กระบวนวิชา 261453 Digital Image Processing
รายงานวิชา 261453

จัดทำโดย

นาย พีรวิชญ์ สมัย
รหัสนักศึกษา 650610876

เสนอ
รศ.ดร.ศันสนีย์ เอื้อพันธ์วิริยะกุล

รายงานนี้เป็นส่วนหนึ่งของวิชา261453
ภาคเรียนที่ 2/2568

มหาวิทยาลัยเชียงใหม่

คำนำ

รายงานนี้เป็นส่วนหนึ่งของรายวิชา261453 Digital Image Processing โดยมีวัตถุประสงค์เพื่อใช้การแจกแจงของHistogram ของภาพ, การใช้ Algebraic Operation เป็นการดำเนินการพื้นฐานบวกลบคูณหารกับภาพinput ในchannel r g b เพื่อเน้นจุดที่สำคัญของภาพ ,ศึกษาการประยุกต์ใช้ของ Histogram Equalization การดำเนินการแบบจุด(Point Operation) และการกู้คืนภาพของวัตถุที่ถูกบิดโดยใช้พื้นฐานของBilinear interpolation

ในส่วนการพัฒนาโค้ดทั้งหมด ดำเนินการด้วยภาษา Python โดยในบางข้อนั้นไม่มีการเรียกใช้ไลบรารีภายนอก เพื่อให้ผู้จัดทำเข้าใจพื้นฐานของการประมวลผลภาพแบบดิจิตอลได้อย่างลึกซึ้ง

1. Histogram and Object Moment Let the binary image f(x,y) of an object be defined by

$$f(x,y)=\begin{cases}1 & \text{if the pixel belongs to the object}\\ 0 & \text{if the pixel belongs to the background}\end{cases}$$

The pq-moment m_{pq} of an object is defined as

$$m_{pq}=\iint_D x^p y^q f(x,y)dxdy \qquad \text{where } p,q=0,1,2,...$$

Where $\hat{x}=m_{10}/m_{00}$ and $\hat{y}=m_{01}/m_{00}$ (These two quantities represent the coordinates of the "center of mass" of the object.) In the discrete case, the integrals are replaced by summations. Normalized moments are defined by

$$\eta_{pq}=\frac{\mu_{pq}}{\mu_{00}^{[(p+q)/2]+1}} \quad \text{for } p+q=2,3,...$$

In theory, the quantity

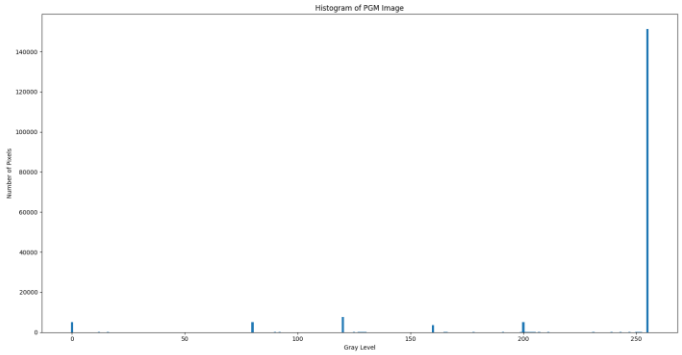
$$\emptyset_1=\eta_{20}+\eta_{02}$$

invariant to translation, rotation and scaling of the object. In this project, you will compute this quantity for various translated, rotated and scaled version of shape and check its invariance. The name of the image containing the various version of the object is "scaled_shapes.pgm" (image size = 325x553). Each version of the object has been given a different "color" (i.e., gray value). Your tasks are the following:

1. Compute the histogram of the image and determine how many objects are in the image and the gray level of each.

<u>ผลการคำนวณ Histogram</u>

เมื่อนำภาพ scaled_shapes.pgm มาพล็อต Histogram โดยให้แกนตั้งคือจำนวนของ pixels แกนนอนคือค่าของ gray level ซึ่งมีค่าตั้งแต่ 0-255 โดยที่ 0 คือภาพที่มีค่าน้ำหนักต่ำสุด (สีดำ) และ 255 เป็นค่าของสีขาวนั้นจะได้ผลดังรูป



Histogram ของภาพ scaled_shapes.pgm

วิเคราะห์ผล

จาก Histogram ที่ได้จะให้ค่าสีขาวเป็นพื้นหลังของวัตถุ ซึ่งจะได้ว่า255เป็นค่าของพื้นหลังสีขาว ส่วนค่าอื่นๆที่ปรากฏใน Histogram ณ ค่าของ gray level ที่ต่างกันนั้นจะเป็นค่า gray level ของวัตถุอื่นๆในภาพ จากผลของHistogramที่ได้ ได้แก่ 0, 80, 120, 160, 200 ซึ่งมีทั้งหมด 5 ค่าที่แตกต่างกันของแต่ละจุดยอด ดังนั้นภาพนี้จะมีวัตถุทั้งหมด 5 วัตถุที่ค่าสีที่แตกต่างกันของภาพ scaled_shapes.pgm

2. Write a procedure to compute the (central)moment of an object given its gray level and use this procedure to compute the central moments μ_{20} and μ_{02}. Using this value, compute ϕ_1 and verify its invariance (proof that ϕ_1 is constant). You can look at the pdf file (CompHw1.pdf) if the equations are not clear.

invariant moment ตัวที่ 1 (ϕ_1) ที่คำนวณจาก central moment μ_{20} และ μ_{02} หลังจากทำ normalization แล้ว เป็นค่าที่ไม่เปลี่ยนแปลงต่อการเลื่อนตำแหน่ง การหมุน และการย่อ–ขยายของวัตถุ ดังนั้น ϕ_1 จึงสามารถใช้เป็นคุณลักษณะ (feature) สำหรับอธิบายและเปรียบเทียบรูปร่างของวัตถุในภาพได้อย่างมีประสิทธิภาพ

