

Image Processing Mid Term 2025

Shaharear Emon C213079

Find the optimal threshold of the following image using Otsu method.

| | | | |
|---|---|---|---|
| 0 | 1 | 4 | 0 |
| 0 | 2 | 1 | 2 |
| 2 | 1 | 4 | 4 |
| 0 | 2 | 3 | 1 |

Otsu's method একটি স্বায়ংক্রিয় ধ্রেশহোল্ডিং টেকনিক, যা ইমেজকে ফুইটি প্রেণিতে (foreground এবং background) বিভক্ত করার জন্য ব্যবহৃত হয়। এটি ক্লাসের মধ্যে ভ্যারিয়েন্স সর্বাধিক (maximum between-class variance) হয় এমন ধ্রেশহোল্ড নির্বাচন করে।

দেওয়া ইমেজ:

$$\begin{bmatrix} 0 & 1 & 4 & 0 \\ 0 & 2 & 1 & 2 \\ 2 & 1 & 4 & 4 \\ 0 & 2 & 3 & 1 \end{bmatrix}$$

ধাপ ২: পিক্সেল মানগুলোর ফ্রিকোয়েলি (Histogram) বের করা

প্রতিটি পিক্সেল মান (0-4) এর কতবার উপস্থিত হয়েছে তা বের করতে হবে।

| পিক্সেল মান (i) | ফ্রিকোয়েলি ($f(i)$) |
|---------------------|------------------------|
| 0 | 4 |
| 1 | 4 |
| 2 | 4 |
| 3 | 1 |
| 4 | 3 |

মোট পিক্সেলের সংখ্যা:

$$N = 4 + 4 + 4 + 1 + 3 = 16$$

ধাপ ৩: সম্ভাব্যতা (Probability) নির্ণয়

প্রত্যেক পিক্সেল মানের সম্ভাব্যতা নির্ণয় করতে হবে:

$$P(i) = \frac{f(i)}{N}$$

| পিক্সেল মান (i) | ফ্রিকোয়েন্সি ($f(i)$) | Probability ($P(i)$) |
|---------------------|--------------------------|-------------------------|
| 0 | 4 | $\frac{4}{16} = 0.25$ |
| 1 | 4 | $\frac{4}{16} = 0.25$ |
| 2 | 4 | $\frac{4}{16} = 0.25$ |
| 3 | 1 | $\frac{1}{16} = 0.0625$ |
| 4 | 3 | $\frac{3}{16} = 0.1875$ |

ধাপ ৪: Mean (μ) বের করা

Mean বা গড় মান বের করতে হবে:

$$\mu_T = \sum(i \times P(i))$$

$$\begin{aligned}\mu_T &= (0 \times 0.25) + (1 \times 0.25) + (2 \times 0.25) + (3 \times 0.0625) + (4 \times 0.1875) \\ &= 0 + 0.25 + 0.5 + 0.1875 + 0.75 = 1.6875\end{aligned}$$

ধাপ ৫: প্রতিটি সম্ভাব্য খ্রেশহোল্ডের জন্য ওজন ও Mean বের করা

আমরা প্রতিটি সম্ভাব্য খ্রেশহোল্ড (T) এর জন্য **Background (ω_1, μ_1) এবং Foreground (ω_2, μ_2) বের করব।

$$T = 0$$

- Background (ω_1): $P(0) = 0.25$
- Foreground (ω_2): $1 - \omega_1 = 1 - 0.25 = 0.75$
- Background Mean (μ_1):

$$\mu_1 = \frac{0 \times 0.25}{0.25} = 0$$

- Foreground Mean (μ_2):

$$\mu_2 = \frac{(1 \times 0.25) + (2 \times 0.25) + (3 \times 0.0625) + (4 \times 0.1875)}{0.75} = 2.25$$

- Between-class Variance:

$$\sigma_B^2 = \omega_1 \omega_2 (\mu_1 - \mu_2)^2 = (0.25)(0.75)(0 - 2.25)^2 = 0.25 \times 0.75 \times 5.0625 = 0.9492$$

$T = 1$

- **Background (ω_1):** $P(0) + P(1) = 0.25 + 0.25 = 0.5$
- **Foreground (ω_2):** $1 - 0.5 = 0.5$
- **Background Mean (μ_1):**

$$\mu_1 = \frac{(0 \times 0.25) + (1 \times 0.25)}{0.5} = 0.5$$

- **Foreground Mean (μ_2):**

$$\mu_2 = \frac{(2 \times 0.25) + (3 \times 0.0625) + (4 \times 0.1875)}{0.5} = 2.75$$

