้แสดงให้เห็นได้ดังต่อไปนี้

ภาพพิมพ์ลายน้ำแบบมองเห็นได้ (visible watermarked) จะกระทำโดยการเขียน เติม หรือแทรกข้อมูลลงในภาพโดยตรงซึ่งข้อดีของการทำภาพพิมพ์ลายน้ำแบบนี้ก็คือการใช้แสดงสิทธิ์การเป็นเจ้าของไฟล์รูปภาพ และยังช่วยยับยั้งการละเมิดลิขสิทธิ์ในเบื้องต้นได้อีกด้วยเพราะเมื่อบุคคลอื่นเห็นสัญลักษณ์ความเป็นเจ้าของภาพแล้วก็คงจะต้องคิดหนักพอสมควรในการจะนำไฟล์รูปภาพไปใช้หรือไปแก้ไขไม่เหมือนกับไฟล์รูปภาพที่เห็นแล้วไม่มีอะไรที่บ่งบอกความมีเจ้าของอยู่ ส่วนข้อเสียของการทำภาพพิมพ์ลายน้ำแบบนี้ก็คือจะทำให้คุณภาพของไฟล์รูปภาพลดลงไปอันเนื่องมาจากการใส่ภาพพิมพ์ลายน้ำลงไปนั่นเอง อย่างไรก็ตามการทำภาพพิมพ์ลายน้ำแบบมองเห็นได้ก็ยังเป็นที่นิยมเป็นอย่างมากเนื่องจากมีความง่ายและไม่ซับซ้อนมากในการทำ และถ้ามีการกระทำในการกำจัดภาพพิมพ์ลายน้ำออกก็อาจจะส่งผลต่อคุณภาพของรูปภาพนั้น ๆ ด้วย

ภาพพิมพ์ลายน้ำแบบมองไม่เห็น (invisible watermarked) เป็นการใส่สัญญาณภาพพิมพ์ลายน้ำลงไปในภาพหลักซึ่งในกระบวนการในการใส่จะใส่ key ซึ่งเป็นข้อมูลลับของเจ้าของไฟล์รูปภาพซึ่งอาจจะเป็นตัวเลขหรือไฟล์รูปภาพเล็ก ๆ ก็ได้ ลงไปด้วยในการเข้ารหัสสัญญาณเพื่อป้องกันไม่ให้ผู้ไม่หวังดีสามารถกู้หรือแก้ไขภาพพิมพ์ลายน้ำได้โดยไม่ได้รับอนุญาตจากเจ้าของ ซึ่งคนที่ไม่รู้ค่า key จะไม่สามารถกู้ภาพพิมพ์ลายน้ำได้ ซึ่งจะมีแต่เจ้าของภาพเท่านั้นที่จะรู้ข้อมูลที่ซ่อนอยู่ภายในไฟล์รูปภาพนั้น ข้อดีของการทำภาพพิมพ์ลายน้ำแบบนี้คือจะเป็นการใช้อ้างสิทธิ์ในไฟล์รูปภาพในภายหลังจากที่มีผู้ละเมิดลิขสิทธิ์แล้วเจ้าของภาพก็จะใช้การกู้กลับคืนภาพพิมพ์ลายน้ำเพื่อยืนยันการเป็นเจ้าของไฟล์รูปภาพ และการทำภาพพิมพ์ลายน้ำด้วยวิธีมองไม่เห็นนี้จะทำให้คุณภาพของไฟล์รูปภาพดีกว่าการใช้วิธีการทำภาพพิมพ์ลายน้ำแบบมองเห็นได้ด้วย

โดยทั่วไปวิธีการทำภาพพิมพ์ลายน้ำดิจิตอลมีด้วยกันสองแบบขึ้นอยู่กับขอบเขตที่ทำการใส่ภาพพิมพ์ลายน้ำดิจิตอล ดังนี้

1. Spatial Domain คือการใส่ภาพพิมพ์ลายน้ำดิจิตอลในสพาเชียลโดเมน (Spatial Domain) โดยจะทำการเลือกพิกเซลที่จะถูกทำการแก้ไขค่าโดยพิจารณาจากตำแหน่งที่พิกเซลนั้นอยู่ในรูปภาพ ซึ่งมีเทคนิคในการใส่ภาพลายน้ำลงในสพาเชียลโดเมนด้วยกันหลายวิธีเช่น LSB(Least Significant Bit Modification) , Correlation-Base Techniques และ CDMA Spead-Spectrum ภาพพิมพ์ลายน้ำที่ถูกกระทำในโดเมนนี้อาจสูญหายไปได้เมื่อผ่านการโจมตีเชิงสัญญาณ เช่นการกรองสัญญาณแต่จะทนต่อการโจมตีเชิงเรขาคณิต เช่นการหมุนภาพ การตัดภาพ เป็นต้น

2. Frequency Domain คือการใส่ภาพพิมพ์ลายน้ำดิจิตอลในฟรีเควนซี (Frequency Domain) โดยพิกเซลที่จะทำการแก้ไขค่าจะเลือกตามความถี่ในการเกิดของพิกเซลนั้น ซึ่งมีเทคนิค

ในการใส่ภาพพิมพ์ลายน้ำลงในฟรีเควนซีโดเมนด้วยกันหลายวิธีเช่น DCT (Discrete cosine transform) , Correlation-Base Teehnique และ CDMA Spead-Spectrum ภาพพิมพ์ลายน้ำที่ถูกกระทำในโดเมนนี้จะทนทานต่อการกรองสัญญาณความถี่ต่ำ การกรองความถี่สูง และการบีบอัดข้อมูลเป็นต้น

### 2.1.3 เทคนิคของภาพพิมพ์ลายน้ำประเภทต่าง ๆ

ตามที่กล่าวมาแล้วว่ามีเทคนิคในการใส่ภาพพิมพ์ลายน้ำลงบนภาพหลักด้วยกัน 2 วิธีคือวิธีการใส่ภาพพิมพ์ลายน้ำในสพาเชียลโดเมน (spatial domain) และฟรีเควนซีโดเมน (frequency domain) ซึ่งแต่ละวิธีการก็จะมีเทคนิคต่าง ๆในการใส่ภาพพิมพ์ลายน้ำซึ่งจะอธิบายคร่าว ๆ ได้ดังนี้

Spatial Domain :

- LSB substitution : โดยวิธีการนี้จะเป็นการแทนที่บิตที่มีความสำคัญน้อยที่สุดในภาพหลักด้วยบิตของภาพลายน้ำ
- Correlation based approach : เป็นวิธีการเปลี่ยนภาพลายน้ำให้เป็นสัญญาณรบกวนและนำไปใส่ลงในภาพพิมพ์ลายน้ำซึ่งในโครงงานนี้ใช้วิธีการนี้ในการทำภาพพิมพ์ลายน้ำที่มองไม่เห็น

Frequency Domain :

- DCT based approaches : เป็นวิธีการแปลงภาพที่มองภาพเป็นสัมประสิทธิ์ของความต่างความถี่ของโคซายน์ซึ่งจะแบ่งภาพออกเป็น 8x8 บล็อกจากนั้นก็จัดการแปลง (transformation) ในแต่ละบล็อกซึ่งมีด้วยกันหลายวิธีได้แก่
  - Mid-band Coefficient Exchange (MBCX)
  - Even-Odd Quantization (EOQ)
  - Differential Energy Watermarking (DEW)
  - CDMA
- Wavelet based approaches : เป็นวิธีการฝังภาพลงในบล็อกในแนวตั้ง (vertieal) LH ของ wavelet transforms ของรูปภาพ
- FFT based approaehes(Discrete Fourier transform based approaehes) : ภาพพิมพ์ลายน้ำที่ใส่ลงไปในภาพหลักจะเป็นสัญญาณ bandlimited ในความถี่การหมุนวนเป็นวงกลมรอบจุดศูนย์กลาง
- Fourier-Mellin transform based approaeh : เป็นวิธีการที่ใช้หลักการสร้าง แผนที่(map) Log Polar ของ FFT ของภาพหลักและฝังข้อมูลลงใน FFT ของ Log Polar Map

**2.1.4 คุณสมบัติที่ภาพพิมพ์ลายน้ำควรมี**

◆ Perccptual Transparcncy : ภาพลายน้ำที่ดีเมื่อนำไปใส่ลงในภาพไม่ควรจะมีผลกระทบต่อคุณภาพของภาพหลัก

◆ Robustness : ความแข็งแรงของภาพพิมพ์ลายน้ำเป็นเกณฑ์ที่ใช้วัดความสามารถของอัลกอริทึมของการฝังภาพพิมพ์ลายน้ำ ภาพพิมพ์ลายน้ำที่ดีควรมีความแข็งแรงทนทานต่อการรบกวนต่าง ๆ เช่น การย่อ-ขยายภาพ การหมุนภาพ การบีบอัดข้อมูล เป็นต้น

◆ Security : ความปลอดภัยจากการถูกผู้อื่นกู้ภาพ

◆ Payload of watermark :จำนวนของข้อมูลที่สามารถใส่ลงในภาพพิมพ์ลายน้ำขึ้นอยู่กับการประยุกต์ใช้งาน

◆ Oblivious vs. Non-oblivious : สามารถใช้อัลกอริทึมในการหาภาพพิมพ์ลายน้ำในภาพหลักที่ใส่ภาพพิมพ์ลายน้ำลงไป

โครงงานนี้จะใช้วิธีการใส่ภาพพิมพ์ลายน้ำลงใน Spatial Domain โดยพิกเซลที่จะนำมาทำการแก้ไขขึ้นอยู่กับตำแหน่งที่ต้องการวางภาพพิมพ์ลายน้ำ ซึ่งในโครงงานนี้จะใส่ภาพพิมพ์ลายน้ำลงใน Spatial Domain ด้วยวิธี Correlation Base Techniques ซึ่งจะใช้สมการในการใส่ภาพพิมพ์ลายน้ำลงในภาพหลักเป็น

$$I_{watermark}(x,y) = \alpha * I_{original}(x,y) + \beta * Watermark(x,y)$$

โดยที่ $I_{watermark}(x,y)$ คือ ภาพผลลัพธ์ที่ได้จากการใส่ภาพพิมพ์ลายน้ำลงในภาพหลัก

$I_{original}(x,y)$ คือ ภาพหลัก (host)ที่นำมาเพื่อใส่ภาพพิมพ์ลายน้ำลงไป

$Watermark(x,y)$ คือภาพพิมพ์ลายน้ำที่นำมาใส่ลงในภาพหลัก

$\alpha$ และ $\beta$ คือ สัมประสิทธิ์ที่มีค่าระหว่าง 0 และ 1

และในภาพพิมพ์ลายน้ำที่มองไม่เห็นจะมีการใส่ key ซึ่งจะเป็น ไฟล์ภาพเล็กๆ (ประมาณ 1x 35 pixels)หรือ เป็นรหัสลับที่เป็นตัวเลข เช่น 110878 ก็ได้ลงไปด้วยเพื่อเป็นการป้องกันการทำลายภาพลายน้ำจากผู้ไม่หวังดี โดยการนำ key นี้ไปเป็นสถานะของการสร้างสัญญาณรบกวนเพื่อนำไปใส่ลงในภาพพิมพ์ลายน้ำ

**2.1.5 การใช้ประโยชน์จากภาพพิมพ์ลายน้ำ**

สามารถแจกแจงเป็นข้อๆ ได้ดังนี้

1. Copyright Protection : เป็นการป้องกันการละเมิดลิขสิทธิ์ เจ้าของข้อมูลสามารถฝังภาพพิมพ์ลายน้ำลงไปเพื่อเป็นตัวแทนของการมีลิขสิทธิ์ (copyright information) ในข้อมูลได้

2. Fingerprinting : ผู้เป็นเจ้าของสามารถใช้วิธีการทำลายนิ้วมือที่พิมพ์ด้วยหมึก (fingerprinting) ซึ่งเจ้าของที่ต้องการคัดลอกผลงานเพื่อขายให้กับผู้ซื้อหลายๆคนซึ่งจะทำให้มีการฝังข้อมูลที่แตกต่างกันลงไปในสินค้าแต่ละชิ้น ซึ่งข้อมูลที่ฝังลงไปนี้อาจจะเป็น serial number หรือ customer id ก็ได้

3. Copy Protection : ภาพพิมพ์ลายน้ำที่ใช้ในตัวตรวจสอบว่าได้มีการอนุญาติให้ผู้คัดลอกมีสิทธิ์คัดลอกข้อมูลนั้นหรือไม่ซึ่งภาพพิมพ์ลายน้ำนี้จะใส่ไว้ในบิตห้ามคัดลอกข้อมูลที่เป็นบิตเดี่ยว (a single copy prohibit bit)

4. Broadcast Monitoring : ภาพพิมพ์ลายน้ำที่ฝังในสัญญาณการติดต่อสื่อสาร ซึ่งระบบเครื่องรับสัญญาณอัตโนมัติ (an automated monitoring system) สามารถที่จะกำหนดได้ว่าการติดต่อนั้นจะกระจายเสียงออกไปหรือไม่ เช่นการจัดโปรแกรมการนำเสนอรายการต่างๆของโทรทัศน์

5. Data Authentication : เป็นการนำภาพพิมพ์ลายน้ำที่อ่อนแอ (fragile watermark) ฝังลงในข้อมูลสามารถช่วยให้แน่ใจได้ว่าข้อมูลจะไม่ถูกเปลี่ยนแปลงโดยวิธีการใดๆจากผู้ใช้

6. Indexing : เป็นการฝังภาพพิมพ์ลายน้ำใน วิดีโอ เมล์ ภาพยนตร์ รายการข่าว สามารถใช้ประโยชน์จากดัชนีข้อมูลได้

7. Data Hiding : การทำภาพพิมพ์ลายน้ำอาจจะใช้วิธีฝังบิตของข้อมูลที่ต่อกันหลายๆบิตลงในข้อมูลเช่นการซ่อนชื่อของเจ้าของลงไปในจดหมายเป็นต้น

8. Medical Safety : ภาพพิมพ์ลายน้ำที่ประกอบไปด้วยชื่อของคนไข้สามารถฝังลงไปใน X-Rays , MRI Scans และผลการตรวจอื่น ซึ่งจะมีประโยชน์ในการช่วยทำให้รู้ว่าผลตรวจเป็นของผู้ป่วยคนใดจะได้ไม่วิเคราะห์ผลกับคนไข้ผิดคนซึ่งจะก่อให้เกิดผลเสียตามมาได้

## 2.2 ความรู้พื้นฐานเกี่ยวกับโหมดสี

1. อาร์จีบี (RGB) เป็นโหมดสีที่ใช้ในตารางการเทียบสีมีสีหลักของโหมดคือ สีแดง สีเขียว และสีน้ำเงิน

2 . อินเด็กซ์คัลเลอร์ (Index color) เป็นโหมดสีที่ใช้ในตารางในการเทียบสี โดยใช้ข้อมูลจำนวน 8 บิตต่อพิกเซล ภาพในโหมดนี้จะแสดงได้สูงสุดเพียง 256 สีต่อพิกเซล ($2^8$) บิต

3. บิตแม็พ (Bitmap) เป็นโหมดสีที่มีการเก็บข้อมูลของสีเพียง 1 บิตต่อพิกเซล ภาพในโหมดนี้สามารถแสดงได้เพียง สีขาวและสีดำ ภาพจะมีความหยาบมากที่สุดแต่ข้อดีคือภาพมีขนาดเล็ก

4 . เกรย์สเกล (Gray scale) เป็นโหมดสำหรับภาพขาวดำ สามารถไล่เฉดสีได้ถึง 256 ลำดับแม้ว่าโหมดนี้จะมีการใช้ข้อมูล 8 บิตต่อพิกเซล ในการเก็บข้อมูลเหมือนโหมดอินเดกซ์คัลเลอร์แต่ก็ไม่ได้ใช้ในตารางการเทียบสี

5. ซีเอ็มวายเค (CMYK) เป็นโหมดสีที่มี channel สีจำนวน 4 สี คือ ฟ้า บานเย็น เหลือง ดำ โดยแต่ละสีเก็บข้อมูล 8 บิตซึ่งก็คือโหมดนี้ต้องใช้ถึง 32 บิตต่อพิกเซล โหมดนี้ใช้มากในกระบวนการพิมพ์ ข้อเสียของโหมดนี้คือไม่สามารถแสดงสีที่เป็นธรรมชาติได้

