**고급 소프트웨어 실습 12주차 보고서**

2반 20171617 김소연

**실습**

**BLOCK SIZE = 16**

  
Nf = 1

  
Nf = 4

  
Nf = 16

  
Nf = 64

  
Nf = 256

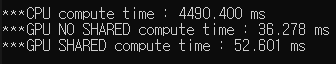
  
Nf = 1024

**BLOCK SIZE = 32**

  
Nf = 1

  
Nf = 4

  
Nf = 16

  
Nf = 64

  
Nf = 256

  
Nf = 1024

**BLOCK SIZE = 128**

  
Nf = 1

  
Nf = 4

  
Nf = 16

  
Nf = 64

  
Nf = 256

  
Nf = 1024

**실행결과 정리 (단위 : milli seconds)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nf | CPU | NO SHARED | SHARED | CPU/NO\_SHARED | CPU/SHARED |
| 1 | 287.59 | 14.43 | 16.84 | 19.93 | 17.07 |
| 4 | 408.593 | 16.802 | 22.224 | 24.31 | 18.38 |
| 16 | 158.927 | 26.475 | 34.669 | 6 | 4.58 |
| 64 | 4480.637 | 68.514 | 93.891 | 65.39 | 47.72 |
| 256 | 16609.518 | 190.5 | 356.379 | 87.18 | 46.6 |
| 1024 | 65307.387 | 597.49 | 1618.717 | 109.3 | 40.34 |

BLOCK\_SIZE = 16

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nf | CPU | NO SHARED | SHARED | CPU/NO\_SHARED | CPU/SHARED |
| 1 | 321.585 | 7.473 | 11.351 | 43.03 | 28.33 |
| 4 | 419.471 | 7.36 | 13.468 | 56.99 | 31.14 |
| 16 | 1145.034 | 13.189 | 15.898 | 86.81 | 72.02 |
| 64 | 4490.4 | 36.278 | 52.601 | 123.77 | 85.36 |
| 256 | 16593.957 | 114.424 | 180.844 | 145.02 | 91.75 |
| 1024 | 65203.348 | 304.617 | 705.825 | 214.05 | 92.37 |

BLOCK\_SIZE = 32

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nf | CPU | NO SHARED | SHARED | CPU/NO\_SHARED | CPU/SHARED |
| 1 | 317.711 | 3.456 | 5.853 | 91.93 | 54.28 |
| 4 | 416.161 | 3.889 | 5.619 | 107 | 74.06 |
| 16 | 1143.106 | 6.919 | 9.667 | 165.21 | 118.24 |
| 64 | 4480.938 | 23.522 | 24.123 | 190.49 | 185.75 |
| 256 | 16609.723 | 91.067 | 86.11 | 182.39 | 192.88 |
| 1024 | 65284.844 | 336.946 | 342.469 | 193.75 | 190.62 |

BLOCK SIZE = 128

모든 BLOCK SIZE와 Nf의 크기에 대하여 GPU를 이용한 결과가 CPU를 이용한 결과보다 그 속도가 빠른 것을 확인할 수 있다.

CPU에서 NOSHARED의 실행시간을 나눈결과와 CPU에서 SHARED의 실행시간을 나눈 결과로 보았을 때, Nf가 작을수록 SHARED의 실행 속도보다 NO SHARED의 실행속도가 빠르나, Nf가 커질수록 SHARED의 실행속도가 NO SHRED의 실행속도보다 빨라지는 것을 확인할 수 있다.