การคำนวณ Raw Moment และจุดศูนย์กลางของวัตถุ
จากโจทย์มีฟังก์ชันภาพของวัตถุเป็น

$$f(x,y)=\begin{cases}1 & \text{if the pixel belongs to the object}\\ 0 & \text{if the pixel belongs to the background}\end{cases}$$

Raw moment ของ (p,q) คือ

$$m_{pq}=\iint_D x^p y^q f(x,y)dxdy \qquad \text{where } p,q=0,1,2,...$$

แปลงจาก continuous raw moment จะได้ Discrete raw moment จะได้

ค่าที่ใช้ในการคำนวณจุดศูนย์กลางของวัตถุ ได้แก่

พื้นที่ของวัตถุ

moment เชิงตำแหน่ง

ตำแหน่งศูนย์กลาง (centroid) ของวัตถุคำนวณได้จาก
Centroid

จาก
Central moment เป็น moment ที่คำนวณรอบจุดศูนย์กลางของวัตถุ โดย

หลังจากนั้นคำนวณหา central moment ลำดับสองโดยให้ จะได้ว่า

Normalized Central Moment
เพื่อให้ moment ไม่ขึ้นกับขนาดของวัตถุ (scale invariant) จึงทำการ normalization ด้วยการหารด้วย m_{00}^2เป็นการ Normalized ค่าของ Moment

$$\eta_{20} = \frac{\mu_{20}}{m_{00}^2}, \eta_{02} = \frac{\mu_{02}}{m_{00}^2}$$

ทำการตรวจสอบความไม่เปลี่ยนแปลงต่อการเลื่อนตำแหน่ง (Translation Invariance)
เมื่อวัตถุถูกเลื่อนตำแหน่ง จุดศูนย์กลางของวัตถุจะเลื่อนตามไปด้วย ทำให้ผลต่างของ $(x - \bar{x})$, $(y - \bar{y})$ไม่เปลี่ยน ดังนั้นค่า central moment μ_{20}และ μ_{02}จะไม่เปลี่ยน ส่งผลให้ ϕ_1คงเดิม

ตรวจสอบความไม่เปลี่ยนแปลงต่อการย่อ–ขยาย (Scale Invariance)
เมื่อวัตถุถูกย่อหรือขยาย central moment μ_{20}และ μ_{02}จะเปลี่ยนตามอัตราส่วนของขนาดวัตถุ อย่างไรก็ตาม เมื่อทำ normalization ด้วย m_{00}^2จะทำให้ค่า η_{20}และ η_{02}ไม่เปลี่ยน ดังนั้นค่า ϕ_1จึงไม่ขึ้นกับ

ขนาดของวัตถุ

ตรวจสอบความไม่เปลี่ยนแปลงต่อการหมุน (Rotation Invariance)
ภายใต้การหมุน ค่า μ_{20}และ μ_{02}อาจเปลี่ยนค่า แต่ผลรวมของ normalized moment ทั้งสองค่า ซึ่งคือ ϕ_1จะยังคงเดิม เนื่องจากเป็นค่าที่แสดงการกระจายของวัตถุรอบจุดศูนย์กลางโดยไม่ขึ้นกับทิศทางของแกน
สรุปผล
จากขั้นตอนการคำนวณ central moment μ_{20}และ μ_{02} และการนำไปคำนวณ invariant moment ตัวที่ 1 (ϕ_1) พบว่า ค่า ϕ_1เป็นค่าคงที่สำหรับวัตถุเดียวกัน แม้ว่าวัตถุจะถูกเลื่อนตำแหน่ง หมุน หรือย่อ–ขยาย ดังนั้น ϕ_1 จึงสามารถใช้เป็นตัวแทนลักษณะรูปร่างของวัตถุในภาพได้
(โปรแกรมคำนวณอยู่ในส่วนของภาคผนวกในฟังก์ชันdef compute_phi1(*image*, *width*, *height*, *gray*):)

ผลการคำนวนค่า ϕ_1ในแต่ละค่า gray level

```
Question 1.2 Results:
Gray Level    phi1
     0      0.301531
    80      0.301193
   120      0.288182
   160      0.264799
   200      0.317671
```

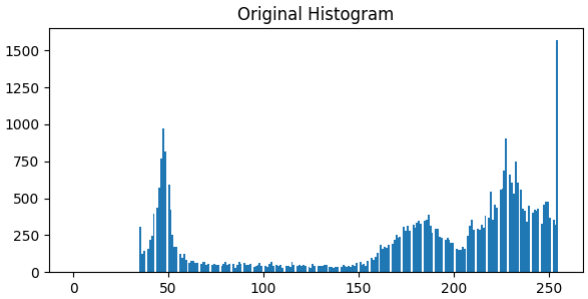
ภาพpgmที่ได้

2. Point Operations This part of the assignment involves two images, “Cameraman.pgm” (200 x 200) and “SEM256_256.pgm” (256 x 256). The image of Cameraman is rather bright and the seed image is dark. Your goal is to improve the appearance and bring out the details using point operations. You can use any software to do histogram equalization. You may also use your own programs if you wish. Your report in this part must include a description of the method you used to enhance the image, and the histograms and images before and after enhancement. Discuss your results and say what is good or bad about them.

