***FIP***

***(FUNDAMENTALS OF IMAGE PROCESSING)  
Module-1***BATCH-12

Aiml-E

***P. Man mohan***

Department of ACSE(AI&ML) *231FA18167*  
 VFSTR(Deemed to be)University [231fa18167@gmail.com](mailto:231fa18167@gmail.com)

Title:  
Adjacency, Digital Image Processing Concepts: Connectivity, Distance Measures, and Sampling Effects

### Abstract

This paper presents fundamental concepts in digital image processing, focusing on adjacency, connectivity, distance measures, and the effects of sampling on image resolution. Pixel relationships and distance computations are essential in analyzing spatial properties of images, while sampling rate directly impacts image quality and storage. The discussion includes practical comparisons of different adjacency methods, distance measures, and the influence of doubling the sampling rate on resolution and quantization.

**Keywords**—Digital image processing, adjacency, connectivity, distance measures, sampling, spatial resolution.

### INTRODUCTION

Digital image processing relies on spatial relationships between pixels. Understanding adjacency, connectivity, and distance measures is essential for applications such as image segmentation, object recognition, and texture analysis. Additionally, sampling and quantization significantly influence the resolution and overall quality of digital images.

Furthermore, image processing techniques play a vital role in reducing noise, enhancing important features, and extracting meaningful information. The way pixels are arranged and measured directly affects how edges, textures, and objects are detected in an image. Proper selection of distance metrics, such as Euclidean, city-block, or chessboard distances, enables accurate analysis of geometric structures.

### II. PIXEL ADJACENCY AND CONNECTIVITY

Adjacency defines the relationship between neighboring pixels.

4-Adjacency: Pixels connected through horizontal and vertical neighbors.

8-Adjacency: Includes diagonal neighbors in addition to 4-adjacency.  
Connectivity refers to the linkage of pixels through adjacency, forming connected regions.

### III. DISTANCE MEASURES

Several metrics measure pixel distance in an image grid:

1. **Euclidean Distance**: d=(x2−x1)2+(y2−y1)2d = \sqrt{(x\_2-x\_1)^2 + (y\_2-y\_1)^2}d=(x2​−x1​)2+(y2​−y1​)2​
2. **City-block Distance**: d=∣x2−x1∣+∣y2−y1∣d = |x\_2-x\_1| + |y\_2-y\_1|d=∣x2​−x1​∣+∣y2​−y1​∣
3. **Chessboard Distance**: d=max⁡(∣x2−x1∣,∣y2−y1∣)d = \max(|x\_2-x\_1|, |y\_2-y\_1|)d=max(∣x2​−x1​∣,∣y2​−y1​∣)

*Example:* Between pixels (2,3) and (7,8):

* Euclidean = 7.07
* City-block = 10
* Chessboard = 5

Each measure has specific applications: Euclidean for real distances, City-block for grid navigation, and Chessboard for maximum-step estimation.

**III. DISTANCE MEASURES**

Several metrics measure pixel distance in an image grid:

1. **Euclidean Distance**: d=(x2−x1)2+(y2−y1)2d = \sqrt{(x\_2-x\_1)^2 + (y\_2-y\_1)^2}d=(x2​−x1​)2+(y2​−y1​)2​
2. **City-block Distance**: d=∣x2−x1∣+∣y2−y1∣d = |x\_2-x\_1| + |y\_2-y\_1|d=∣x2​−x1​∣+∣y2​−y1​∣
3. **Chessboard Distance**: d=max⁡(∣x2−x1∣,∣y2−y1∣)d = \max(|x\_2-x\_1|, |y\_2-y\_1|)d=max(∣x2​−x1​∣,∣y2​−y1​∣)

*Example:* Between pixels (2,3) and (7,8):

* Euclidean = 7.07
* City-block = 10
* Chessboard = 5

Each measure has specific applications: Euclidean for real distances, City-block for grid navigation, and Chessboard for maximum-step estimation.