6. แล็บคัลเลอร์ (LAB color) เป็นโหมดสีที่ให้สีเหมือนจริงมากที่สุด โหมดนี้จะใช้ค่า L (Lightness) แทนความสว่างโดยมีค่าตั้งแต่ 0-100 ค่า A แทนสีเขียวถึงสีแดง และค่า B แทนสีน้ำเงินถึงสีเหลือง ค่าของ A และ B จะมีค่าตั้งแต่ -120 ถึง +120

## 2.3 ไฟล์ฟอร์แม็ตของรูปภาพดิจิตอล (Digital Image File Format)

ไฟล์ฟอร์แม็ตของรูปภาพแบ่งเป็น 2 ประเภทใหญ่ ๆ คือ Bitmapped Format ซึ่งเป็นฟอร์แม็ตที่เก็บข้อมูลดิจิตอลของรูปภาพทั้งหมดและ Vector Format จะเก็บข้อมูลแต่ละองค์ประกอบแยกออกจากกัน ในที่นี้จะขอกล่าวถึงเฉพาะ Bitmapped Format ดังนี้

1. วินโดว์บีเอ็มพีฟอร์แม็ต (Window BMP Format) เป็นรูปแบบไฟล์มาตรฐานที่ใช้กันทั่วไปใน window , dos ไฟล์รูปแบบนี้รองรับโหมดสีแบบ RGB , Indexed color, Gray scale และ Bitmap

2. พีซีเอ็กซ์ฟอร์แม็ต (PCX Format) เป็นรูปแบบที่ใช้กับเครื่อง IBM Compatible ซึ่งโปรแกรมที่ใช้กับเครื่องคอมพิวเตอร์ทั่วไปจะรองรับไฟล์ PCX version5 ไฟล์ PCX รองรับโหมดสีแบบ RGB , Indexed color , Gray scale และ Bitmap

3. จีไอเอฟฟอร์แม็ต (GIF Format: Graphics Interchange Format) เป็นรูปแบบที่ใช้กันมากเพื่อแสดงภาพที่อยู่ในโหมด Indexed color และรูปภาพที่เป็นไฟล์เอกสารแบบ HTML (Hypertext Markup Language) ซึ่งใช้กันมากในเวปไซต์และบริการออนไลน์ต่างๆ ไฟล์รูปแบบนี้จะรองรับโหมดสี Bitmap , Gray scale และ Indexed color

4. ทีไฟเอฟเอฟ (TIFF:Tagged-Image File Format) ถูกใช้ในการแลกเปลี่ยนไฟล์ระหว่างโปรแกรมและ platform ของเครื่องรุ่นต่างๆ ไฟล์ TIFF รองรับโหมดสีแบบ CMYK , RGB , Gray scale , LAB color , Indexed color และ Bitmap

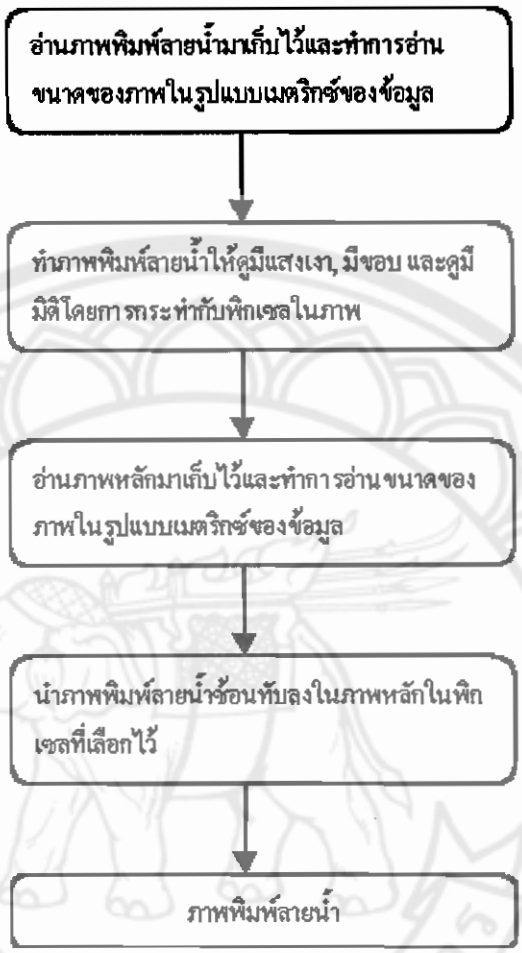
# บทที่ 3

# การออกแบบและการพัฒนา

ในการทำโครงงานในครั้งนี้ ผู้จัดทำได้ดำเนินงานต่างๆ เป็นขั้นตอน ดังต่อไปนี้

- ศึกษา ค้นคว้าการทำภาพพิมพ์ลายน้ำดิจิตอล โดยใช้วิธีการทำภาพพิมพ์ลายน้ำใน สพา-เชียลโดเมน
- ศึกษาการเขียน โปรแกรม เพื่อนำมาใช้ในการเขียนโปรแกรมการสร้างภาพพิมพ์ลายน้ำดิจิตอล โดยการใช้เทคนิคในสพาเชียลโดเมน
- ออกแบบโปรแกรมการสร้างภาพพิมพ์ลายน้ำดิจิตอล โดยการใช้เทคนิคในสพาเชียลโดเมน
- ออกแบบโปรแกรมในส่วนของ Graphic User Interface
- ทำการทดลองโปรแกรมบนระบบ Window ME

## 3.1 การออกแบบและพัฒนาการสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยใช้เทคนิคในสพาเชียลโดเมนแบบที่มองเห็นได้




รูปที่ 3.1 ขั้นตอนการสร้างภาพพิมพ์ลายน้ำแบบมองเห็นได้

ขั้นตอนการสร้างภาพพิมพ์ลายน้ำแบบที่มองเห็นได้ต่อไปนี้จะเป็นการทำให้ภาพพิมพ์ลายน้ำมองดูมีแสงเงา ,มีลักษณะนูน หรือมองเห็นเป็นขอบของภาพซึ่งจะใช้ในภาพพิมพ์ลายน้ำดิจิตอลแบบ Overlapping, Light Overlapping, Edge Only, Harmony และ Three Dimensions ส่วนภาพพิมพ์ลายน้ำแบบ Ordinary Overlapping นั้นไม่ต้องกระทำการใด ๆ กับภาพที่เป็นภาพพิมพ์ลายน้ำเลย ขั้นตอนการอธิบายเป็นดังต่อไปนี้

1. อ่านภาพพิมพ์ลายน้ำมาเก็บไว้ในที่นี้ใช้ภาพขนาด 128x128 pixels

```
[X,map] = imread('logo9_2.bmp','bmp');
```

**2. แปลง class ของภาพพิมพ์ลายน้ำให้เป็น class double เพื่อที่เราจะได้จัดการกับพิกเซลของภาพเพื่อให้ได้ภาพพิมพ์ลายน้ำตามที่ต้องการซึ่ง class เดิมของภาพคือ class uint8**

**X = double(X);**

**3. อ่านค่าขนาดของภาพพิมพ์ลายน้ำซึ่งจะได้เมตริกซ์ของแถวคูณหลักของของภาพลายน้ำในclass double**

**[row,eolumn] = size(X);**

**4. ทำให้ภาพดูมีสักษณะนูนขึ้นด้วยการแบ่งพิกเซลของภาพออกเป็นบล็อกๆที่เท่ากันและก็** จัด การกับพิกเซลเหล่านั้นดังนี้ (row , column) => (i , j)

4.1. ในพิกเซลทางด้านแถวเลือกพิกเซลในตำแหน่งแถวที่ 3 ถึง ตำแหน่งแถวที่จำนวนแถวของภาพทั้งหมดลบด้วย 2 : for i =3 : row-2 ถึงตอนนี้เราจะได้ค่าของแถวเป็น 126 แถวซึ่งจากเดิมมี 128 แถว (จากภาพ 128x128 pixels) ส่วนในพิกเซลของทางด้านหลักเลือกพิกเซลในตำแหน่งหลักที่ 3 ถึงตำแหน่งหลักที่จำนวนหลักทั้งหมดลบด้วย 2 : for j=3 : column-2 ถึงตอนนี้เราจะได้ค่าของหลักเป็น 126 หลักจากเดิม 128 หลัก

4.2. กำหนดให้ค่าคงที่ในการคำนวณหาค่าพิกเซลเป็น 0

4.3. จากนี้เราจะได้วนลูบเพื่อกำหนดให้พิกเซลใดมีค่าเป็น 0 หรือ 1 โดยเราจะกำหนดพิกเซลในลักษณะเยื้องกับภาพเดิม โดยกำหนดพิกเซลของแถวตั้งแต่ แถวที่ i-2 : i-2 กำหนดพิกเซลของหลักตั้งหลักที่ j-2 : j+2 และให้เซตค่าของแถวและหลักที่มีค่าน้อยกว่า 100 ให้เป็น 0 และเซตค่าที่เหลือทั้งหมดให้เป็น 1 ซึ่งพิกเซลที่ถูกกำหนดค่าให้เหล่านี้จะนำเอาไปรวมกับค่าคงที่ จากข้อ 4.2 ในทุกๆค่าของพิกเซลเพื่อหาผลรวมของพิกเซลที่เราสนใจสำหรับเป็นค่าสีของอินพุทในพิกเซลที่เราต้องการ

4.4. จากนั้นเราต้องหาค่าสีสำหรับเป็น อินพุทใน (i , j) ที่จะทำให้เกิดแสงเงาโดยการนำค่าผลรวมของค่าคงที่ที่ได้จากข้อ 4.3 มาหารด้วย 25 ซึ่งเป็นจำนวนพิกเซลและคูณด้วย 255 ซึ่งจะเป็นค่าสีที่จะใช้ใส่ลงในตำแหน่งที่เราจะฝังภาพพิมพ์ลายน้ำลงในภาพหลักทำให้ภาพพิมพ์ลายน้ำและภาพหลักดูกลมกลืนกัน

4.5. จากนั้นเราก็จะมากำหนดค่าให้กับค่าคงที่ซึ่งจะทำให้ภาพพิมพ์ลายน้ำดูเข้มขึ้นหรือจางลงโดยเราสามารถจะกำหนดได้ว่าจะให้ค่าคงที่นี้มีค่ามากกว่าเท่าใดในพิสัยตั้งแต่ 0 ถึง 25 โดยค่ายิ่งต่ำมากๆจะทำให้ภาพยิ่งจางลง ส่วนค่ายิ่งมากจะทำให้ภาพลายน้ำมีลักษณะเข้มขึ้นและกลับค่าของพื้นหลังของภาพพิมพ์ลายน้ำจากสีขาวให้กลายเป็นสีดำโดยกำหนดให้ค่าที่ได้จากการคำนวณ**หาค่าสีสำหรับอินพุทใน ( i , j) ในข้อ 4.4 ให้มีค่าเป็น 0 ดังนี้**

```
Z = (k/25)*255;
if  k>16  % เราสามารถปรับค่านี้เพื่อความเข้มมากน้อยของภาพพิมพ์ลายน้ำ
      Z=0 ;  % กลับค่าสีของพื้นหลังจากขาวเป็นดำ
```

```
end
```

4.6. จากนั้นคัดลอกภาพ ในแบบ binary ไปใส่ใน slide ทั้งสามของภาพ

```
B(i,j,1) = Z;    % slide 1
B(i,j,2) = Z;    % slide 2
B(i,j,3) = Z;    % slide 3
```

4.7. เราจะทำให้เกิดมีแสงและเงาในภาพพิมพ์ลายน้ำเราจะต้องกลับภาพจากดำเป็นขาว และจากขาวเป็นดำและทำการเลื่อนตำแหน่งของภาพ แล้วนำไปลบออกจากค่าของ B ดังนี้

```
D(i,j,1) =B(i,j,1)-(255-logo(i+2,j+2)); % (i +2 , j+2) เป็นการเลื่อนตำแหน่งภาพ
D(i,j,2) = B(i,j,2)-(255-logo(i+2,j+2));
D(i,j,3) =B(i,j,3)-(255-logo(i+2,j+2));
```

ขั้นตอนที่กล่าวมาทั้งหมดตั้งแต่ข้อ 4.1 ถึงข้อ 4.7 นำมาเขียนเป็นโปรแกรมได้ดังนี้

```
for i = 3 : row-2 % considerated pixel at position rows 3 to row-2(amount of 25 pixel)
 for j = 3 : column-2% considerated pixel at position column 3 to row-2
   k=0;
   for n = i-2 : i+2
     for m = j-2 : j+2
      if logo(n,m) <100
         C = 0;
      else
         C = 1;
      end
      k = k+C; % sum for find total pixel at considerated for input color valued to
      % that pixel consider
     end
   end
%find color valued for input in (i,j) that  make shading color
   %:k/25 is total pixel that consider,255 is for will
   %recieve color value input into position's pixel embedded and 10 is vary
   %valued can any change (for adapt shade color)
   Z = (k/25)*255;
% make logo look has dimension,range k = 0-24 <0.5 is invisible>
   if k > 20
```

```
Z = 0;% convert background from whitc to black color
end
%copy binary image to 3 slide image
B(i,j,1) = Z;
B(i,j,2) = Z;
B(i,j,3) = Z;
% goal for make shadow and light on watermark image
% convert image from white to black,black to white ,
% +2 is sliding position of character and
% subtract from B because want shadow & light
%  D(i,j,1) = B(i,j,1)-(255-logo(i+2,j+2));
% D(i,j,2) = B(i,j,2)-(255-logo(i+2,j+2));
%D(i,j,3) = B(i,j,3)-(255-logo(i+2,j+2));
D(i,j,1) = B(i,j,1)-(logo(i,j));
D(i,j,2) = B(i,j,2)-(logo(i,j));
D(i,j,3) = B(i,j,3)-(logo(i,j));
end
end
```

5. ซึ่งเราก็จะได้ไฟล์ของภาพมาด้วยกัน 2 ไฟล์คือ B(i, j, k) และ D(i, j, k) ซึ่งไฟล์ภาพทั้งสองนี้เราจะนำไปทำภาพพิมพ์ลายน้ำที่ใส่ลงใน Host Image ซึ่งไฟล์ภาพทั้งสองนี้สามารถนำมารวมกันและคูณด้วยค่าสัมประสิทธิ์ที่คงที่ค่าใดๆเพื่อให้เป็นภาพพิมพ์ลายน้ำที่สวยงามได้ ซึ่งเราต้องทำการ save ไฟล์ภาพพิมพ์ลายน้ำทั้งสองไฟล์นี้เก็บไว้เพื่อไว้ใช้ในขั้นตอนการกู้กลับคืนภาพพิมพ์ลายน้ำ

```
imwrite(uint8(B),'Bmarked1.bmp','bmp');
imwrite(uint8(D),'Dmarked1.bmp','bmp');
```

6. หลังจากได้ภาพพิมพ์ลายน้ำตามต้องการแล้วเราก็จะนำมาฝังลงในภาพหลัก โดยการอ่านภาพหลักเข้ามาโดยไฟล์ภาพที่จะนำมากระทำการต่อกัน (บวกหรือ ลบ) จะต้องมีมิติที่เท่ากันและต้องเป็น class double เนื่องจากที่ได้ลองเขียนโปรแกรมดูแล้ว จะรู้ว่า class uint8 ซึ่งเป็น class **ของไฟล์ภาพไม่สามารถนำมาบวกหรือลบกันได้ต้องทำให้เป็น class double**

```
[Y,map]=imread('lena.bmp','bmp');
Y=double(Y);
```

7. จากนั้นก็ทำการเลือกตำแหน่งในภาพหลักที่จะใส่ภาพพิมพ์ลายน้ำลงไปตามสมการต่อไปนี้

$$W = \alpha * I_w + \beta * I_H ;$$

โดยที่ W คือภาพที่ได้จากการใส่ภาพพิมพ์ลายน้ำลงในภาพหลักแล้ว

$\alpha$ และ $\beta$ คือค่าคงที่ใดๆที่เรานำมาปรับให้ภาพดูคมชัดหรือจาง ซึ่งจากการทดสอบโปรแกรมแล้วได้ผลดี(เพื่อที่จะให้ภาพ พิมพ์ลายน้ำนี้กู้ภาพกลับคืนได้) ก็คือให้ค่า $\alpha$ มีค่าต่ำๆ และค่าของ $\beta$ มีค่าสูงโดยที่ค่าของทั้งสองจะต้องอยู่ระหว่าง 0-1 เท่านั้น