- **Between-class Variance:**

$$\sigma_B^2 = (0.5)(0.5)(0.5 - 2.75)^2 = 0.5 \times 0.5 \times 5.0625 = 1.2656$$

$T = 2$

- **Background (ω_1):** $P(0) + P(1) + P(2) = 0.75$
- **Foreground (ω_2):** $1 - 0.75 = 0.25$
- **Background Mean (μ_1):**

$$\mu_1 = \frac{(0 \times 0.25) + (1 \times 0.25) + (2 \times 0.25)}{0.75} = 1$$

- **Foreground Mean (μ_2):**

$$\mu_2 = \frac{(3 \times 0.0625) + (4 \times 0.1875)}{0.25} = 3.6$$

- **Between-class Variance:**

$$\sigma_B^2 = (0.75)(0.25)(1 - 3.6)^2 = 0.75 \times 0.25 \times 6.76 = 1.2656$$

$T = 3$

ধাপ ১: Background এবং Foreground ওজন (ω_1, ω_2) নির্ণয়

$T = 3$ অর্থাৎ pixel values ≤ 3 হবে Background এবং > 3 হবে Foreground।

Background (ω_1)

Background-এর সম্ভাবনা (probability):

$$\omega_1 = P(0) + P(1) + P(2) + P(3)$$

$$= 0.25 + 0.25 + 0.25 + 0.0625 = 0.8125$$

Foreground (ω_2)

$$\omega_2 = 1 - \omega_1 = 1 - 0.8125 = 0.1875$$

ধাপ ২: Background এবং Foreground এর Mean (μ_1, μ_2) নির্ণয়

Background Mean (μ_1)

$$\begin{aligned}\mu_1 &= \frac{(0 \times 0.25) + (1 \times 0.25) + (2 \times 0.25) + (3 \times 0.0625)}{0.8125} \\ &= \frac{0 + 0.25 + 0.5 + 0.1875}{0.8125} = \frac{0.9375}{0.8125} = 1.154\end{aligned}$$

Foreground Mean (μ_2)

$$\mu_2 = \frac{(4 \times 0.1875)}{0.1875} = 4$$

ধাপ ৩: Between-Class Variance (σ_B^2) বের করা

$$\begin{aligned}\sigma_B^2 &= \omega_1 \omega_2 (\mu_1 - \mu_2)^2 \\ &= (0.8125)(0.1875)(1.154 - 4)^2 \\ &= (0.8125)(0.1875)(8.085) \\ &= 1.231\end{aligned}$$

$T = 3$ এর জন্য Between-Class Variance = 1.231 ✓

যেহেতু $T = 2$ এর জন্য Between-Class Variance = 1.2656 ছিল, যা বেশি, তাই $T = 2$ -ই অপ্টিমাল থ্রেশহোল্ড হবে। ⚡

Formula for Otsu Method

$$w_f = \frac{\text{freq. No. of pixel}_1 + \text{freq.}_2 + \dots + \text{freq.}_n}{\text{Total No. of pixel}}$$

$$\mu_f = \frac{(\text{gray level}_1 \times \text{No. of pixel}_1) + \dots + (\text{gray level}_n \times \text{No. of pixel}_n)}{\text{Total No. of pixel}}$$

$$\sigma_f^2 = \frac{\{(\text{gray level}_1 - \mu_f)^2 \times \text{freq. / pixel}_1\} + \dots + \{(\text{gray level}_n - \mu_f)^2 \times \text{freq. / pixel}_n\}}{\text{Total No. of pixel}}$$

$$\sigma_w^2 = w_b \sigma_b^2 + w_f \sigma_f^2$$

$$w_b/w_f = \frac{\text{No. of pixel}_1}{\text{Total No. pixel}}$$

optimal/minimum class variance will be the threshold value.

Low Pass Filter

① Given a 4×4 original image with 3 bits/pixel.

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 3 | 3 | 0 | 0 |
| 0 | 0 | 6 | 4 | 2 | 0 |
| 0 | 2 | 6 | 3 | 4 | 0 |
| 0 | 1 | 3 | 3 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |

[Used zero padding].

Performing low pass filtering:

Low pass filtering formula, $R = \frac{1}{9} \sum_{i=1}^9 z_i$.