1.นำรูปตัวอย่าง pgm มาอ่านภาพและหา Histogram ทั้ง 2 รูป

ภาพ Cameraman.pgm

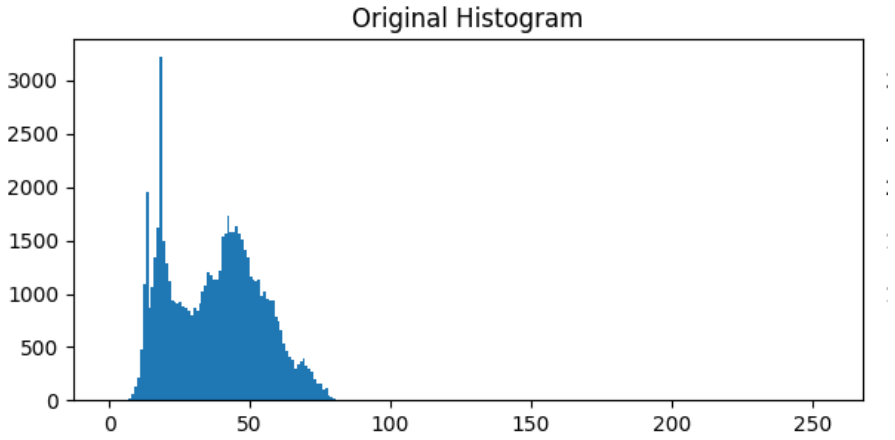
ภาพ Cameraman.pgm ก่อนทำHistogram equalization



การแจกแจงของ Histogram ของ Cameraman.pgm ก่อนทำHistogram equalization

จะได้ว่าภาพCameraman.pgmมีค่าความเข้มของสีมากระหว่างช่วง150-250 และจุดที่มืดที่สุดจะไม่ใช่ค่าความเข้มที่เป็น 0 (จาก Histogram)

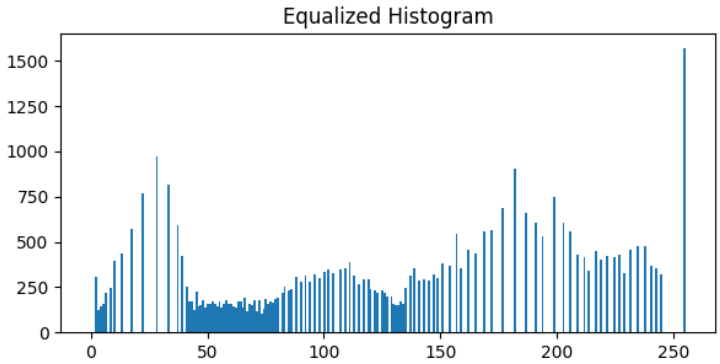
ภาพ SEM256_256.pgm
ก่อนทำHistogram equalization



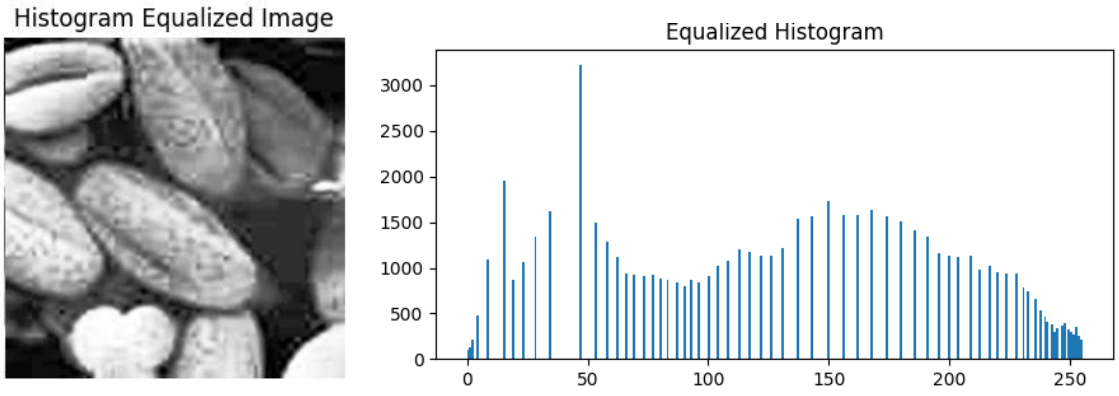
การแจกแจงของ Histogram ของ SEM256_256.pgm ก่อนทำHistogram equalization

ผลการทำ Histogram Equalization

Cameraman.pgm



ภาพ SEM256_256.pgm

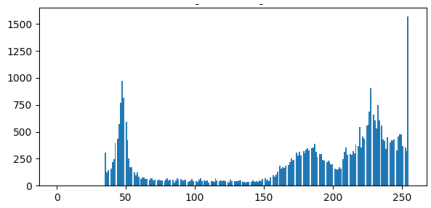
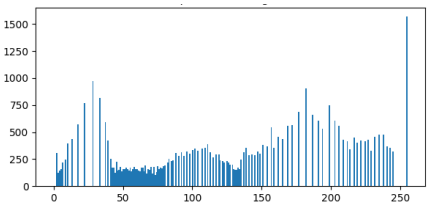
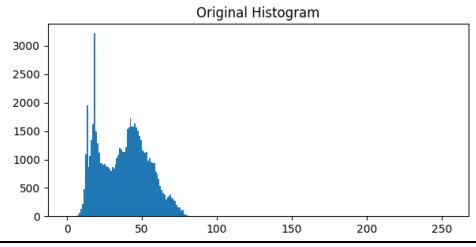
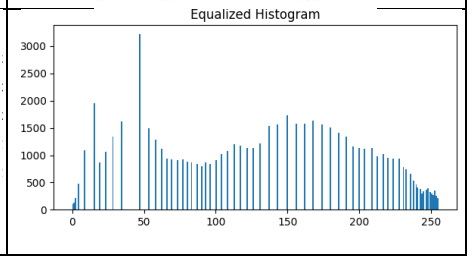


