Performance Analysis Report

Introduction

This report presents the results and analysis of the CUDA convolution program implemented with two memory management techniques: **Global Memory** and **Shared Memory**. The primary objective of this study was to evaluate and compare the performance of these implementations (Q1 and Q2) under varying conditions, including different grid shapes, block sizes, and image dimensions. The experiments involved timing the execution for each configuration, generating performance graphs, and visualizing the input and output images for selected cases.

Experimental Setup

Parameters

- Input Images:
  - `360x360.png`
  - `720x720.png`
  - `1200x1200.png`
- **Block Sizes:** 16, 32, 64
- Grid Shapes:
  - 10x10 (Square)
  - 20x10 (Rectangular)
  - 15x30 (Vertical Rectangular)
  - 40x40 (Large Square)
- **Repetitions:** Each test was executed multiple times, and the average execution time was recorded.

Execution Environment

- CUDA-enabled GPU
- The shared memory limit was queried using the CUDA runtime API to ensure configurations did not exceed the hardware limitations.

Results and Observations

Input and Output Images

Selected input images and their processed outputs are included for visualization. These demonstrate the edge detection operation performed by the convolution program using a predefined kernel.

Performance Graphs

Graphs were plotted for each image size, block size, and grid shape. Additionally, an average performance graph was generated to provide an overview of the execution times across all configurations.

Observations:

1. **Global vs. Shared Memory:**
    - For smaller images (`360x360`), **Global Memory** slightly outperformed **Shared Memory** on average, demonstrating marginally lower execution times.
    - For larger images (`1200x1200`), **Shared Memory** showed better average performance, particularly in configurations with smaller block sizes, due to more efficient memory usage.
2. **Impact of Block Sizes:**
    - Increasing block sizes generally reduced the execution time, as larger blocks allow for better utilization of the GPU's computational resources. However, diminishing returns were observed beyond a block size of 32, particularly for smaller images.
3. **Grid Shapes:**
    - Square grids (e.g., 10x10, 40x40) exhibited better performance consistency compared to rectangular grids (e.g., 15x30, 20x10), likely due to more balanced thread allocation.
4. **Shared Memory Limitations:**
    - Configurations exceeding the available shared memory size led to increased execution times, as indicated in the graphs. This was particularly evident for the `1200x1200` image with block size 64.

Detailed Performance Analysis

Execution Times

The table below summarizes the execution times for selected configurations:

Image	Mode	Block Size	Grid Shape	Execution Time
1200x1200.png	Global	16	10x10	0.354594
1200x1200.png	Shared	16	10x10	0.318115
1200x1200.png	Global	16	20x10	0.384134
1200x1200.png	Shared	16	20x10	0.392553
1200x1200.png	Global	16	15x30	0.388721
1200x1200.png	Shared	16	15x30	0.381481
1200x1200.png	Global	16	40x40	0.448429

Image	Mode	Block Size	Grid Shape	Execution Time
1200x1200.png	Shared	16	40x40	0.444042
1200x1200.png	Global	32	10x10	0.407794
1200x1200.png	Shared	32	10x10	0.419043
1200x1200.png	Global	32	20x10	0.407055
1200x1200.png	Shared	32	20x10	0.416744
1200x1200.png	Global	32	15x30	0.449242
1200x1200.png	Shared	32	15x30	0.438928
1200x1200.png	Global	32	40x40	0.578744
1200x1200.png	Shared	32	40x40	0.587728
1200x1200.png	Global	64	10x10	0.330577
1200x1200.png	Shared	64	10x10	0.33407
1200x1200.png	Global	64	20x10	0.335946
1200x1200.png	Shared	64	20x10	0.325879
1200x1200.png	Global	64	15x30	0.31959
1200x1200.png	Shared	64	15x30	0.329105
1200x1200.png	Global	64	40x40	0.350522
1200x1200.png	Shared	64	40x40	0.340536
720x720.png	Global	16	10x10	0.184352
720x720.png	Shared	16	10x10	0.197168
720x720.png	Global	16	20x10	0.215853
720x720.png	Shared	16	20x10	0.211938
720x720.png	Global	16	15x30	0.235018
720x720.png	Shared	16	15x30	0.239089
720x720.png	Global	16	40x40	0.292546
720x720.png	Shared	16	40x40	0.292703
720x720.png	Global	32	10x10	0.213173
720x720.png	Shared	32	10x10	0.224618
720x720.png	Global	32	20x10	0.234298
720x720.png	Shared	32	20x10	0.253632
720x720.png	Global	32	15x30	0.26872

