Coding Assignment 3

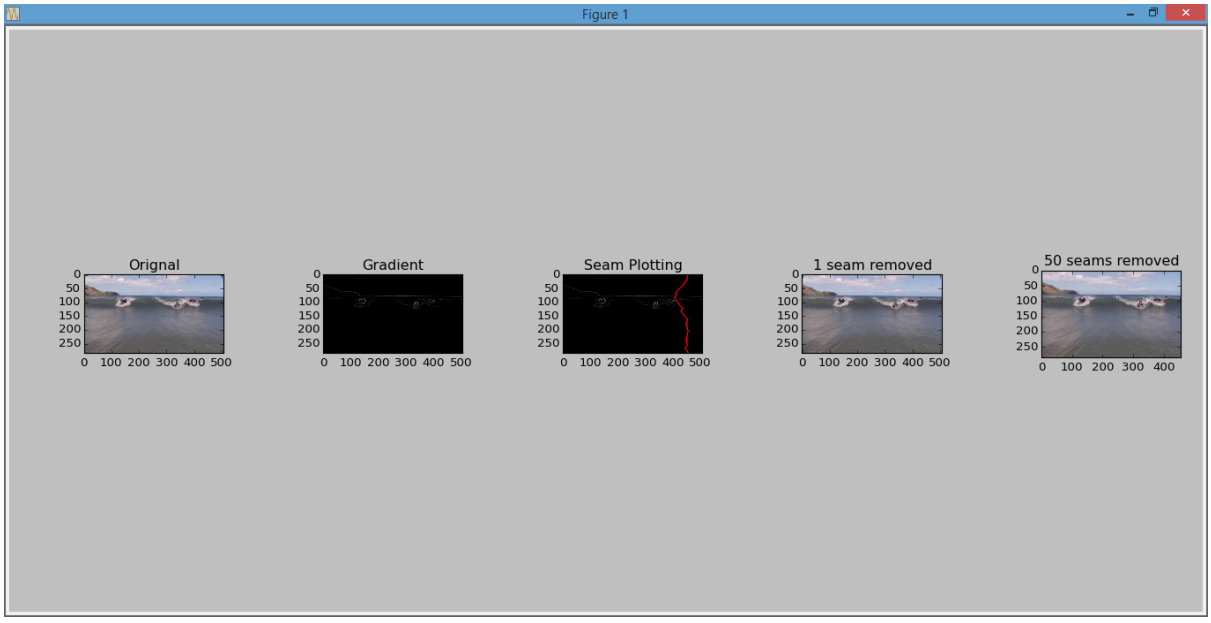
Posting ID: 0970-503

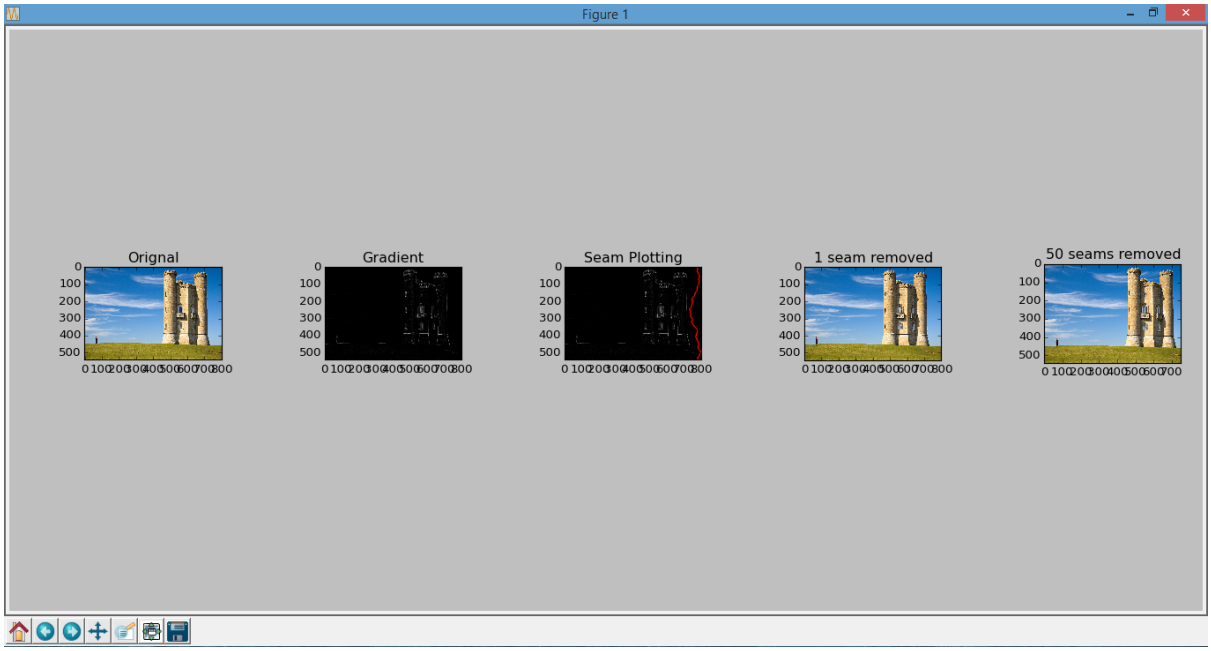
1. Reflection

In this coding assignment, I am able to successfully complete seam carving of a given image.
Seam carving is an approach to change the aspect ratio of an image without stretching any of the important objects depicted in it. The seam carving algorithm uses the **dual gradient energy** function as a disruption measure.
The code displays the minimum cost of the each seam removed and shows the image after one seam removal and after removing 50 seams. The output also displays the image after calculating dual gradient energy and the image after finding the first seam.

During this coding assignment, I was able to learn the following things:
a. Manipulation of images using **skimage** and **numpy** python package.
b. Calculating the energy of each pixel using **hsobel** and **vsobel** function.
c. Identified and removed a vertical seam of minimum energy, using the **dynamic programming** algorithm for shortest-path problem.