Kernel Sliding & Calculation:

◆ উদাহরণ: প্রথম পিক্সেল (উপরে বাঁয়ে) গাণিতিক হিসাব:

পিক্সেল উইন্ডো:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & 6 \end{bmatrix}$$

গড় বের করা:

$$\frac{0 + 0 + 0 + 0 + 1 + 3 + 0 + 0 + 6}{9} = \frac{10}{9} \approx 1$$

◆ দ্বিতীয় পিক্রেল:

পিক্রেল উইন্ডো:

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 3 & 3 \\ 0 & 6 & 4 \end{bmatrix}$$

গড় বের করা:

$$\frac{0 + 0 + 0 + 1 + 3 + 3 + 0 + 6 + 4}{9} = \frac{17}{9} \approx 2$$

Operations!

$$\textcircled{1} (3+6+1)/9 = 10/9 = 1.11.$$

$$\textcircled{2} (1+3+3+6+4)/9 = 17/9 = 1.89$$

$$\textcircled{4} (3+3+6+4+2)/9 = 2.$$

$$\textcircled{5} (3+4+2)/9 = 1$$

$$\textcircled{3} (3+2+6+6)/9 = 1.89$$

$$\textcircled{6} (1+3+3+6+4+2+6+5) \frac{1}{9} = 3.33$$

$$\textcircled{7} (3+3+6+4+2+6+5+4) \frac{1}{9} = 3.67$$

$$\textcircled{8} (3+4+2+5+4) \frac{1}{9} = 2$$

$$\textcircled{9} (6+2+6+1+3) \frac{1}{9} = 2$$

$$\textcircled{10} (6+4+2+6+5+1+3+2) \frac{1}{9} = 3.33$$

$$\textcircled{11} (6+4+2+6+5+4+3+3) \frac{1}{9} = 3.67$$

$$\textcircled{12} (9+2+5+4) \frac{1}{9} = 1.67$$

$$\textcircled{13} (2+6+1+3) \frac{1}{9} = 1.33$$

$$\textcircled{14} (2+6+5+1+3+3) \frac{1}{9} = 2.22$$

$$\textcircled{15} (6+5+4+3+3) \frac{1}{9} = 2.33$$

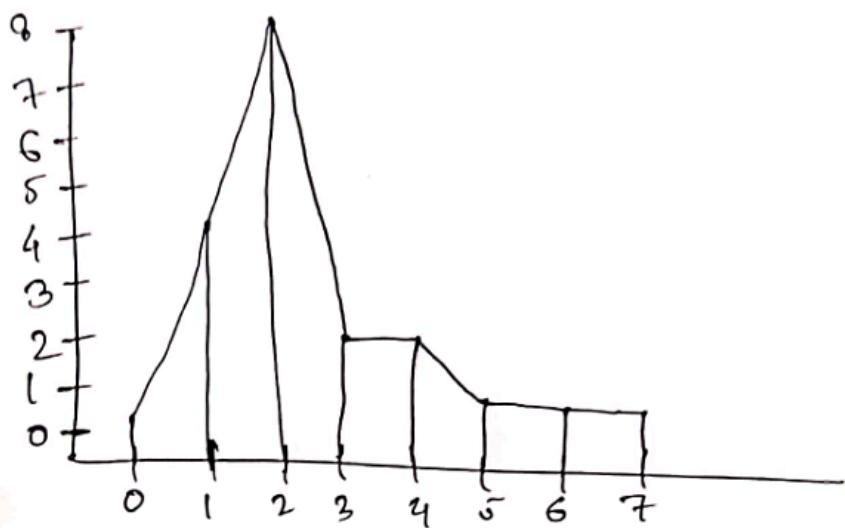
$$\textcircled{16} (5+4+3) \frac{1}{9} = 1.33$$

Updated image after low-pass filtering:

| | | | |
|------|------|------|------|
| 1.41 | 1.89 | 2 | 1 |
| 1.89 | 3.33 | 3.67 | 2. |
| 2 | 3.33 | 3.67 | 1.67 |
| 1.33 | 2.22 | 2.33 | 1.33 |

⑪ Analyzing the differences of both image:

plotting histogram for low pass filtered image.