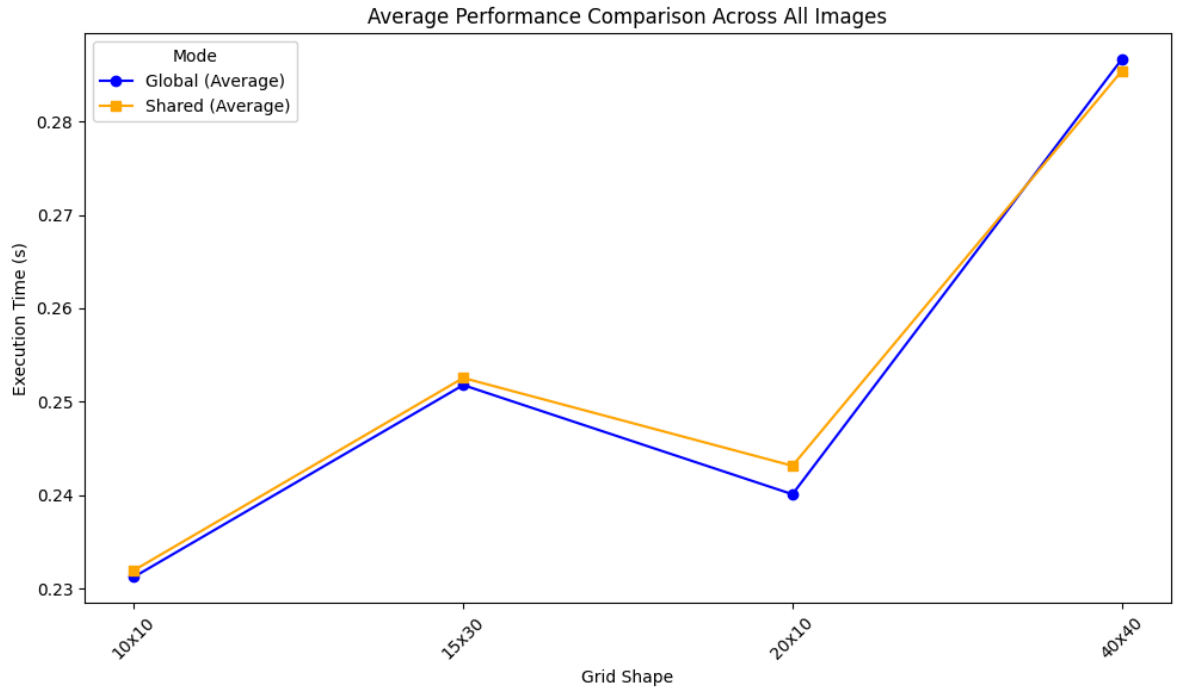
Image	Mode	Block Size	Grid Shape	Execution Time
720x720.png	Shared	32	15x30	0.278269
720x720.png	Global	32	40x40	0.30706
720x720.png	Shared	32	40x40	0.289049
720x720.png	Global	64	10x10	0.175593
720x720.png	Shared	64	10x10	0.189094
720x720.png	Global	64	20x10	0.180711
720x720.png	Shared	64	20x10	0.188569
720x720.png	Global	64	15x30	0.180849
720x720.png	Shared	64	15x30	0.17895
720x720.png	Global	64	40x40	0.179587
720x720.png	Shared	64	40x40	0.18737
360x360.png	Global	16	10x10	0.1429
360x360.png	Shared	16	10x10	0.132494
360x360.png	Global	16	20x10	0.13847
360x360.png	Shared	16	20x10	0.134141
360x360.png	Global	16	15x30	0.149855
360x360.png	Shared	16	15x30	0.145945
360x360.png	Global	16	40x40	0.147874
360x360.png	Shared	16	40x40	0.152055
360x360.png	Global	32	10x10	0.151838
360x360.png	Shared	32	10x10	0.147836
360x360.png	Global	32	20x10	0.144499
360x360.png	Shared	32	20x10	0.145641
360x360.png	Global	32	15x30	0.148586
360x360.png	Shared	32	15x30	0.154877
360x360.png	Global	32	40x40	0.152695
360x360.png	Shared	32	40x40	0.153852
360x360.png	Global	64	10x10	0.120376
360x360.png	Shared	64	10x10	0.125131
360x360.png	Global	64	20x10	0.119868

Image	Mode	Block Size	Grid Shape	Execution Time
360x360.png	Shared	64	20x10	0.119156
360x360.png	Global	64	15x30	0.12549
360x360.png	Shared	64	15x30	0.126292
360x360.png	Global	64	40x40	0.122447
360x360.png	Shared	64	40x40	0.121316

Key Findings:

1. **Consistency Across Sizes:** Shared memory performed well for medium configurations but was slightly slower for smaller images on average.
2. **Memory Access Patterns:** Shared memory's performance advantage is closely tied to optimized memory access patterns. The overhead introduced by shared memory management becomes significant when limits are exceeded.

Average Performance of Two Mode in Graph



Conclusion

This study highlighted the trade-offs between **Global Memory** and **Shared Memory** implementations. While shared memory can yield significant performance improvements for medium-sized workloads, its benefits diminish for smaller workloads due to marginal latency differences. Careful tuning of grid and block dimensions is crucial for leveraging the advantages of shared memory effectively.

Usage

In main directory you can write:

```
make help
```

for information to compile and run all programs and tests.