2. Testing Output





3. Static Analysis / Compilation Output

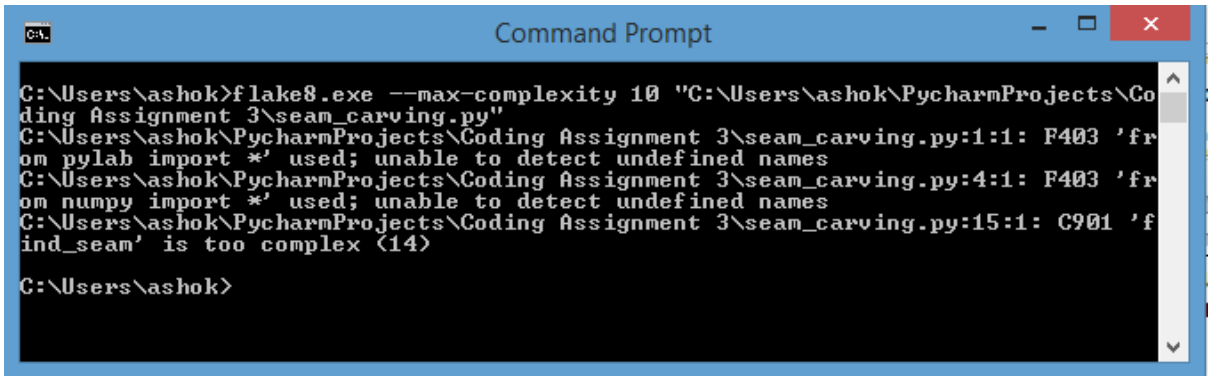