จากผลการทำ Histogram Equalization ซึ่งเป็นการดำเนินการแบบจุดของรูปอินพุตเพื่อให้ Output Histogram มีความถี่เท่ากันในทุกค่าสีเท่านั้นจะได้ว่า

ภาพของช่างภาพ(Cameraman.pgm) จุดที่เข้มที่สุดของภาพจะกระจายออกไปที่บริเวณอื่นเนื่องจากผลของการกระจายของค่าสีของการทำHistogram Equaliozation ทำให้บริเวณของพื้นหลังจากเดิมที่มีค่าสีไปทางอ่อนนั้นกลับมาเข้มขึ้น โดยค่าความเข้มจะกระจายออกมาจากการทำ Histogram equalization ของ input image

ภาพของเมล็ดกาแฟ(SEM256_256.pgm) จุดที่เข้มที่สุดของภาพจะกระจายออกไปที่บริเวณอื่นเนื่องจากผลของการกระจายของค่าสีของการทำHistogram Equalization ทำให้บริเวณของพื้นหลังจากเดิมที่มีค่าสีไปทางอ่อนนั้นกลับมาเข้มขึ้นเช่นเดียวกัน โดยค่าความเข้มจะกระจายออกมาจากการทำ Histogram equalization ของ input image แต่จะมีความต่างกับ Cameraman.pgm ในส่วนของวัตถุเมล็ดกาแฟที่มีความสว่างขึ้นจากเดิม เนื่องจาก Histogram กระจายความถี่ออกไปในทุกๆค่าสีอย่างสม่ำเสมอ

ตารางเปรียบเทียบรูป ก่อน-หลังทำ Histogram Equalization

image	Before histogram equalization	After Histogram equalization
Cameraman.pgm		
Histogram		
SEM256_256.pgm		
Histogram		

3. Algebraic Operations You will find the red, green and blue component images of 414 x 800 outdoor scene. The three color channels of this color image are called "SanFranPeak_red.pgm" (red), "SanFranPeak_green.pgm" (green), and "San_FranPeak_blue.pgm" (blue). You task is to combine the images using algebraic operations so that the different parts of the image are emphasized. Let g be the green image, r be the red image, and b be the blue image. Some traditional favorite are 2g-r-b (excess green) and r-b (red-blue difference). Actually (r+b+g)/3 creates a gray-level (intensity) image. The excess green image generally emphasizes trees and grass. You can invent your own algebraic combinations to bring out different objects. Your report on this part must include description of the algebraic operation you used to bring out different objects, and the images after the operation. Discuss your results and say what is good or bad about them.

<u>หลักการของ Algebraic Operations ในภาพสี</u>

ภาพสีแบบ RGB สามารถมองว่าเป็นภาพระดับเทาสามภาพที่ซ้อนกัน โดยแต่ละช่องสีมีข้อมูลเกี่ยวกับวัตถุแตกต่างกัน เช่น

- ช่องสีเขียว (g) มักมีค่าสูงในบริเวณพืช ใบไม้ และหญ้า
- ช่องสีแดง (r) เด่นในบริเวณดิน อาคาร หรือวัตถุที่สะท้อนแสงแดง
- ช่องสีน้ำเงิน (b) เด่นในบริเวณท้องฟ้า น้ำ หรือเงา

การนำช่องสีเหล่านี้มาบวก ลบ หรือถ่วงน้ำหนัก จะช่วยเน้นหรือลดทอนวัตถุประเภทต่าง ๆ ได้

<u>ผลการทดลอง</u>

จากการทำ Algebraic Operations โดยมี Algebraic Operation ที่ต่างกันนั้น จะให้ภาพ output คือ

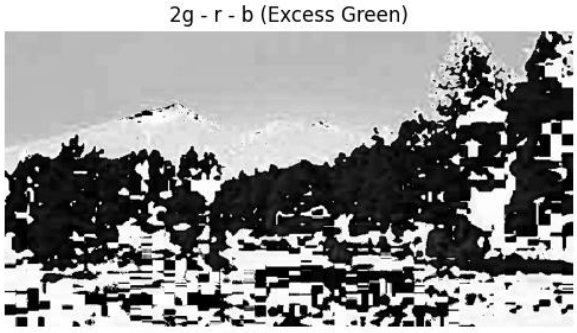
1) Excess Green Image (2g – r – b)

การคำนวณภาพ excess green มีสมการคือ

$$E_g = 2g - r - b$$

สมการนี้เพิ่มน้ำหนักให้ช่องสีเขียว และลบอิทธิพลของสีแดงและสีน้ำเงินออก(เทอม-r-b) โดยเพิ่มค่าสีของ Channel สีเขียวเป็น2เท่าให้ชัดเจนมากขึ้น และหักลบออกด้วยค่าของสีแดงและสีน้ำเงิน

ผลที่ได้



จะได้ว่าบริเวณที่มีพืช เช่น ต้นไม้และใบหญ้า จะปรากฏสว่างและเด่นชัดในลักษณะของภาพที่สลับสีขาวดำบริเวณอื่นๆและท้องฟ้าจะถูกลดความเด่นลง

ข้อดี

เหมาะสำหรับการเน้นพืชพรรณทีมีค่าของสีเขียวชัดเจน และวิธีนี้จะสามารถแยกพื้นที่ทสีเขียวที่มีมากๆออกจากฉากหลังได้ชัดเจน