$I_w$ คือภาพพิมพ์ลายน้ำที่จะนำมาใส่ในภาพหลัก

$I_H$ คือภาพหลักหรือ Host Image

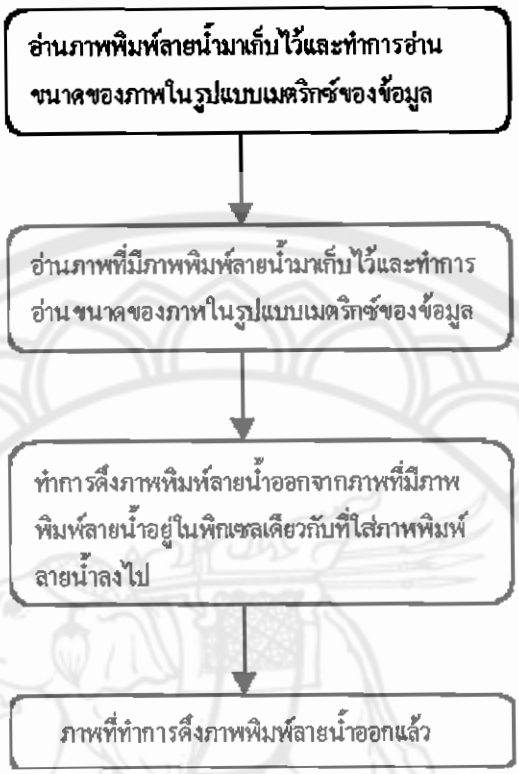
ซึ่งจะแสดงเป็นโปรแกรมได้ดังต่อไปนี้

```
[row1,column1,slide1]=size(B);
%****** P and Q are position that put watermark in host image *****
P=60; %select for move ROW   (P<(Row of background - Row of Text))
Q=110; %select for move COLUMN(Q<(Column of background - Column of Text))
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
          T(i,j,k)=A*B(i,j,k) + B*D(i,j,k); % A,B can change
        W(i+P,j+Q,k)=(α *T(i,j,k))+( β*Y(i+P,j+Q,k));
    end
  end
end
```

8. ได้ไฟล์รูปภาพที่มีการฝังภาพพิมพ์ลายน้ำลงไปแล้ว ทำการ save ไว้เพื่อใช้งาน

```
imwrite(uint8(W),'marked6.bmp','bmp');
```

## 3.2 การออกแบบและพัฒนาการกู้คืนภาพพิมพ์ลายน้ำดิจิตอลโดยเทคนิคในสพาเชียลโดเมนแบบที่มองเห็นได้




รูปที่ 3.2 ขั้นตอนการกู้กลับคืนภาพลายน้ำแบบมองเห็นได้

เป็นการทำกลับกับการทำภาพพิมพ์ลายน้ำที่มองเห็นได้ก็คือเราอ่านภาพพิมพ์ลายน้ำและภาพ ที่ต้องการทำการกู้ กลับคืนขึ้นมาจากนั้นก็ทำกลับสมการของการฝังภาพลายน้ำลงบนภาพหลักในตำแหน่งที่ใส่ภาพลายน้ำลงในภาพหลักดังสมการต่อไปนี้

$$I_H = ( W - \alpha * I_W ) / \beta ;$$

1. ทำการอ่านภาพพิมพ์ลายน้ำที่ได้จากกระทำในกระบวนการสร้างภาพพิมพ์ลายน้ำซึ่งเป็นไฟล์ภาพสองไฟล์คือ B(i, j, k) และ D(i, j, k) มาเก็บไว้

```
[B,map]=imread('Bmarked1.bmp','bmp');
[D,map]=imread('Dmarked1.bmp','bmp');
```

2. แปลง class ของภาพพิมพ์ลายน้ำให้เป็น class double เพื่อที่เราจะได้จัดการกับพิกเซลของภาพเพื่อให้ได้ภาพพิมพ์ลายน้ำตามที่ต้องการซึ่ง class เดิมของภาพคือ class uint8 และอ่านขนาดของไฟล์ภาพในรูปแบบเมตริกซ์ของข้อมูล

```
B=double(B);
D=double(D);
[row,column,slide]=size(D);  % can read from  B or D
```

3. ทำการอ่านไฟล์ภาพที่มีภาพพิมพ์ลายน้ำฝังอยู่

```
[W,map]=imread('markcd6.bmp','bmp');
```

4. แปลง class ของไฟล์ภาพให้เป็น class double

```
W=doublc(W);
```

5. ทำการดึงภาพพิมพ์ลายน้ำออกจากไฟล์ภาพโดยกำหนดตำแหน่งให้ตรงกับตำแหน่งที่ใส่ภาพพิมพ์ลายน้ำลงไป

```
P=60; %sclect for move ROW   (P<(Row of background - Row of Text))
Q=110; %select for move COLUMN(Q<(Column of background - Column of Text))
Y=W;
for k=1:3
   for i=1:row
         for j=1:column
      T(i,j,k)=(A*B(i,j,k))+(B*D(i,j,k));
      Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-( α*T(i,j,k)))/ β;
       end
    cnd
end
```

6. ได้ภาพที่ดึงภาพพิมพ์ลายน้ำออกแล้ว

## 3.3 การออกแบบและพัฒนาการสร้างภาพพิมพ์ลายน้ำดิจิตอลโดยเทคนิคในสพาเชียลโดเมนแบบที่มองไม่เห็น


รูปที่ 3.3 ขั้นตอนการสร้างภาพลายน้ำแบบมองไม่เห็น

มีขั้นตอนการพัฒนาดังต่อไปนี้

1. กำหนดค่าให้กับค่า เกรน(gain factor) สำหรับใช้ในการฝังภาพพิมพ์ลายน้ำลงบนภาพหลักและเซตขนาดของบล็อกในภาพหลักซึ่งจะถูกใช้ในแต่ละบิตของภาพพิมพ์ลายน้ำซึ่งในโปรแกรมนี้เซตให้บล็อกมีขนาด 16

```
k=20;          % set the gain factor for embedding
blocksize=16;  % set the size of the block in cover to be used for each bit in watermark
```

2. อ่านภาพหลักเข้ามาเก็บไว้และทำการอ่านขนาดของภาพหลักซึ่งจะได้เมตริกซ์ของแถวคูณกับหลัก และทำการกำหนดขนาดของภาพพิมพ์ลายน้ำที่ใหญ่ที่สุดที่สามารถนำมาฝังลงในภาพหลักได้ซึ่งทั้งภาพหลักและภาพพิมพ์ลายน้ำต้องมีมิติและขนาดของเมตริกซ์ที่สอดคล้องกันซึ่งในโปรแกรมการฝังภาพพิมพ์ลายน้ำนี้กำหนดให้ขนาดใหญ่ที่สุดของภาพพิมพ์ลายน้ำมีค่าเท่ากับขนาดของแถวและหลักของภาพหลักคูณกันและหารด้วยขนาดของบล็อกยกกำลัง 2

```
[host,map]=imread('pout.tif','tif');
host=double(host);
% determine size of watermarked image
 [Mc,Nc]=size(host);
% determine maximum message size based on host, and  blocksize
max_logo=Mc*Nc/(blocksize^2);
```

3. อ่านภาพพิมพ์ลายน้ำเข้ามาเก็บไว้และทำการอ่านขนาดภาพของภาพพิมพ์ลายน้ำซึ่งขนาดของภาพพิมพ์ลายน้ำที่ใช้ทดสอบโปรแกรมจะใช้ขนาด 29 x12 พิกเซลซึ่งจะเข้ากันได้กับขนาดของภาพหลักโดยใช้ภาพหลักขนาด 400x400 พิกเซลและทำการแปลงเมตริกซ์ของภาพพิมพ์ลายน้ำให้เป็นเวกเตอร์

```
[logo,map]=imread('_logo16_1.bmp','bmp');
logo=double(logo);
 [Mm,Nm]=size(logo);
% reshape the message to a vector
logo=round(reshape(logo,Mm*Nm,1)./256);
% pad the message out to the maximum message size with 0's
logo_vector=ones(1,max_logo);
logo_vector(1:length(logo))=logo;
```

4. กำหนดค่า key สำหรับการสร้างสัญญาณรบกวนเพื่อใส่เข้าไปในภาพพิมพ์ลายน้ำเพื่อป้องกันการลักลอบการกู้ภาพคืนจากบุคคลอื่นที่ไม่ได้รับอนุญาติซึ่งคนที่ไม่รู้ค่าของ key นี้จะไม่สามารถทำการกู้ภาพคืนได้ ซึ่งค่า key นี้มีประโยชน์อย่างมากในการใช้เพื่อพิสูจน์การอ้างสิทธิ์ใน

ภาพของตนเอง ซึ่ง key นี้เราจะกำหนดเป็นตัวเลขหลาย ๆ หลักก็ได้เช่น 45613 หรือจะใช้เป็นแบบรูปภาพขนาดเล็กๆแทนก็ได้เช่นไฟล์ภาพขนาด 1x35 พิกเซล

```
key=14641;  % assumed key
rand('state',key);
```

หรือจะใช้การเรียกไฟล์ภาพ ขนาด 1x35 พิกเซลดังนี้

```
[key,map]=imread('_key.bmp','bmp');
key=double(key)./256;
rand('state',key);
```

5. สร้างสัญญาณรบกวนเพื่อนำไปใส่ในภาพลายน้ำ ด้วยการสุ่มข้อมูลในเมตริกซ์ของขนาดของบล็อกที่เราได้กำหนดไว้ในตอนแรกนำค่าเหล่านั้นมาลบออกด้วย 0.5 และคูณด้วย 2 จากนั้นทำให้ผลลัพธ์ที่ได้มาให้เป็นจำนวนเต็มที่มีค่าใกล้เคียงกับค่าผลลัพธ์นั้น

```
pn_sequence_one=round(2*(rand(blocksize,blocksize)-0.5));
pn_sequence_zero=round(2*(rand(bloeksize,bloeksize)-0.5));
% find two highly un-correlated PN (PN=Pseudo-Random Noise) sequenccs
while (corr2(pn_sequence_one,pn_sequence_zero) > -0.1)
   pn_sequence_one=round(2*(rand(blocksize,bloeksize)-0.5));
   pn_sequence_zero=round(2*(rand(blocksize,blocksize)-0.5));
end
```

6. สร้างหน้ากากของภาพลายน้ำ (watermark mask) ขึ้นมาโดยการกระทำกับบล็อกของภาพพิมพ์ลายน้ำซึ่งในตอนนี้ภาพพิมพ์ลายน้ำจะมีลักษณะเป็นเวกเตอร์โดยกระทำกับบิตของ 0 ในเวกเตอร์ของภาพพิมพ์ลายน้ำ ซึ่งก็คือถ้าบิตใดๆของภาพพิมพ์ลายน้ำมีค่าเป็น 0 ก็จะใส่สัญญาณรบกวนลงไปส่วนในบิตอื่นที่ไม่มีค่าเป็น 0 ก็จะทำการใส่ค่า 0 ลงไปแทนที่ค่าเดิมจากนั้นก็เคลื่อนย้ายบล็อกที่อยู่ถัดไปไปตาม ค่าของ x โดยกำหนดให้ค่า x มีค่าเป็น 1 ถ้าสิ้นสุดแถวแล้วก็ให้เคลื่อนย้ายไปยังบล็อกถัดไป เมื่อสิ้นสุดกระบวนการนี้เราก็จะได้ภาพพิมพ์ลายน้ำที่จะนำไปใส่ลงในภาพหลัก

```
% process the image in blocks
% first construct the global watermark mask
x=1;
y=1;
for (kk = 1:length(logo_vector))
% if logo bit contains zero, add PN sequence to that portion of mask
  if (logo_vector(kk) == 0)
     watermark(y:y+blocksize-1,x:x+blocksize-1) = pn_sequence_zero;
```

```
% otherwise mask is filled with zeros
  else
    watermark(y:y+blocksize-1,x:x+blocksize-1) = pn_sequence_one;
  end
% move to next block of mask along x; If at end of row, move to next row
  if (x+blocksize) >= Nc
    x=1;
    y=y+blocksize;
  else
    x=x+blocksize;
  end
end
```

7. ทำการฝังภาพพิมพ์ลายน้ำลงบนภาพหลักด้วยสมการ

Watermarked = Host + k*Watermark

เมื่อ k คือ gain factor

8. ได้ภาพพิมพ์ลายน้ำดิจิตอลแบบที่มองไม่เห็น(Invisible digital watermarking) ทำการ save ไฟล์ภาพไว้ใช้งาน

```
Watermarked2=uint8(Host+k*Watermark);
imwrite(Watermarked2,'watermarked1.bmp','bmp'); % save output
```

## 3.4 การออกแบบและพัฒนาการกู้กลับคืนภาพลายน้ำดิจิตอลโดยเทคนิคในสพาเชียลโดเมนแบบมองไม่เห็น


รูปที่ 3.4 ขั้นตอนการกู้กลับคืนภาพลายน้ำแบบมองไม่เห็น

เป็นการทำกลับกับการฝังภาพลายน้ำลงในภาพหลักและจะต้องใส่ key ให้ถูกต้องด้วยไม่เช่นนั้นแล้วก็จะไม่ได้ภาพพิมพ์ลายน้ำที่ถูกต้องออกมา

มีขั้นตอนการทำงานดังต่อไปนี้

1. กำหนดขนาดบล็อกเท่ากับการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็น

```
blocksize=16;
```

2. อ่านไฟล์ภาพที่มีภาพพิมพ์ลายน้ำฝังอยู่ และอ่านขนาดของไฟล์ภาพ

```
[watermarked,map]=imread('watermarked1.bmp','bmp');
watermarked=double(watermarked);
% determine size of watermarked image
 [Mw,Nw]=size(watermarked);
% determine maximum possible logo size in object
  max_logo=Mw*Nw/(blocksize^2);
```

3. อ่านไฟล์ภาพพิมพ์ลายน้ำมาเก็บไว้และอ่านขนาดของไฟล์ภาพ

```
% read in original watermark
 [orig_watermark,map]=imread('_logo16_1.bmp','bmp');
orig_watermark=double(orig_watermark);
% determine size of original watermark
 [Mo,No]=size(orig_watermark);
```

4. ใส่ key ให้ตรงกับ key ที่ใช้ในขั้นตอนการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็น

```
key=14641;  % assumed key
rand('state',key);
```

หรือ

```
[key,map]=imread('cpe_key.bmp','bmp');
key=double(key)./256;
rand('state',key);
```

5. สร้างสัญญาณรบกวนเช่นเดียวกับในขั้นตอนการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็น

```
% gencrate PN sequences to designate "1" and "0"
watermark_one=round(2*(rand(blocksize,bloeksize)-0.5));
watermark_zero=round(2*(rand(bloeksize,blocksize)-0.5));
% find two highly un-correlated PN sequenees
while (corr2(watermark_one,watermark_zero) > -0.1)
   watermark_one=round(2*(rand(blocksize,blocksize)-0.5));
   watermark_zero=round(2*(rand(blocksize,blocksize)-0.5));
end
```

```
% pad logo out to maximum logo size with ones
logo_vector=ones(max_logo,1);
```

6. นำสัญญาณรบกวนที่ได้ไปเปรียบเทียบกับสัญญาณรบกวนในภาพพิมพ์ลายน้ำและทำการกำจัดสัญญาณรบกวนออกจากภาพพิมพ์ลายน้ำแล้วดึงภาพพิมพ์ลายน้ำออกมาจากไฟล์ภาพ

```
x=1;
y=1;
for (kk = 1:length(logo_vector))
  % caleulate eorrelations for both PN sequences
  correlation_one(kk)=corr2(watermarked(y:y+bloeksize-1,x:x+blocksize-
  1),watermark_one);
  correlation_zero(kk)=corr2(watermarked(y:y+bloeksize-1,x:x+bloeksize-
1),watermark_zero);
        % choose which ever correlation is higher
          if correlation_one(kk) > correlation_zero(kk)
            logo_vector(kk)=1;
          else
            logo_vector(kk)=0;
          end
        % move on to next block. At and of row move to next row
          if (x+blocksize) >= Nw
            x=1;
            y=y+blocksize;
          else
            x=x+blocksize;
          end
        end
```

7. ทำการแปลงเวกเตอร์ให้เป็นเมตริกซ์

```
logo=reshape(logo_vector(1:Mo*No),Mo,No);
```

8. จะได้ไฟล์ของภาพพิมพ์ลายน้ำ

## 3.5 ขั้นตอนการออกแบบการทำงานของ Graphic User Interface (GUI)


รูปที่ 3.5 ขั้นตอนการทำงานของ GUI

ขั้นตอนการทำงานของ GUI มีดังต่อไปนี้

1. ให้ผู้ใช้เลือกว่าต้องการสร้างภาพพิมพ์ลายน้ำแบบใด ซึ่งจะมีรูปแบบการทำภาพพิมพ์ลายน้ำให้เลือกทั้งหมด 7 ตัวเลือกคือ Ordinary, Overlapping, Light Overlapping, Edge Only, Harmony, 3D และ Invisible Embossing ซึ่งเมื่อผู้ใช้เลือกแล้วโปรแกรมก็จะแสดงผลการทำภาพพิมพ์ลายน้ำนั้นๆ ออกมา