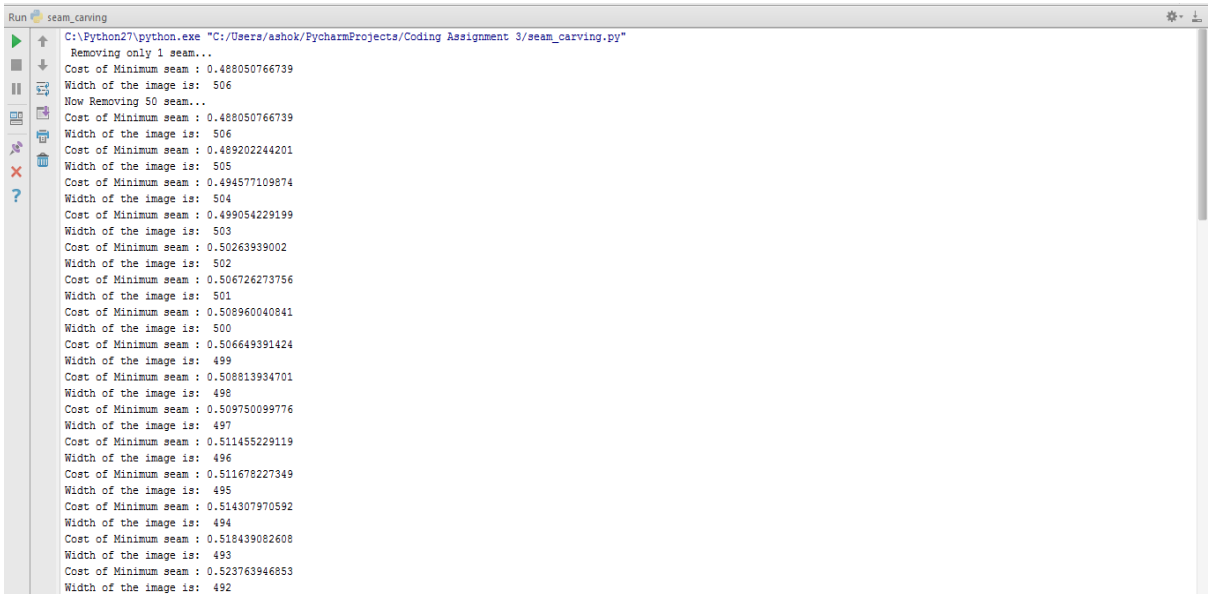
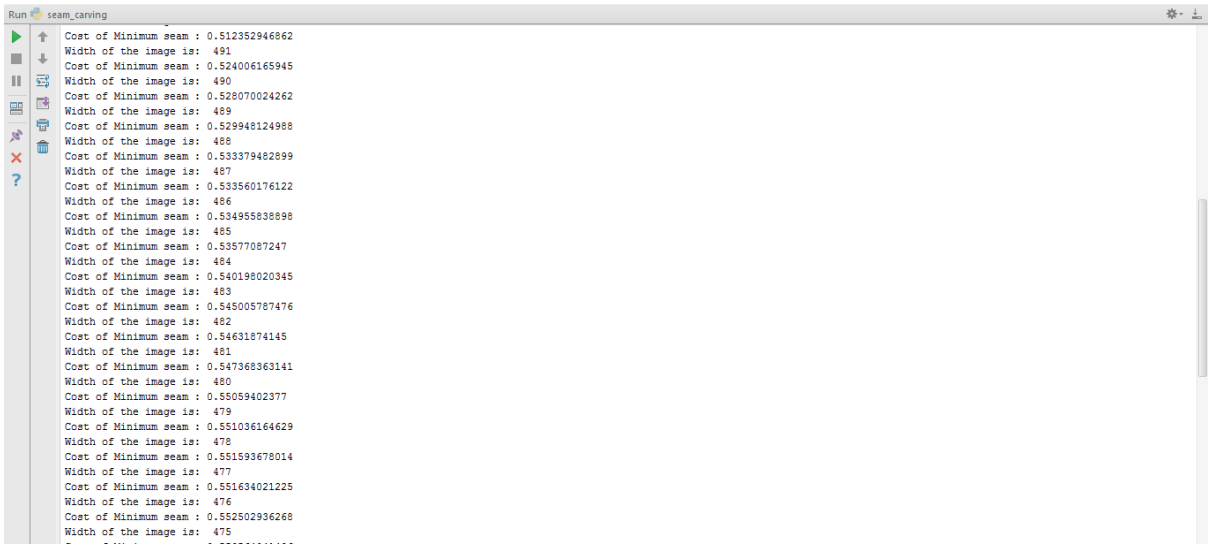


Image 1:


