Table

Description automatically generated

Answer

ทำการ Normalization ก่อนด้วย 255.00 เนื่องจากข้อมูลเป็น 8 bit / pixel จะได้

A picture containing text, shoji, building

Description automatically generated

หลังจากทำการคูณด้วย 63 เนื่องจากต้องการลดระดับของ Pixel จาก 8 bit / pixel จะได้ Array ที่ระดับ 6 bit / pixel ดังนี้

A picture containing shoji, building

Description automatically generated

Text

Description automatically generated

A picture containing text, laying

Description automatically generated

Answer - Linear Function จะใช้เป็น Pl = 1 \* Pi + 5; โดยที่ Pi คือ Pixel ณ ตำแหน่ง i โดยที่ จะ

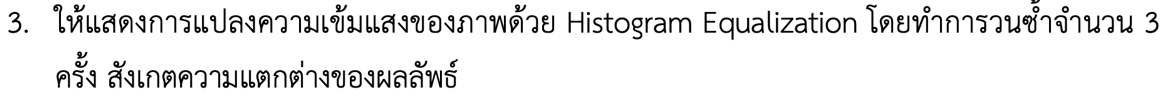
เป็นรูปต้นฉบับ และ Pl จะเป็น Pixel ใหม่ที่ได้จาก Linear Function

* Power law Function จะใช้เป็น Pp = 1 \* Pi \*\* 1.4; โดยที่ Pi คือ Pixel ณ

ตำแหน่ง โดยที่ จะเป็นรูปต้นฉบับ และ Pp จะเป็น Pixel ใหม่ที่ได้จาก Power law Function

โดยการใช้ Linear Function จะทำให้รูปภาพสว่างมากขึ้นเนื่องจากเกิดการ shift ของค่า pixel ในแต่ละ pixel ซึ่งโซนสีดำเกิดจาก ค่าที่เกินระดับ 255 ตัว OpenCV จะทำการปัดเป็น 0 ทันที และ Power law Function จะเป็นการทำให้เห็นรายละเอียด High level component มากขึ้นแต่ก็จะทำให้ Low level component มีรายละเอียดน้อยลงด้วย โดยจะเห็นได้ว่ารูปที่ทำการ Power law Function จะทำให้เห็น รายละเอียด ของ กำแพง มากขึ้นและ สุดท้าย Histogram Equalization จะทำให้รูปภาพมี Contrast ที่มากขึ้นทำให้เห็น รายละเอียดของรูปภาพมากขึ้นเช่นกัน

โดยสรุปแล้ว Linear Function จะทำให้เกิดการ Shift ของ pixel ขึ้นอยู่กับสมการที่เราใช้ โดยจะทำให้เกิดความสว่างขึ้นเหมาะ กับรูปที่มีความมืดค่อนข้างเยอะ ส่วน Power law Function จะเป็นการ Stretch low level/ high level และ Compress low level/ high level ขึ้นอยู่กับค่า Gamma ที่เราใช้ โดยจากคำตอบจะใช้เป็น Gamma = 1.4 ซึ่งเป็นการ Compress low level และ Stretch high level ซึ่งถ้าเราต้องการเป็น Detail ของภาพมากขึ้นเราสามารถใช้ เทคนิคนี้ในการปรับปรุงรูปภาพได้ขึ้นอยู่กับว่า เราต้องการ Compress หรือ Stretch Component นี้ แต่ข้อเสีย มันไม่สามารถเลือก Component เดียวถ้าเราทำการ Stretch Component หนึ่ง อีก Component ก็จะโดน Compress เป็นต้น และ สุดท้าย Histogram Equalization จะเป็นการปรับ Contrast ของรูปภาพ โดยเทคนิคจะทำให้ Histogram ของรูปอยู่ ใน Uniform มากที่สุด จะช่วยให้ Contrast มีค่ามากขึ้นและจะเป็นรายละเอียดมากขึ้น แต่ข้อเสียคือ รูปภาพที่ ในส่วนที่เป็น High level component จะโดน ขยายออกมากขึ้นจะเห็นได้ว่าจากรูปส่วนที่เป็นแสงอาทิตย์จะสว่างมาก



A picture containing text

Description automatically generated

Answer โดยเมื่อทำการ Histogram Equalization รอบแรกจะทำให้ Histogram ของรูปภาพมีความ Uni-from มากขึ้น หรือ กระจายตัวเท่ากันมากขึ้น แต่เมื่อทำการ Histogram Equalization ในรอบถัดไปเรื่อยๆจะไม่เห็นถึงความแตกต่างมากเนื่องจาก การทำ Histogram Equalization เป็นการทำให้ค่าเกิดการกระจายตัวเท่า ๆ กัน เมื่อทำการ Histogram Equalization รอบถัดไปก็จะได้ค่าที่มีความใกล้เคียงกับค่าเดิม เพราะ เราทำให้ค่าเกิดการกระจายตัวเท่าๆ กันมากที่สุดแล้ว พอทำ Histogram Equalization ก็ทำให้ค่าเปลี่ยนแปลงเพียงเล็กหน่อย จนทำให้ไม่เป็นเห็นถึงความแตกต่างเลย