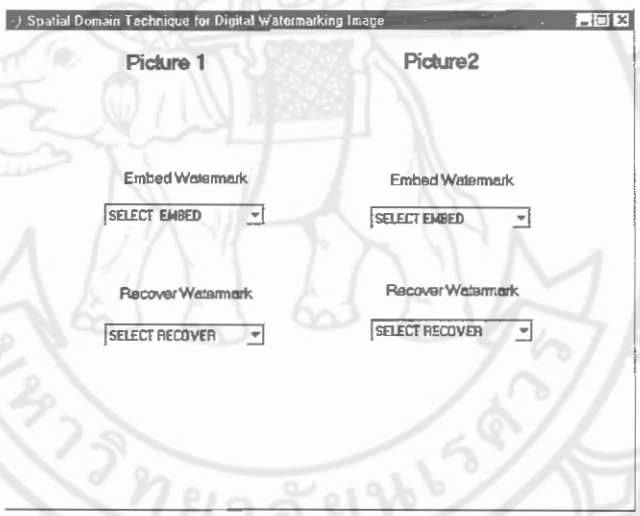
2. ส่วนที่สองเป็นส่วนที่ให้ผู้ใช้เลือกการกู้กลับคืนภาพพิมพ์ลายน้ำแบบต่างๆ เช่นเดียวกับการสร้างภาพพิมพ์ลายน้ำ ซึ่งก่อนที่ผู้ใช้จะเลือกการกู้กลับคืนภาพพิมพ์ลายน้ำนั้นผู้ใช้ต้องเลือกให้โปรแกรมทำการสร้างภาพพิมพ์ลายน้ำก่อนเพื่อที่จะได้นำภาพผลลัพธ์ของการสร้างภาพพิมพ์ลายน้ำมาเป็นอินพุทของการกู้กลับคืนภาพพิมพ์ลายน้ำ ซึ่งเมื่อผู้ใช้เลือกแล้วโปรแกรมก็จะแสดงผล**ลัพธ์ของการกู้กลับคืนภาพพิมพ์ลายน้ำ**

# บทที่ 4
# ผลการทดลอง

ผู้จัดทำโครงงานได้ทำการทดสอบการสร้างภาพพิมพ์ลายน้ำโดยใช้เทคนิคในสพาเชียลโดเมน (spatial domain) ได้ผลการทดสอบดังต่อไปนี้

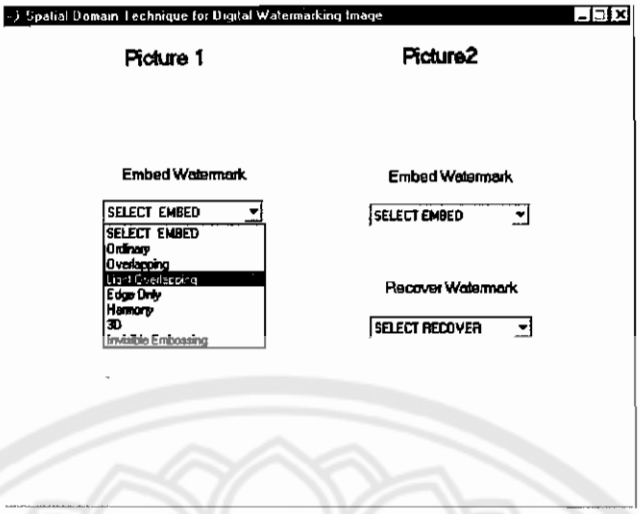
## 4.1 การทดสอบโปรแกรม ส่วนติดต่อผู้ใช้ (Graphic User Interface (GUI))

การทดสอบทำโดยการรันไฟล์ที่ชื่อ test_present.m ซึ่งเป็นโปรแกรมในส่วนของการติดต่อผู้ใช้ เพื่อให้ผู้ใช้งานใช้โปรแกรมได้สะดวกซึ่งในส่วนของโปรแกรมภาพพิมพ์ลายน้ำต่าง ๆ ผู้เขียนได้ใส่ภาพหลักและภาพพิมพ์ลายน้ำที่ใช้ในการทดสอบเข้าไปในโปรแกรมแล้วซึ่งได้มีการนำภาพหลักและภาพพิมพ์ลายน้ำมาทดสอบการสร้างภาพพิมพ์ลายน้ำโปรแกรมละ 2 ภาพซึ่งหน้าต่างของส่วนติดต่อผู้ใช้ แสดงดังรูปที่ 4.1



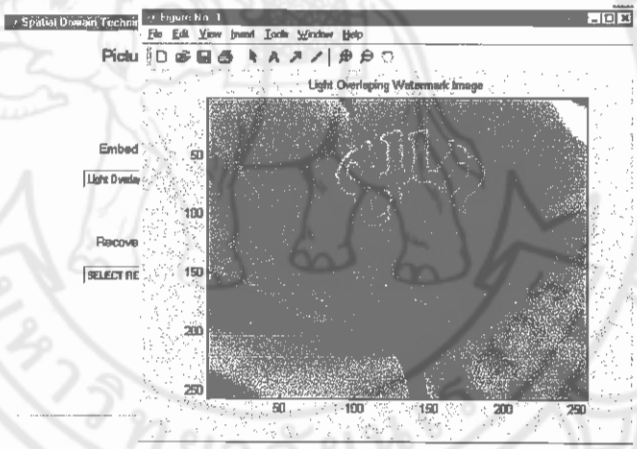

รูปที่ 4.1 หน้าต่างของส่วนติดต่อผู้ใช้

ในส่วนของการติดต่อผู้ใช้ นี้จะให้ผู้ใช้เลือกการทำภาพพิมพ์ลายน้ำ (SELECT EMBED) ซึ่งจะเลือกจาก ป๊อบอัพเมนู :Embed watermark และเลือกการทำกู้กลับคืนภาพพิมพ์ลายน้ำ (SELECT RECOVER) โดยเลือกจาก ป๊อบอัพเมนู : Recover Watermark ซึ่งจะแสดงผลได้ดังรูปที่ 4.2 ถึงรูปที่ 4.5



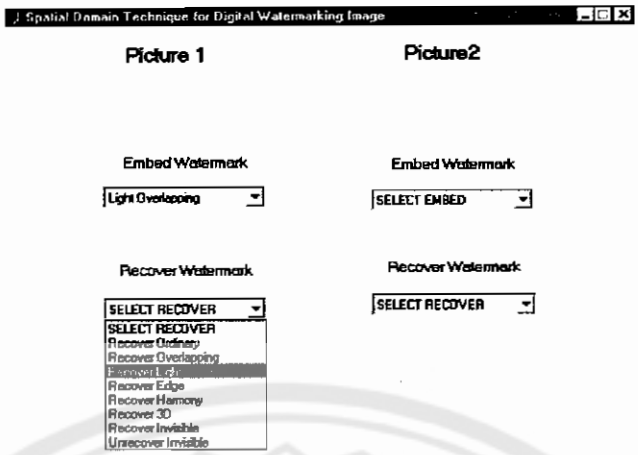

รูปที่ 4.2 แสดงการเลือกการทำภาพพิมพ์ลายน้ำใน Picture1

รูปที่ 4.2 นี้เป็นการแสดงภาพการเลือกการทำภาพพิมพ์ลายน้ำใน Picture1 ซึ่งในที่นี้แสดงการเลือกการทำภาพ ไลท์โอเวอร์แล็ปปิ้ง ซึ่งจะให้ผลดังรูปที่ 4.3



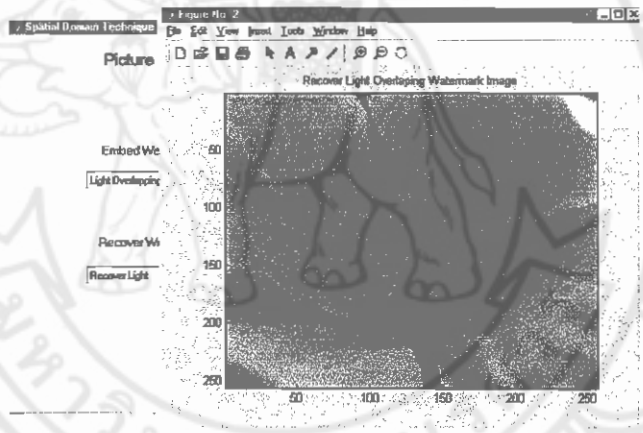

รูปที่ 4.3 แสดงผลของการเลือกการทำภาพพิมพ์ลายน้ำใน Picture1

รูปที่ 4.4 จะแสดงการเลือกการทำการกู้กลับคืนภาพพิมพ์ลายน้ำ




รูปที่ 4.4 แสดงการเลือกการทำการกู้กลับคืนภาพใน Picture1

รูปที่ 4.4 นี้เป็นการแสดงภาพการเลือกการกู้กลับคืนภาพพิมพ์ลายน้ำใน Picture1 ซึ่งในที่นี้แสดงการเลือกการทำการกู้กลับคืนภาพ ไลท์โอเวอร์แล็ปปิ้ง (Recover Light Overlapping) ซึ่งจะให้ผลดังรูปที่ 4.5




รูปที่ 4.5 แสดงผลการเลือกการทำกู้กลับคืนภาพใน Picture1

จากที่ได้แสดงการทดสอบโปรแกรมในส่วนของการติดต่อผู้ใช้ข้างต้น จะแสดงในส่วนของ Picture1 ให้คู่ซึ่งผู้จัดทำขอละเว้นการแสดงในส่วนของ Picture2 เนื่องจากว่ามีลักษณะการใช้งานเหมือนกันเพียงแต่เปลี่ยนภาพหลักและภาพที่จะนำมาทำภาพพิมพ์ลายน้ำใหม่

ต่อไปเป็นการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบมองเห็นได้และแบบมองไม่เห็นที่สามารถทำการกู้กลับคืนได้ซึ่งได้ผลลัพธ์ตามตารางที่ 4.1 และตารางที่ 4.2

ตารางที่ 4.1 ผลการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบมองเห็นได้ที่สามารถกู้กลับคืนได้

| ชนิด | $\alpha^*$ | $\beta^*$ | หมายเหตุ |
|---|---|---|---|
| ออดินารี โอเวอร์แล็ปปิ้ง | 0.06 - 0.8 | 0.2 - 0.9 | ใช้ค่า $\alpha \propto 1/\beta$ |
| โอเวอร์แล็ปปิ้ง | 0.01 - 0.15 | 0.99 ↑ | ค่า $\beta$ ที่แนะนำ คือ 0.999 |
| ไลท์ โอเวอร์แล็ปปิ้ง | 0.02 - 0.2 | 0.99 ↑ | ค่า $\beta$ ที่แนะนำ คือ 0.999 |
| เอด โอนรี | 0.05 - 0.2 | 0.99 ↑ | ค่า $\beta$ ที่แนะนำ คือ 0.999 |
| ฮาร์โมนี | 0.03 - 0.3 | 0.99 ↑ | ค่า $\beta$ ที่แนะนำ คือ 0.999 |
| ทรี ไดเมนชัน | 0.02 - 0.1 | 0.99 ↑ | ค่า $\beta$ ที่แนะนำ คือ 0.999 |

*ค่าของ $\alpha$ และ $\beta$ มีค่าระหว่าง 0 และ 1

ตารางที่ 4.2 ผลการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็นที่สามารถกู้กลับคืนได้

| K (gain factor) | ความคมชัดภาพ | การกู้กลับคืน | หมายเหตุ |
|---|---|---|---|
| 0.1 — 5 | คมชัดมาก | ไม่สมบูรณ์ | ค่า K (gain factor) นี้อาจจะใช้ไม่ได้กับทุกภาพ แต่ส่วนใหญ่ก็จะมีค่าใกล้เคียงค่าเหล่านี้ |
| 6 ↑ | ไม่คมชัด จะมองเห็นสัญญาณรบกวนบนภาพ | สมบูรณ์ | |

## 4.2 การทดสอบการสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำที่มองเห็นได้

ในการแสดงผลการทดสอบการสร้างภาพพิมพ์ลายน้ำในส่วนของภาพพิมพ์ลายน้ำที่สามารถมองเห็นนี้ผู้จัดทำจะขอเสนอผลการสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ ไปพร้อม ๆ กันโดยจะแสดงในแต่ละแบบของภาพพิมพ์ลายน้ำตามหัวข้อที่ 4.2.1 ถึง 4.2.6 ต่อไป

### 4.2.1 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำในแบบ ออดินารี โอเวอร์แล็ปปิ้ง (Ordinary Overlapping)

การสร้างภาพพิมพ์ลายน้ำแบบ ออดินารี โอเวอร์แล็ปปิ้ง นี้เป็นการนำเอาภาพพิมพ์ลายน้ำไปใส่ในภาพหลักโดยไม่ต้องมีการเปลี่ยนแปลงใด ๆ ในภาพพิมพ์ลายน้ำโดยจะอ่านในส่วนทั้ง

หมดของภาพพิมพ์ลายน้ำรวมถึงพื้นหลังของภาพพิมพ์ลายน้ำด้วยแล้วนำไปซ้อนทับลงบนภาพหลัก ซึ่งในการทดสอบนี้ผู้เขียนได้ทำการทดสอบหลายครั้งด้วยกันซึ่งในแต่ละครั้งจะทำการปรับค่าของ $\alpha$ และ $\beta$ ในสมการการสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำดังต่อไปนี้

สมการการสร้างภาพพิมพ์ลายน้ำ แบบมองเห็นได้

$$I_{watermarked} = \alpha * I_{watermark} + \beta * I_{host} \quad \text{.....สมการที่ 1.}$$

สมการการกู้กลับคืนภาพพิมพ์ลายน้ำ แบบมองเห็นได้

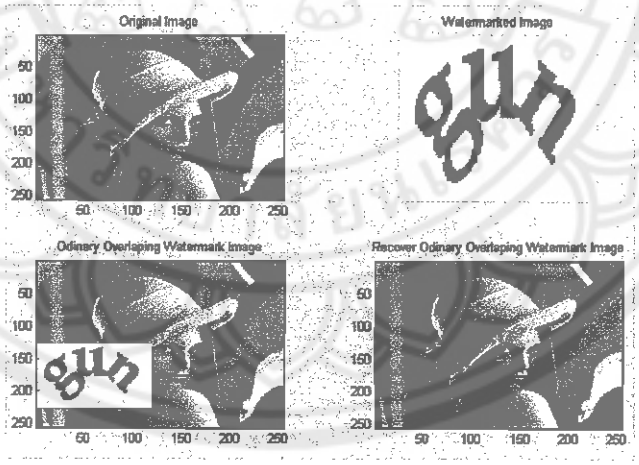
$$I_{recover} = (I_{watermarked} - \alpha * I_{watermark}) / \beta \quad \text{.....สมการที่ 2.}$$

โดยที่ ค่าของ $\alpha$ และ $\beta$ มีค่าอยู่ระหว่าง 0-1

ซึ่งจากการทดสอบการสร้างภาพพิมพ์ลายน้ำแบบออดินารี โอเวอร์แล็ปปิ้ง ในหลายๆ ครั้ง และปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดูทำให้ได้ข้อมูลดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.06 - 0.8$ ซึ่งค่า $\alpha = 0.06$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ออดินารี โอเวอร์แล็ปปิ้ง ที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha = 0.8$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ออดินารี โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

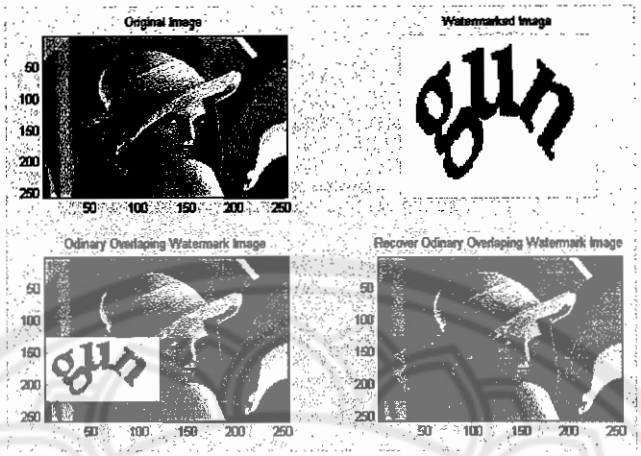
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.2 - 0.9$ ซึ่งค่า $\beta = 0.2$ จะทำให้มองเห็นพื้นหลังของภาพพิมพ์ลายน้ำชัดขึ้นซึ่งเป็นผลให้ภาพพิมพ์ลายน้ำมีลักษณะที่เข้มที่สุดที่สามารถกู้กลับคืนได้ และค่า $\beta = 0.9$ จะทำให้มองเห็นพื้นหลังของภาพพิมพ์ลายน้ำจางลงซึ่งจะมีผลทำให้ภาพพิมพ์ลายน้ำมีลักษณะที่จางที่สุดที่ตามองเห็นที่สามารถกู้กลับคืนได้ซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำที่เข้มที่สุดที่สามารถกู้กลับคืนได้มาแสดงดังรูปที่ 4.6