Run seam_carving
C:\Python27\python.exe "C:/Users/ashok/PycharmProjects/Coding Assignment 3/seam_carving.py"
Removing only 1 seam...
Cost of Minimum seam : 0.488050766739
Width of the image is: 506
Now Removing 50 seam...
Cost of Minimum seam : 0.488050766739
Width of the image is: 506
Cost of Minimum seam : 0.489202244201
Width of the image is: 505
Cost of Minimum seam : 0.494577109874
Width of the image is: 504
Cost of Minimum seam : 0.499054229199
Width of the image is: 503
Cost of Minimum seam : 0.50263939002
Width of the image is: 502
Cost of Minimum seam : 0.506726273756
Width of the image is: 501
Cost of Minimum seam : 0.508960040841
Width of the image is: 500
Cost of Minimum seam : 0.506649391424
Width of the image is: 499
Cost of Minimum seam : 0.508813934701
Width of the image is: 498
Cost of Minimum seam : 0.509750099776
Width of the image is: 497
Cost of Minimum seam : 0.511455229119
Width of the image is: 496
Cost of Minimum seam : 0.511678227349
Width of the image is: 495
Cost of Minimum seam : 0.514307970592
Width of the image is: 494
Cost of Minimum seam : 0.518439082608
Width of the image is: 493
Cost of Minimum seam : 0.523763946853
Width of the image is: 492


Run seam_carving
Cost of Minimum seam : 0.512352946862
Width of the image is: 491
Cost of Minimum seam : 0.524006165945
Width of the image is: 490
Cost of Minimum seam : 0.528070024262
Width of the image is: 489
Cost of Minimum seam : 0.529948124988
Width of the image is: 488
Cost of Minimum seam : 0.533379482899
Width of the image is: 487
Cost of Minimum seam : 0.533560176122
Width of the image is: 486
Cost of Minimum seam : 0.534955838898
Width of the image is: 485
Cost of Minimum seam : 0.53577087247
Width of the image is: 484
Cost of Minimum seam : 0.540198020345
Width of the image is: 483
Cost of Minimum seam : 0.545005787476
Width of the image is: 482
Cost of Minimum seam : 0.54631874145
Width of the image is: 481
Cost of Minimum seam : 0.547368363141
Width of the image is: 480
Cost of Minimum seam : 0.55059402377
Width of the image is: 479
Cost of Minimum seam : 0.551036164629
Width of the image is: 478
Cost of Minimum seam : 0.551593678014
Width of the image is: 477
Cost of Minimum seam : 0.551634021225
Width of the image is: 476
Cost of Minimum seam : 0.552502936268
Width of the image is: 475