Median Filtering

| | | | |
|---|---|---|---|
| 1 | 3 | 3 | 0 |
| 0 | 6 | 4 | 2 |
| 2 | 6 | 5 | 4 |
| 1 | 3 | 3 | 0 |

Step 1: Image Padding

যেহেতু 3×3 ফিল্টার প্রয়োগ করতে হবে, তাই আমরা Zero Padding ব্যবহার করবো। Original 4×4 Image কে Padding করে 6×6 বানাই—

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 3 & 3 & 0 & 0 \\ 0 & 0 & 6 & 4 & 2 & 0 \\ 0 & 2 & 6 & 5 & 4 & 0 \\ 0 & 1 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: Kernel Sliding & Calculation

প্রতিটি 3×3 উইন্ডো-এর উপর মিডিয়ান বের করবো।

প্রথম পিণ্ডেল (উপরে বাঁয়ে) গাণিতিক হিসাব:

◆ 3×3 Window:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & 6 \end{bmatrix}$$

◆ Sort করলে:

0, 0, 0, 0, 0, 1, 3, 6

◆ Median = 0 (মাঝের মান)

দ্বিতীয় পিণ্ডেল:

◆ 3×3 Window:

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 3 & 3 \\ 0 & 6 & 4 \end{bmatrix}$$

◆ Sort:

0, 0, 0, 1, 3, 3, 4, 6

◆ Median = 1

Performing median filtering:

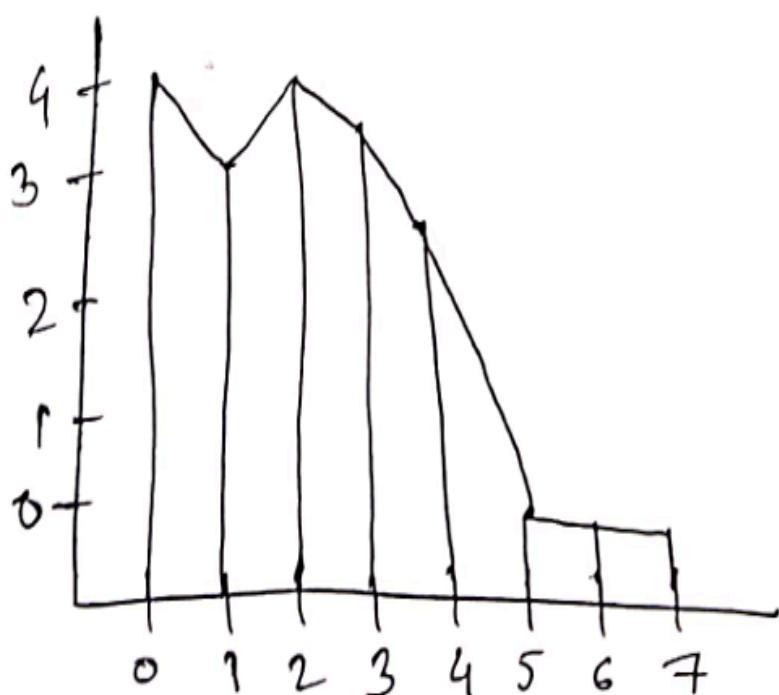
Operation:

- | | |
|------------------------------------|-------------------------------------|
| ① 0 0 0 0 0 1 3 6 ↓ median | ⑩ 0 1 2 3 3 4 5 6 6 ↓ median. |
| ② 0 0 0 0 1 3 3 4 6 ↓ median | ⑪ 0 2 3 3 4 4 5 6 6 ↓ median |
| ③ 0 0 0 0 2 3 3 4 6 ↓ median | ⑫ 0 0 0 0 2 3 4 4 5 ↓ median |
| ④ 0 0 0 0 0 0 2 3 4 ↓ median | ⑬ 0 0 0 0 0 1 2 3 6 ↓ median |
| ⑤ 0 0 0 0 1 2 3 6 6 ↓ median | ⑭ 0 0 0 1 2 3 3 5 6 ↓ median |
| ⑥ 0 1 2 3 3 4 5 6 6 ↓ median | ⑮ 0 0 0 0 3 3 4 5 6 ↓ median. |
| ⑦ 0 2 3 3 4 4 5 6 6 ↓ median | ⑯ 0 0 0 0 0 3 4 5 - ↓ median. |
| ⑧ 0 0 0 0 2 3 4 4 5 ↓ median | |
| ⑨ 0 0 0 0 1 2 3 6 6 ↓ median | |

Updated image after median filtering:

| | | | |
|---|---|---|---|
| 0 | 1 | 2 | 0 |
| 1 | 3 | 4 | 2 |
| 1 | 3 | 4 | 2 |
| 0 | 2 | 3 | 0 |

plotting histogram for median filtered image.



