FYP Project musings:

milestones

* Compiled several examples for the Altera/Intel SDK on the snowball departmental server. It takes a long time to compile.
* Examined the CrowdSim program for Intel OpenCL 2.0

Crowdsim analysis:

Interestingly the biggest part of it is an external library created by a team at the University of North Carolina- RVO2. It’s for the “Reciprocal Collision Avoidance for Real-Time Multi-Agent Simulation”. In other words, this is the thing that does the crowd simulation. This is located in the RVO2 folder- it can be replaced with a (newer?) version from the library homepage which contains a makefile/CMakeList that can be used to compile it for linux- not much work to be done here as it is in C++.

Oddly enough compiling with my machine (OpenCL 2.1 is supported) works, however when running the program, it can’t compile the OpenCL code. Errors such as this appear:

1:31:9: error: unknown type name 'float2'; did you mean 'float'?

1:56:5: error: unknown type name 'uint'; did you mean 'int'?

1:78:5: error: unknown type name 'uint'; did you mean 'int'?

1:79:5: error: unknown type name 'uint'; did you mean 'int'?

1:84:5: error: unknown type name 'uint'; did you mean 'int'?

1:85:5: error: unknown type name 'uint'; did you mean 'int'?

Which is strange as these are standard OpenCL types defined since OpenCL 1.1 (I think?). Perhaps it can’t find the correct OpenCL run-time?

It seems like these errors were caused by the OpenCL compiler having an include directive that included the working directory. For some reason this prevented the compiler from being able to use the proper CL header files/libs.

The source files (w/o RVO2 library):

**basic.cpp**

So this appears to implement basic reusable functions in the code, things like is\_number, etc.

But more importantly it also implements functions which work with aligned pointers, such as malloc and free. Need to look into this.

**cmdoptions.cpp**

This file contains the display for any command line flags, for help (display all) and if any invalid parameters are specified. Not very interesting.

**cmdparser.cpp**

Similar to cmdoptions, this one works the handling of cmd flags.

**main.cpp**

Point of entry.

**oclobject.cpp**

It describes a class called OpenCLBasic, and presumably handles the initialization of the OpenCL context.

**RenderGL.cpp**

This is as the name suggests, the file that handles the rendering of the simulation. Unfortunately it is mostly rigged up for windows so we will exclude this file for now.

**SimulateCL.cpp**

The original Intel code has multiple calls to windows specific headers (windows.h, io.h), which are used specificially for the OpenGL rendering and creation of the application window.

Thankfully we can get rid of this as we don’t have a monitor to output to.

However it seems as though the program uses OpenGL (find out why?) even when the graphics disabled flag is enabled.

libglu1-mesa-dev (install the OpenGL development headers/libraries)

libglew-dev (Install the OpenGL Wrangler Extension)

Usually RVO2 compiles to a library which we can then link, but in this case the original template just includes the RVO source files so I have kept it that way.

http://stackoverflow.com/questions/12122631/difference-between-opengl-files-glew-h-and-gl-h-glu-h

Now works under Linux (Lubuntu 16.04).

alex@alex-Latitude-E7240:~/CrowdSim/CrowdSim$ ./CrowdSim\_CL --no-graphics

Simulating 400 agents using OpenCL

Platforms (1):

[0] Intel(R) OpenCL [Selected]

Devices (1):

[0] Intel(R) Core(TM) i7-4600U CPU @ 2.10GHz [Selected]

Type of OpenCL device: CPU

[ INFO ] Successfully queried SVM capabilities of the device devices[0].

[ INFO ] The device supports required SVM capabilities: fine grain buffer.

[ INFO ] Called ShareBuffers.

[ INFO ] Created RVOSimulator.

new page, size = 2097152

[ INFO ] SimulateCL::Init finishes successfully

Simulation of 400 agents completed correctly after 1783 frames

Total kernel time 0.579363 sec, worst frame 0.688076 ms, average frame 0.324937 ms

**Segfaults when no graphics is not on. Find out why (or make it default!).**

Now time to port to OpenCL 1.2.

