Frequency Domain Filtering Using OpenCV

What we need to get started with OpenCv...
We need to import few libraries given below and are available in **Google Colab**, independent installations may be required for other platforms.
1. Imports required

```
import cv2
from google.colab.patches import cv2_imshow
import numpy as np
import matplotlib.pyplot as plt
```

2. Next we import an image and use a simple Edge Preserving Filter.
Domain Filtering - Frequency Domain Filters are used for smoothing and sharpening of image by removal of high or low frequency components. Sometimes it is possible of removal of very high and very low frequency.

```
#read image
src = cv2.imread(r'/content/P.png', cv2.IMREAD_UNCHANGED)
DF = cv2.edgePreservingFilter(src, flags=1, sigma_s=60, sigma_r=0.4)
print("Domain Filter - Cartoonify")
cv2_imshow(numpy.hstack((src, DF)))
```

3. A- Smoothing of Image and other Domain Filters.
Frequency Domain Filters are used for smoothing and sharpening of images by removal of high or low-frequency components.
Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function.

```
#apply guassian blur on src image
dst = cv2.GaussianBlur(src,(5,5),cv2.BORDER_DEFAULT)
print("Gaussian Smoothing")
cv2_imshow(numpy.hstack((src, dst)))
```

Gaussian Smoothing

Mean Filter

```
kernel = np.ones((10,10),np.float32)/25
dst2 = cv2.filter2D(src,-1,kernel)
print("Mean Filter")
cv2_imshow(numpy.hstack((src, dst2)))
```

Mean Filter

Median Filter

```
#Median Filter
dst3 = cv2.medianBlur(src,5)
print("Median Filter")
```

```
Median Filter
```

4. B- Sharpening Filter -
Bilateral filter

```
#Bilateral filter
print("Bilateral Filter")
dst4 = cv2.bilateralFilter(src, 60, 60, 60)
cv2_imshow(numpy.hstack((src, dst4)))
```

5. C- Frequency Band Filter
Low Pass

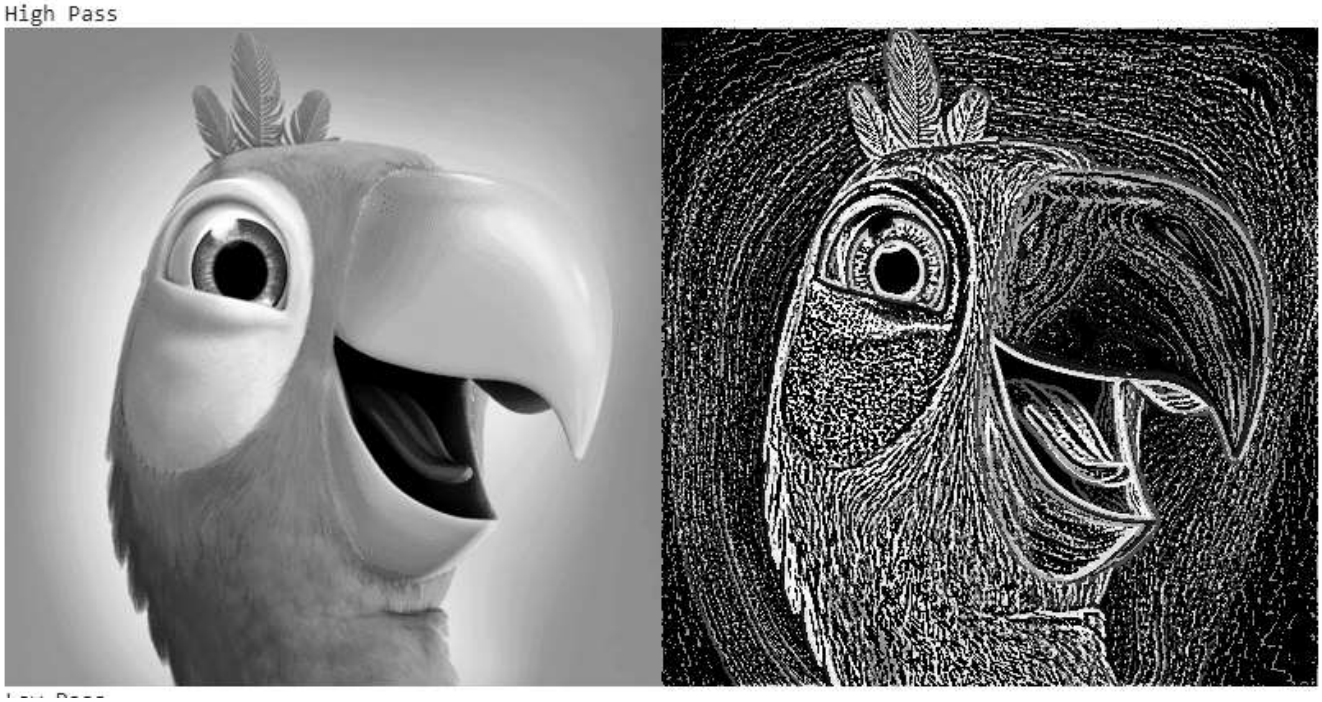
```
#low pass filter
Lp = cv2.filter2D(src,-1, kernel)
Lp = src - Lp
print("Low Pass")
cv2_imshow(numpy.hstack((src, Lp)))
```

Low Pass

High Pass

```
Hp = src - dst
filtered = filtered + 127*numpy.ones(src.shape, numpy.uint8)
print("High Pass")
cv2_imshow(numpy.hstack((src, Hp)))
```

High Pass



Frequency filters process an image in the frequency domain. The image is Fourier transformed, multiplied with the filter function and then re-transformed into the spatial domain. Attenuating high frequencies results in a smoother image in the spatial domain, attenuating low frequencies enhances the edges.


alcatraz714

OpenCV image processing using Python

Rishi Saxena ・ Sep 9 ・ 4 min read

#python #jupyter #opencv

Thanks for Reading...

36 0

Categories:

Python

Tags:

Python Opencv Jupyter

Share What You Think

Post Comment

Show Comments

Related Posts

NodeJS vs Python

App Development Practices To Follow For Your Business In 2021

Creating a Podcast of book summaries and articles in PDF in Portuguese with AI in Python!

Getting started with Docker & Flask.

21 must-read books for kids, beginner, advanced & interview with PDF

Weekly Challenge 140

Logistic Regression

OOP in Python