고급소프트웨어실습1

Lecture 12 과제

반 : 4

학번 : 20171669

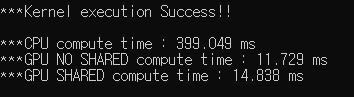
이름 : 이재영

1. 실습
2. 실습1

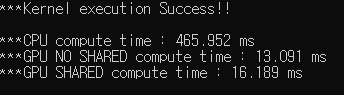
**두 프로그램 GPU-Code-NO-SM와 GPU-Code-SM의 성능 차이를 분석하기 위하여, 주어진 쓰레드 블럭 크기에 대하여 중복성을 의미하는 nf 값을 변화시켜가면서, 두 버전의 시간을 측정한 후 왜 그러한 결과/차이가 산출되었는지 분석하라.**

<Block Size = 16>

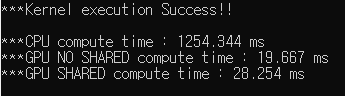
Nf = 1



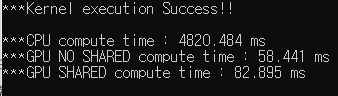
Nf = 4



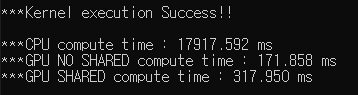
Nf = 16



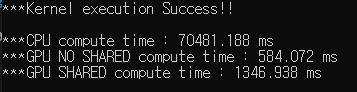
Nf = 64



Nf = 256

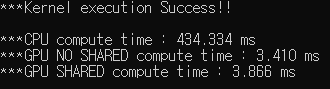


Nf = 1024

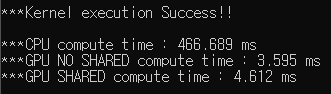


<Block Size = 64>

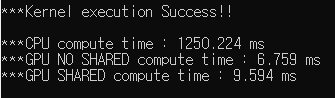
Nf = 1



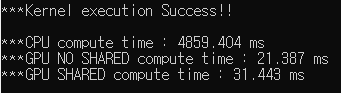
Nf = 4



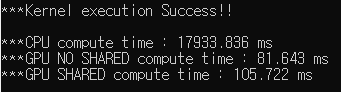
Nf = 16



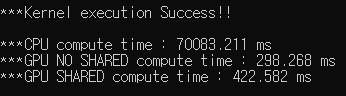
Nf = 64



Nf = 256



Nf = 1024

****

<Block Size = 16>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 4 | 16 | 64 | 256 | 1024 |
| CPU | 399.049 | 465.952 | 1254.344 | 4820.484 | 17917.592 | 70481.188 |
| NO SHARED | 11.729 | 13.091 | 19.667 | 58.441 | 171.858 | 584.072 |
| SHARED | 14.838 | 16.189 | 28.254 | 82.895 | 317.950 | 1346.938 |

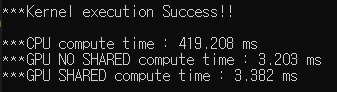
<Block Size = 64>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 4 | 16 | 64 | 256 | 1024 |
| CPU | 434.334 | 466.689 | 1250.224 | 4859.404 | 17933.836 | 70083.211 |
| NO SHARED | 3.410 | 3.595 | 6.759 | 21.387 | 81.643 | 299.268 |
| SHARED | 3.866 | 4.612 | 9.594 | 31.443 | 105.722 | 422.582 |

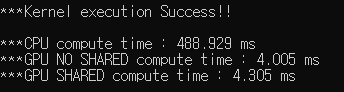
* Block Size를 16과 64를 분석해본 결과 일정한 Block size까지는 Block Size가 커질수록 전체적인 CPU/GPU의 수행속도가 빨라진 것을 확인할 수 있다. 그리고 각각의 같은 Block size에서는 GPU가 memory를 SHARED, NO SHARED 하는 경우를 확인해보면 위의 실험결과에서는 SHARED가 느린 것을 확인할 수 있는데 이는 실험과정의 오류가 아니라 kernel영역에서 연산을 하는 경우 예외 처리를 위한 for문을 사용해서 나타난 결과로 for문을 없애면서 kernel을 최적화하면 SHARED가 빠르다는 것을 알 수 있다.
* 실습1 – 최적화

<Block Size = 1024>

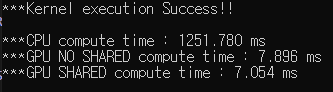
Nf = 1



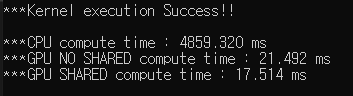
Nf = 4



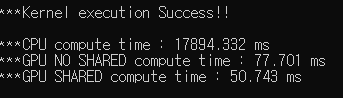
Nf = 16



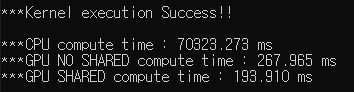
Nf = 64



Nf = 256



Nf = 1024



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 4 | 16 | 64 | 256 | 1024 |
| CPU | 419.208 | 488.929 | 1251.780 | 4859.320 | 17894.332 | 70323.273 |
| NO SHARED | 3.203 | 4.005 | 7.896 | 21.492 | 77.701 | 267.965 |
| SHARED | 3.382 | 4.305 | 7.054 | 17.514 | 50.743 | 193.910 |