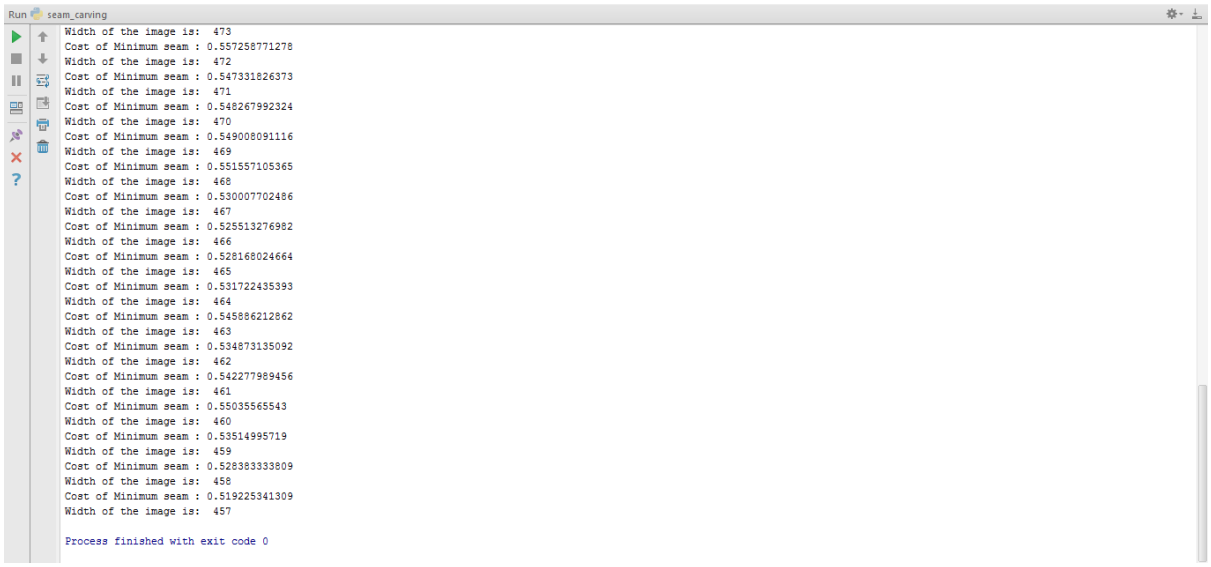
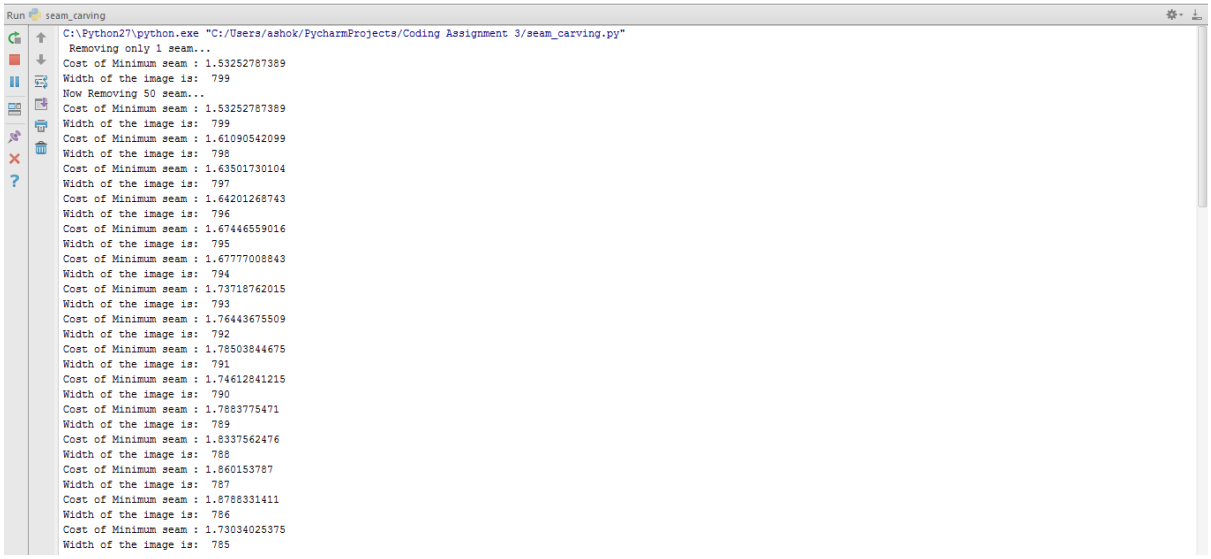