High Pass Filter

A 4×4 original image is given with 3 bits/pixel.

| | | | |
|---|---|---|---|
| 1 | 3 | 3 | 0 |
| 0 | 6 | 4 | 2 |
| 2 | 6 | 5 | 4 |
| 1 | 3 | 3 | 0 |

◆ High-Pass Filtering ইমেজের edge এবং sharp details সংরক্ষণ করে। এটি noise বৃদ্ধি করতে পারে, তবে ইমেজের গুরুত্বপূর্ণ বৈশিষ্ট্য তুলে ধরে।

আমরা 3×3 High-Pass Filter Kernel (Laplacian Kernel) ব্যবহার করবো—

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Step 1: Image Padding

প্রথমে Zero Padding করে ইমেজটিকে 6×6 বানাই—

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 3 & 3 & 0 & 0 \\ 0 & 0 & 6 & 4 & 2 & 0 \\ 0 & 2 & 6 & 5 & 4 & 0 \\ 0 & 1 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: Kernel Sliding & Calculation

প্রত্যেক 3×3 উইndow-তে High-Pass Kernel প্রয়োগ করবো।

প্রথম পিঙ্কেল (উপরে বাঁয়ে) হিসাব:

◆ 3×3 Window:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & 6 \end{bmatrix}$$

◆ Kernel প্রয়োগ:

$$(0 \times -1) + (0 \times -1) + (0 \times -1) + (0 \times -1) + (1 \times 8) + (3 \times -1) + (0 \times -1) + (0 \times -1) + (6 \times -1) \\ = 0 + 0 + 0 + 0 + 8 - 3 + 0 + 0 - 6 = -1$$

দ্বিতীয় পিক্সেল হিসাব:

◆ 3×3 Window:

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 3 & 3 \\ 0 & 6 & 4 \end{bmatrix}$$

◆ Kernel প্রয়োগ:

$$\begin{aligned} (0 \times -1) + (0 \times -1) + (0 \times -1) + (1 \times -1) + (3 \times 8) + (3 \times -1) + (0 \times -1) + (6 \times -1) + (4 \times -1) \\ = 0 + 0 + 0 - 1 + 24 - 3 + 0 - 6 - 4 = 10 \end{aligned}$$

◆ তৃতীয় পিক্সেল (Row: 1, Col: 3) হিসাব:

3×3 Window:

$$\begin{bmatrix} 0 & 0 & 0 \\ 3 & 3 & 0 \\ 6 & 4 & 2 \end{bmatrix}$$

Kernel প্রয়োগ:

$$\begin{aligned} (0 \times -1) + (0 \times -1) + (0 \times -1) + (3 \times -1) + (3 \times 8) + (0 \times -1) + (6 \times -1) + (4 \times -1) + (2 \times -1) \\ = 0 + 0 + 0 - 3 + 24 - 0 - 6 - 4 - 2 = 9 \end{aligned}$$

এইভাবে সম্পূর্ণ হিসাব করলে, **High-Pass Filtered Image** পাওয়া যাবে:

$$\begin{bmatrix} -1 & 10 & 7 & -8 \\ 5 & 18 & 9 & -6 \\ 4 & 11 & 3 & -5 \\ -1 & 5 & 0 & -9 \end{bmatrix}$$

সারাংশ

- High-Pass Filtering ইমেজের edge এবং details সংরক্ষণ করে।
- Kernel প্রয়োগ করে প্রতিটি পিক্সেলের নতুন মান বের করি।
- Negative মান থাকলে, Normalization করতে হয় (0-255 স্কেলে)।

Normalization Process:

Step 3: Normalization (Scaling to 0-255)

যেহেতু কিছু মান নেগেটিভ এসেছে, আমরা Normalization করবো। New Range: 0 থেকে 255

নতুন স্কেল:

$$\text{New Value} = \frac{\text{Pixel} - \text{Min}}{\text{Max} - \text{Min}} \times 255$$

এভাবে সব পিক্সেল 0-255 স্কেলে নিয়ে আসতে পারবো।

Normalization Step-by-Step

আমরা High-Pass Filtering শেষে একটি নতুন ম্যাট্রিক্স পেয়েছি যেখানে কিছু নেগেটিভ ভ্যালু আছে। চূড়ান্ত ইমেজ 0 থেকে 255 স্কেলে আনতে Normalization করতে হবে।

Step 1: Given Filtered Image (Before Normalization)

আমরা আগের High-Pass Filtering এ পেয়েছি—

$$\begin{bmatrix} -1 & 10 & 9 & -6 \\ 5 & 18 & 9 & -6 \\ 4 & 11 & 3 & -5 \\ -1 & 5 & 0 & -9 \end{bmatrix}$$

Step 2: Min-Max Normalization Formula

$$\text{New Value} = \frac{\text{Pixel} - \text{Min}}{\text{Max} - \text{Min}} \times 255$$

