EE604 Assignment 4

Name: Deependra Chansoliya **Roll no: 180227**

For Rome.jpg

Algorithm	Speed(sec)	PSNR	SSIM
anyf	7	16.8145	0.0865
mybf (original)	56.34	NA	NA
mybf (your speed improved)	14.85	21.6335	0.7512
rtbf (para-1: good accuracy)	0.93	17.8102	0.4738
rtbf (para-2: good speed)	7.36	12.7531	0.7369
rtbf (para-3: best para)	184.54	24.6554	0.9012

For IITK.jpg

Algorithm	Speed(sec)	PSNR	SSIM
anyf	50.50	18.8082	0.8287
mybf (original)	420.52	NA	NA
mybf (your speed improved)	106.16	28.4160	0.9195
rtbf (para-1: good accuracy)	10.21	27.7145	0.7991
rtbf (para-2: good speed)	17.47	26.6123	0.7596
rtbf (para-3: best para)	12.63	26.3752	0.8665

1. **Any Filter:**

```
fastNlMeansDenoisingColored(P1,P2,P3,P4,P5,P6);
```

Parameters:
P1 – Source Image Array
P2 – Destination Image Array
P3 – Size in pixels of the template patch that is used to compute weights: 10
P4 – Size in pixels of the window to compute a weighted average for the given pixel: 10
P5 – Parameter regulating filter strength for luminance component: 7
P6 – Same as above but for color components // Not used in a grayscale image: 21

Original	Result
	
	

2. Bilateral filter: mybf

I have downsampled the image then I used the filter and the again upsampled the image. mybf filter runs faster on downsampled images.

4. Analysis

	sigmaspatial:	sigmarange
For good accuracy	100	0.2
For good speed	10	0.2
For optimal parameters for high speed as well as maintaining acceptable denoising performance	75	0.2

5. Application

Parameters
Blur_value = 7, Line_size = 7

Algorithm

```
 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
 gray_blur = cv2.medianBlur(gray, blur_value)
 edges = cv2.adaptiveThreshold(gray_blur, 255,
cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY, line_size, blur_value)

 lab = cv2.cvtColor(img,cv2.COLOR_BGR2LAB)
 l, a, b = cv2.split(lab)
 l=l/100.0
 img = img.astype("float32")
 for i in range(5):
   l = rt_ft(l, 75.0, 0.2)
   out_img=l*100.0
   out_img= np.uint8(out_img)
   out_img= cv2.merge((out_img,a,b))
   result=cv2.cvtColor(out_img,cv2.COLOR_LAB2BGR)
   out_img= cv2.bitwise_and(out_img, out_img, mask=edges)
```

6. Segmentation

Code in zip file

Result Image