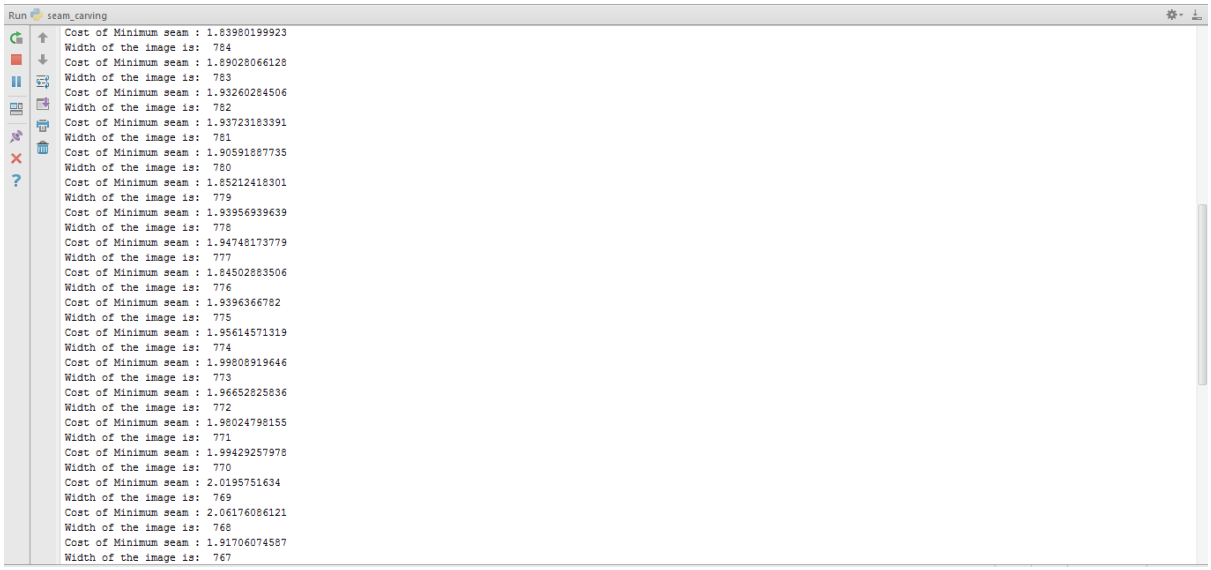
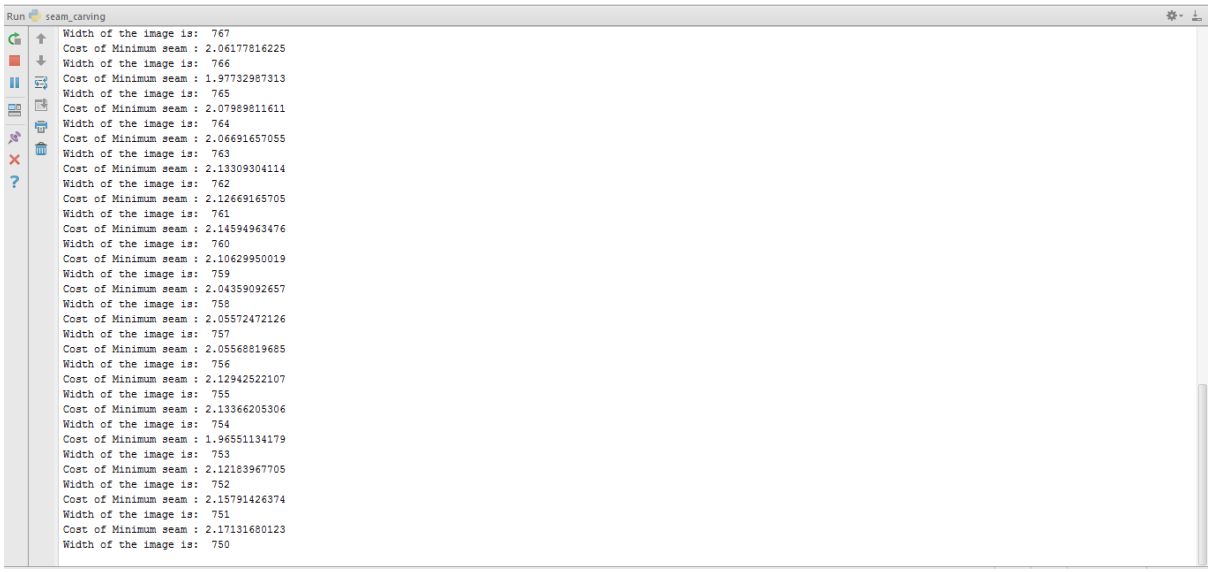