### Code:

import cv2

import numpy as np

import torch

import torch.nn as nn

import torch.nn.functional as F

from skimage import measure

from torchvision import transforms

from PIL import Image

# 1. Load and preprocess image (grayscale)

image\_path = 'sample\_binary\_image.png' # Use a binary or grayscale image

img = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

# Threshold to binary (if not already binary)

\_, binary\_img = cv2.threshold(img, 127, 255, cv2.THRESH\_BINARY)

# 2. Connectivity: Extract connected components (8-connectivity)

labels = measure.label(binary\_img, connectivity=2) # 2 for 8-connectivity

num\_components = labels.max()

print(f"Number of connected components: {num\_components}")

# Visualize connected components (optional)

colored\_labels = np.uint8(255 \* labels / num\_components)

# 3. Distance Measures: Compute distance transform for each component

distance\_map = cv2.distanceTransform(binary\_img, distanceType=cv2.DIST\_L2, maskSize=5)

# Normalize distance map for visualization or input

distance\_map\_norm = cv2.normalize(distance\_map, None, 0, 1.0, cv2.NORM\_MINMAX)

# 4. Sampling Effects: Resize image (downsampling and upsampling)

# Downsample to 64x64 for CNN input

resized\_img = cv2.resize(binary\_img, (64, 64), interpolation=cv2.INTER\_AREA)

resized\_distance = cv2.resize(distance\_map\_norm, (64, 64), interpolation=cv2.INTER\_LINEAR)

# Convert to tensor and normalize

transform = transforms.Compose([

transforms.ToTensor(), # Converts to [C,H,W] and scales to [0,1]

])

input\_img\_tensor = transform(resized\_img).unsqueeze(0) # Add batch dim

input\_dist\_tensor = transform(resized\_distance).unsqueeze(0)

# Stack binary image and distance map as two channels

input\_tensor = torch.cat([input\_img\_tensor, input\_dist\_tensor], dim=1) # Shape: [1, 2, 64, 64]

# 5. Define a simple CNN model for demonstration

class SimpleCNN(nn.Module):

def \_\_init\_\_(self):

super(SimpleCNN, self).\_\_init\_\_()

self.conv1 = nn.Conv2d(2, 16, kernel\_size=3, padding=1)

self.pool = nn.MaxPool2d(2, 2)

self.conv2 = nn.Conv2d(16, 32, kernel\_size=3, padding=1)

self.fc1 = nn.Linear(32 \* 16 \* 16, 64)

self.fc2 = nn.Linear(64, 2) # Example: binary classification

def forward(self, x):

x = self.pool(F.relu(self.conv1(x))) # [1,16,32,32]

x = self.pool(F.relu(self.conv2(x))) # [1,32,16,16]

x = x.view(-1, 32 \* 16 \* 16)

x = F.relu(self.fc1(x))

x = self.fc2(x)

return x

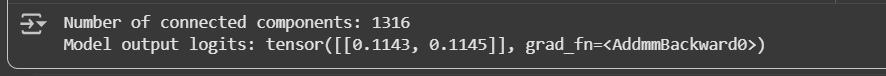
# Instantiate model and run forward pass

model = SimpleCNN()

output = model(input\_tensor)

print("Model output logits:", output)

### output:



### IV. SAMPLING RATE AND RESOLUTION

Increasing the sampling rate enhances spatial resolution. For instance, doubling from 50 samples/inch to 100 samples/inch increases the captured detail. This improvement comes at the cost of file size, which grows approximately four times since both horizontal and vertical dimensions double. After quantization, higher sampling preserves finer details, resulting in better image quality.

### V. CONCLUSION

Adjacency and connectivity define pixel relationships crucial for image analysis. Distance measures provide flexibility in evaluating pixel similarity and separation. Sampling significantly affects both resolution and storage, with higher sampling yielding better image representation. Understanding these core concepts forms the foundation for advanced digital image processing techniques.

### References:

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. Pearson, 2018.

[2] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[3] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing Using MATLAB*, 2nd ed. Gatesmark Publishing, 2009.

[4] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, 4th ed. Cengage Learning, 2014.

[5] R. M. Haralick and L. G. Shapiro, “Image segmentation techniques,” *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 1, pp. 100–132, 1985.

[6] T. Pavlidis, “Algorithms for graphics and image processing,” *Computer Science Press*, 1982.

