Tables

10W:7H

| | NP | DP |
|---|---|---|
| 1 | 0.03543 | 0.00199 |
| 2 | 0.03690 | 0.00099 |
| 3 | 0.03416 | 0.00100 |
| Average | 0.03550 | 0.00133 |

8W:8H

| | NP | DP |
|---|---|---|
| 1 | 0.10272 | 0.00099 |
| 2 | 0.07730 | 0.00099 |
| 3 | 0.11768 | 0.00199 |
| Average | 0.09923 | 0.00132 |

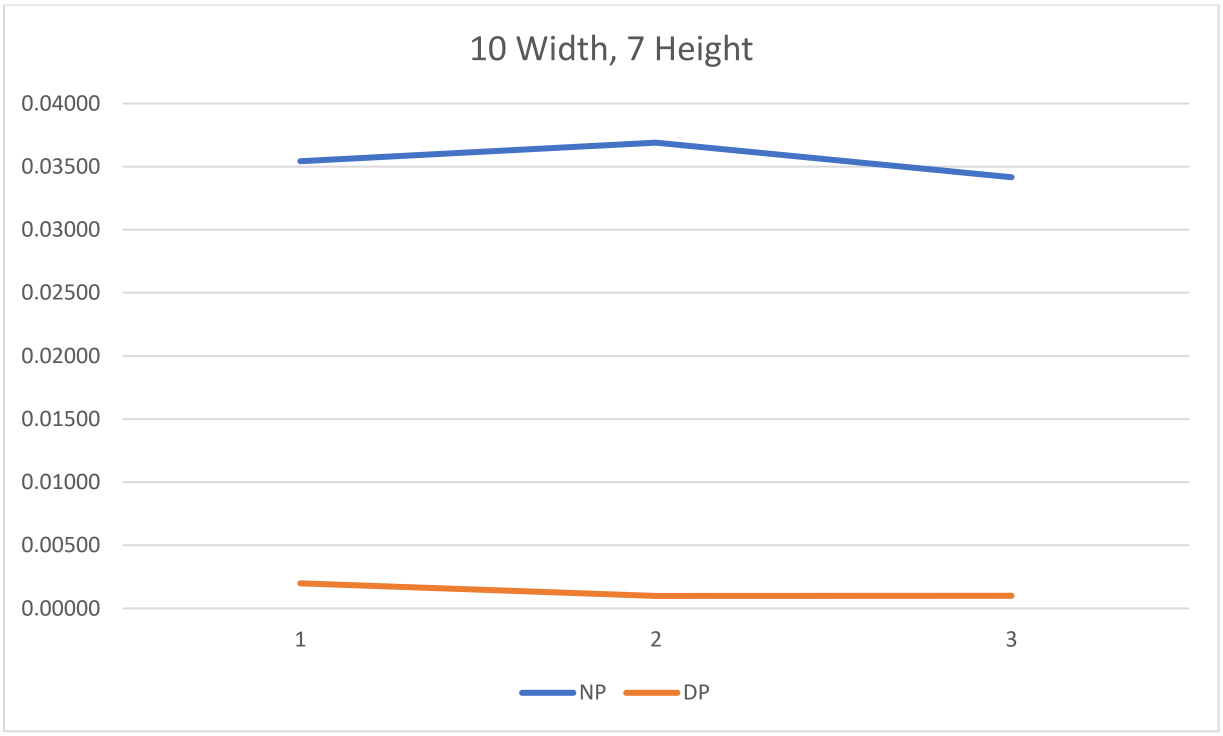
10W:10H

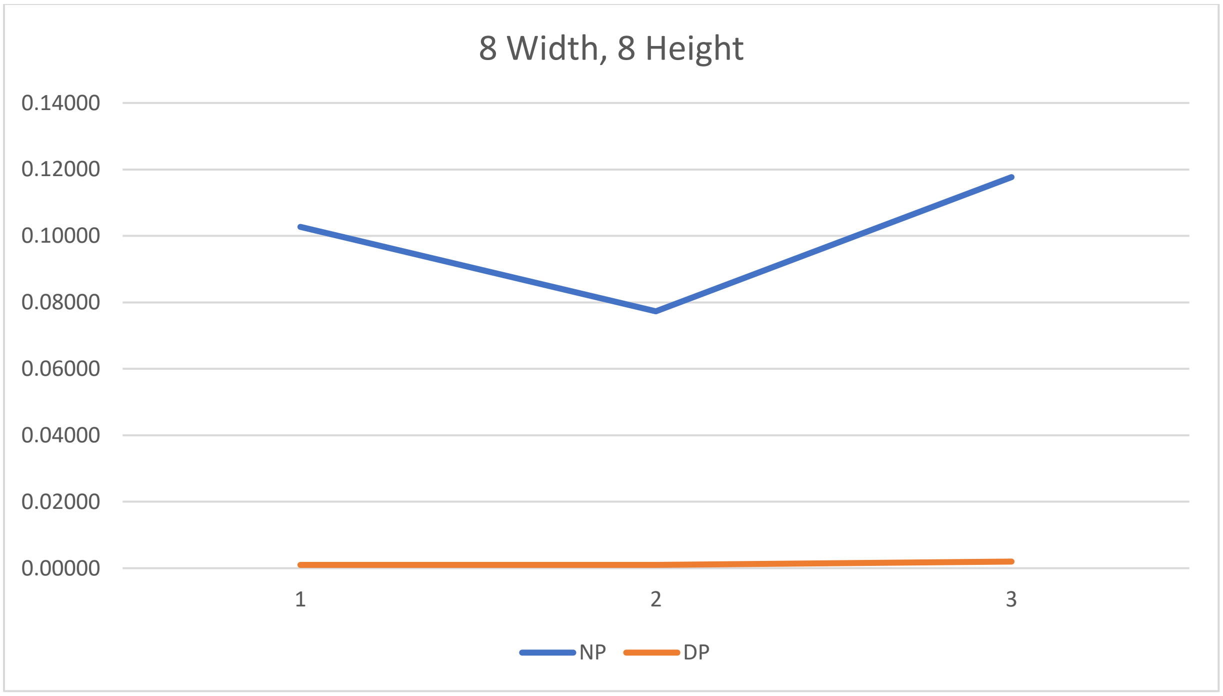
| | NP | DP |
|---|---|---|
| 1 | 1.00900 | 0.00099 |
| 2 | 0.88957 | 0.00099 |
| 3 | 0.92283 | 0.00099 |
| Average | 0.94047 | 0.00099 |