◆ Min Value = -9

◆ Max Value = 18

Step 3: Normalization Calculation for Each Pixel

👉 Pixel = -1

$$\frac{(-1 - (-9))}{(18 - (-9))} \times 255 = \frac{8}{27} \times 255 = 75.56 \approx 76$$

👉 Pixel = 10

$$\frac{(10 - (-9))}{(18 - (-9))} \times 255 = \frac{19}{27} \times 255 = 179.44 \approx 179$$

👉 Pixel = 9

$$\frac{(9 - (-9))}{(18 - (-9))} \times 255 = \frac{18}{27} \times 255 = 170$$

👉 Pixel = -6

$$\frac{(-6 - (-9))}{(18 - (-9))} \times 255 = \frac{3}{27} \times 255 = 28.33 \approx 28$$

👉 Pixel = 5

$$\frac{(5 - (-9))}{(18 - (-9))} \times 255 = \frac{14}{27} \times 255 = 132.22 \approx 132$$

👉 Pixel = 18

$$\frac{(18 - (-9))}{(18 - (-9))} \times 255 = \frac{27}{27} \times 255 = 255$$

👉 Pixel = 0

$$\frac{(0 - (-9))}{(18 - (-9))} \times 255 = \frac{9}{27} \times 255 = 85$$

Step 4: Final Normalized Image

$$\begin{bmatrix} 76 & 179 & 170 & 28 \\ 132 & 255 & 170 & 28 \\ 123 & 188 & 113 & 38 \\ 76 & 132 & 85 & 0 \end{bmatrix}$$

Final Summary

- ✓ Normalization করে নেগেটিভ ভ্যালুগুলো সরিয়ে 0-255 স্কেলে আনা হয়েছে।
- ✓ Min-Max Normalization ব্যবহার করা হয়েছে।
- ✓ Edge Enhancement Image প্রস্তুত হয়ে গেল! 🚀

এভাবে High-Pass Filtered Image কে সঠিকভাবে Normalize করা হয়! 💯

Histogram

A 5x5 bits/pixel original image is given by (4 bits/pixel)

| | | | | |
|----|----|----|----|----|
| 15 | 12 | 8 | 9 | 14 |
| 12 | 12 | 12 | 14 | 11 |
| 13 | 13 | 10 | 9 | 10 |
| 15 | 12 | 10 | 12 | 11 |
| 13 | 14 | 13 | 13 | 14 |

2(b):

- i. Apply histogram equalization to the image by rounding the resulting image pixels to integer.
- ii. Sketch the histograms of the original image and the histogram-equalized image.
- iii. Why histogram equalization not produce a perfectly flat histogram?

$$\begin{aligned} \text{max-value} &= 25 \\ 2^4 &= 16 > 25 \\ \therefore L &= 16 \Rightarrow \text{interval} = [0, 25] \end{aligned}$$

histogram equalization:

$$N(q) = \max \left\{ 0, \text{round} \left(\frac{2^L \cdot c(q)}{n} \right) - 1 \right\}$$

