IMAGE FILTERING USING PYTHON & OPENCV

1. Introduction

Image Processing computer science ka ek important field hai jisme digital images par different operations perform kiye jaate hain.
Is project me humne **Python** aur **OpenCV** library ka use karke image par different **filters** apply kiye hain jaise:

- Grayscale Filter
- Blur Filter
- Edge Detection Filter

Is project ka main goal image ko analyze karna aur uski visual quality ko improve ya extract karna hai.

2. Objectives

Is project ke objectives ye hain:

- Image ko Python me load karna
- Image par different filters apply karna
- Original aur filtered images ko compare karna
- OpenCV aur Matplotlib libraries ka practical use samajhna

3. Tools & Technologies Used

- **Python**
- **Google Colab**
- **OpenCV (cv2)**
- **NumPy**
- **Matplotlib**

4. Libraries Description

- **OpenCV:** Image processing aur computer vision ke liye use hoti hai
- **NumPy:** Image data ko array ke form me handle karta hai
- **Matplotlib:** Images ko visualize/display karne ke liye use hoti hai

5. Methodology / Working

Step 1: Install Required Libraries

```
!pip install opencv-python matplotlib numpy
```

Step 2: Upload Image

User apni image Google Colab par upload karta hai:

```
from google.colab import files
uploaded = files.upload()
```

Step 3: Read & Display Original Image

- Image OpenCV ke through read ki jaati hai
- BGR se RGB format me convert kiya jata hai

```
img = cv2.imread(image_path)
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

Step 4: Apply Grayscale Filter

Grayscale image sirf intensity show karti hai, color information remove ho jaati hai:

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

Step 5: Apply Blur Filter

Blur filter noise reduce karta hai aur image smooth banata hai:

```
blur = cv2.GaussianBlur(img_rgb, (15,15), 0)
```

Step 6: Apply Edge Detection Filter

Edge detection image ke important boundaries ko detect karta hai:

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

Step 7: Display All Images Together

Original aur filtered images ko ek hi window me display kiya jata hai for comparison.

6. Results

- **Original Image:** Actual uploaded image
- **Grayscale Image:** Black & white version
- **Blur Image:** Smooth aur noise-free image
- **Edge Detection:** Image ke sharp edges clearly visible

Isse image analysis aur feature extraction easily ho jaata hai.

7. Applications

- Face Detection
- Medical Imaging
- Object Detection
- Security & Surveillance
- Computer Vision Projects

8. Advantages

- Easy to implement
- Fast processing
- Real-time applications ke liye useful
- Open-source libraries ka use

9. Limitations

- High-quality images ke liye zyada processing power chahiye
- Lighting condition image result ko affect karti hai

10. Conclusion

Is project me humne successfully Python aur OpenCV ka use karke image filtering perform ki.
Different filters ke through image ke features ko analyze aur enhance kiya gaya.
Ye project beginners ke liye image processing seekhne ka best example hai.