ข้อเสีย

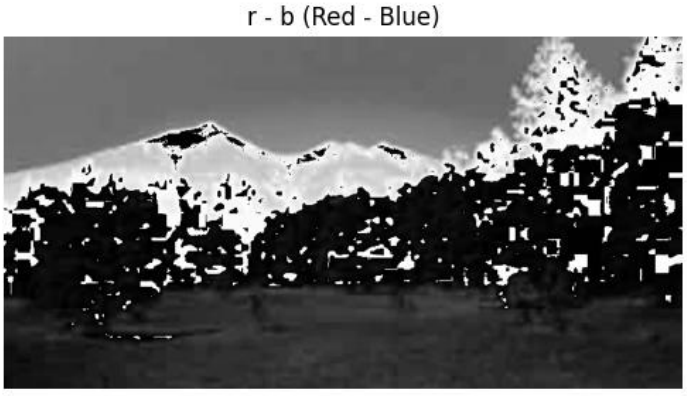
หากเจอวัตถุที่ไม่ใช่พืช(มีองค์ประกอบของสีเขียวน้อยหรือมีองค์ประกอบของีอื่นผสมอยู่)จะมืดลงหรือปรากฏค่าสีเทาทีผลของ algebraic operation และส่งผลรายละเอียดในบริเวณอื่นเกิดการ
สูญหาย ในกรณีที่เป็นวัตถุสีเขียวที่มีองค์ประกอบของสีอื่นอยู่ด้วยก็จะถูกลดทอนลงไปให้ค่าเข้าใกล้สีดำไปด้วย

2.Red–Blue Difference Image (r – b)

การหาผลต่างระหว่างช่องสีแดงและสีน้ำเงินมีสมการคือ

$$D_{rb} = r - b$$

ผลที่ได้



ผลที่ได้

ช่วยเน้นวัตถุที่มีองค์ประกอบสีแดงมากกว่าสีน้ำเงิน พื้นดินที่มีโทนสีแดงจะเด่นขึ้น และท้องฟ้าและเงาซึ่งมักมีสีน้ำเงินจะถูกลดทอนลง

ข้อดี

แยกอาคารหรือวัตถุที่มีโทนสีแดงได้ดี ช่วยลดอิทธิพลของท้องฟ้าในการจำแนกวัตถุ

ข้อเสีย

พืชพรรณอาจไม่ได้ถูกเน้นให้เด่นขึ้นและจะถูกลดค่าลงเป็นสีดำเนื่องจากมีค่าของสีน้ำเงินมากกว่าสีแดง เกิดค่าติดลบ ทำให้บางครั้งในการทำงานต้องมีการปรับช่วงค่า (normalization)ให้เหมาะสม

3.การสร้างภาพระดับเทา (Intensity Image)

การหาผลของการสร้างภาพระดับสีเทา มีสมการคือ

$$I = \frac{r + g + b}{3}$$

ผลที่ได้

(r + g + b) / 3 (Grayscale)

จะได้ว่าภาพสีเทา (intensity image) สามารถคำนวณได้จากค่าเฉลี่ยของทั้งสามช่องสี ภาพที่ได้จะแสดงความสว่างโดยรวมของฉาก โดยไม่เน้นสีใดเป็นพิเศษ เหมาะสำหรับการดูโครงสร้างโดยรวมของภาพ แต่ไม่สามารถแยกวัตถุจากสีได้ชัดเจน

ข้อดี

แสดงความสว่างของฉากได้ดี ไม่มีการเน้นสีใดสีหนึ่งจนมากเกินไป สามารถมองเห็นโครงสร้างโดยรวมของภาพได้ชัดเจน และเห็นระยะของวัตถุได้ชัดเจนมากขึ้น

ข้อเสีย

รายละเอียดเชิงสีหายไป ไม่เหมาะสำหรับแยกวัตถุที่อาศัยความแตกต่างของสีในกรณีที่วัตถุมีค่าสีใกล้เคียงกันมากๆ ทำให้เกิดปัญหาในการมองโครงสร้างโดยรวมของภาพได้

4.การผสมเชิงพีชคณิตอื่น ๆ (ที่ผู้ทดลองออกแบบ)

ตัวอย่างเช่น

$$E_{br} = b - r$$

b - r (Sky Emphasis)

ภาพที่ได้จะแสดงความสว่างของท้องฟ้า โดยจะเน้นให้ท่องฟ้าเป็นค่าสีเทา และแยกวัตถุระยะหน้าและระยะกลาง(ในที่นี้คือต้นไม้และภูเขา)ไว้ที่ค่าน้ำหนักเข้มสุดและอ่อนสุด

ข้อดี

แสดงความสว่างของวัตถุระยะหน้าได้ดี และรายละเอียดหลักของวัตถุระยะหน้าไม่สูญหาย

ข้อเสีย

ความสว่างของวัตถุที่ต้องการแยกจะมองได้ยาก เนื่องจากอยู่ในช่วงของgrayscale ทำให้มองวัตถุอื่นที่ไม่ต้องการแยกได้ง่ายกว่าวัตถุหลักทำให้บางครั้งเกิดความผิดพลาดได้