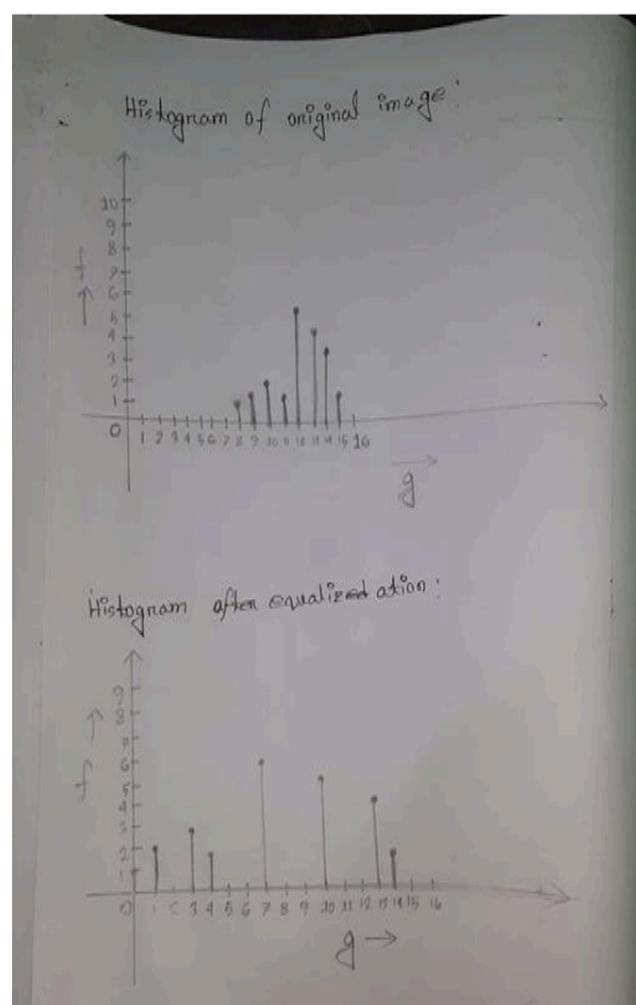
$$n = \sum f = 25$$

$$2^L = 46.46$$

| q | f | c | N | New-frequency |
|-----|-----|-----|-----|---------------|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 |
| 8 | 4 | 1 | 0 | 1 |
| 9 | 2 | 3 | 1 | 2 |
| 10 | 3 | 6 | 3 | 3 |
| 11 | 2 | 8 | 4 | 2 |
| 12 | 6 | 14 | 7 | 6 |
| 13 | 5 | 19 | 10 | 5 |
| 14 | 4 | 23 | 13 | 4 |
| 15 | 2 | 25 | 14 | 2 |

$\sum f = 25$

2(c)(ii): Answer



প্রশ্নে 5×5 ম্যাট্রিক্সের একটি প্রেক্ষেল ইমেজ দেওয়া হয়েছে যেখানে প্রতিটি পিক্সেল 8-বিট গভীরতা বিশিষ্ট (মানে 0-15 এর মধ্যে মান থাকতে পারে)।

প্রশ্ন অনুযায়ী, আমাদের করতে হবে—

1. Histogram Equalization প্রয়োগ করা এবং নতুন পিক্সেল মান বের করা।
2. মূল ইমেজ ও Histogram Equalized ইমেজের হিস্টোগ্রাম আঁকা।
3. কেন Histogram Equalization একটি পুরোপুরি ফ্ল্যাট হিস্টোগ্রাম তৈরি করে না? তা ব্যাখ্যা করা।

Step 1: Histogram তৈরি করা

প্রথমে মূল ইমেজের প্রতিটি পিক্সেল ভ্যালুর ফ্রিকোয়েন্সি (বার কতবার এসেছে) গণনা করা হয়।

ছবির দ্বিতীয় অংশে (হাতে লেখা কাগজে) একটি টেবিল তৈরি করা হয়েছে, যেখানে—

- প্রথম কলামে "z" (পিক্সেল ভ্যালু)
- দ্বিতীয় কলামে "f" (ফ্রিকোয়েন্সি বা সংখ্যার পরিমাণ)
- তৃতীয় কলামে "c" (কিউমুলেটিভ ফ্রিকোয়েন্সি)
- চতুর্থ কলামে "N(q)" (নতুন সমতলকৃত মান)
- পঞ্চম কলামে "New Frequency" (নতুন মান অনুযায়ী নতুন ফ্রিকোয়েন্সি)

$$N(g) = \max \left\{ 0, \text{round} \left(\frac{2^l \cdot c(g)}{n} \right) - 1 \right\}$$

- Where, $N(g)$ = new gray level of the pixel.
- $C(g)$ = cumulative pixel count upon old gray level g .
- Round = rounding operation to nearest integer value.
- 2^l = number of gray levels.
- N = total number of pixels.

A 4x4 original image is given with 3 bits/pixel.

| | | | |
|---|---|---|---|
| 1 | 3 | 4 | 0 |
| 0 | 7 | 4 | 2 |
| 2 | 6 | 7 | 4 |
| 1 | 2 | 3 | 1 |

- i) Perform Prewitt and Sobel operator on the image (Use padding)
- ii) Analyze the differences of both images.

Solution:

Prewitt Edge Detection

Input Image :

| | | | | |
|----|----|---|---|---|
| 10 | 9 | 9 | 4 | 0 |
| 0 | 6 | 6 | 2 | 2 |
| 5 | 9 | 8 | 4 | 3 |
| 7 | 5 | 5 | 4 | 3 |
| 8 | 10 | 8 | 5 | 0 |