Run seam_carving
Width of the image is: 473
Cost of Minimum seam : 0.557258771278
Width of the image is: 472
Cost of Minimum seam : 0.547331826373
Width of the image is: 471
Cost of Minimum seam : 0.548267992324
Width of the image is: 470
Cost of Minimum seam : 0.549008091116
Width of the image is: 469
Cost of Minimum seam : 0.551557105365
Width of the image is: 468
Cost of Minimum seam : 0.530007702486
Width of the image is: 467
Cost of Minimum seam : 0.525513276982
Width of the image is: 466
Cost of Minimum seam : 0.528168024664
Width of the image is: 465
Cost of Minimum seam : 0.531722435393
Width of the image is: 464
Cost of Minimum seam : 0.545886212862
Width of the image is: 463
Cost of Minimum seam : 0.534873135092
Width of the image is: 462
Cost of Minimum seam : 0.542277989456
Width of the image is: 461
Cost of Minimum seam : 0.55035565543
Width of the image is: 460
Cost of Minimum seam : 0.53514995719
Width of the image is: 459
Cost of Minimum seam : 0.528383333809
Width of the image is: 458
Cost of Minimum seam : 0.519225341309
Width of the image is: 457
Process finished with exit code 0

Image 2 :


Run seam_carving
C:\Python27\python.exe "C:/Users/ashok/PycharmProjects/Coding Assignment 3/seam_carving.py"
Removing only 1 seam...
Cost of Minimum seam : 1.53252787389
Width of the image is: 799
Now Removing 50 seam...
Cost of Minimum seam : 1.53252787389
Width of the image is: 799
Cost of Minimum seam : 1.61090542099
Width of the image is: 798
Cost of Minimum seam : 1.63501730104
Width of the image is: 797
Cost of Minimum seam : 1.64201268743
Width of the image is: 796
Cost of Minimum seam : 1.67446559016
Width of the image is: 795
Cost of Minimum seam : 1.67777008843
Width of the image is: 794
Cost of Minimum seam : 1.73718762015
Width of the image is: 793
Cost of Minimum seam : 1.76443675509
Width of the image is: 792
Cost of Minimum seam : 1.78503844675
Width of the image is: 791
Cost of Minimum seam : 1.74612841215
Width of the image is: 790
Cost of Minimum seam : 1.7883775471
Width of the image is: 789
Cost of Minimum seam : 1.8337562476
Width of the image is: 788
Cost of Minimum seam : 1.860153787
Width of the image is: 787
Cost of Minimum seam : 1.8788331411
Width of the image is: 786
Cost of Minimum seam : 1.73034025375
Width of the image is: 785


Run seam_carving
Cost of Minimum seam : 1.83980199923
Width of the image is: 784
Cost of Minimum seam : 1.89028066128
Width of the image is: 783
Cost of Minimum seam : 1.93260284506
Width of the image is: 782
Cost of Minimum seam : 1.93723183391
Width of the image is: 781
Cost of Minimum seam : 1.90591887735
Width of the image is: 780
Cost of Minimum seam : 1.85212418301
Width of the image is: 779
Cost of Minimum seam : 1.93956939639
Width of the image is: 778
Cost of Minimum seam : 1.94748173779
Width of the image is: 777
Cost of Minimum seam : 1.84502883506
Width of the image is: 776
Cost of Minimum seam : 1.9396366782
Width of the image is: 775
Cost of Minimum seam : 1.95614571319
Width of the image is: 774
Cost of Minimum seam : 1.99808919646
Width of the image is: 773
Cost of Minimum seam : 1.96652825836
Width of the image is: 772
Cost of Minimum seam : 1.98024798155
Width of the image is: 771
Cost of Minimum seam : 1.99429257978
Width of the image is: 770
Cost of Minimum seam : 2.0195751634
Width of the image is: 769
Cost of Minimum seam : 2.06176086121
Width of the image is: 768
Cost of Minimum seam : 1.91706074587
Width of the image is: 767



