	Result	Size	Time	Cycles	GPU	SM Frequency	Process	Attributes
Current	585 - thresholding_kernel	(120, 68, 1)x(16, 16, 1)	88 us	126,197	0 - NVIDIA GeForce GTX 1660 Ti	1.43 Ghz	[11724] image	

Summary | Details | Source | Context | Comments | Raw | Session

Compare | Tools | View | Export

This table shows all results in the report. Use the column headers to sort the results in this report. Double-click a result to see detailed metrics. Double-click on demangled names to rename it.

ID	Estimated Speedup [%]	Function Name	Demangled Name	Duration [ms] (6.45 ms)	Runtime Improvement [ms] (4.27 ms)	Compute Throughput [%]	Memory Throughput [%]	# Reg
3	53.06	gradient_kernel	gradient_kernel(flo...	2.83	1.50	87.15	7.58	
4	63.03	edge_kernel	edge_kernel..char..	0.18	0.11	28.70	36.97	
5	48.34	thresholding_kernel	thresholding_kernel..	0.09	0.04	32.00	51.66	

The following performance optimization opportunities were discovered for this result. Follow the rule links to see more context on the Details page.
Note: *Speedup estimates provide upper bounds for the optimization potential of a kernel assuming its overall algorithmic structure is kept unchanged.*

Long Scoreboard Stalls
Est. Speedup: 48.34%

On average, each warp of this workload spends 18.3 cycles being stalled waiting for a scoreboard dependency on a L1TEX (local, global, surface, texture) operation. Find the instruction producing the data being waited upon to identify the culprit. To reduce the number of cycles waiting on L1TEX data accesses verify the memory access patterns are optimal for the target architecture, attempt to increase cache hit rates by increasing data locality (coalescing), or by changing the cache configuration. Consider moving frequently used data to shared memory. This stall type represents about 75.9% of the total average of 24.1 cycles between issuing two instructions.

Achieved Occupancy
Est. Speedup: 28.81%

The difference between calculated theoretical (100.0%) and measured achieved occupancy (71.2%) can be the result of warp scheduling overheads or workload imbalances during the kernel execution. Load imbalances can occur between warps within a block as well as across blocks of the same kernel. See the CUDA Best Practices Guide for more details on optimizing occupancy.

Uncoalesced Global Accesses
Est. Speedup: 27.60%

This kernel has uncoalesced global accesses resulting in a total of 179382 excessive sectors (29% of the total 622557 sectors). Check the L2 Theoretical Sectors Global Excessive table for the primary source locations. The CUDA Programming Guide has additional information on reducing uncoalesced device memory accesses.