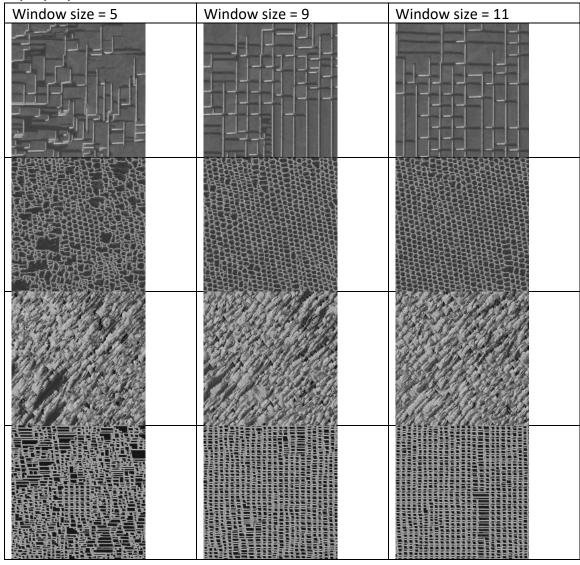
# Assignment 2 bf289

# **Texture Synthesis and Image Inpainting**

## • Texture Synthesis

Totally, the performance for texture synthesis will be better with the increment of the window size. We first use window size parameter to 5, 9, 11 pixels to do the texture synthesis, and the T5 still did not perform well, so we also use window size 15, 21 to do texture on T5, which is apparently better than previous parameters.

T1, T2, T3, T4



Window size = 5	Window size = 9	Window size = 11
	rioub at our a sit in Das locarian, what do not hooked for the root and a sit in Das locarian, who would be for the root and a sit in Das locarian, who would be for this property of the root of the	thought with a cong consecutivida Discone through the condition of the con
Window size = 15	Window size = 21	
use Mool Actors. 1873 Lew	b Ionical-left a Lewa Lex He left to Leva Lex nob ril; loonicall. Heft nd itse left at not it last 'pp'' al copy!" ast fal comd itself, at not it last fall. He ca ruses thedian Al Frootis, "ast fal dais Hef chia he fal coms," as Housed it last now self; last nov cal conica last fall, Heft a rin Al He fall. He fal fal comd it!" 'litica Lewa she Heft "litio rin Al I : "the left a ringing quest fal conica Leva Lex left I. He years of Montica Lewa fall. Hound traelf, a selfiselit!" 'hat, "thise was ton was House I ef fall." ill, at ng ratow selfitical Footse Phil. Heft a rind rs al Frish "thaf, "a initiates romat hest nd ticars ew self, at "thâr, quous years dais 'Hed itse Ho imaging roomedaires' and ripp?" Trippiped itsd s of Mat ngitop? Thouse Fitts years come left nging room Al. He fal. He peft a roow st "the Ho drans," al p. Fis dang quest nom?Led ticaths do	

And here is the running time for different images and different window size.

```
Texture :T1.gif Windows Size :5 Time :129.313119173 seconds
Texture :T1.gif Windows Size :9 Time :193.906089067 seconds
Texture :T1.gif Windows Size :11 Time :251.079768181 seconds
Texture :T2.gif Windows Size :5 Time :88.852558136 seconds
Texture :T2.gif Windows Size :9 Time :121.760793924 seconds
Texture :T2.gif Windows Size :11 Time :157.60354805 seconds
Texture :T3.gif Windows Size :5 Time :89.0545659065 seconds
Texture :T3.gif Windows Size :9 Time :120.837645054 seconds
Texture: T3.qif Windows Size: 11 Time: 161.412675858 seconds
Texture :T4.gif Windows Size :5 Time :70.8592510223 seconds
Texture :T4.gif Windows Size :9 Time :86.9934499264 seconds
Texture :T4.qif Windows Size :11 Time :115.035634995 seconds
Texture :T5.gif Windows Size :5 Time :411.663320065 seconds
Texture :T5.gif Windows Size :9 Time :568.823732138 seconds
Texture :T5.gif Windows Size :11 Time :694.184584856 seconds
Texture :T5.gif Windows Size :15 Time :988.866293192 seconds
Texture :T5.gif Windows Size :21 Time :1520.98673487 seconds
```

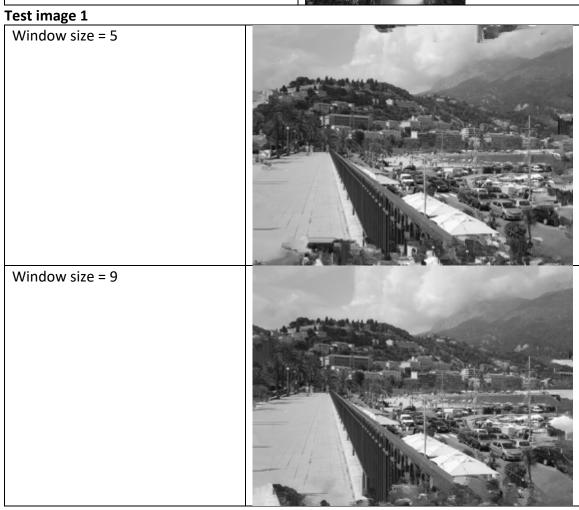
### Image Inpainting

Make some changes on the code in question 1, make the black region as the unfilled part, and to find the best matches in the rest part. And we also use the window size parameter to 5, 9, 11 to do the image inpainting.