Interestingly my device is only OpenCL 1.2 but supports SVM- presumably as it is a CPU device.

NAME: Intel(R) Core(TM) i7-4600U CPU @ 2.10GHz

VENDOR: Intel(R) Corporation

PROFILE: FULL\_PROFILE

VERSION: OpenCL 1.2 (Build 25)

EXTENSIONS: cl\_khr\_icd cl\_khr\_global\_int32\_base\_atomics cl\_khr\_global\_int32\_extended\_atomics cl\_khr\_local\_int32\_base\_atomics cl\_khr\_local\_int32\_extended\_atomics cl\_khr\_byte\_addressable\_store cl\_khr\_depth\_images cl\_khr\_3d\_image\_writes cl\_intel\_exec\_by\_local\_thread cl\_khr\_spir cl\_khr\_fp64

DRIVER\_VERSION: 1.2.0.25

Type: CPU

EXECUTION\_CAPABILITIES: Kernel Native

GLOBAL\_MEM\_CACHE\_TYPE: Read-Write (2)

CL\_DEVICE\_LOCAL\_MEM\_TYPE: Global (2)

SINGLE\_FP\_CONFIG: 0x7

QUEUE\_PROPERTIES: 0x3

VENDOR\_ID: 32902

MAX\_COMPUTE\_UNITS: 4

MAX\_WORK\_ITEM\_DIMENSIONS: 3

MAX\_WORK\_GROUP\_SIZE: 8192

PREFERRED\_VECTOR\_WIDTH\_CHAR: 1

PREFERRED\_VECTOR\_WIDTH\_SHORT: 1

PREFERRED\_VECTOR\_WIDTH\_INT: 1

PREFERRED\_VECTOR\_WIDTH\_LONG: 1

PREFERRED\_VECTOR\_WIDTH\_FLOAT: 1

PREFERRED\_VECTOR\_WIDTH\_DOUBLE: 1

MAX\_CLOCK\_FREQUENCY: 2100

ADDRESS\_BITS: 64

MAX\_MEM\_ALLOC\_SIZE: 4182107136

IMAGE\_SUPPORT: 1

MAX\_READ\_IMAGE\_ARGS: 480

MAX\_WRITE\_IMAGE\_ARGS: 480

IMAGE2D\_MAX\_WIDTH: 16384

IMAGE2D\_MAX\_HEIGHT: 16384

IMAGE3D\_MAX\_WIDTH: 2048

IMAGE3D\_MAX\_HEIGHT: 2048

IMAGE3D\_MAX\_DEPTH: 2048

MAX\_SAMPLERS: 480

MAX\_PARAMETER\_SIZE: 3840

MEM\_BASE\_ADDR\_ALIGN: 1024

MIN\_DATA\_TYPE\_ALIGN\_SIZE: 128

GLOBAL\_MEM\_CACHELINE\_SIZE: 64

GLOBAL\_MEM\_CACHE\_SIZE: 262144

GLOBAL\_MEM\_SIZE: 16728428544

MAX\_CONSTANT\_BUFFER\_SIZE: 131072

MAX\_CONSTANT\_ARGS: 480

LOCAL\_MEM\_SIZE: 32768

ERROR\_CORRECTION\_SUPPORT: 0

PROFILING\_TIMER\_RESOLUTION: 1

ENDIAN\_LITTLE: 1

AVAILABLE: 1

COMPILER\_AVAILABLE: 1

MAX\_WORK\_GROUP\_SIZES: 8192 8192 8192

---------------------------------------------------------------------

\*\*\* Kernel compilation resulted in non-empty log message.

\*\*\* Set environment variable CL\_HELPER\_PRINT\_COMPILER\_OUTPUT=1 to see more.

\*\*\* NOTE: this may include compiler warnings and other important messages

\*\*\* about your code.

\*\*\* Set CL\_HELPER\_NO\_COMPILER\_OUTPUT\_NAG=1 to disable this message.

0.000012 s

0.009980 GB/s

GOOD