Prewitt Kernel :

x - direction

| | | |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

y - direction

| | | |
|----|----|----|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| -1 | -1 | -1 |

with x-direction:

| | | | | |
|----|----|---|---|---|
| 10 | 9 | 9 | 4 | 0 |
| 0 | 6 | 6 | 2 | 2 |
| 5 | 9 | 8 | 4 | 3 |
| 7 | 5 | 5 | 4 | 3 |
| 8 | 10 | 8 | 5 | 0 |

x

| | | |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

$$\begin{aligned} 10 \times (-1) + 9 \times 0 + 9 \times 1 + (0) \times (-1) + 6 \times 0 + 6 \times 1 + 5 \times (-1) \\ + 9 \times 0 + 8 \times 1 \\ = 8 \end{aligned}$$

| | | |
|---|-----|-----|
| 8 | -14 | -18 |
| 7 | -10 | -11 |
| 1 | -11 | -15 |

= 1x

with y-direction:

| | | | | |
|----|----|---|---|---|
| 10 | 9 | 9 | 4 | 0 |
| 0 | 6 | 6 | 2 | 2 |
| 5 | 9 | 8 | 4 | 3 |
| 7 | 5 | 5 | 4 | 3 |
| 8 | 10 | 8 | 5 | 0 |

| | | |
|----|----|----|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| -1 | -1 | -1 |

| | | |
|---|---|---|
| 8 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

| | | |
|----|----|----|
| -6 | 1 | -2 |
| -5 | 0 | -2 |
| -4 | -2 | 2 |

= 1y

Thresholding

$$\text{Magnitude} = \sqrt{x^2 + y^2}$$

| | | |
|----|----|----|
| 10 | 14 | 18 |
| 9 | 10 | 11 |
| 4 | 11 | 15 |

$$\text{Threshold} = (10 + 14 + 18 + 9 + 10 + 11 + 4 + 11 + 15) / 9 = 11.2$$

$$x > 11 \Rightarrow 1$$

$$x \leq 11 \Rightarrow 0$$

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 1 |

Edge

| | | |
|---|---|---|
| 2 | 2 | 2 |
| 0 | 1 | 2 |
| 2 | 2 | 2 |

Sobel Edge Detection

Input Image:

| | | | | |
|----|----|---|---|---|
| 10 | 9 | 9 | 4 | 0 |
| 0 | 6 | 6 | 2 | 2 |
| 5 | 9 | 8 | 4 | 3 |
| 7 | 5 | 5 | 4 | 3 |
| 8 | 10 | 8 | 5 | 0 |

Sobel Kernel

| | | |
|----|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

x-direction

| | | |
|----|----|----|
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

y-direction

| | | |
|----|-----|-----|
| 14 | -18 | -22 |
| 10 | -15 | -16 |
| -1 | -12 | -17 |

1x

Exp of aggregation:

$$10 \times (-1) + 9 \times 0 + 9 \times 1 \\ 0 \times (-2) + 6 \times 0 + 6 \times 2 \\ + 5 \times (-1) + 9 \times 0 + 8 \times 1 \\ = (14) \quad (1,1)$$

form

Input Image x y-direction kernel

| | | |
|----|----|----|
| 0 | 2 | -2 |
| -4 | 1 | -4 |
| -5 | -2 | 1 |

$$\text{Magnitude} = \sqrt{(ix)^2 + (iy)^2}$$

Magnitude

| | | |
|----|----|----|
| 15 | 18 | 22 |
| 11 | 15 | 16 |
| 5 | 12 | 17 |

Threshold

$$\begin{aligned} & (15+18+22+11+15+16 \\ & +5+12+17)/9 \\ & = 15 \end{aligned}$$

Edge

$$x > \text{threshold} \Rightarrow 1$$

$$x \leq \text{threshold} \Rightarrow 0$$

| | | |
|---|---|---|
| 0 | 1 | 1 |
| 0 | 0 | 1 |
| 0 | 0 | 1 |

3. Prewitt ও Sobel অপারেটরের পার্থক্য

- Prewitt অপারেটর এজ ডিটেকশনে তুলনামূলকভাবে নরম (less sharp)।
- Sobel অপারেটর বেশি ওজন দেয় মধ্যবর্তী সারির পিক্সেলগুলোর মানকে, যার ফলে এটি প্রবণ (more prominent) এজ শনাক্ত করতে পারে।
- Sobel অপারেটরে Gx এবং Gy-এর মান বড় হয়, কারণ এটি কেন্দ্রীয় পিক্সেলগুলোর উপর বেশি জোর দেয়।
- Sobel এর Gradient Magnitude বেশি ফলে এটি বেশি শক্তিশালী এজ শনাক্ত করতে পারে।

উপসংহার:

- Prewitt অপারেটর সাধারণ কিন্তু কম শক্তিশালী।
- Sobel অপারেটর ধারালো (sharper) এবং এজ ডিটেকশনে বেশি কার্যকর।
- Sobel অপারেটর বেশি ব্যবহৃত হয় কোনো ইমেজের গুরুত্বপূর্ণ বৈশিষ্ট্য বের করতে।

