Deepam Sarmah
2020050
deepam20050@iiitd.ac.in

GPU Code Design:

- I created blocks each consisting of 32 threads by 32 threads (in x and y direction i.e, a 2D block). Each of these threads would perform computation of a single (x, y) of the image in a similar manner to the CPU code given.
- I create a grid consisting of N / block.x by N / block.y blocks (here block.x and block.y equals 32). I do this so that the total number of threads in x direction and total number of threads in y direction are equal to the number of locations of x and number of locations of y respectively corresponding to the CPU code.
- In the kernel launch, each thread that executes first computes the location (x, y) using the blockDim, blockIdx and threadIdx for the respective x and y coordinates. After that it runs the code similar to finding the number of iterations for a given (x, y) same as in the CPU code. The only difference being that I use the CUDA library's thrust::complex library instead of std::complex.
- I added the qualifiers __host__ __device__ to HSVtoRGB() function so that both the CPU and GPU can call the HSVtoRGB() function.
- For the compare_CPU_GPU():
    - I used the method of total sum of squares (SST) and explained sum of squares (SSE) which I learnt in the Econometrics course to find the sum of squared errors.
    - If y_i denotes the CPU time of i[th] element of the CPU image, $\bar{y}$ denotes the mean of CPU times and $\hat{y}_i$ denotes the GPU time of i[th] element of the GPU image then $SST = \sum_{i=1}^{nelem} (y_i - \bar{y})^2$ and $SSE = \sum_{i=1}^{nelem} (\hat{y}_i - \bar{y})^2$. Then for a good metric I decided that $\frac{SSE}{SST} \geq 99\%$ i.e, the accuracy between the values should be greater than or equal to 99%. This way I could avoid the precision errors of CPU and GPU time.

Results:

N	CPU (in ms)	GPU (in ms)
256	53.24099	0.287936
512	207.472443	0.639904
1024	829.247314	1.753152
2048	3290.394531	5.975808
4096	13133.09766	23.1744

Images are in the format:
<X> <N> Julia 0.285 + i0.010.png
Where X = 0 if CPU image and X = 1 if GPU image
And N = 256, 512, 1024, 2048, 4096

All result images are in the .zip file

1. N = 256

```
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$ ./Assignment01
Usage: ./Assignment01 <real> <imag>
Where <real> and <imag> form the complex seed for the Julia set.
Performing Julia set computation on CPU... done in 53.240990 milliseconds.
Performing Julia set computation on GPU... done in 0.287936 milliseconds.
CPU-GPU results do match!
Image saved as: 0 256 Julia 0.285 + i0.010.png
Image saved as: 1 256 Julia 0.285 + i0.010.png
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$
```

2. N = 512

```
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$ ./Assignment01
Usage: ./Assignment01 <real> <imag>
Where <real> and <imag> form the complex seed for the Julia set.
Performing Julia set computation on CPU... done in 207.472443 milliseconds.
Performing Julia set computation on GPU... done in 0.639904 milliseconds.
CPU-GPU results do match!
Image saved as: 0 512 Julia 0.285 + i0.010.png
Image saved as: 1 512 Julia 0.285 + i0.010.png
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$
```

3. N = 1024

```
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$ ./Assignment01
Usage: ./Assignment01 <real> <imag>
Where <real> and <imag> form the complex seed for the Julia set.
Performing Julia set computation on CPU... done in 829.247314 milliseconds.
Performing Julia set computation on GPU... done in 1.753152 milliseconds.
CPU-GPU results do match!
Image saved as: 0 1024 Julia 0.285 + i0.010.png
Image saved as: 1 1024 Julia 0.285 + i0.010.png
deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$
```