Table

Description automatically generated

Answer

ทำการ Histogram Equalization ตารางที่ 4.1 ก่อน จะได้

Table, calendar

Description automatically generated

ต่อมาทำการ Histogram Equalization ตารางที่ 4.2 จะได้

Table, calendar

Description automatically generated

นำผล Histogram Equalization ของ ตาราง 4.1 กับ 4.1 มา Mapping จะได้

Chart, calendar

Description automatically generated

หลังจาการทำ Mapping แล้วนำค่า Pixel level ใหม่ไปเทียบกับจำนวน Pixel กับ ตารางที่ 4.1 จะได้ดังนี้

Calendar

Description automatically generated

เมือเทียบ Histogram ใหม่ที่จะเป็นดังนี้ (ซ้ายจะเป็นต้นฉบับ ส่วนขวาจะเป็นผลลัพธ์จากทำ Histogram Matching )

Chart

Description automatically generated

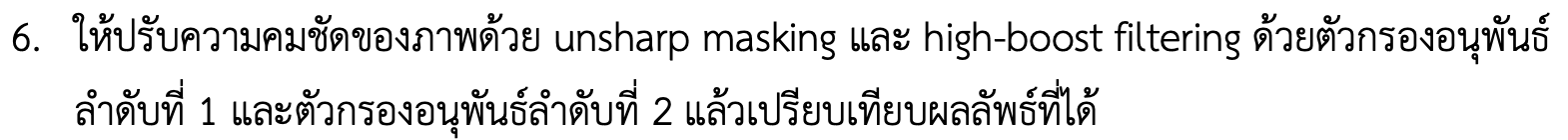
Text

Description automatically generated

Answer



จะเห็นได้ว่า การที่เราใช้ Mean filter ขนาด 5x5 จะได้ผลลัพธ์ที่รูปภาพจะมีความ Blur มากกว่าใช้ Mean filter ขนาด 3x3 2 รอบ เนื่องมากจาก Mean filter ที่มีขนาดใหญ่ขึ้น ก็จะเป็นการเอาจำนวน Pixel ณ ตำแหน่ง I มาเฉลี่ยกันมากขึ้น ก็จะทำให้ภาพมีความ Smooth หรือ Blur มากขึ้น แต่ถ้าใช้ Mean filter ขนาด 3x3 มากกว่า 2 รอบก็อาจให้ผลลัพธ์ที่ Blur มากกว่าการใช้ Mean filter ขนาด 5x5 เพียง 1 รอบ เพราะ มันก็คือการเอาค่าเฉลี่ยมาเฉลี่ยซ้ำเรื่อย ๆ แต่ยังไงก็ตาม ถ้าทำการเพิ่มครั้งในการ Mean filter ไปกับ Size mean filter เทียบกันก็ไม่สามารถทำให้ได้ผลลัพธ์ Blur ที่มากกว่าของการเพิ่มจำนวน Size ของ Mean filter ได้ จากการทดลอง



Answer

จากสมการของ อนุพันธ์ลำดับที่ 1 ถ้าเราทำการ หาทั้ง Forward และ Backward เพื่อหา Gradient ทั้งสองฝั่ง โดย สมการที่ (1) คือ Forward และ สมการที่ (2) คือ Backward เมื่อนำสมการมารวมกันจะได้ ดังรูปภาพข้างล่าง โดยจะแสดง Kernel ที่ได้ด้วย [ซ้ายคือ X – axis และ ขวา คือ Y - axis]

Chart, scatter chart

Description automatically generated

โดยเมื่อทำการเอา 1D Gaussian Filter มา Convolution กับ X – derivative กับ จะได้ Kernel ใหม่ และ ทำการเอา 1D Gaussian Filter มา Convolution กับ Y – derivative กับ จะได้ Kernel ใหม่ โดยต้องทำการปรับเลขให้จำนวนเต็มให้หมดก่อนเนื่องจากจำนวน Pixel ต้องเป็นจำนวนเต็ม ดังภาพข้างล่าง [ซ้ายคือ X – axis และ ขวา คือ Y - axis]

Diagram

Description automatically generated with medium confidence

ชึ่งผลลัพธ์ที่เราได้คือ Sobel filter ที่จะเป็นการเอาข้อมูลของรูปภาพในส่วน High-level component ออกมา

A picture containing text

Description automatically generated

โดยทำการนำรูปภาพที่ต้องการมา Convolution ด้วย Sobel Filter ทั้ง X และ Y axis เพื่อดึงข้อมูล Hight-level component ของทั้ง X และ Y axis จะได้ดังรูปภาพข้างบนข้างต้นในหัวข้อ “First derivative”