4. Source Code

```
1   from pylab import *
2   from skimage import img_as_float
3   from skimage.filters import sobel_h, sobel_v
4   from numpy import *
5
6
7   def dual_gradient_energy(img):
8       R = img[:, :, 0]
9       G = img[:, :, 1]
10      B = img[:, :, 2]
11      return sobel_h(R)**2 + sobel_v(R)**2 + sobel_h(G)**2 + \
12          sobel_v(G)**2 + sobel_h(B)**2 + sobel_v(B)**2
13
14
15  def find_seam(img):
16      x = dual_gradient_energy(img)
17      x = x[1:-1, 1:-1]
18      w = len(x[0])
19      h = len(x)
20      newarr = zeros((h, w))
21      index = zeros((h, w), dtype=int32)
22      seam = zeros(len(x), dtype=int32)
23
24      for i in range(0, h):
25          for j in range(0, w):
```

```
            if i == 0:
                newarr[i][j] = x[i][j]
                index[i][j] = 0
            else:
                newarr[i][j] = Infinity
                index[i][j] = -1

    for i in range(1, h):
        for j in range(0, w):
            shortest = Infinity
            Imin = -1
            if j is not 0:
                if newarr[i-1][j-1] < shortest:
                    shortest = newarr[i-1][j-1]
                    Imin = j-1
            if newarr[i-1][j] < shortest:
                shortest = newarr[i-1][j]
                Imin = j

            if j != w-1:
                if newarr[i-1][j+1] < shortest:
                    shortest = newarr[i-1][j+1]
                    Imin = j+1

            newarr[i][j] = shortest + x[i][j]
            index[i][j] = Imin

    min = Infinity
    tmin = -1

    for j in range(0, w):
        shortest = newarr[h-1][j]
        if shortest < min:
            min = shortest
            tmin = j

    seam = zeros(h, dtype=int32)
    seam[h-1] = tmin
    for i in range(h-2, -1, -1):
        seam[i] = index[i+1][seam[i+1]]
    seam = insert(seam, 0, seam[0])
    seam = append(seam, seam[h-1])
    print "Cost of Minimum seam :", min
    return seam


def plot_seam(img, seam):
    x = dual_gradient_energy(img)
    s = []
    for i in range(0, len(x)):
```

```
        s.append((seam[i], i))
    plt.tight_layout()
    plt.plot(*zip(*s), color='r')
    plt.imshow(x)


def remove_seam(img, seam):
    img = img_as_float(img)
    img = img.tolist()
    seam = seam.tolist()
    for i in range(0, len(img)):
        del img[i][seam[i]]
    plt.imshow(img)
    print "Width of the image is: ", len(img[0])
    return img


def remove_multiple_pixels(img, count):
    for i in range(0, count):
        img = remove_seam(img, find_seam(img))
        img = img_as_float(img)
    plt.imshow(img)


if __name__ == '__main__':
    img = imread("test1.png")
    img = img_as_float(img)
    print " Removing only 1 seam..."
    seam1 = find_seam(img)
    figure()
    gray()
    subplot(1, 5, 1)
    imshow(img)
    title("Orignal")
    subplot(1, 5, 2)
    imshow(dual_gradient_energy(img))
    title("Gradient")
    subplot(1, 5, 3)
    plot_seam(img, seam1)
    title("Seam Plotting")
    subplot(1, 5, 4)
    remove_seam(img, seam1)
    title("1 seam removed")
    print "Now Removing 50 seam..."
    subplot(1, 5, 5)
    remove_multiple_pixels(img, 50)
    title("50 seams removed")
    plt.imsave("carved.jpg", img)
    show()
```

Revised rubric for coding assignments.

This is a 5-point rubric for coding projects. Graders should refrain from using fractional points (they are a pain to defend), choose the one one number that best reflects the assignment. For assignments with multiple parts, choose the lowest scoring part.

This rubric is based on the idea that students submit PDF write-ups with their coding assignment. Write-ups *must* be PDF's with the source code so that graders can quickly view them annotate them using blackboard. The rubric does not address specific learning objectives — the assumption is that by completing the assignment the student has implicitly demonstrated some set objectives in addition to coding.

0 points — Student does not submit **all** parts of the assignment, meaning *both* a **PDF** writeup (all sections) that includes source code and output of testing, as well as a .**zip** file with source code.

2 points — The code does not run or does not *appear* to be able to run. The code it much longer than it should be, or does not appear to follow the scaffolding provided. The grader can but **does not have to verify that it does not run**, it is the student's responsibility to provide a writeup that is sufficiently convincing. Student may not appeal by coming after the fact and showing that code runs on their machine.
When grading, the grader should indicate portions of the code by annotating the writeup that are suspicious.

3 points — The code runs or looks like it would run, but the student has not shown via their writeup that it produces the correct result on reasonable inputs. Or, the student has implemented algorithms using approaches other than the ones indicated in the assignment, or the implementation has the wrong asymptotic complexity or that demonstrates a lack of understanding of the assignments objectives. The grader can, but **does not have to run the code to verify correctness** — it is the student's responsibility to make a convincing case that the output and the algorithm is correct.
When grading, the grader should indicate by annotating the write-up where results

4 points — The code runs or appears to run correctly, but has readability or style issues. The student has not demonstrated that their code has passed style guidelines, or the student's implementation appears to be unnecessarily complex (even though it looks like it works).
When grading, the grader should indicate the style problems.

5 points — No issues that we can spot.