การอภิปรายผล

จากการทดลองพบว่า algebraic operations สามารถเน้นวัตถุในภาพได้อย่างมีประสิทธิภาพโดยไม่ต้องใช้วิธีที่ซับซ้อนมาก ซึ่งวิธีนี้หลักๆนั้นใช้เพียงการดำเนินการพื้นฐานบวกลบคูณและหาร ทำให้เห็นว่าการเลือกสมการมีผลอย่างมากต่อผลของภาพที่ได้ หากเลือกสมการไม่เหมาะสม อาจทำให้รายละเอียดบางส่วนสูญหาย หรือเกิด noise สูงในบางบริเวณ(จำแนกภาพที่มีสีต่างกันได้ยาก) ดังนั้นการเลือกใช้สมการควรพิจารณาจากลักษณะของฉากและวัตถุที่ต้องการเน้นเป็นหลัก

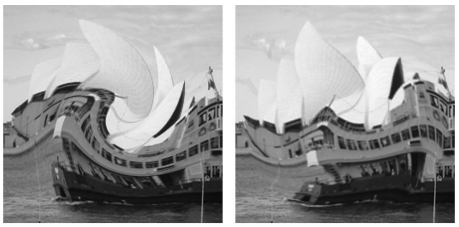
สรุป

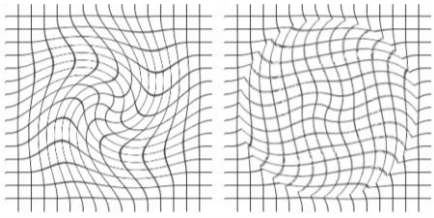
การรวมภาพสีด้วย algebraic operations เป็นวิธีที่ง่ายและมีประสิทธิภาพในการเน้นวัตถุประเภทต่าง ๆ ในภาพสี เช่น พืชพรรณ อาคาร หรือท้องฟ้า โดยสมการแต่ละแบบมีข้อดีและข้อจำกัดแตกต่างกัน การเลือกใช้สมการที่เหมาะสมตามวัตถุปรางค์การใช้งานจะช่วยให้ได้ผลลัพธ์ที่สอดคล้องกับวัตถุประสงค์ของการวิเคราะห์ภาพมากที่สุด

4. Geometric Operations You are given a 256 x 256 grid image called “grid.pgm” and a distorted version “distgrid.pgm”. Use these images to derive a set of spatial transformations between chosen control regions. Use the set of spatial transformations to “undistort” poor old “Lenna”. The distorted Lenna may be found in “disOperaHouse.pgm”. Note that you may also need an interpolation algorithm. You may use as many control regions as you wish, depending on your patience. You report on this part must include a description of the transformation and interpolation method you used, the number of control regions used, and the images before and after the operation. Discuss your results and say what is good or bad about them

Your report should consist of a brief description of the algorithms/formulas used, print-out of the results, a discussion of the results, and a well documented program listing.

จะได้ผลการทำ Geometric Operations คือ





การหมุนภาพแบบเฉพาะบริเวณ (Local Rotation) ด้วย Bilinear Interpolation

ในงานนี้ได้พัฒนาโปรแกรมสำหรับการแปลงภาพแบบเรขาคณิต ซึ่งทางผู้จัดทำได้ทำเป็นการหมุนภาพเฉพาะบริเวณใกล้จุดศูนย์กลางของภาพ (Local Rotation) แทนการหมุนทั้งภาพ วิธีนี้ช่วยให้เกิดเอฟเฟกต์การบิดของภาพเฉพาะจุดที่สนใจ ขณะที่บริเวณอื่นของภาพยังคงสภาพเดิม

<u>ขั้นตอนแรก</u> โปรแกรมทำการอ่านภาพชนิด PGM แบบ P5 ซึ่งเป็นภาพระดับสีเทา จากนั้นกำหนดจุดศูนย์กลางของภาพโดยคำนวณจากความกว้างและความสูงของภาพ เพื่อใช้เป็นจุดอ้างอิงในการคำนวณการหมุน

สำหรับพิกเซลแต่ละตำแหน่ง จะคำนวณระยะห่างจากจุดศูนย์กลางของภาพ หากพิกเซลนั้นอยู่ภายนอกรัศมีที่กำหนด จะไม่ทำการหมุนและคัดลอกค่าความเข้มจากภาพต้นฉบับโดยตรง แต่หากพิกเซลอยู่ภายในรัศมีดังกล่าว จะทำการคำนวณมุมหมุนซึ่งมีค่าเปลี่ยนตามระยะจากจุดศูนย์กลาง จะได้ว่าพิกเซลที่อยู่ใกล้จุดศูนย์กลางจะมีมุมหมุนมากที่สุด และมุมหมุนจะค่อย ๆ ลดลงจนเป็นศูนย์เมื่อเข้าใกล้ขอบเขตของรัศมีที่กำหนด ซึ่งช่วยให้การเปลี่ยนแปลงของภาพมีความต่อเนื่องและไม่เกิดขอบเขตที่แข็งกระด้าง

ในด้านของการคำนวณตำแหน่งพิกเซลใหม่จะใช้หลักการ Backward Mapping โดยพิจารณาตำแหน่งพิกเซลในภาพผลลัพธ์ แล้วย้อนกลับไปหาตำแหน่งที่สอดคล้องกันในภาพต้นฉบับ วิธีนี้ช่วยหลีกเลี่ยงปัญหาการเกิดช่องว่าง (holes) ในภาพผลลัพธ์ เนื่องจากทุกพิกเซลในภาพปลายทางจะได้รับค่าความเข้ม

เนื่องจากตำแหน่งพิกเซลที่คำนวณได้หลังการหมุนมักไม่เป็นจำนวนเต็ม จึงใช้วิธี Bilinear Interpolation เปลี่ยนค่าของจำนวนจริงให้เป็นจำนวนเต็ม เพื่อประมาณค่าความเข้มของพิกเซล โดยอาศัยค่าความเข้มของพิกเซลรอบข้างจำนวนสี่จุดและถ่วงน้ำหนักตามระยะในแนวแกน x และ y วิธีนี้ช่วยให้ภาพที่ได้มีความเรียบเนียนและลดการเกิดลักษณะเป็นบล็อก(sawtooth effect) ได้บางจุด