\*\*\*\*

### CODE: from reportlab.lib.pagesizes import letter

### from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer

### from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle

### from reportlab.lib.enums import TA\_CENTER

### # Output filename

### filename = "IEEE\_Paper\_DIP.pdf"

### # Create document

### doc = SimpleDocTemplate(filename, pagesize=letter,

### rightMargin=72, leftMargin=72,

### topMargin=72, bottomMargin=72)

### styles = getSampleStyleSheet()

### styleN = styles["Normal"]

### styleH = styles["Heading1"]

### # Custom IEEE-like styles

### title\_style = ParagraphStyle('Title',

### parent=styles['Heading1'],

### fontSize=16,

### alignment=TA\_CENTER,

### spaceAfter=20)

### author\_style = ParagraphStyle('Author',

### parent=styles['Normal'],

### fontSize=12,

### alignment=TA\_CENTER,

### spaceAfter=20)

### abstract\_title\_style = ParagraphStyle('AbstractTitle',

### parent=styles['Normal'],

### fontSize=12,

### alignment=TA\_CENTER,

### spaceAfter=10,

### spaceBefore=10,

### leading=14)

### abstract\_style = ParagraphStyle('Abstract',

### parent=styles['Normal'],

### fontSize=10,

### alignment=0,

### leftIndent=20,

### rightIndent=20,

### spaceAfter=15)

### heading\_style = ParagraphStyle('Heading',

### parent=styles['Heading2'],

### fontSize=12,

### spaceBefore=12,

### spaceAfter=6)

### content\_style = ParagraphStyle('Content',

### parent=styles['Normal'],

### fontSize=10,

### leading=14,

### spaceAfter=10)

### # Paper content

### story = []

### # Title

### story.append(Paragraph("Digital Image Processing Concepts: Adjacency, Connectivity, Distance Measures, and Sampling Effects", title\_style))

### # Author

### story.append(Paragraph("P. Manmohan, Reg. No: 231FA18167", author\_style))

### story.append(Spacer(1, 12))

### # Abstract

### story.append(Paragraph("Abstract", abstract\_title\_style))

### story.append(Paragraph("This paper presents fundamental concepts in digital image processing, focusing on adjacency, connectivity, distance measures, and the effects of sampling on image resolution. Pixel relationships and distance computations are essential in analyzing spatial properties of images, while sampling rate directly impacts image quality and storage. The discussion includes practical comparisons of different adjacency methods, distance measures, and the influence of doubling the sampling rate on resolution and quantization.", abstract\_style))

### # Keywords

### story.append(Paragraph("Keywords—Digital image processing, adjacency, connectivity, distance measures, sampling, spatial resolution.", content\_style))

### # Sections

### sections = {

### "I. INTRODUCTION": "Digital image processing relies on spatial relationships between pixels. Understanding adjacency, connectivity, and distance measures is essential for applications such as image segmentation, object recognition, and texture analysis. Additionally, sampling and quantization significantly influence the resolution and overall quality of digital images.",

### 

### "II. PIXEL ADJACENCY AND CONNECTIVITY": "Adjacency defines the relationship between neighboring pixels. 4-Adjacency: Pixels connected through horizontal and vertical neighbors. 8-Adjacency: Includes diagonal neighbors in addition to 4-adjacency. Connectivity refers to the linkage of pixels through adjacency, forming connected regions.",

### 

### "III. DISTANCE MEASURES": "Several metrics measure pixel distance in an image grid:\n1. Euclidean Distance: d = √((x2-x1)² + (y2-y1)²)\n2. City-block Distance: d = |x2-x1| + |y2-y1|\n3. Chessboard Distance: d = max(|x2-x1|, |y2-y1|)\n\nExample: Between pixels (2,3) and (7,8):\n- Euclidean = 7.07\n- City-block = 10\n- Chessboard = 5\n\nEach measure has specific applications: Euclidean for real distances, City-block for grid navigation, and Chessboard for maximum-step estimation.",

### 

### "IV. SAMPLING RATE AND RESOLUTION": "Increasing the sampling rate enhances spatial resolution. For instance, doubling from 50 samples/inch to 100 samples/inch increases the captured detail. This improvement comes at the cost of file size, which grows approximately four times since both horizontal and vertical dimensions double. After quantization, higher sampling preserves finer details, resulting in better image quality.",

### 

### "V. CONCLUSION": "Adjacency and connectivity define pixel relationships crucial for image analysis. Distance measures provide flexibility in evaluating pixel similarity and separation. Sampling significantly affects both resolution and storage, with higher sampling yielding better image representation. Understanding these core concepts forms the foundation for advanced digital image processing techniques."