, ,	 	0
Test Image 1		Test image 2
ICSt IIIIage I		TCSt IIIIage 2







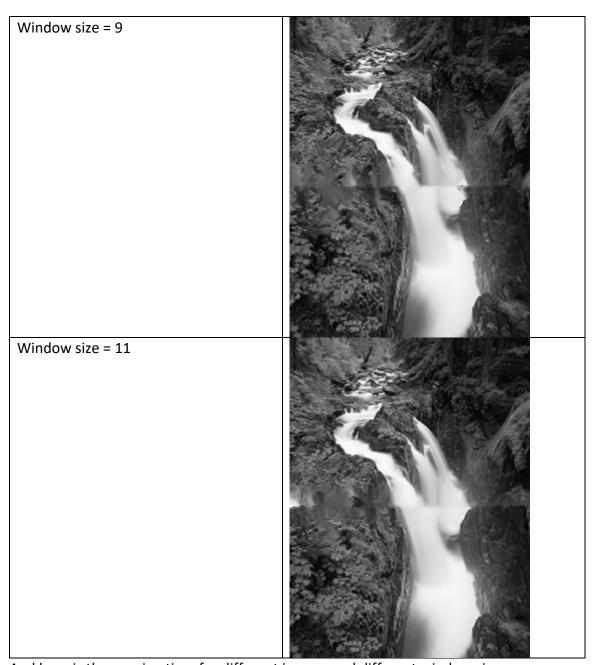
Window size = 11



# Test image 2

Window size = 5





#### And here is the running time for different images and different window size.

Image :test\_im1.bmp Windows Size :5 Time :1086.48941088 seconds
Image :test\_im1.bmp Windows Size :9 Time :1680.17870116 seconds
Image :test\_im1.bmp Windows Size :11 Time :2082.07063198 seconds

Image :test\_im2.bmp Windows Size :5 Time :568.088115931 seconds
Image :test\_im2.bmp Windows Size :9 Time :864.564464092 seconds
Image :test\_im2.bmp Windows Size :11 Time :1060.98718214 seconds

# • Object Removal

Creminis, Perez and Toyama's algorithm has better performance, which has higher quality than the Efros and Leung's algorithm, but both way cost so much time to remove object from the image.



Efros and Leung's algorithm



Creminis, Perez and Toyama's algorithm



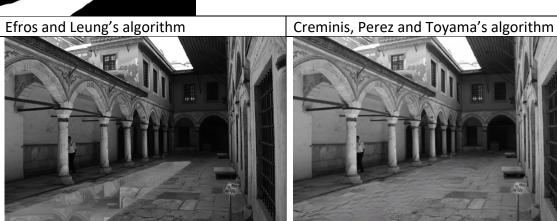
Efros and Leung's algorithm



Creminis, Perez and Toyama's algorithm





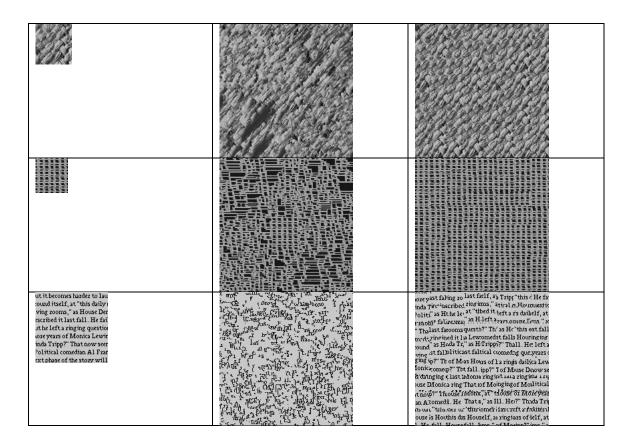


# • Image Quilting

#### Quality

The Efros and Freeman's method is much better than the Efros and Leung's method when we use the same image at the same window size 5, which means the patch size is 25.

texture	Efros and Leung's	Efros and Freeman's
	Algorithm	Algorithm



#### **Running Time**

For the running time, the Efros and Freeman's method is not always bettre than the Efros and Leung's method. For different images, the result is different, for T1 to T4, the Efros and Freeman's method is better since these texture is easy and Efros and Leung's method works on one pixel each time, which is slower than one patch each time. And for T5, which is a hard texture, although one patch is quicker than one pixel, but the overlapping cost much time, which makes the total time longer.

Running time for Efros and Freeman's method

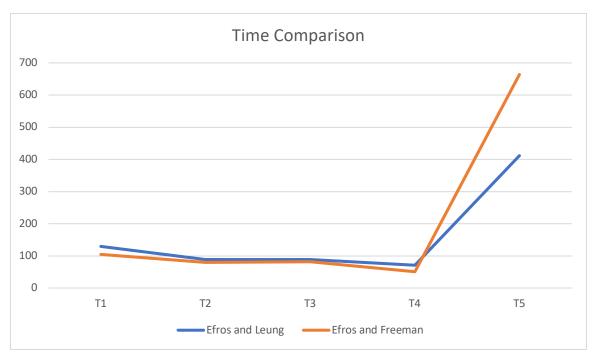
Texture :T1.gif Time :104.02799201 seconds

Texture :T2.gif Time :79.6215689182 seconds

Texture :T3.gif Time :81.5578610897 seconds

Texture :T4.gif Time :50.6514790058 seconds

Texture :T5.gif Time :664.048991919 seconds



The implementation of the Efros and Freeman's Algorithm is obtained from https://github.com/veslam/Exemplar-Based-Inpaining-Python.

#### How to Run

The texture synthesis and the image inpainting are implemented in efros.py, and object removal is implemented in exemplarBasedInpainting.py, and to get the results for first three questions, just run the main.py, which has included all these parts.

The image quilting is implemented in PathBasedSynthesis.py, you can just run this file to get the result for the last question.