Image Processing with Python — Complete Comprehensive Tutorial

This tutorial is written step-by-step for **students, teachers, and beginners**, and gradually moves to **intermediate & practical level**. Examples use **Python 3.x** and are suitable for academic projects, final-year projects, and real applications.

---

1. What is Image Processing?

Image processing means **performing operations on an image** to: - Enhance it - Extract useful information - Analyze patterns - Prepare it for Machine Learning / AI

Common Applications

- Face recognition
- Medical imaging (X-ray, MRI)
- CCTV & surveillance
- Object detection
- Document scanning
- Lost & Found systems (your recent project 😉)

---

2. Image Processing vs Computer Vision

Image Processing	Computer Vision
Image enhancement	Understanding image content
Noise removal	Object recognition
Filtering	Scene understanding
Feature extraction	AI-based decision making

---

3. Required Python Libraries

Install once:

```
pip install opencv-python pillow matplotlib numpy scikit-image
```

Libraries Purpose

- **OpenCV (cv2)** → Core image processing
- **Pillow (PIL)** → Simple image handling
- **NumPy** → Matrix operations
- **Matplotlib** → Visualization
- **Scikit-Image** → Advanced algorithms

---

4. Reading & Displaying Images

Using OpenCV

```
import cv2
img = cv2.imread('image.jpg')
cv2.imshow('Image', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Using Pillow

```
from PIL import Image
img = Image.open('image.jpg')
img.show()
```

---

5. Image Properties

```
import cv2
img = cv2.imread('image.jpg')
print(img.shape)   # (height, width, channels)
print(img.size)    # total pixels
print(img.dtype)  # data type
```

---

6. Color Spaces

BGR → RGB

```
rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

RGB → Grayscale

```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

RGB → HSV

```
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

---

7. Image Resizing & Cropping

```
resized = cv2.resize(img, (300, 300))
cropped = img[50:200, 100:300]
```

---

8. Image Rotation & Flipping

```
rotated = cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)
flipped = cv2.flip(img, 1)  # 1=horizontal, 0=vertical
```

---

9. Drawing on Images

```
cv2.rectangle(img, (50,50), (200,200), (0,255,0), 2)
cv2.circle(img, (150,150), 40, (255,0,0), -1)
cv2.putText(img, 'Hello', (50,300), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)
```

---

10. Image Arithmetic Operations

Addition & Subtraction

```
result = cv2.add(img, img)
result = cv2.subtract(img, img)
```

Bitwise Operations

```
bit_and = cv2.bitwise_and(img, img)
```

---

11. Image Thresholding

Simple Threshold

```
_, thresh = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
```

Adaptive Threshold

```
thresh = cv2.adaptiveThreshold(gray,255,
cv2.ADAPTIVE_THRESH_MEAN_C,
cv2.THRESH_BINARY,11,2)
```

---

12. Image Filtering (Noise Removal)

Gaussian Blur

```
blur = cv2.GaussianBlur(img, (5,5), 0)
```

Median Blur

```
median = cv2.medianBlur(img, 5)
```

---

13. Edge Detection

Canny Edge Detector

```
edges = cv2.Canny(gray, 100, 200)
```

---

14. Morphological Operations

```
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5,5))
dilation = cv2.dilate(thresh, kernel)
erosion = cv2.erode(thresh, kernel)
```

---

15. Contour Detection

```
contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
for cnt in contours:
    cv2.drawContours(img, [cnt], -1, (0,255,0), 2)
```

---

16. Feature Detection (Basics)

ORB Feature Detector

```
orb = cv2.ORB_create()
kp, des = orb.detectAndCompute(gray, None)
img_kp = cv2.drawKeypoints(img, kp, None)
```

---

17. Histogram & Equalization

```
hist = cv2.calcHist([gray],[0],None,[256],[0,256])
equalized = cv2.equalizeHist(gray)
```

---

18. Image Segmentation

K-Means Segmentation

```
import numpy as np
Z = img.reshape((-1,3))
Z = np.float32(Z)
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)
```

```
K = 3
_, labels, centers = cv2.kmeans(Z, K, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)
centers = np.uint8(centers)
segmented = centers[labels.flatten()].reshape(img.shape)
```

---

19. Saving Images

```
cv2.imwrite('output.jpg', img)
```

---

20. Image Metadata (Advanced – PIL)

```
from PIL import Image, PngImagePlugin
img = Image.open('image.png')
meta = PngImagePlugin.PngInfo()
meta.add_text('user_id', '123')
meta.add_text('status', 'lost')
img.save('output.png', pnginfo=meta)
```

---

21. Mini Projects (Recommended)

1. Face detection system
2. Document scanner
3. Bone fracture detection
4. Lost & Found image system
5. OCR-based text reader

---

22. Image Processing → AI Roadmap (For You)

Since you want **AI/ML Engineer** path:

Image Processing → Computer Vision → ML → CNN → YOLO → Transformers

---

23. Tools for Practice

- OpenCV

- Google Colab
- Kaggle datasets
- LabelImg
- MediaPipe

---

24. Final Advice

- Practice with real images
- Visualize every step
- Combine with Flask / Android
- Move towards CNN once basics are strong

If you want, I can: 👉🏽Convert this into **PDF / Lab Manual** 👉🏽Create **MCQs + Viva questions** 👉🏽Design **university-level syllabus**👉🏽 Build **real projects step-by-step**

Just tell me 👍