Tecnológico de Monterrey

Campus Santa Fe

Programación Multinúcleo

Assignment 2

By Luis Carlos Arias Camacho

Objective.

The purpose of this assignment is to develop a software that blurs an image in opencv with secuencial implementation, OpenMP and GPU implementation.

- Sequentially on CPU.
- On Parallel on CPU.
- On Parallel on GPU.

With this information we are going to be able to learn and comprehend some of the basics of multithreading in CPU and GPU.

Test Specifications
All the test were made on the server that the professor gave us access to test our programs, that is because I do not have a Laptop with an Nvidia GPU (MacBook Pro 2012).

Also in the assignment there was specified that we should use a 5x5 image blur matrix, but we also implemented a 11x11 matrix blur so the image can be more details. In some cases the 5x5 image blur setted some pixels to red, blue or yellow.

Files in the Assignment and Functionality

1. **common.h**
   This file just defines the SAFE CALLS for the device call in GPU

2. **imageBlur_CPU.cpp**
   This file has the secuential and OpenMP implementation.

   To compile it use:
   g++ -o image_Blur_CPU imageBlur_CPU.cpp `pkg-config --cflags --libs opencv` -std=c++11 -fopenmp

3. **imageBlur_GPU.cu**
   This file has the GPU implementation

   To compile use:
   make

Testing:

Secuential And OpenMP

```
A01364808@alien1-lab:~/.../Assignement_2$  ./image_Blur_CPU
Test on CPU
Margin 2 -- Total pixels for blur matrix 25.000000
Image blur elapsed 40.211605 ms in CPU with a blur matrix of 5x5
Margin 5 -- Total pixels for blur matrix 121.000000
Image blur elapsed 159.594055 ms in CPU with a blur matrix of 11x11


Test on OpenMP
Margin 2 -- Total pixels for blur matrix 25.000000
Image blur elapsed 23.401752 ms in OpenMP with a blur matrix of 11x11
Margin 5 -- Total pixels for blur matrix 121.000000
Image blur elapsed 47.222393 ms in OpenMP with a blur matrix of 11x11
```

In GPU

```
A01364808@alien1-lab:~/.../Assignement_2$ ./imageBlur_GPU.exe
Test on GPU
Input image step: 1920 rows: 400 cols: 640
blur_kernel<<<(40, 25) , (16, 16)>>>
Image blur elapsed 0.012878 ms in GPU with a blur matrix of 5x5
Input image step: 1920 rows: 400 cols: 640
blur_kernel<<<(40, 25) , (16, 16)>>>
Image blur elapsed 0.007259 ms in GPU with a blur matrix of 11x11
```

Results.

Miliseconds.

	Secuential	OpenMP	GPU
5X5 blur matrix	40.21 ms	23.40 ms	0.012 ms
11x11 blur matrix	159.59 ms	47.22 ms	0.007 ms

Speedups.

	Secuential	OpenMP	GPU
Secuential	**1**	**1.72**	**3,350**
OpenMP	**0.58**	**1**	**1,950**
GPU	**0.0002**	**0.0005**	**1**

Conclusion.

GPU in this particular implementations is almost 3,500 times faster than the sequential implementation. Here we can see why programming in GPU is the next step of the programming world.