รูปที่ 4.6 ภาพพิมพ์ลายน้ำออดินารี โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.6 เป็นภาพพิมพ์ลายน้ำ ออดินารี โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้ โดยมีค่า $\alpha = 0.8$ และ $\beta = 0.2$

ในส่วนของการสร้างภาพพิมพ์ลายน้ำ ออดินารี โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้กลับคืนได้ ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.7




รูปที่ 4.7 ภาพพิมพ์ลายน้ำออดินารี โอเวอร์แล็ปปิ้ง ที่กู้กลับคืนไม่ได้

รูปที่ 4.7 เป็นภาพพิมพ์ลายน้ำ ออดินารี โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.7$ และ $\beta = 0.8$

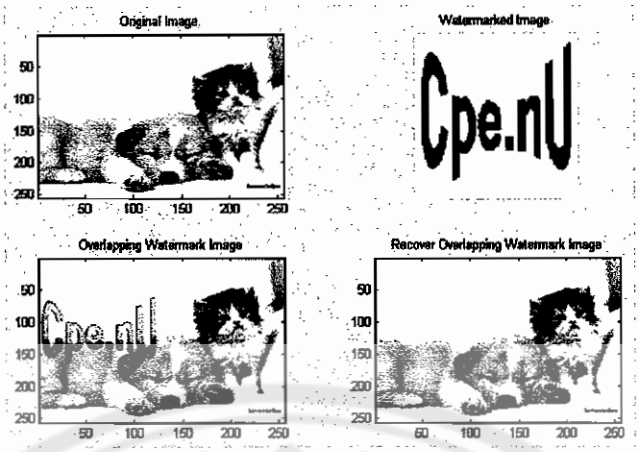
### 4.2.2 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบ โอเวอร์แล็ปปิ้ง (Overlapping)

การสร้างภาพพิมพ์ลายน้ำแบบโอเวอร์แล็ปปิ้ง นี้เป็นการอ่านภาพพิมพ์ลายน้ำมาเฉพาะตัวภาพพิมพ์ลายน้ำโดยไม่สนใจพื้นหลังแล้วนำไปซ้อนทับลงบนภาพหลัก ซึ่งการทดสอบผู้เขียนได้ทำการทดสอบโดยการปรับเปลี่ยนค่าของ $\alpha$ และ $\beta$ ในหลายค่าด้วยกันตามสมการที่1. และสมการที่ 2. ข้างต้น เพื่อให้ได้ค่าที่ดีที่สุดในการทำภาพพิมพ์ลายน้ำแบบ โอเวอร์แล็ปปิ้ง

ซึ่งจากการทดสอบในหลายๆ ครั้งและปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดูทำให้ได้ข้อมูลที่ดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.01 - 0.15$ ซึ่งค่า $\alpha = 0.01$ จะทำให้ได้ภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้งที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha = 0.15$ จะทำให้ได้ภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

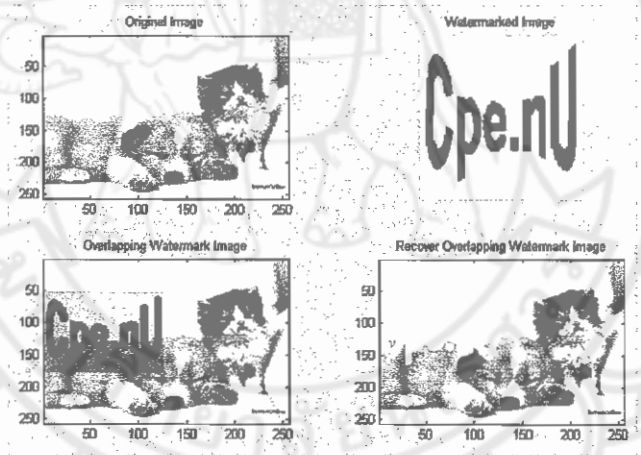
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ต้องมีค่ามากกว่า 0.99 แต่ไม่ถึง 1 ซึ่งค่ายิ่งค่า $\beta$ มีค่ามาก ๆ จะทำให้ภาพมองดูคมชัดสวยงามมากขึ้นและสามารถกู้กลับคืนได้ ส่วนค่าที่น้อยกว่านี้คือค่าที่ ต่ำกว่าค่า 0.99 จะทำให้มองเห็นภาพพื้นหลังของภาพพิมพ์ลายน้ำเป็นสีเทาไปจนถึงสีดำซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำที่เข้มที่สุดที่สามารถกู้กลับคืนได้มาแสดงดังรูปที่ 4.8




รูปที่ 4.8 ภาพพิมพ์ลายน้ำโอเวอร์แล็ปปิ้งที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.8 นั้นเป็นภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้ โดยมีค่า $\alpha = 0.15$ และ $\beta = 0.999$

ในส่วนของการสร้างภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้งที่ไม่สามารถกู้กลับคืนได้ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.9




รูปที่ 4.9 ภาพพิมพ์ลายน้ำ โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้คืนได้

รูปที่ 4.9 เป็นภาพพิมพ์ลายน้ำโอเวอร์แล็ปปิ้งที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.5$ และ $\beta = 0.8$

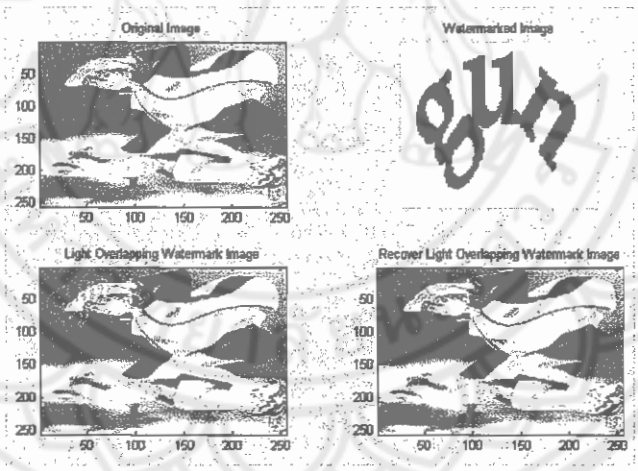
**4.2.3 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบ ไลท์โอเวอร์แล็ปปิ้ง (Light Overlapping)**

การสร้างภาพพิมพ์ลายน้ำแบบ ไลท์โอเวอร์แล็ปปิ้ง นี้คือการทำให้ภาพพิมพ์ลายน้ำมีแสงเงาโดยไม่สนใจพื้นหลังของภาพพิมพ์ลายน้ำแล้วนำมาซ้อนทับลงบนภาพหลักแล้วทำการปรับค่าคงที่เพื่อทำให้ภาพพิมพ์ลายน้ำดูสวยงาม ซึ่งผู้เขียนได้ทำการทดสอบหลายครั้งโดยการปรับเปลี่ยนค่าของ $\alpha$ และ $\beta$ ในหลายค่าด้วยกันตามสมการที่1.และสมการที่ 2. ข้างต้น เพื่อให้ได้ค่าที่ดีที่สุดในการทำภาพพิมพ์ลายน้ำแบบ ไลท์โอเวอร์แล็ปปิ้ง

ซึ่งจากการทดสอบในหลายๆ ครั้งและปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดูทำให้ได้ข้อมูลที่ดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.02 - 0.2$ ซึ่งค่า $\alpha = 0.02$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ไลท์โอเวอร์แล็ปปิ้ง ที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha = 0.2$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ไลท์โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

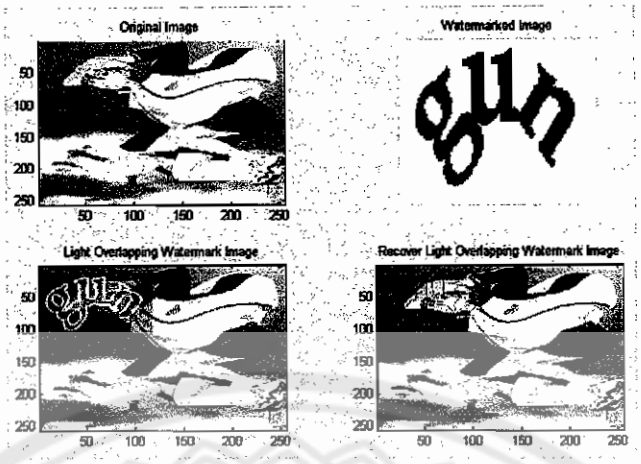
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ต้องมีค่ามากกว่า 0.99 แต่ไม่ถึง 1 ซึ่งค่ายิ่งค่า $\beta$ มีค่ามาก ๆ จะทำให้ภาพมองดูคมชัดสวยงามมากขึ้นและสามารถกู้กลับคืนได้ส่วนค่าที่น้อยกว่านี้คือค่าที่ ต่ำกว่าค่า 0.99 จะทำให้มองเห็นภาพพื้นหลังของภาพพิมพ์ลายน้ำเป็นสีเทาไปจนถึงสีดำซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำที่เข้มที่สุดที่สามารถกู้กลับคืน ได้มาแสดงดังรูปที่ 4.10




รูปที่ 4.10 ภาพพิมพ์ลายน้ำไลท์โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.10 เป็นภาพพิมพ์ลายน้ำ ไลท์โอเวอร์แล็ปปิ้ง ที่เข้มที่สุดที่สามารถกู้กลับคืนได้โดยมีค่า $\alpha = 0.2$ และ $\beta = 0.999$

ในส่วนของภาพพิมพ์ลายน้ำ ไลท์โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้กลับคืนได้ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.11




ภาพที่ 4.11 ภาพพิมพ์ลายน้ำไลท์โอเวอร์แล็ปปิ้ง ที่กู้กลับคืนไม่ได้

รูปที่ 4.11 เป็นภาพพิมพ์ลายน้ำ ไลท์โอเวอร์แล็ปปิ้ง ที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.6$ และ $\beta = 0.8$

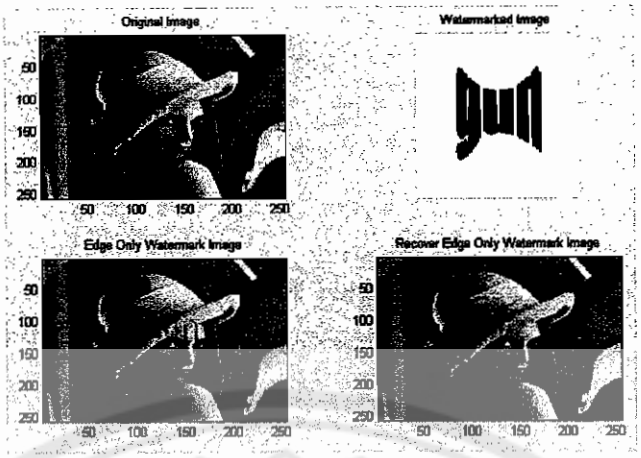
**4.2.4 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบ เอดโอนรี (Edge Only)**

การสร้างภาพพิมพ์ลายน้ำแบบ เอดโอนรี นี้เป็นการอ่านมาเฉพาะขอบของภาพพิมพ์ลายน้ำ โดยไม่สนใจพื้นหลังของภาพพิมพ์ลายน้ำและนำมาซ้อนทับลงบนภาพหลัก ซึ่งผู้เขียนได้ทำการทดสอบหลายครั้งโดยการปรับเปลี่ยนค่าของ $\alpha$ และ $\beta$ ในหลายค่าด้วยกันตามสมการที่1.และสมการที่ 2. ข้างต้น เพื่อให้ได้ค่าที่ดีที่สุดในการทำภาพพิมพ์ลายน้ำแบบ เอดโอนรี

ซึ่งจากการทดสอบในหลายๆ ครั้งและปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดูทำให้ได้ข้อมูลดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.05 - 0.2$ ซึ่งค่า $\alpha = 0.05$ จะทำให้ได้ภาพพิมพ์ลายน้ำ เอดโอนรี ที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha = 0.2$ จะทำให้ได้ภาพพิมพ์ลายน้ำ เอดโอนรี ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

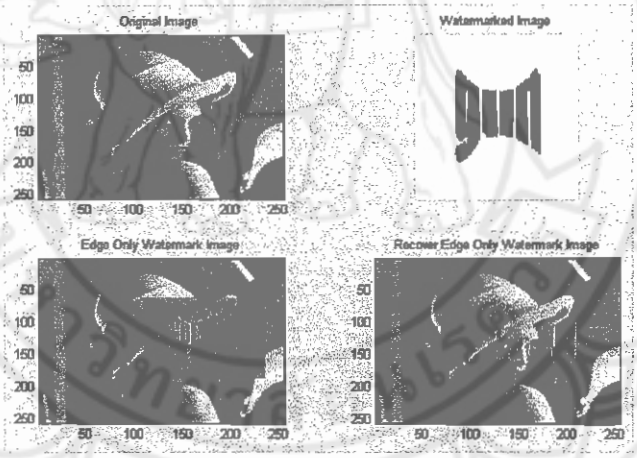
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ต้องมีค่ามากกว่า 0.99 แต่ไม่ถึง 1 ซึ่งค่ายิ่งค่า $\beta$ มีค่ามาก ๆ จะทำให้ภาพมองดูคมชัดสวยงามมากขึ้นและสามารถกู้กลับคืนได้ส่วนค่าที่น้อยกว่านี้คือค่าที่ ต่ำกว่าค่า 0.99 จะทำให้มองเห็นภาพพื้นหลังของภาพพิมพ์ลายน้ำเป็นสีเทาไปจนถึงสีดำซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำที่เข้มที่สุดที่สามารถกู้กลับคืนได้มาแสดงดังรูปที่ 4.12




รูปที่ 4.12 ภาพพิมพ์ลายน้ำเอดโอนรี ที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.12 เป็นภาพพิมพ์ลายน้ำ เอดโอนรี ที่เข้มที่สุดที่สามารถกู้กลับคืนได้โดยมีค่า $\alpha = 0.2$ และ $\beta = 0.999$

ในส่วนของการสร้างภาพพิมพ์ลายน้ำ เอดโอนรี ที่ไม่สามารถกู้กลับคืนได้ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.13




รูปที่ 4.13 ภาพพิมพ์ลายน้ำ เอดโอนรี ที่ไม่สามารถกู้กลับคืนได้

รูปที่ 4.13 เป็นภาพพิมพ์ลายน้ำ Edge Only ที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.5$ และ $\beta = 0.8$

### 4.2.5 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบ ฮาร์โมนี (Harmony)

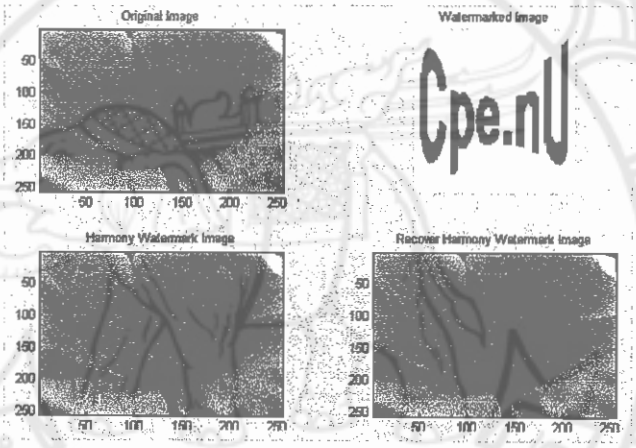
การสร้างภาพพิมพ์ลายน้ำแบบ ฮาร์โมนี เป็นการทำให้ภาพพิมพ์ลายน้ำโปร่งใสเมื่อซ้อนทับลงไปในภาพหลัก แล้วจะมองดูเหมือนมีสีกลมกลืนกับภาพหลัก ในการทดสอบผู้เขียนได้