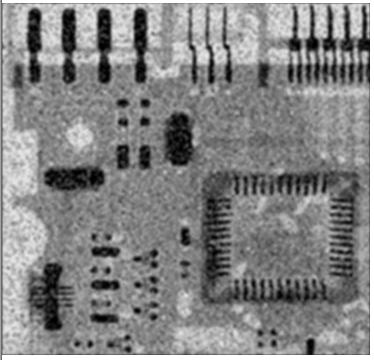
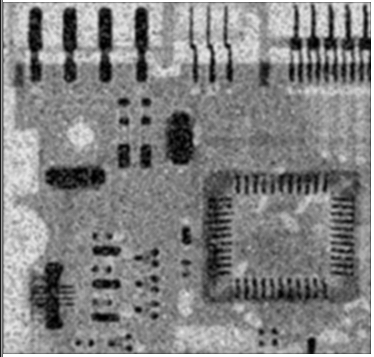
이는 SHARED에서 nf가 클수록 buffer에 접근하는 횟수가 더 많아지기 때문에, NO SHARED에 직접 data에 접근하는 것 보다 시간을 많이 아낄 수 있었던 것으로 보인다. Nf가 작은 경우는 buffer에 data를 올리는 시간 대비 사용하는 시간에 대하여 효율이 좋지 않기 때문에 오히려 SHARED가 NO SHARED보다 오래걸린 것으로 이해할 수 있다.

또한 BLOCK SIZE가 커질수록 SHAREd와 NO SHARED의 속도가 둘 다 빨라지는 것을 확인할 수 있는데, 이는 본인의 GPU에서 128개의 블락으로 나누는 것이 가장 효율이 높기 때문인 것으로 보인다. 하지만, 이전 실습으로 미루어보았을 때 BLOCK SIZE가 계속 커진다고 해서 효율이 계속 좋아지진 않을 것이다.

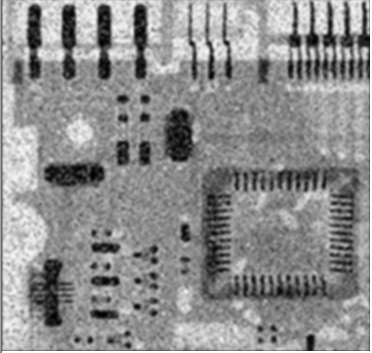
**숙제**

2주차 실습에서 진행하였던 Gaussian filtering을 CUDA를 이용해 GPU 위에서 실행할 수 있도록 함수를 구현해주었다. 이 때, thread간에 Buffer를 공유하는 Shared version의 코드와, 공유하지 않는 No Shared version을 함께 구현하여, CPU, GPU-shared, GPU-No shared의 3가지 버전으로 실행하였을 때 그 속도의 차이를 살펴보고자 하였다.

BLOCK SIZE를 바꾸어가며 그 속도의 차이를 관찰하였는데, 그에 따른 결과 이미지는 모두 같았고, 이는 아래와 같다.

**◀GPU - CUDA No shared, GPU - CUDA Shared Buffer▶**  


**CPU**



BLOCK SIZE를 달리하였을 때의 속도의 차이는 아래와 같았다.

**단위: milli seconds**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | BLOCK SIZE | 1 | 4 | 16 | 32 | 44 | 88 |
| GPU | SHARED | 11 | 11 | 12 | 12 | 12 | 11 |
|  | NO\_SHARED | 14 | 13 | 13 | 13 | 14 | 13 |
| CPU |  | 16 | 16 | 16 | 16 | 16 | 16 |

※BLOCK SIZE가 2의 제곱수 때 GPU의 BLOCK을 나누는 것이 좀 더 효율적이라고 생각하나, 이미지의 height가 32 \* 11이었기 때문에, 그 약수로 BLOCK SIZE를 나누어주었다.

모든 BLOCK SIZE에 대하여, GPU를 이용한 Gaussian Filtering의 연산이 CPU를 이용한 Gaussian Filtering보다 빠른 것을 확인할 수 있었다. 또한, Buffer를 이용해 한 Block 내에서의 thread끼리 데이터를 공유하여 연산한 SHARED version이, 각각의 thread별로 데이터에 직접 접근한 No SHARED version보다 조금 더 빠르거나 비슷한 속도를 보이는 것을 확인할 수 있다.