Final Technical Report

Matrix Multiplication with Tiling and Shared Memory

Alejandro Herce Bernal

December 2018

1. Introduction

Template matching is a technique in digital image processing for finding small parts of an image which match a template image. It can be used in manufacturing as a part of quality control, a way to navigate a mobile robot, or as a way to detect edges in images.

The main challenges in the template matching task are; occlusion, detection of non-rigid transformations, illumination and background changes, background clutter and scale changes.

For this project, we're using template matching, specifically the sum of absolute differences with CUDA and OpenMP, varying the block sizes. SAD consists in finding the difference between the template image (the object you are searching on the other image) against the search area (the same size of the template) to see the differences. Then, the smallest difference is where the object is located on the image.

$$SAD(x, y) = \sum_{i=0}^{T_{\text{rows}}} \sum_{j=0}^{T_{\text{cols}}} \text{Diff}(x+i, y+j, i, j)$$

2. Development

A. Technical specifications

- CPU
  - Intel Core i7-4790k
  - Frequency: 4.4 Ghz
  - Cores: 4
  - Logic cores: 8
  - L1 cache: 256 KB
  - L2 cache: 1.0 MB
  - L3 cache: 8.0 MB

- GPU
  - Nvidia GeForce GTX 980 Ti
  - Memory: 6 GB GDDR5
  - CUDA Cores: 2816
  - GPU Clock Speed: 1.24 GHz
  - Memory Clock Speed: 3505 MHz

B. Testing

For the testing, different configurations were selected to measure the difference in performance. CPU with OpenMP and GPU with CUDA.

Different configurations were tested for GPUs, varying the tile size and the block size.

For each of the settings, 10 tests were executed and the execution time was averaged.

A. GPU Testing

For GPU testing, different block sizes were selected with the same formula for the grid.

All times below are in milliseconds. For CPU, OpenMP was used with all threads available. An average was calculated based on 10 tests of each configuration.

Block Size	CPU	GPU
8x8	28.86	0.0176
16x16	28.86	0.0165
32x32	28.86	0.0161

The most consistent and fast results were with tile and block size of 16x16, but tilled offering not much of a difference between sizes 16 and 32.

B. Comparison and Speedup

For speedup times, we have the following information:

	GPU time	CPU	Speedup
8x8	0.0176	28.86	1,639.77
16x16	0.0165	28.86	1,749.09
32x32	0.0161	28.86	1,792.54

3. Conclusions

Even when the program returns successfully the location of the object, it may have 1 or 2 pixels of difference. This is because it contains an IF clause that is not ready for multiple concurrent threads. The solution to this problem should be storing the SADs in an array of objects, and then using Thrust to obtain the minimum SAD. The SAD object will contain the exact location of the object.