ทดสอบการปรับค่าของ $\alpha$ และ $\beta$ ในหลายค่าด้วยกันตามสมการที่1.และสมการที่ 2. ข้างต้น เพื่อให้ได้ค่าที่ดีที่สุดในการทำภาพพิมพ์ลายน้ำแบบ ฮาร์โมนี

ซึ่งจากการทดสอบในหลายๆ ครั้งและได้ลองปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดู ทำให้ได้ข้อมูลดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า 0.03 – 0.3 ซึ่งค่า $\alpha$ = 0.03 จะทำให้ได้ภาพพิมพ์ลายน้ำ ฮาร์โมนีที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha$ = 0.3 จะทำให้ได้ภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

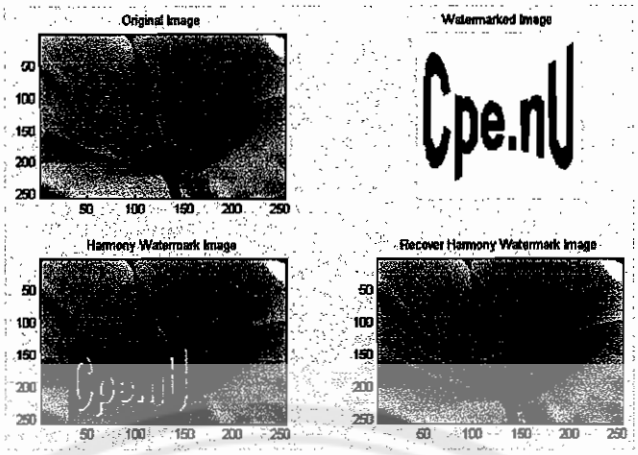
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ต้องมีค่ามากกว่า 0.99 แต่ไม่ถึง 1 ซึ่งค่ายิ่งค่า $\beta$ มีค่ามาก ๆ จะทำให้ภาพมองดูคมชัดสวยงามมากขึ้นและสามารถกู้กลับคืนได้ส่วนค่าที่น้อยกว่านี้คือค่าที่ ต่ำกว่าค่า 0.99 จะทำให้มองเห็นภาพพื้นหลังของภาพพิมพ์ลายน้ำเป็นสีเทาไปจนถึงสีดำซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำที่เข้มที่สุดที่สามารถกู้กลับคืนได้มาแสดงดังรูปที่ 4.14




รูปที่ 4.14 ภาพพิมพ์ลายน้ำ ฮาร์โมนีที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.14 เป็นภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่เข้มที่สุดที่สามารถกู้กลับคืนได้โดยมีค่า $\alpha$ = 0.3 และ $\beta$ = 0.999

ในส่วนของการสร้างภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่ไม่สามารถกู้กลับคืนได้ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.15




รูปที่ 4.15 ภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่กู้กลับคืนไม่ได้

รูปที่ 4.15 เป็นภาพพิมพ์ลายน้ำ ฮาร์โมนี ที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.7$ และ $\beta = 0.8$

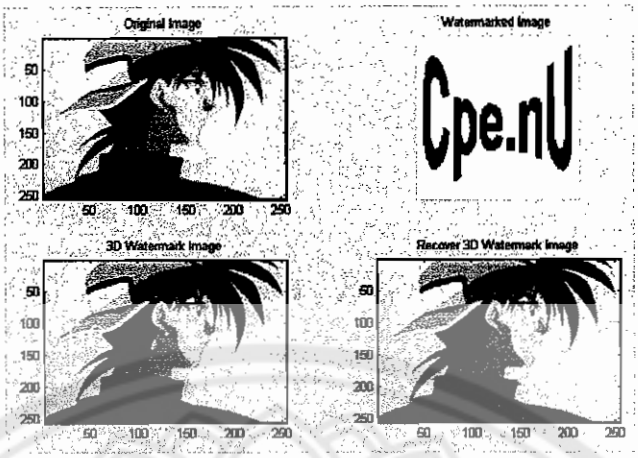
**4.2.6 การสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบ ทรีไดเมนชัน (Three Dimension)**

ภาพพิมพ์ลายน้ำแบบ ทรีไดเมนชัน เป็นภาพพิมพ์ลายน้ำที่เกิดจากการทำให้พิกเซลของภาพใน slide ของค่าสี RGB (3 slide) เหลื่อมกันคือการทำให้ขอบของภาพด้านใดด้านหนึ่งสูงกว่าอีกด้านที่เหลือ จากตัวอย่างโปรแกรมที่พัฒนาขึ้นให้ขอบด้านซ้ายหนากว่าด้านขวา เมื่อนำภาพพิมพ์ลายน้ำนี้ไปซ้อนทับบนภาพหลักจะทำให้ดูมีลักษณะเป็นภาพนูน 3 มิติ ผู้เขียนได้ทำการทดสอบการสร้างโดย การปรับค่าของ $\alpha$ และ $\beta$ ในหลายค่าด้วยกันตามสมการที่1.และสมการที่ 2. ข้างต้น เพื่อให้ได้ค่าที่ดีที่สุดในการทำภาพพิมพ์ลายน้ำแบบ ทรีไดเมนชัน

ซึ่งจากการทดสอบในหลายๆ ครั้งและได้ลองปรับเปลี่ยนค่า $\alpha$ และ $\beta$ ดู ทำให้ได้ข้อมูลดังต่อไปนี้

ค่าของ $\alpha$ นั้นเราสามารถใช้ได้ตั้งแต่ค่า $0.02 - 0.1$ ซึ่งค่า $\alpha = 0.02$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่จางที่สุดที่ตามองเห็นได้ที่สามารถกู้กลับคืนได้ และค่า $\alpha = 0.1$ จะทำให้ได้ภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่เข้มที่สุดที่สามารถกู้กลับคืนได้

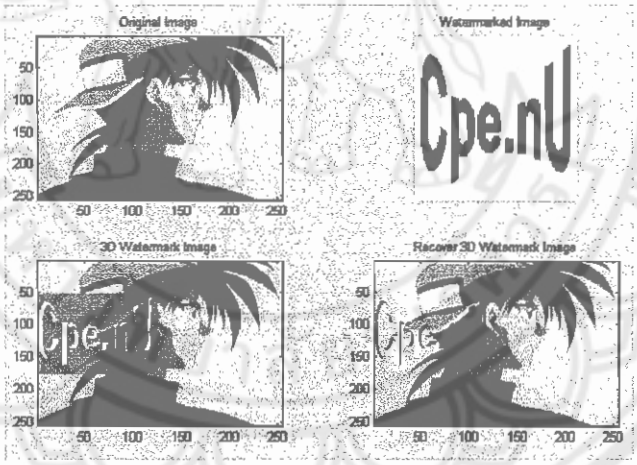
ค่าของ $\beta$ นั้นเราสามารถใช้ได้ต้องมีค่ามากกว่า 0.99 แต่ไม่ถึง 1 ซึ่งค่ายิ่งค่า $\beta$ มีค่ามาก ๆ จะทำให้ภาพมองดูคมชัดสวยงามมากขึ้นและสามารถกู้กลับคืนได้ส่วนค่าที่น้อยกว่านี้คือค่าที่ ต่ำกว่าค่า 0.99 จะทำให้มองเห็นภาพพื้นหลังของภาพพิมพ์ลายน้ำเป็นสีเทาไปจนถึงสีดำซึ่งผู้เขียนได้นำภาพพิมพ์ลายน้ำแบบ ทรีไดเมนชันที่เข้มที่สุดที่สามารถกู้กลับคืนได้มาแสดงดังรูปที่ 4.16




รูปที่ 4.16 ภาพพิมพ์ลายน้ำทรีไดเมนชัน ที่เข้มที่สุดที่กู้กลับคืนได้

รูปที่ 4.16 เป็นภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่เข้มที่สุดที่สามารถกู้กลับคืนได้โดยมีค่า $\alpha = 0.1$ และ $\beta = 0.999$

ในส่วนของการสร้างภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่ไม่สามารถกู้กลับคืนได้ซึ่งเกิดจากการปรับค่าของ $\alpha$ และ $\beta$ ที่นอกเหนือจากที่กล่าวมาจะเป็นดังรูปที่ 4.17




รูปที่ 4.17 ภาพพิมพ์ลายน้ำ ทรีไดเมนชัน ที่ไม่สามารถกู้กลับคืนได้

รูปที่ 4.17 เป็นภาพพิมพ์ลายน้ำ ทรีไดเมนชันที่ไม่สามารถกู้กลับคืนได้ ซึ่งใช้ค่า $\alpha = 0.5$ และ $\beta = 0.8$

## 4.3 การทดสอบการสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำแบบที่มองไม่เห็น

ในการแสดงผลการทดสอบการสร้างภาพพิมพ์ลายน้ำในส่วนของภาพพิมพ์ลายน้ำที่ไม่สามารถมองเห็นได้นี้ผู้จัดทำจะขอเสนอผลการสร้างภาพพิมพ์ลายน้ำและการกู้กลับคืนภาพพิมพ์ลายน้ำ ไปพร้อม ๆ กันเช่นเดียวกับภาพพิมพ์ลายน้ำที่มองเห็นได้ซึ่งจะแสดงดังต่อไปนี้

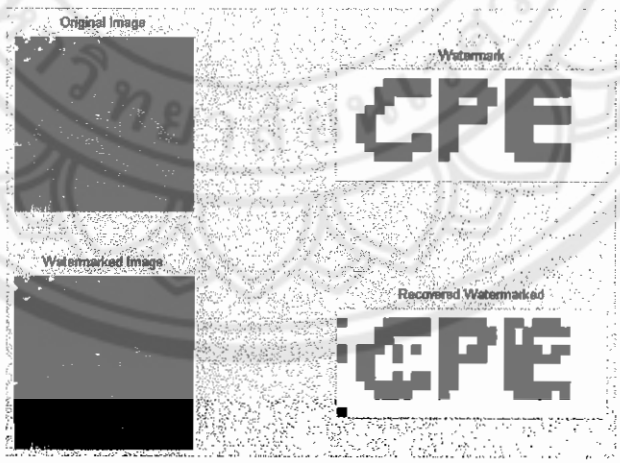
สมการที่ใช้ในการใส่ภาพพิมพ์ลายน้ำแบบมองไม่เห็นนี้จะแตกต่างจากสมการของการใส่ภาพพิมพ์ลายน้ำแบบมองเห็นได้นิดหน่อยโดยสมการของภาพลายน้ำที่มองไม่เห็นจะใช้ค่า gain factor (k) เป็นสัมประสิทธิ์ของสมการและจะคูณในสมการเฉพาะตรงส่วนของภาพพิมพ์ลายน้ำเท่านั้นดังนี้

$$Watermarked = Host + k*Watermark \quad \text{.....สมการที่ 3.}$$

เมื่อ k คือ gain factor และมีค่ามากกว่า 0

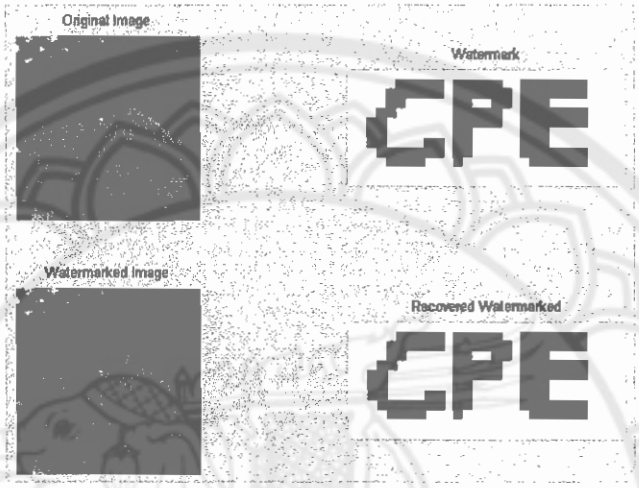
ค่าของ gain factor ที่ใช้ในการสร้างภาพพิมพ์ลายน้ำแบบมองไม่เห็นนี้มีความสำคัญอย่างมากต่อสมการคือมีผลต่อความคมชัดของภาพและมีผลต่อการกู้คืนภาพด้วยโดยที่ค่าของ gain factor ที่น่าสนใจมีดังนี้

ค่า gain factor ต่ำๆ จะทำให้ภาพคมชัดมากทำให้ภาพดูสวยงามแต่การกู้กลับคืนภาพพิมพ์ลายน้ำจะไม่สมบูรณ์เท่าที่ควรคือจะทำให้ภาพพิมพ์ลายน้ำที่กู้กลับคืนมามีสัญญาณรบกวนติดมาแต่ก็สามารถมองออกได้ว่าภาพพิมพ์ลายน้ำมีลักษณะอย่างไร ค่า gain factor นี้คือค่า gain factor ที่มีค่าระหว่าง 0 ถึง 5 ($0 < k < 6$) ซึ่งเราจะแสดงภาพการทำภาพพิมพ์ลายน้ำและการกู้คืนได้ดังรูปที่ 4.18 ซึ่งเป็นภาพที่เกิดจากการใช้ค่า gain factor = 1



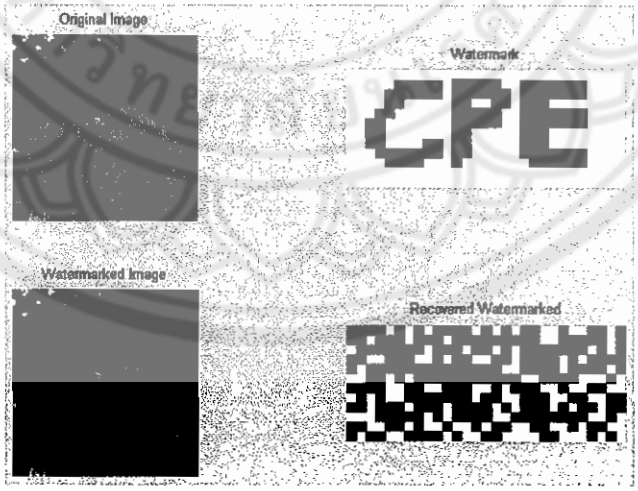

รูปที่ 4.18 ภาพพิมพ์ลายน้ำ k=1

ในทางกลับกันค่า gain factor สูงๆ จะทำให้ภาพคมชัดน้อยทำให้ภาพดูมัวไม่สวยงามแต่การกู้ภาพพิมพ์ลายน้ำจะสมบูรณ์มากจะไม่มีสัญญาณรบกวนติดมาเลย ค่า gain factor นี้คือค่า gain factor ที่มีค่าตั้งแต่ 6 ขึ้นไป (k>=6 ) แต่ทั้งนี้ก็ไม่ได้หมายความว่าค่านี้จะใช้ได้กับภาพพิมพ์ลายน้ำและภาพหลักทุกภาพซึ่งบางภาพอาจต้องใช้ค่า gain factor สูงกว่านี้จึงจะกู้คืนภาพพิมพ์ลายน้ำได้สมบูรณ์ซึ่งเราจะแสดงภาพการทำภาพพิมพ์ลายน้ำและการกู้คืนได้ดังรูปที่ 4.19 ซึ่งเป็นรูปภาพที่เกิดจากการใช้ค่า gain factor = 20




รูปที่ 4.19 ภาพพิมพ์ลายน้ำ k=20

ส่วนภาพพิมพ์ลายน้ำที่กู้คืนกลับไม่ได้เนื่องจากใส่ค่า key ไม่ถูกต้องจะแสดงได้ดังรูปที่ 4.20




รูปที่ 4.20 ภาพพิมพ์ลายน้ำที่กู้คืนกลับไม่ได้

# บทที่ 5
# บทสรุป

## 5.1 สรุปผล

1. การ สร้างภาพพิมพ์ลายน้ำได้ผลดีในระดับหนึ่ง

2. การสร้างภาพพิมพ์ลายน้ำแบบที่มองเห็นได้ยังไม่สามารถป้องกันการกู้กลับคืนภาพได้ดีเท่าที่ควร

3. การสร้างภาพพิมพ์ลายน้ำแบบที่มองไม่เห็นสามารถป้องกันการกู้กลับคืนภาพได้เป็นอย่างดี

4. การสร้างภาพพิมพ์ลายน้ำนี้ยังไม่สามารถเลือกภาพพิมพ์ลายน้ำและภาพหลักเพื่อนำมาทำภาพพิมพ์ลายน้ำได้ ต้องใส่ภาพลงไปในโปรแกรมเลย