4. N = 2048

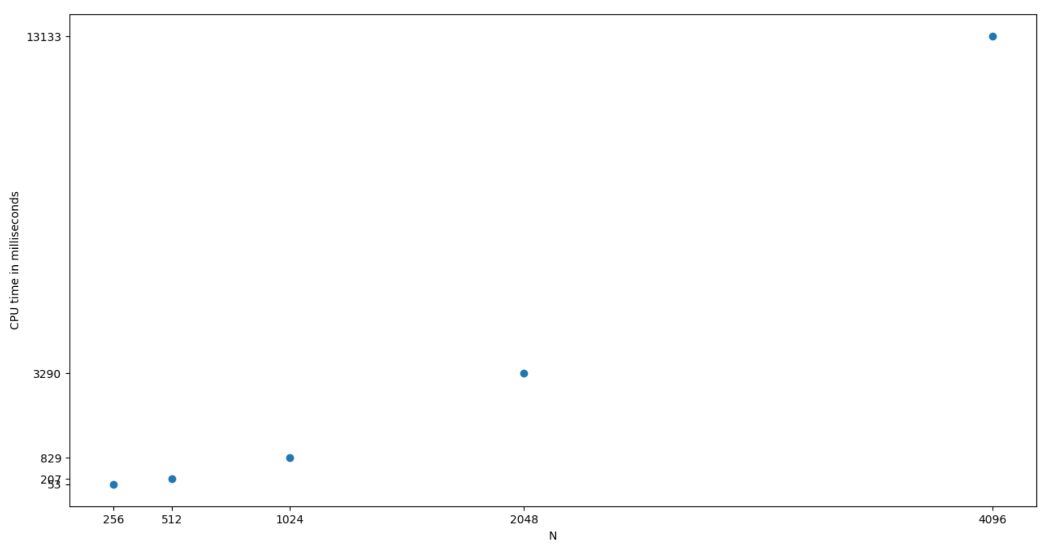
```
● deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$ ./Assignment01
  Usage: ./Assignment01 <real> <imag>
  Where <real> and <imag> form the complex seed for the Julia set.
  Performing Julia set computation on CPU... done in 3290.394531 milliseconds.
  Performing Julia set computation on GPU... done in 5.975808 milliseconds.
  CPU-GPU results do match!
  Image saved as: 0 2048 Julia 0.285 + i0.010.png
  Image saved as: 1 2048 Julia 0.285 + i0.010.png
○ deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$
```

5. N = 4096

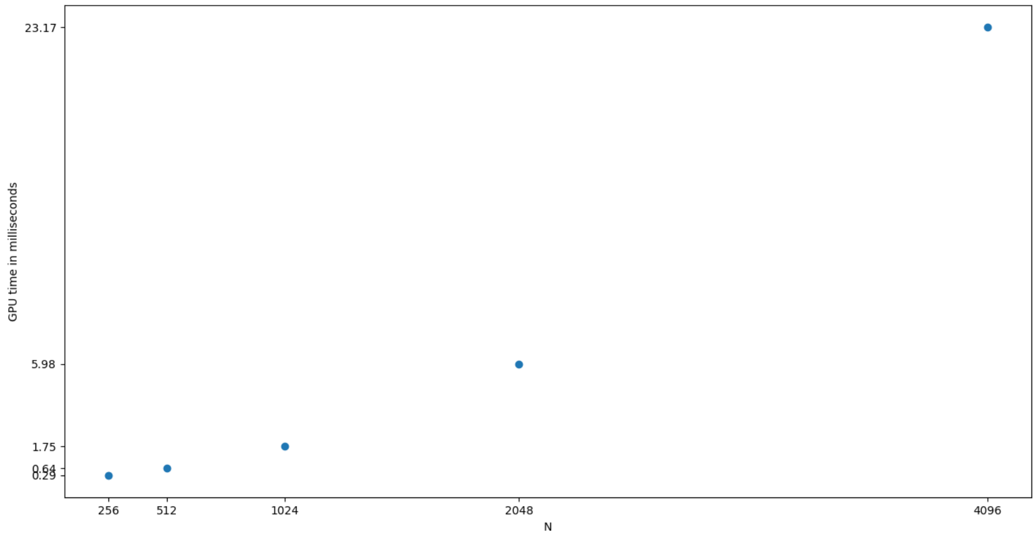
```
● deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$ ./Assignment01
  Usage: ./Assignment01 <real> <imag>
  Where <real> and <imag> form the complex seed for the Julia set.
  Performing Julia set computation on CPU... done in 13133.097656 milliseconds.
  Performing Julia set computation on GPU... done in 23.174400 milliseconds.
  CPU-GPU results do match!
  Image saved as: 0 4096 Julia 0.285 + i0.010.png
  Image saved as: 1 4096 Julia 0.285 + i0.010.png
○ deepam20050@newton:~/Desktop/Assignment 1/Assignment01_code$
```

Plots:

- CPU time (CPU timings plot.png)



- GPU time (GPU timings plot.png)



References:

1. https://thrust.github.io/doc/group__complex__numbers.html