ต่อมาโดยทำการ Unsharp Marking โดยให้

Fhp = เป็น High level component หลังจากทำ Sobel ทั้ง x และ y

Imo = รูปภาพต้นฉบับ

โดยเมื่อทำการใช้สมการนี้

Imum= Imo  + Fhp

จะได้ผลลัพธ์ดังภาพในหัวข้อ “Unsharp Masking” และ ทำการทำ Highboost Filtering โดยให้สมการดังนี้

Imum= Imo  + 2 \* Fhp

จะได้ผลลัพธ์ดังภาพในหัวข้อ “Highboost Filtering”

โดยจากสมการ อนุพันธ์ลำดับที่ 2 ทำการหาทั้ง X และ Y Axis แล้วนำสมการมารวมกัน แล้วนำมาเขียนเป็นรูป Kernel จะได้ดังรูปภาพข้างล่าง

A picture containing table

Description automatically generated

ชึ่งผลลัพธ์ที่เราได้คือ Laplacian filter ที่จะเป็นการเอาข้อมูลของรูปภาพในส่วน High-level component ออกมา

A picture containing text

Description automatically generated

โดยทำการนำรูปภาพที่ต้องการมา Convolution ด้วย Laplacian Filter เพื่อดึงข้อมูล Hight-level component ของรูป จะได้ดังรูปภาพข้างบน ในหัวข้อ “Second derivative”

ต่อมาโดยทำการ Unsharp Marking โดยให้

Fhp = เป็น High level component หลังจากทำ Laplacian ทั้ง x และ y

Imo = รูปภาพต้นฉบับ

โดยเมื่อทำการใช้สมการนี้

Imum= Imo  + Fhp

จะได้ผลลัพธ์ดังภาพในหัวข้อ “Unsharp Masking” และ ทำการทำ Highboost Filtering โดยให้สมการดังนี้

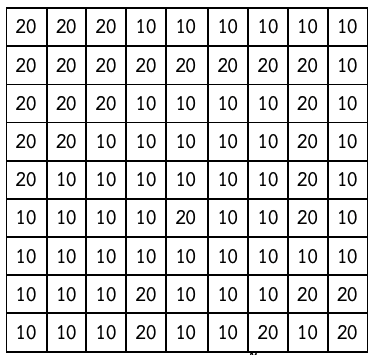
Imum= Imo  + 2 \* Fhp

จะได้ผลลัพธ์ดังภาพในหัวข้อ “Highboost Filtering”

โดยสรุปแล้วเราจะเห็นได้ว่าการใช้ อนุพันธ์ลำดับที่ 1 ในการทำ Unsharp Masking จะได้ผลลัพธ์ดีกว่า อนุพันธ์ลำดับที่ 2 โดยจะให้ Detail ที่ชัดเจนมากว่า แต่ในส่วน Highboost Filtering อนุพันธ์ลำดับที่ 2 จะให้ Detail ที่ดีกว่า อนุพันธ์ลำดับที่ 1

A picture containing text, clock

Description automatically generated



ทำการนำ Array ต้นฉบับ มา Convolution กับ Filter ที่ 1 จะได้ดังรูปภาพข้างล่าง โดยทำการตัด Pixel ที่น้อยว่า 0 ให้เป็น 0 แล้วใช้ Adjacent เพื่อหา Region

Table

Description automatically generated with low confidence

ทำการนำ Array ต้นฉบับ มา Convolution กับ Filter ที่ 2 จะได้ดังรูปภาพข้างล่าง โดยทำการตัด Pixel ที่น้อยว่า 0 ให้เป็น 0 แล้วใช้ Adjacent เพื่อหา Region

A picture containing electronics

Description automatically generated

ทำการนำ Array ต้นฉบับ มา Convolution กับ Filter ที่ 3 จะได้ดังรูปภาพข้างล่าง โดยทำการตัด Pixel ที่น้อยว่า 0 ให้เป็น 0 แล้วใช้ Adjacent เพื่อหา Region

A picture containing text, crossword puzzle

Description automatically generated

Chart

Description automatically generated with medium confidence

ทำการนำ Array ต้นฉบับ มา Convolution กับ Filter ที่ 4 จะได้ดังรูปภาพข้างล่าง โดยทำการตัด Pixel ที่น้อยว่า 0 ให้เป็น 0 แล้วใช้ Adjacent เพื่อหา Region

A picture containing text, electronics, tiled

Description automatically generated

Chart, table

Description automatically generated

จะเห็นได้ว่า Filter ที่ 3 กับ 4 จะให้ Region สีเหลือง ที่ใหญ่กว่า Filter 2 อันแรก ซึ่งแสดงว่าน่าจะเป็นขอบของรูปทั้งหมด โดยถ้านำ Filter ที่ 4 มา Masking กับ Region สีเหลือง กับรูปภาพต้นฉบับ จะได้ดังรูปภาพข้างล่าง

A picture containing calendar

Description automatically generated

Qr code

Description automatically generated with low confidence

Qr code

Description automatically generated

A picture containing background pattern

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

Text

Description automatically generated

A picture containing clothing, person, green, colorful

Description automatically generated