## 5.2 ปัญหาและแนวทางแก้ไข

1. การสร้างภาพพิมพ์ลายน้ำนี้ยังมีข้อจำกัดในการใช้ไฟล์ฟอร์แม็ตของภาพซึ่งไฟล์ฟอร์แม็ตบางไฟล์อาจใช้ไม่ได้กับการสร้างภาพพิมพ์ลายน้ำนี้ การนำไปใช้งานควรจะใช้กับไฟล์ฟอร์แม็ตที่เข้ากันได้เพื่อให้การสร้างภาพพิมพ์ลายน้ำได้ผลดีที่สุด

2. ขนาดและมิติของภาพที่นำมาใช้ในการสร้างภาพพิมพ์ลายน้ำยังมีข้อจำกัดในการใช้อยู่ การนำไปใช้งานจึงควรให้ภาพหลักและภาพที่นำมาซ้อนทับมีขนาดและมิติที่เข้ากันได้

3. ค่าคงที่ต่างๆ ที่ได้จากการทดสอบโปรแกรมนี้อาจใช้ไม่ได้ดีกับภาพอื่นๆ ดังนั้นเมื่อนำการสร้างภาพพิมพ์ลายน้ำนี้ไปใช้จึงควรทดลองปรับค่าเหล่านี้เพื่อหาค่าที่ดีที่สุด

# เอกสารอ้างอิง

[1] http://www-nt.e-technik.uni-erlangen.de/~hartung/watermakinglinks.html

[2] http://www.watermarkingworld.com

[3] http://www.kmutt.ac.th/organization/Research/Intellect/watermark.htm

[4] http://www.cpe.kmutt.ac.th/lab/mcl/Introduction.htm

[5] http://www.student.nu.ac.th/u41320300/watermark.html

[6] Donald Hearn & M.Pauline Baker, *Computer Graphics* , Prentice-Hall, Englewood Cliffs, New Jersey

# ภาคผนวก ก

# Source Code Program

## test_present.m

```
function varargout = test_present(varargin)
% TEST_PRESENT Application M-file for test_present.fig
%    FIG = TEST_PRESENT launch test_present GUI.
%    TEST_PRESENT('callback_name', ...) invoke the named callback.
% Last Modified by GUIDE v2.0 27-Jun-2003 02:11:44
if nargin == 0  % LAUNCH GUI
   fig = openfig(mfilename,'reuse');
   % Use system color scheme for figure:
   set(fig,'Color',get(0,'defaultUicontrolBackgroundColor'));
   % Generate a structure of handles to pass to callbacks, and store it.
   handles = guihandles(fig);
   guidata(fig, handles);
   if nargout > 0
            varargout{1} = fig;
   end
elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK
   try
            if (nargout)
                     [varargout{1:nargout}] = feval(varargin{:}); % FEVAL switchyard
            else
                     feval(varargin{:}); % FEVAL switchyard
            end
   catch
            disp(lasterr);
   end
end
```

```
% --------------------------------------------------------------------
function varargout = popupmenu1_Callback(h, eventdata, handles, varargin)
val=get(h,'Value');
string_list=get(h,'String');
selected_string=string_list{val};
switch seleeted_string
case 'SELECT EMBED',

case 'Ordinary',
   ordinary;
case 'Overlapping',
   overlap;
case 'Light Overlapping',
   lightover;
case 'Edge Only',
   edge;
case 'Harmony',
   harmony;
case '3D',
   three_D;
case 'Invisible Embossing',
   invisible;
end

guidata(h,handles)
% --------------------------------------------------------------------
function varargout = popupmenu2_Callback(h, eventdata, handles, varargin)
val=get(h,'Value');
string_list=get(h,'String');
selected_string=string_list{val};
switch selected_string
case 'SELECT RECOVER',
```

```
case 'Recover Ordinary',
   recover_ordi
case 'Recover Overlapping',
   recover_over
case 'Recover Light',
   recover_light
case 'Recover Edge',
   recover_edge
case 'Recover Harmony',
   rccover_har
case 'Recover 3D',
   recover_three
case 'Recover Invisible',
   rccover_invis
case 'Unrecover Invisible',
   unrecover_invis
end

guidata(h,handles)
% --------------------------------------------------------------------
function varargout = popupmenu3_Callback(h, eventdata, handles, varargin)
val=get(h,'Value');
string_list=get(h,'String');
selected_string=string_list{val};
switch selected_string
case 'SELECT EMBED',

case 'Ordinary',
   ordinary2
case 'Overlapping',
   overlap2
```

```
case 'Light Overlapping',
   lightover2
case 'Edge Only',
   edge2
case 'Harmony',
   harmony2
case '3D',
   three_D2
case 'Invisible Embossing',
   invisible2
end

guidata(h,handles)
% --------------------------------------------------------------------
function varargout = popupmenu4_Callback(h, eventdata, handles, varargin)
val=get(h,'Value');
string_list=get(h,'String');
selected_string=string_list{val};
switch selcetcd_string
case 'SELECT RECOVER',

case 'Recover Ordinary',
   reeover_ordinay2
case 'Reeover Overlapping',
   reeover_over2
case 'Recover Light',
   reeover_light2
case 'Recover Edge ',
   recover_edge2
case 'Recover Harmony',
   recover_har2
case 'Recover 3D',
```

```
   recover_three2
case 'Recover Invisible',
   recover_invis
case 'Unrecover Invisible',
   unrecover_invis
end

guidata(h,bandles)
```

**ordinary.m**

```
% ordinary overlaping image (watermarking image)
close all
clear all;
[X,map]=imread('logo9_2.bmp','bmp');
logo=X(:,:,1);
% Convert data in arrays of class double
logo=double(logo);
% read image size
[row,column]=size(logo);
%Read host image
%***************
[Y,map]=imread('lena.bmp','bmp');
Y=double(Y);
[Mc,Nc]=size(Y);
% Convert data in arrays of class double
% read image size of logo image
[row,column,slide]=size(logo);
%****** P and Q are position that put watermark in bost image *****
P=125; %select for move ROW   (P<(Row of background - Row of Text))
Q=0; %select for move COLUMN(Q<(Column of background - Column of Text))
```

```
W=Y;
for k=1:3
  for i=1:row
    for j=1:column
       % Equation of inserting watermarked in host image
      W(i+P,j+Q,k)=0.80*logo(i,j)+0.20*Y(i+P,j+Q,k) ;
    end
  end
end
%W=W/2;
figure(1)
image(uint8(W));
title('Odinary Overlaping Watermark Image');
%%% Save file
imwrite(uint8(W),'marked2.bmp','bmp');
```

recover_ordi.m

```
%close all
clear all;
[A,map]=imread('lena.bmp','bmp');
[X,map]=imread('logo9_2.bmp','bmp');
% Convert data in arrays of class double
logo=double(X);
% read image size
%Read host image
%***************
[W,map]=imread('marked2.bmp','bmp');
% Convert data in arrays of class double
W=double(W);
% read image size of logo image
```

```
[row,column,slide]=size(logo);
%****** P and Q are position that put watermark in host image *****
P=125; %select for move ROW  (P<(Row of background - Row of Text))
Q=0; %select for move COLUMN(Q<(Column of baekground - Column of Text))
Y=W;
for k=1:3
  for i=1:row
    for j=1:column
       % Equation of inserting watermarked in host image
         Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-(0.80*logo(i,j)))/0.8;
    end
  end
end
%colormap(map);
%Y=Y/2;
figure(2)
image(uint8(Y));
title('Recover Odinary Overlaping Watermark Image');
%%% Save file
%imwrite(uint8(Y),'marked2.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(X);
title('Watermarked Image');
subplot(2,2,3);image(uint8(W));
title('Odinary Overlaping Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover Odinary Overlaping Watermark Image');
```

MISSING

```
if k >18
Z = 0;% convert background from white to black color
end
%copy binary image to 3 slide image
B(i,j,1) = Z;
B(i,j,2) = Z;
B(i,j,3) = Z;
% +2 is sliding position of character and
D(i,j,1) = (255-logo(i+1,j+1));
D(i,j,2) =(255-logo(i+1,j+1));
D(i,j,3) = (255-logo(i+1,j+1));
 end
end
%*****************************************************************
% Show output after manipulates logo image to shower and light
%*****************************************************************
% Convert data in arrays of elass unit8
B=uint8(B);
D=uint8(D);
imwrite(B,'Bmarked4.bmp','bmp');
imwrite(D,'Dmarked4.bmp','bmp');
%***************
%Read host image
%***************
[Y,map]=imread('cat_dog.jpg','jpg');
% Convert data in arrays of elass double
B=double(B);
D=double(D);
Y=double(Y);
% read image size of logo image
[row1,column1,slide1]=size(B);
%****** P and Q are position that put watermark in host image *****
```

```
P=50; %select for move ROW   (P<(Row of background - Row of Text))
Q=0; %select for move COLUMN(Q<(Column of background - Column of Text))
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
      % Equation of inserting watermarked in host image
     T(i,j,k)=(-2.5*B(i,j,k))-(1.2*D(i,j,k));
     W(i+P,j+Q,k)=(0.5*T(i,j,k))+(0.8*Y(i+P,j+Q,k));
    end
  end
end
%figure(1)
%image(uint8(T));
%imwrite(uint8(T),'Tmarked.bmp','bmp');
figure(1)
image(uint8(W));
title('Overlapping Watermark Image')
%save file
imwrite(uint8(W),'marked3.bmp','bmp');
```

recover_over.m

```
%close all
clear all;
[A,map]=imread('cat_dog.jpg','jpg');
[X,map]=imread('logo4.bmp','bmp');
[B,map]=imread('Bmarked4.bmp','bmp');
[D,map]=imread('Dmarked4.bmp','bmp');
%[D,map]=imread('Tmarked.bmp','bmp');
% Convert data in arrays of class double
B=double(B);
```

```
D=double(D);
% read image size
%Read host image
%***************
[W,map]=imread('marked3.bmp','bmp');
% Convert data in arrays of class double
W=double(W);
% read image size of logo image
[row,column,slide]=size(D);
%****** P and Q are position that put watermark in host image *****
P=50; %select for move ROW   (P<(Row of background - Row of Text))
Q=0; %select for move COLUMN(Q<(Column of background - Column of Text))
Y=W;
for k=1:3
  for i=1:row
    for j=1:column
        % Equation of inserting watermarked in host image
        T(i,j,k)=(-2.5*B(i,j,k))-(1.2*D(i,j,k));
      Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-(0.5*T(i,j,k)))/0.8;
    end
  end
end
figure(2)
image(uint8(Y));
title(' Recover Overlapping Watermark Image');
%%% Save file
%imwrite(uint8(Y),'marked2.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(X);
title('Watermarked Image');
```

```
subplot(2,2,3);image(uint8(W));
title('Overlapping Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover Overlapping Watermark Image');
```

**lightover.m**

```
%light overlaping image
%close all
clear all;
[X,map]=imread('logo9_2.bmp','bmp');
% Convert data in arrays of class double
logo=double(X);
% read image size
[row,column]=size(logo);
% Perform embossing watermark image
% 1) devide the host-image into equal-size block
% 2) manipulate those pixels in block (i,j)
for i = 3 : row-2 % considerated pixel at position rows 3 to row-2(amount of 25 pixel)
  for j = 3 : column-2% considerated pixel at position column 3 to row-2
   k=0;
   for n = i-2 : i+2
     for m = j-2 : j+2
       if logo(n,m) <100
         C = 0;
       else
         C = 1;
       end
       k = k+C; % sum for find total pixel at considerated for input color valucd to
       % that pixel consider
     end
   end
```

```
%find color valued for input in (i,j) that  make shading color
%:k/25 is total pixel that consider,255 is for will
%recieve eolor value input into position's pixel embedded and 10 is vary
%valued can any change (for adapt shade color)
Z = (k/25)*255;
% make logo look has dimension,range k = 0-24
if k > 20
Z = 0;% convert background from white to black color
end
%copy binary image to 3 slide image
B(i,j,1) = Z;
B(i,j,2) = Z;
B(i,j,3) = Z;
 end
end
% Convert data in arrays of class unit8
B=uint8(B);
%figure(1)
%subplot(2,2,1);colormap(map);image(B);
imwrite(B,'Bmarked5.bmp','bmp');
%***************
%Rcad host image
%***************
[Y,map]=imread('Goldflowers.jpg','jpg');
% Convert data in arrays of class doublc
B=double(B);
Y=double(Y);
% read image size of logo image
[row1,column1,slide1]=size(B);
P=20; %select for move ROW   (P<(Row of background - Row of Text))
Q=80; %sclcct for move COLUMN(Q<(Column of background - Column of Text))
```

```
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
      T(i,j,k)=1.2*B(i,j,k);
      W(i+P,j+Q,k)=(0.2*T(i,j,k))+(0.999*Y(i+P,j+Q,k));
    end
  end
end
%%% Save file
%%%% Show output after embbed logo image to host image
figure(1)
image(uint8(W));
title('Light Overlaping Watermark Image')
%save file
imwrite(uint8(W),'marked4.bmp','bmp');
%imwrite(uint8(Y),'d:\MATLAB6p1\work\gun\Image\Image_output\marked4.jpg','jpg');
```

**recover_light.m**

```
%close all
clear all;
[A,map]=imread('Goldflowers.jpg','jpg');
[K,map]=imread('logo9_2.bmp','bmp');
[X,map]=imread('Bmarked5.bmp','bmp');
% Convert data in arrays of class double
logo=double(X);
% read image size
%Read host image
%***************
[W,map]=imread('marked4.bmp','bmp');
```

```
% Convert data in arrays of class double
W=double(W);
% read image size of logo image
[row,column,slide]=size(logo);
%****** P and Q are position that put watermark in host image *****
P=20; %selcct for move ROW   (P<(Row of background - Row of Text))
Q=80; %seleet for move COLUMN(Q<(Column of baekground - Column of Text))
Y=W;
for k=1:3
  for i=1:row
    for j=1:column
      % Equation of inserting watermarked in host image
       T(i,j,k)=1.2*logo(i,j,k);
      Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-(0.2*T(i,j,k)))/0.999;
    end
  end
end
%colormap(map);
figure(2)
image(uint8(Y));
title('Recover Light Overlaping Watermark Image');
%%% Save file
%imwrite(uint8(Y),'OKmarked2.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(K);
title('Watermarked Image');
subplot(2,2,3);image(uint8(W));
title('Light Overlapping Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover Light Overlappint Watermark Image');
```

**edge.m**

```
close all
clear all;
[X,map]=imread('logo5.bmp','bmp');%logo max=128 ,very small don't edge
logo=X(:,:,1);
% Convert data in arrays of class double
logo=double(logo);
% read image size
[row,column]=size(logo);
% Perform embossing watermark image
% 1) devide the host-image into equal-size block
% 2) manipulate those pixels in block (i,j)
for i = 3 : row-2 % considerated pixel at position rows 3 to row-2(amount of 25 pixel)
  for j = 3 : column-2% considerated pixel at position column 3 to row-2
    k=0;
    for n = i-2 : i+2
      for m = j-2 : j+2
        if logo(n,m) <100
          C = 0;
        else
          C = 1;
        end
        k = k+C; % sum for find total pixel at considerated for input color valued to
        % that pixel consider
      end
    end
    %find color valued for input in (i,j) that  make shading color
    %:k/25 is total pixel that consider,255 is for will
    %recicve color value input into position's pixel embedded and 10 is vary
    %valued can any change (for adapt shade color)
    Z = (k/25)*255;
```

```
% make logo look has dimension,range k = 0-24 <0.5 is invisible>
if k > 20
Z = 0;% convert background from white to black color
end
%copy binary image to 3 slide image
B(i,j,1) = Z;
B(i,j,2) = Z;
B(i,j,3) = Z;
% goal for make shadow and light on watermark image
% convert image from white to black,black to white ,
% +2 is sliding position of character and
% subtract from B because want shadow & light
%  D(i,j,1) = B(i,j,1)-(255-logo(i+2,j+2));
% D(i,j,2) = B(i,j,2)-(255-logo(i+2,j+2));
%D(i,j,3) = B(i,j,3)-(255-logo(i+2,j+2));
D(i,j,1) = B(i,j,1)-(logo(i,j));
D(i,j,2) = B(i,j,2)-(logo(i,j));
D(i,j,3) = B(i,j,3)-(logo(i,j));
  end
end
% Show output after manipulates logo image to shower and light
% Convert data in arrays of class unit8
B=uint8(B);
D=uint8(D);
%figure(4)
%subplot(2,2,1);colormap(map);image(B);
%subplot(2,2,2);colormap(map);image(D);
imwrite(uint8(B),'Bmarked1.bmp','bmp');
imwrite(uint8(D),'Dmarked1.bmp','bmp');
%Read host image
[Y,map]=imread('lena.bmp','bmp');
```

```
% Convert data in arrays of elass double
B=double(B);
D=double(D);
Y=double(Y);
% read image size of logo image
[row1,column1,slide1]=size(B);
%****** P and Q are position that put watermark in host image *****
P=60; %select for move ROW   (P<(Row of background - Row of Text))
Q=110; %select for move COLUMN(Q<(Column of background - Column of Text))
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
      % Equation of inserting watermarked in host image
      T(i,j,k)=-1.5*B(i,j,k)+2.4*D(i,j,k);
      W(i+P,j+Q,k)=(0.3*T(i,j,k))+(0.999*Y(i+P,j+Q,k));
    end
  end
end
%%%% Show output after embbed logo imagc to host image
figure(1)
image(uint8(W));
title('Edge Only Watermark Image')
%save file
imwrite(uint8(W),'marked6.hmp','bmp');
```

recover_edge.m

```
%close all
clear all;
[A,map]=imread('lena.bmp','bmp');
[X,map]=imrcad('logo5.bmp','bmp');
```

```
[B,map]=imread('Bmarked1.bmp','bmp');
[D,map]=imread('Dmarked1.bmp','bmp');
%[D,map]=imread('Tmarked.bmp','bmp');
% Convert data in arrays of class double
B=double(B);
D=double(D);
% read image size
%Read host image
%***************
[W,map]=imread('marked6.bmp','bmp');
% Convert data in arrays of class double
W=double(W);
% read image size of logo image
[row,column,slide]=size(D);
%****** P and Q are position that put watermark in host image *****
P=60; %select for move ROW   (P<(Row of background - Row of Text))
Q=110; %seleet for move COLUMN(Q<(Column of baekground - Column of Text))
Y=W;
for k=1:3
  for i=1:row
    for j=1:column
      T(i,j,k)=(-1.5*B(i,j,k))+(2.4*D(i,j,k));
      Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-(0.3*T(i,j,k)))/0.999;
    end
  end
end
%colormap(map);
%Y=Y/2;
figure(2)
image(uint8(Y));
titlc(' Recover Edge Only Watermark Image');
%%% Save file
```

```
%imwrite(uint8(Y),'marker3.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(X);
title('Watermarked Image');
subplot(2,2,3);image(uint8(W));
title('Edge Only Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover Edge Only Watermark Image');
```