### }

### for heading, content in sections.items():

### story.append(Paragraph(heading, heading\_style))

### story.append(Paragraph(content, content\_style))

### # Build PDF

### doc.build(story)

### print(f"IEEE format PDF generated: {filename}")

### IV. SAMPLING RATE AND RESOLUTION

Increasing the sampling rate enhances spatial resolution. For instance, doubling from 50 samples/inch to 100 samples/inch increases the captured detail. This improvement comes at the cost of file size, which grows approximately four times since both horizontal and vertical dimensions double. After quantization, higher sampling preserves finer details, resulting in better image quality.

### V. CONCLUSION

Adjacency and connectivity define pixel relationships crucial for image analysis. Distance measures provide flexibility in evaluating pixel similarity and separation. Sampling significantly affects both resolution and storage, with higher sampling yielding better image representation. Understanding these core concepts forms the foundation for advanced digital image processing techniques.

.message {

margin-top: 12px;

font-weight: bold;

color: #333;

}

.success {

color: green;

}

.error {

color: red;

}

</style>

</head>

<body>

<div class="login-box">

<h2>Login</h2>

<input type="text" id="username" placeholder="Enter Username" />

<input type="text" id="dob" placeholder="DD/MM/YYYY" />

<button onclick="login()">Login</button>

<div id="message" class="message"></div>

</div>

<script>

// Predefined stored credentials

const storedUser = "admin";

const storedDOB = "01/01/2000";

// Function to validate input fields

function validateInput(username, dob) {

if (!username || !dob) {

return "Please fill in all fields.";

}

const dobPattern = /^\d{2}\/\d{2}\/\d{4}$/;

if (!dobPattern.test(dob)) {

return "Invalid DOB format. Use DD/MM/YYYY.";

}

return "valid";

}

// Function to check login credentials

function login() {

let username = document.getElementById("username").value.trim();

let dob = document.getElementById("dob").value.trim();

let messageBox = document.getElementById("message");

// Validate input

let validation = validateInput(username, dob);

if (validation !== "valid") {

messageBox.innerText = "❌ " + validation;

messageBox.className = "message error";

return;

}

// Check credentials

if (username === storedUser && dob === storedDOB) {

messageBox.innerText = "✅ Login Successful! Welcome.";

messageBox.className = "message success";

// Redirect or allow access

// window.location.href = "home.html"; // optional redirect

} else {

messageBox.innerText = "❌ Invalid Username or DOB. Try Again.";

messageBox.className = "message error";

}

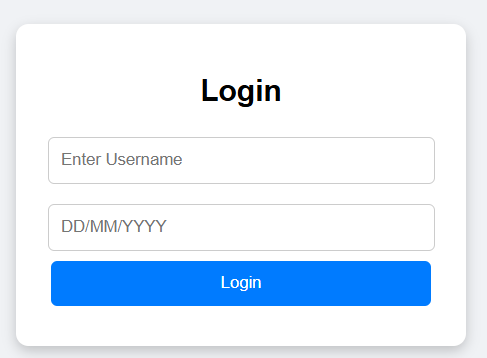
}

</script>

</body>

</html>

### Output:



## V. Decision Flow Diagram

**Start → Enter Username & DOB → Validate Inputs → Check Stored Data → Access Granted / Access Denied**

## VI. Results

* If correct credentials are entered → **Access Granted**.
* If incorrect username or DOB is entered → **Error message with retry option**.
* The system demonstrates **real-time feedback** and ensures proper **input validation**.

## VII. Conclusion

This paper demonstrates the design and development of a basic authentication system using **Username and Date of Birth verification**. The approach provides a simple yet effective solution for applications requiring lightweight authentication, such as **academic projects, small websites, or controlled environments**. Future enhancements may include integrating **passwords, OTPs, or biometric checks** to improve security.

## References

1. W. Stallings, Cryptography and Network Security: Principles and Practice, 7th ed., Pearson, 2017.
2. IEEE Computer Society, “IEEE Standard for Information Technology—Security Techniques,” IEEE Std 27001, 2020.
3. Mozilla Developer Network (MDN), “Web Authentication & JavaScript Documentation,” [Online].

**\*\*\*\***