จากผลการทดลองพบว่า ภาพที่ได้หลังการประมวลผลมีการหมุนและกู้คืนได้เฉพาะบริเวณใกล้จุดศูนย์กลางอย่างเท่านั้น ในขณะที่บริเวณรอบนอกของภาพยังคงลักษณะเดิม แสดงให้เห็นว่าสามารถควบคุมพื้นที่และความรุนแรงของการหมุนได้แต่วิธีนี้ไม่สามารถกู้คืนภาพที่บิดได้ทั้งหมด

ภาคผนวก

```
import matplotlib.pyplot as plt
from readpgm import read_pgm_p5_no_lib

# อ่านภาพ PGM
pgm = read_pgm_p5_no_lib('scaled_shapes.pgm')

data_2d = pgm['data']
maxval = pgm['maxval']

# รวม pixel เป็น 1D
pixels = []
for row in data_2d:
    for p in row:
        pixels.append(p)

# ===== สร้าง histogram แบบไม่พึ่ง plt =====
hist = [0] * (maxval + 1)
for p in pixels:
    hist[p] += 1

# ===== นับจำนวน object จาก histogram =====
object_count = 0
object_gray_levels = []

for gray in range(0, 255):   # ไม่รวม background
    if hist[gray] > 3400:
        object_count += 1
        object_gray_levels.append(gray)

print("Number of objects =", object_count)
print("Object gray levels =", object_gray_levels)

# ===== plot histogram =====
plt.figure()
plt.bar(range(maxval + 1), hist, width=1.0)
plt.title('Histogram of PGM Image')
plt.xlabel('Gray Level')
```

```
plt.ylabel('Number of Pixels')
plt.show()

#question 1.2
height = len(data_2d)
width = len(data_2d[0])

def compute_phi1(image, width, height, gray):
    # ---- raw moments ----
    m00 = 0
    m10 = 0
    m01 = 0

    for y in range(height):
        for x in range(width):
            if image[y][x] == gray:
                m00 += 1
                m10 += x
                m01 += y

    # center of mass
    x_hat = m10 / m00
    y_hat = m01 / m00

    # ---- central moments ----
    mu20 = 0.0
    mu02 = 0.0

    for y in range(height):
        for x in range(width):
            if image[y][x] == gray:
                mu20 += (x - x_hat) ** 2
                mu02 += (y - y_hat) ** 2

    # ---- normalized moments ----
    eta20 = mu20 / (m00 ** 2)
    eta02 = mu02 / (m00 ** 2)
```

```
    # invariant
    phi1 = eta20 + eta02
    return phi1

print("\nQuestion 1.2 Results:")
print("Gray Level   phi1")

phi1_values = []

for gray in object_gray_levels:
    phi1 = compute_phi1(data_2d, width, height, gray)
    phi1_values.append(phi1)
    print(f"{gray:5d}      {phi1:.6f}")
```

ข้อ2

```
#use Cameraman image
import cv2
import matplotlib.pyplot as plt

# อ่านภาพ PGM
img = cv2.imread("Cameraman.pgm", cv2.IMREAD_GRAYSCALE)
img1 = cv2.imread("SEM256_256.pgm", cv2.IMREAD_GRAYSCALE)
# img = cv2.imread("SEM256_256.pgm", cv2.IMREAD_GRAYSCALE)

# Histogram Equalization
eq = cv2.equalizeHist(img)

# แสดงภาพ + histogram
plt.figure(figsize=(12,6))

plt.subplot(2,2,1)
plt.imshow(img, cmap='gray')
plt.title("Original Image")
plt.axis("off")
```

```
plt.subplot(2,2,2)
plt.imshow(eq, cmap='gray')
plt.title("Histogram Equalized Image")
plt.axis("off")

plt.subplot(2,2,3)
plt.hist(img.flatten(), bins=256, range=(0,255))
plt.title("Original Histogram")

plt.subplot(2,2,4)
plt.hist(eq.flatten(), bins=256, range=(0,255))
plt.title("Equalized Histogram")

###

# plt.subplot(2,2,1)
# plt.imshow(img1, cmap='gray')
# plt.title("Original Image")
# plt.axis("off")

# plt.subplot(2,2,2)
# plt.imshow(eq, cmap='gray')
# plt.title("Histogram Equalized Image")
# plt.axis("off")

# plt.subplot(2,2,3)
# plt.hist(img1.flatten(), bins=256, range=(0,255))
# plt.title("Original Histogram")

# plt.subplot(2,2,4)
# plt.hist(eq.flatten(), bins=256, range=(0,255))
# plt.title("Equalized Histogram")

plt.tight_layout()
plt.show()
```

ข้อที่2

```
#use SEM image
import cv2
import matplotlib.pyplot as plt

# อ่านภาพ PGM
img = cv2.imread("SEM256_256.pgm", cv2.IMREAD_GRAYSCALE)

# Histogram Equalization
eq = cv2.equalizeHist(img)

# แสดงภาพ + histogram
plt.figure(figsize=(12,6))

plt.subplot(2,2,1)
plt.imshow(img, cmap='gray')
plt.title("Original Image")
plt.axis("off")

plt.subplot(2,2,2)
plt.imshow(eq, cmap='gray')
plt.title("Histogram Equalized Image")
plt.axis("off")

plt.subplot(2,2,3)
plt.hist(img.flatten(), bins=256, range=(0,255))
plt.title("Original Histogram")

plt.subplot(2,2,4)
plt.hist(eq.flatten(), bins=256, range=(0,255))
plt.title("Equalized Histogram")


plt.tight_layout()
plt.show()
```