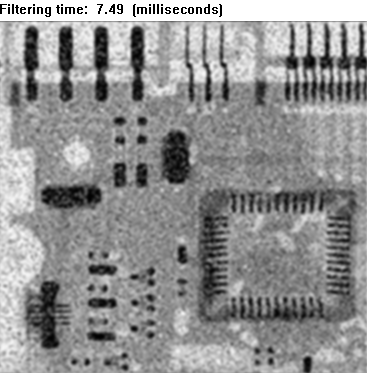
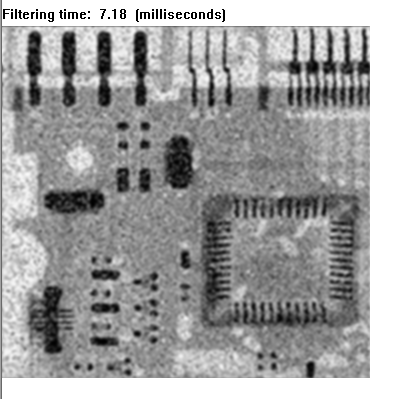
* kernel에서 for문을 사용한 것을 for문을 없애서 최적화를 하고 Block Size를 1024로 한 결과를 보여준다. 표를 살펴보면 Nf가 1, 4인 경우는 buffer에 data를 올리는 시간의 효율이 안 좋기 때문에 SHARED가 NO SHARED보다 느리다는 것을 볼 수 있고 아니라면 SHARED가 NO SHARED가 빠르다는 것을 확인할 수 있다.

1. 숙제

* **이차원 영상을 읽어 들인 후 주어진 크기의 Gaussian filter를 적용한 영상을 출력해주는 프로그램을 작성하라. 특히 아래에 기술한 세 가지 형태의 프로그램을 작성한 후 다양한 쓰레드 블럭 크기에 따른 수행 시간을 비교하라.**

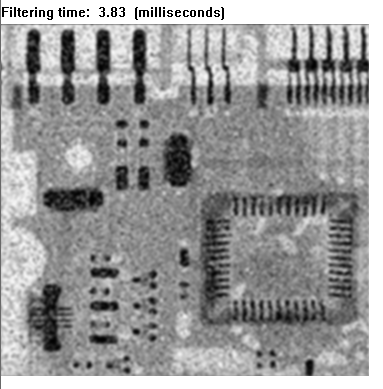
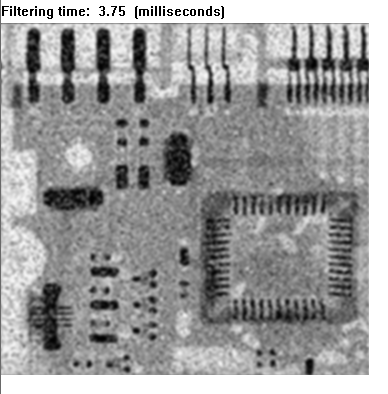
<Block Size 16>

* No Shared / Shared



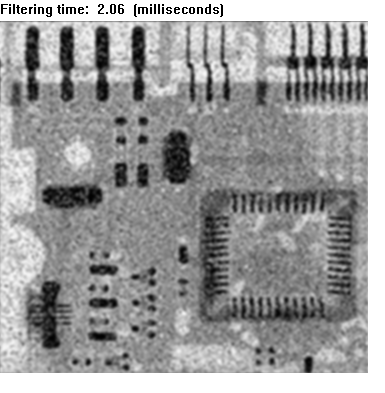
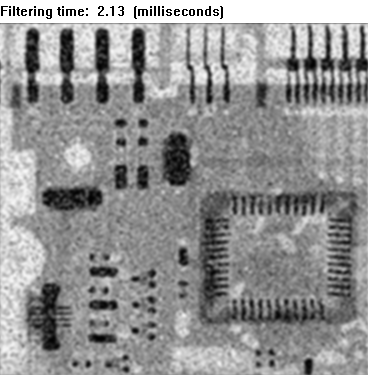
<Block Size 32>

* No Shared / Shared



<Block Size 64>

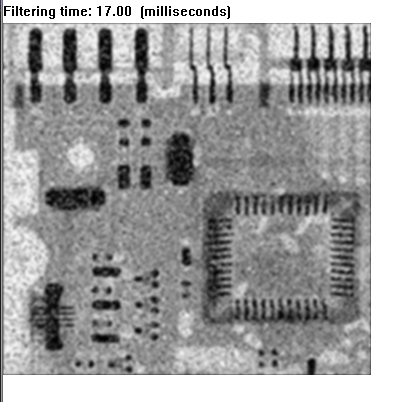
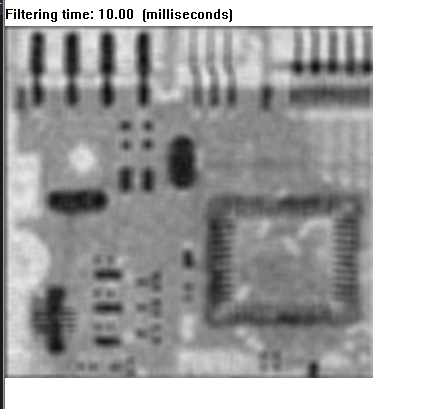
* No Shared / Shared



* Block size가 커질수록 연산속도가 줄어드는 것을 확인할 수 있고 Block size가 커질수록 연산이 빨라지는 정도가 낮아지는 것을 확인할 수 있다. 실습1에서의 방법처럼 kernel에서 최적화한 결과 Block Size가 64인 경우에는 SHARED가 빠른 것을 확인할 수 있고 더 작은 경우는 추가적인 최적화 방법을 구현한다면 SHARED가 빠르도록 보일 수 있다.

|  |  |  |
| --- | --- | --- |
| Block Size | NO SHARED | SHARED |
| 16 | 7.18 ms | 7.49 ms |
| 32 | 3.75 ms | 3.83 ms |
| 64 | 2.13 ms | 2.06 ms |

* GaussianFilteringCPU / Blur

****

* GPU를 사용한 Gaussian Filtering의 연산이 CPU를 사용한 경우보다 빠르다는 것을 확인할 수 있다.