**harmony.m**

```
%Harmony watermarked image
close all
clear all;
[X,map]=imread('logo4.bmp','bmp');
logo=X(:,:,1);
% Convert data in arrays of class double
logo=double(logo);
% read image size
[row,column]=size(logo);
% Perform embossing watermark image
% 1) devide the host-image into equal-size bloek
% 2) manipulate those pixels in block (i,j)
for i = 3 : row-2 % considerated pixel at position rows 3 to row-2(amount of 25 pixel)
  for j = 3 : column-2% considerated pixel at position column 3 to row-2
    k=0;
    for n = i-2 : i+2
      for m = j-2 : j+2
        if logo(n,m) <100
          C = 0;
```

```
        else
          C =1;
        end
        k = k+C; % sum for find total pixel at considerated for input color valued to
        % that pixel consider
      end
    end
    %find color valued for input in (i,j) that  make shading color
    %:k/25 is total pixel that consider,255 is for will
    %reeieve color value input into position's pixel embedded and 10 is vary
    %valued can any change (for adapt shade color)
    Z = (k/25)*255;
    % make logo look has dimension,range k = 0-25
    if k >16
    Z = 0;% convert background from white to black color
    end
    %copy binary image to 3 slide image
    B(i,j,1) = Z;
    B(i,j,2) = Z;
    B(i,j,3) = Z;
    % goal for make shadow and light on watermark image
    % convert image from white to blaek,black to white ,
    % +2 is sliding position of eharacter and
    % subtraet from B because want shadow & light
    D(i,j,1) =B(i,j,1)-(255-logo(i+2,j+2));
    D(i,j,2) = B(i,j,2)-(255-logo(i+2,j+2));
    D(i,j,3) =B(i,j,3)-(255-logo(i+2,j+2));
  end
end
% Show output after manipulates logo image to shower and light
% Convert data in arrays of class unit8
B=uint8(B);
```

```
D=uint8(D);
%figure(1)
%subplot(2,2,1);colormap(map);image(B);
%subplot(2,2,2);colormap(map);image(D);
imwrite(B,'Bmarked2.bmp','bmp');
imwrite(D,'Dmarked2.bmp','bmp');
%***************
%Read host image
%***************
[Y,map]=imread('CLOUD.jpg','jpg');
% Convert data in arrays of class double
B=double(B);
D=double(D);
Y=double(Y);
% read image size of logo image
[row1,column1,slide1]=size(B);
%****** P and Q are position that put watermark in host image *****
P=40; %select for move ROW   (P<(Row of background - Row of Text))
Q=50; %select for move COLUMN(Q<(Column of background - Column of Text))
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
      % Equation of inserting watermarked in host image
      T(i,j,k)=(-0.4*B(i,j,k))+(1.4*D(i,j,k));
      W(i+P,j+Q,k)=(0.5*T(i,j,k))+(0.8*Y(i+P,j+Q,k));
      %********
    end
  end
end
%%%% Show output after embbed logo image to host image
figure(1)
```

```
image(uint8(W));
title('Harmony Watermark Image')
%save file
imwrite(uint8(W),'marked7.bmp','bmp');
```

**recover_har.m**

```
%close all
clear all;
[A,map]=imread('CLOUD.jpg','jpg');
[X,map]=imread('logo4.bmp','bmp');
[B,map]=imread('Bmarked2.bmp','bmp');
[D,map]=imread('Dmarked2.bmp','bmp');
%[D,map]=imread('Tmarked.bmp','bmp');
% Convert data in arrays of class double
B=double(B);
D=double(D);
% read image size
%Read host image
%***************
[W,map]=imread('marked7.bmp','bmp');
% Convert data in arrays of class double
W=double(W);
% read image size of logo image
[row,column,slide]=size(D);
%****** P and Q are position that put watermark in host image *****
P=40; %select for move ROW   (P<(Row of background - Row of Text))
Q=50; %seleet for move COLUMN(Q<(Column of background - Column of Text))
Y=W;
for k=1:3
  for i=1:row
```

```
    for j=1:column
        % Equation of inserting watermarked in host image
        T(i,j,k)=(-0.4*B(i,j,k))+(1.4*D(i,j,k));
        Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-(0.5*T(i,j,k)))/0.8;
    end
  end
end
%colormap(map);
%Y=Y/2;
figure(2)
image(uint8(Y));
title(' Recover Harmony Watermark Image');
%%% Save file
%imwrite(uint8(Y),'marker2.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(X);
title('Watermarked Image');
subplot(2,2,3);image(uint8(W));
title('Harmony Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover Harmony Watermark Image');
```

**three_D.m**

```
% 3 Dimension watermarked image
close all
clear all;
[X,map]=imread('logo4.bmp','bmp');
% Convert data in arrays of class double
logo=double(X);
```

```
% read image size
[row,eolumn]=size(logo);
% Perform embossing watermark image
% 1) devide the host-image into equal-size block
% 2) manipulate those pixels in block (i,j)
for i = 3 : row-2 % considerated pixel at position rows 3 to row-2(amount of 25 pixel)
  for j = 3 : column-2% considerated pixel at position column 3 to row-2
    k=0;
    for n = i-2 : i+2
      for m = j-2 : j+2
        if logo(n,m) <100
          C = 0;
        else
          C = 1;
        end
        k = k+C; % sum for find total pixel at considerated for input color valued to
        % that pixel consider
      end
    end
    %find color valued for input in (i,j) that  make shading color
    %:k/25 is total pixel that consider,255 is for will
    %recieve color value input into position's pixel embedded and 10 is vary
    %valued can any change (for adapt shade color)
    Z = (k/25)*255;
    % make logo look has dimension,range k = 0-24
    if k>24
    Z = 0;% convert background from white to black color
    end
    %copy binary image to 3 slidc image
    B(i,j,1) = Z;
    B(i,j,2) = Z;
```

```
    B(i,j,3) = Z;
    % goal for make shadow and light on watermark image
    % convert image from white to black,black to white ,
    % +2 is sliding position of character and
    % subtract from B because want shadow & light
    D(i,j,1) = B(i,j,1)-(logo(i+2,j+2));
    D(i,j,2) = B(i,j,2)-(logo(i+2,j+2));
    D(i,j,3) = B(i,j,3)-(logo(i+2,j+2));
  end
end
% Convert data in arrays of class unit8
D=uint8(D);
%figure(1)
%subplot(2,2,1);colormap(map);image(D);
imwrite(uint8(D),'Dmarked7.bmp','bmp');
D=double(D);
%***************
%Read host image
%***************
[Y,map]=imread('rega.jpg','jpg');
% Convert data in arrays of class double
Y=double(Y);
% read image size of logo image
[row1,column1,slide1]=size(D);
%****** P and Q are position that put watermark in host image *****
P=50; %select for move ROW   (P<(Row of background - Row of Text))
Q=0; %select for move COLUMN(Q<(Column of background - Column of Text))
W=Y;
for k=1:3
  for i=1:row1
    for j=1:column1
      % Equation of inserting watermarked in host image
```

MISSING

```
Q=0; %select for move COLUMN(Q<(Column of baekground - Column of Text))
Y=W;
for k=1:3
  for i=1:row
    for j=1:eolumn
       % Equation of inserting watermarked in host image
       T(i,j,k)=2.4*logo(i,j,k);
        Y(i+P,j+Q,k)=(W(i+P,j+Q,k)-0.08*T(i,j,k))/0.999;
    end
  end
end
%eolormap(map);
figure(2)
image(uint8(Y));
title('Recover 3-D Watermark Image');
%%% Save file
%imwrite(uint8(Y),'marked2.bmp','bmp');
figure(3)
subplot(2,2,1);image(uint8(A));
title('Original Image');
subplot(2,2,2);imshow(K);
title('Watermarked Image');
subplot(2,2,3);image(uint8(W));
title('3D Watermark Image');
subplot(2,2,4);image(uint8(Y));
title('Recover 3D Watermark Image');
```

**invisible.m**

```
%       Invisible Watermark embedding
close all;
clear all;
```

```
k=20;          % set the gain factor for embedding
blocksize=16;  % set the size of the block in cover to be used for each bit in watermark
% read in the host
[host,map]=imread('pout.tif','tif');
host=double(host);
% determine size of watermarked image
[M_h,N_h]=size(host);
% determine maximum logo size based on host, and blocksize
max_logo=M_h*N_h/(blocksize^2);
% read in the logo image
[logo,map]=imread('_logo16_1.bmp','bmp');
logo=double(logo);
[M_p,N_p]=size(logo);
% reshape the logo to a vector (from matrix)
logo=round(reshape(logo,Mm*Nm,1)./256);
% check that the logo isn't too large for cover
if (length(logo) > max_logo)
   error('Logo too large to fit in host')
end
% pad the logo out to the maximum logo size with 0's
logo_vector=ones(1,max_logo);
logo_vector(1:length(logo))=logo;
% read in key for PN generator
%[key,map]=imread('_key.hmp','bmp');
%key=double(key)./256;
key=14641;  % assumed key
% reset MATLAB's PN generator to state "key"
rand('state',key);
% generate PN sequences to designate "1" and "0"
pn_one=round(2*(rand(blocksize,bloeksize)-0.5));
pn_zero=round(2*(rand(blocksize,hlocksize)-0.5));
% find two highly un-correlated PN sequences
```

```
while (corr2(pn_one,pn_zero) > -0.1)
  pn_one=round(2*(rand(blocksize,blocksize)-0.5));
  pn_zero=round(2*(rand(blocksize,blocksize)-0.5));
end
% process the image in blocks
% first construct the global watermark mask
x=1;
y=1;
for (kk = 1:length(logo_vector))
  % if logo bit contains zero, add PN sequence to that portion of mask
  if (logo_vector(kk) == 0)
    watermark(y:y+blocksize-1,x:x+blocksize-1) = pn_zero;

  % otherwise mask is filled with zeros
  else
    watermark(y:y+blocksize-1,x:x+blocksize-1) = pn_one;
  end
  % move to next block of mask along x; If at end of row, move to next row
  if (x+blocksize) >= N_h
    x=1;
    y=y+blocksize;
  else
    x=x+blocksize;
  end
end

% add watermark mask to host image using gain factor k
watermarked1=host+k*watermark;
watermarked2=uint8(host+k*watermark);
% write the watermarked image out to a file
%save file
imwrite(watermarked2,'invis_watermark.bmp','bmp'); % output
```

```
% display watermarked image
figure(1)
imshow(watermarked2,[])
title('Watermarked Image')
```

**recover_invisible.m**

```
%          Invisible Watermark Recovery
clear all;
blocksize=16;     % set the size of the block in cover to be used for each bit in watermark
% read in the watermarked object
[host,map]=imread('pout.tif','tif');
[watermarked,map]=imread('invis_watermark.bmp','bmp');
watermarked=double(watermarked);
% determine size of watermarked image
[M_w,N_w]=size(watermarked);
% determine maximum possible logo size in object
max_logo=M_w*N_w/(blocksize^2);
% read in original watermark
[logo1,map]=imread('_logo16_1.bmp','bmp');
[orig_watermark,map]=imread('_logo16_1.bmp','bmp');
orig_watermark=double(orig_watermark);
% determine size of original watermark
[M_o,N_o]=size(orig_watermark);
% read in key for PN generator
%[key,map]=imread('cpe_key.bmp','bmp');
%key=double(key)./256;
key=14641;  % assumed key
% reset MATLAB's PN generator to state "key"
rand('state',key);
% generate PN sequences to designate "1" and "0"
watermark_one=round(2*(rand(blocksize,blocksize)-0.5));
```

```
watermark_zero=round(2*(rand(blocksize,blocksize)-0.5));
% find two highly un-correlated PN sequences
while (corr2(watermark_one,watermark_zero) > -0.1)
   watermark_one=round(2*(rand(blocksize,blocksize)-0.5));
   watermark_zero=round(2*(rand(blocksize,blocksize)-0.5));
end
% pad message out to maximum logo size with ones
logo_vector=ones(max_logo,1);
% process the image in blocks
% for each block determine it's correlation with base pn sequence
x=1;
y=1;
for (kk = 1:length(logo_vector))
   % calculate correlations for both PN sequences
   correlation_one(kk)=corr2(watermarked(y:y+blocksize-1,x:x+blocksize-
1),watermark_one);
   correlation_zero(kk)=corr2(watermarked(y:y+blocksize-1,x:x+blocksize-
1),watermark_zero);
   % choose which ever correlation is higher
   if correlation_one(kk) > correlation_zero(kk)
      logo_vector(kk)=1;
   else
      logo_vector(kk)=0;
   end

   % move on to next block. At and of row move to next row
   if (x+blocksize) >= N_w
      x=1;
      y=y+blocksize;
   else
      x=x+blocksize;
   end
```

```
end
% reshape the logo
logo=reshape(logo_vector(1:M_o*N_o),M_o,N_o);
% display the recovered logo
figure(2)
imshow(logo,[])
title('Recovered Watermarked')
figure(3)
subplot(2,2,1);imsbow(uint8(host));
title('Original Image');
subplot(2,2,2);imshow(logo1);
title('Watermark');
subplot(2,2,3);imshow(watermarked,[]);
title('Watermarked Image');
subplot(2,2,4);imshow(logo,[]);
title('Recovered Watermarked')
```

# Index Source Code


| | | หน้า |
|---|---|---|
| test_present.m | (GUI) | 42 |
| ordinary.m | (Ordinary Overlapping) | 46 |
| recover_ordi.m | (กู้กลับคืน Ordinary Overlapping) | 47 |
| overlap.m | (Overlapping) | 49 |
| recover_over.m | (กู้กลับคืน Overlapping) | 51 |
| lightover.m | (Light Overlapping) | 53 |
| reeover_light.m | (กู้กลับคืน Light Overlapping) | 55 |
| edge.m | (Edge Only) | 57 |
| recover_edge.m | (กู้กลับคืน Edge Only) | 59 |
| harmony.m | (Harmony) | 61 |
| recover_har.m | (กู้กลับคืน Harmony) | 64 |
| three_D.m | (Three Dimension) | 65 |
| recover_three.m | (กู้กลับคืน Three Dimension) | 68 |
| invisible.m | (ภาพพิมพ์ลายน้ำที่มองไม่เห็น) | 69 |
| recover_invis.m | (กู้กลับคืน ภาพพิมพ์ลายน้ำที่มองไม่เห็น) | 72 |