ข้อที่3

```
import matplotlib.pyplot as plt
import numpy as np

# ---------- PGM P5 reader (no external library) ----------
def read_pgm_p5(filename):
    with open(filename, 'rb') as f:
        # Read header
        assert f.readline().strip() == b'P5'
        line = f.readline()
        while line.startswith(b'#'):
            line = f.readline()

        width, height = map(int, line.split())
        maxval = int(f.readline())

        # Read image data
        img = np.frombuffer(f.read(), dtype=np.uint8)
        img = img.reshape((height, width))

    return img, maxval

# ---------- Normalize for display ----------
def normalize(img):
    img = img.astype(float)
    min_val = img.min()
    max_val = img.max()
    if max_val - min_val == 0:
        return np.zeros_like(img)
    return (img - min_val) / (max_val - min_val)

# ---------- Read RGB component images ----------
r, _ = read_pgm_p5("SanFranPeak_red.pgm")
g, _ = read_pgm_p5("SanFranPeak_green.pgm")
b, _ = read_pgm_p5("SanFranPeak_blue.pgm")
```

```
# ---------- Algebraic combinations ----------
gray = (r + g + b) / 3
excess_green = 2*g - r - b
red_blue_diff = r - b
veg_balance = g - (r + b) / 2
sky_emphasis = b - r

# ---------- Normalize results ----------
gray_n = normalize(gray)
eg_n = normalize(excess_green)
rb_n = normalize(red_blue_diff)
vb_n = normalize(veg_balance)
sky_n = normalize(sky_emphasis)

# ---------- Plot results ----------
plt.figure(figsize=(15, 10))

plt.subplot(2, 3, 1)
plt.imshow(gray_n, cmap='gray')
plt.title("(r + g + b) / 3 (Grayscale)")
plt.axis("off")

plt.subplot(2, 3, 2)
plt.imshow(eg_n, cmap='gray')
plt.title("2g - r - b (Excess Green)")
plt.axis("off")

plt.subplot(2, 3, 3)
plt.imshow(rb_n, cmap='gray')
plt.title("r - b (Red - Blue)")
plt.axis("off")

plt.subplot(2, 3, 4)
plt.imshow(vb_n, cmap='gray')
plt.title("g - (r + b) / 2")
plt.axis("off")

plt.subplot(2, 3, 5)
```

```
plt.imshow(sky_n, cmap='gray')
plt.title("b - r (Sky Emphasis)")
plt.axis("off")

plt.tight_layout()
plt.show()
```

ข้อที่4

```
import numpy as np
import matplotlib.pyplot as plt
import os

# ------------------------------
# Read PGM
# ------------------------------
def read_pgm_p5(filename):
    if not os.path.exists(filename):
        img = np.zeros((300, 400), dtype=np.uint8)
        for y in range(300):
            for x in range(400):
                if ((y // 40) + (x // 40)) % 2 == 0:
                    img[y, x] = 255
        return img

    with open(filename, 'rb') as f:
        assert f.readline().strip() == b'P5'
        line = f.readline()
        while line.startswith(b'#'):
            line = f.readline()
```

```
        w, h = map(int, line.split())
        f.readline()
        img = np.frombuffer(f.read(), dtype=np.uint8).reshape(h, w)
    return img

# --------------------------------
# Bilinear interpolation
# --------------------------------
def bilinear(img, x, y):
    h, w = img.shape
    if x < 0 or x >= w - 1 or y < 0 or y >= h - 1:
        return 0

    x0 = int(np.floor(x))
    y0 = int(np.floor(y))
    dx = x - x0
    dy = y - y0

    return (
        (1 - dx) * (1 - dy) * img[y0, x0] +
        dx * (1 - dy) * img[y0, x0 + 1] +
        (1 - dx) * dy * img[y0 + 1, x0] +
        dx * dy * img[y0 + 1, x0 + 1]
    )

# --------------------------------
# Local rotation (center only)
# --------------------------------
def local_rotate_ccw(src, max_angle_deg=30, radius=120):
    h, w = src.shape
    dst = np.zeros_like(src)

    cx = w / 2
    cy = h / 2

    for y in range(h):
        for x in range(w):
            dx = x - cx
```

```
            dy = y - cy
            r = np.sqrt(dx*dx + dy*dy)

            if r > radius:
                dst[y, x] = src[y, x]
                continue

            # weight decreases with distance
            wgt = 1 - r / radius
            theta = np.deg2rad(max_angle_deg * wgt)

            cos_t = np.cos(theta)
            sin_t = np.sin(theta)

            # backward mapping (CCW)
            src_x = cos_t * dx + sin_t * dy + cx
            src_y = -sin_t * dx + cos_t * dy + cy

            dst[y, x] = bilinear(src, src_x, src_y)

    return dst

# ------------------------------
# Main
# ------------------------------
img = read_pgm_p5("disOperaHouse.pgm")

out = local_rotate_ccw(
    img,
    max_angle_deg=-89,  # มุมสูงสุดตรงกลาง
    radius= 120         # รัศมีที่มีผล
)

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(img, cmap='gray')
plt.title("Original")
plt.axis("off")
```

```
plt.subplot(1, 2, 2)
plt.imshow(out, cmap='gray')
plt.title("Local Rotation (Center Only, CCW)")
plt.axis("off")

plt.tight_layout()
plt.show()
```