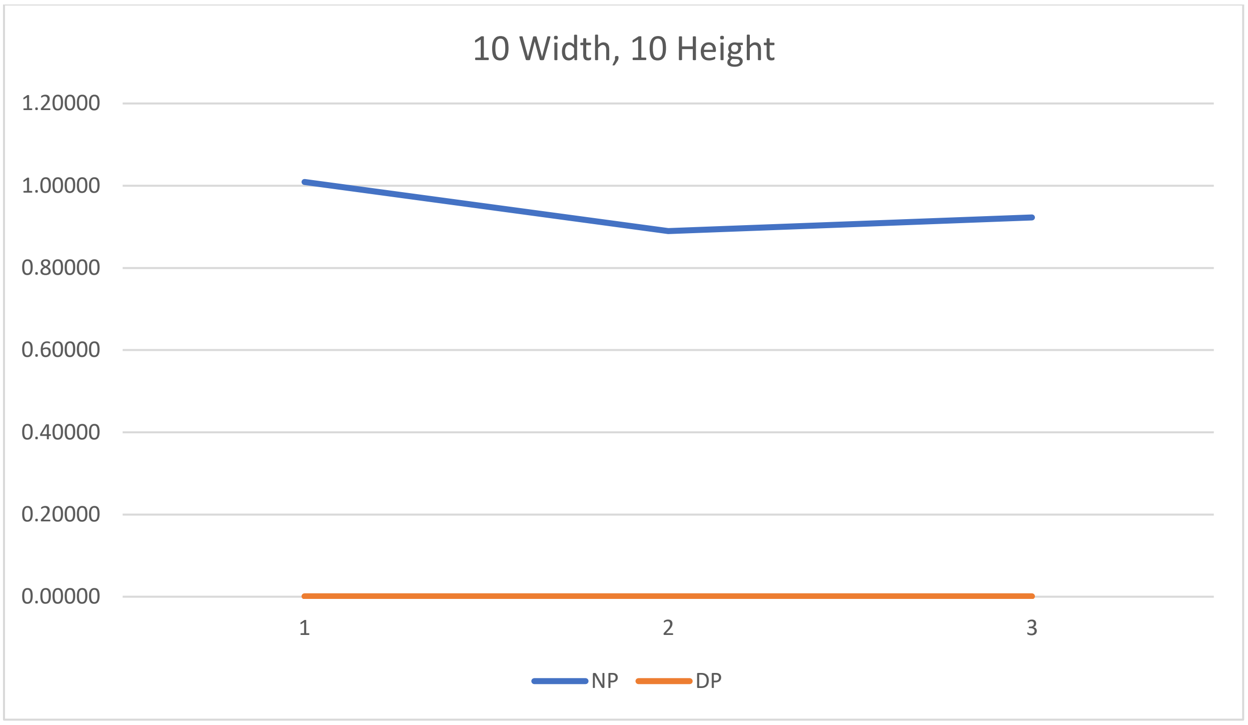
12W:12H

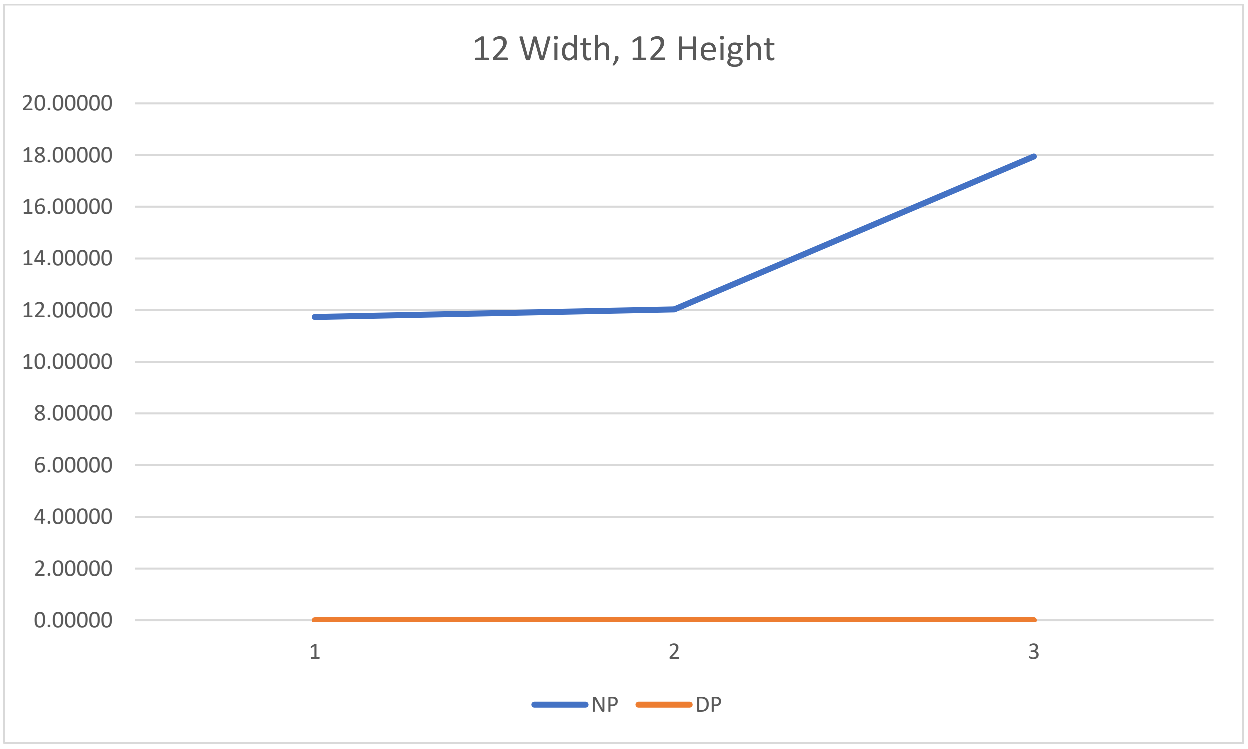
| | NP | DP |
|---|---|---|
| 1 | 11.73538 | 0.00099 |
| 2 | 12.03426 | 0.00099 |
| 3 | 17.94492 | 0.00299 |
| Average | 13.90485 | 0.00166 |

Graphs


10 Width, 7 Height
0.04000
0.03500
0.03000
0.02500
0.02000
0.01500
0.01000
0.00500
0.00000
1
2
3
NP
DP


8 Width, 8 Height
0.14000
0.12000
0.10000
0.08000
0.06000
0.04000
0.02000
0.00000
1
2
3
NP
DP


10 Width, 10 Height
1.20000
1.00000
0.80000
0.60000
0.40000
0.20000
0.00000
1
2
3
NP
DP


12 Width, 12 Height
20.00000
18.00000
16.00000
14.00000
12.00000
10.00000
8.00000
6.00000
4.00000
2.00000
0.00000
1
2
3
NP
DP

## Analysis

The naïve approach uses a recursive solution to check every possible seam that could be created and then find the one with the lowest possible energy. While doing the benchmarking, I noticed the number of recursions were so great, that the memory on my computer could do not computations on large or even relatively small pictures. We created 4 randomly generated pictures each with their own pixel sizes, with the max of 12 pixels both in height and width.

The naïve approach would always grow exponentially and considering that it will compute O($n^3$). It's exponential growth and the way it finds and stores all seams, shows this to be true. For the dynamic programming approach, the time didn't always grow or change but it does not mean it is on constant time. The algorithm follows that each pixel will store the lowest possible energy it can store and the location of the index that has the lowest energy that is also it's neighbor. This shows that the dynamic approach is O(n). It goes to each pixel and stores the lowest energy from the neighbor, only going through n possibilities.

The recurrence relation of both implementations follows three elements: initialization, maintenance and termination. For the naïve approach, initialization is followed by gathering the location of the current pixel, the energy it contains and the path before it. The recurrence continues finding the next pixel in its path. The call stack stops when the current recurrence has a pixel location that is at the bottom of the picture. The current path is then added to a list of seams and then the recurrent function goes back in it's call stack to go to the next pixel to create a seam. The termination process is complete when all possible seams have been found and the seam with the smallest energy is chosen.

The recurrence relation of the dynamic programming approach initializes by starting at the top left corner of the picture and finding the lowest potential energy. Once it does that, it will go through each pixel finding the lowest potential energy from its neighbors. It terminates once it reached every pixel and find the smallest potential energy of a seam.