

CUDA Tools

Marc-Antoine Le Guen

Timing nvprof

- línea de comando
- `nvprof ./sumArraysOnGPU-timer`

```
./sumArraysOnGPU-timer Starting... Using Device 0: Tesla M2070 ==17770== NVPROF is profiling process 17770, command: ./sumArraysOnGPU-timer Vector size 16777216 sumArraysOnGPU <<<16384, 1024>>> Time elapsed 0.003266 sec Arrays match.
```

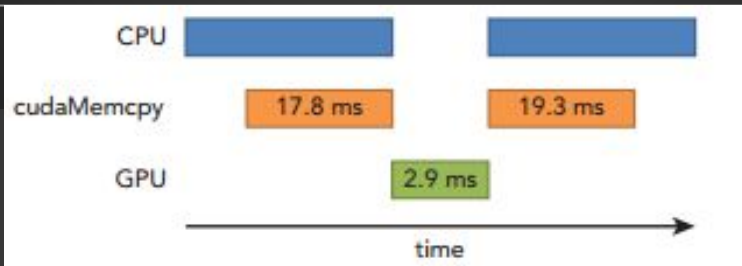
```
==17770== Profiling application: ./sumArraysOnGPU-timer
```

```
==17770== Profiling result: Time(%) Time Calls Avg Min Max Name
```

```
70.35% 52.667ms 3 17.556ms 17.415ms 17.800ms [CUDA memcpy HtoD]
```

```
25.77% 19.291ms 1 19.291ms 19.291ms 19.291ms [CUDA memcpy DtoH]
```

```
3.88% 2.9024ms 1 2.9024ms 2.9024ms 2.9024ms sumArraysOnGPU (float*, float*, int)
```



Manejar errores

- cudaMalloc
- cudaHostAlloc
- cudaMemcpy
- cudaEventCreate
- cudaStreamCreate
- cuda*****
- Cada una de estas funciones regresan un valor de tipo `cudaError_t`



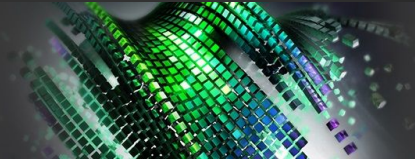
Herramientas

LIBRARY NAME	DOMAIN
NVIDIA cuFFT	Fast Fourier Transforms
NVIDIA cuBLAS	Linear Algebra (BLAS Library)
CULA Tools	Linear Algebra
MAGMA	Next generation Linear Algebra
IMSL Fortran Numerical Library	Mathematics and Statistics
NVIDIA cuSPARSE	Sparse Linear Algebra
NVIDIA CUSP	Sparse Linear Algebra and Graph Computations
AccelerEyes ArrayFire	Mathematics, Signal and Image Processing, and Statistics
NVIDIA cuRAND	Random Number Generation
NVIDIA NPP	Image and Signal Processing
NVIDIA CUDA Math Library	Mathematics
Thrust	Parallel Algorithms and Data Structures
HiPLAR	Linear Algebra in R
Geometry Performance Primitives	Computational Geometry
Paralution	Sparse Iterative Methods
AmgX	Core Solvers

Herramientas

Para la instalación y uso de la librería

Profesional CUDA Programming Chapt. 8



Herramientas

- CUFFT
- Una de las dos librerías incluidas en el CUDA toolkit
- Fast Fourier Transform
 - 2D - 3D transform hasta 16, 384 elementos por dimensión
 - 1D - 128 millones de elementos
- Librería de uso libre



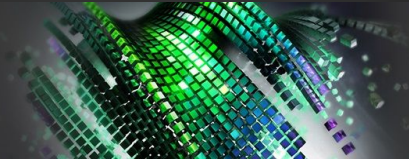
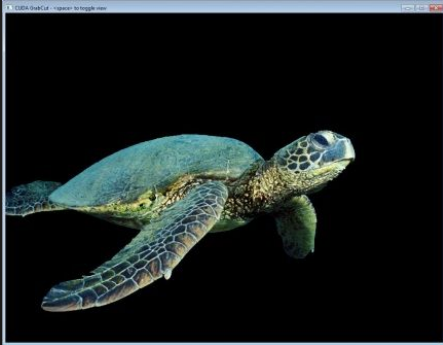
Herramientas

- CUBLAS (Basic Linear Algebra Subprograms)
- Segunda librería del CUDA toolkit
- 152 Rutinas de Álgebra lineal
 - Vectores
 - Matrices
- Single / Double precision
- Real / Complex data
- Fácil de acceso para los usuarios de BLAS



Herramientas

- Nvidia Performance Primitives
- Image, video and signal processing
 - 1900 image processing primitives
 - 600 signal processing primitives



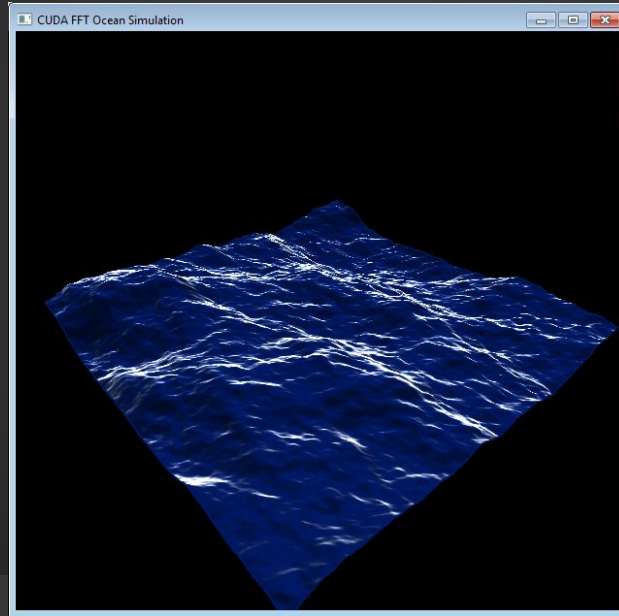
Herramientas

- Nvidia GPU Computing SDK
 - Samples
 - Simple
 - Utilities
 - Graphics
 - Finance
 - Imaging
 - Simulation
 - Advanced
 - Librerías CUDA



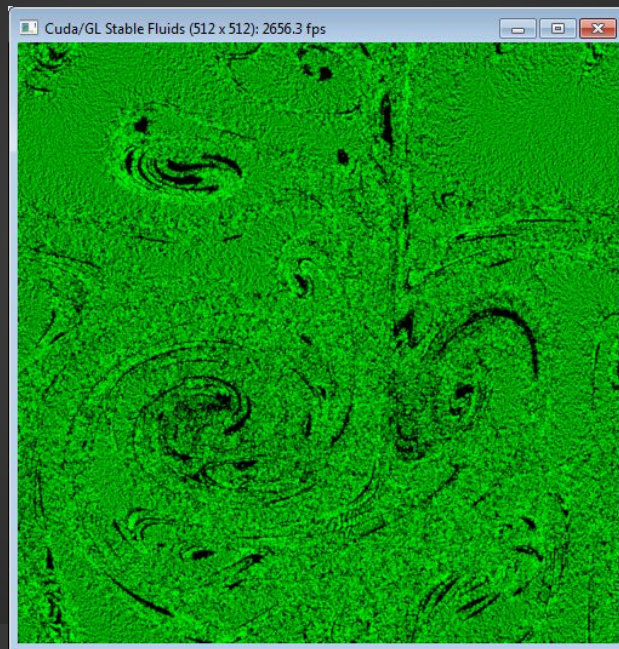
Herramientas

- Ocean FFT



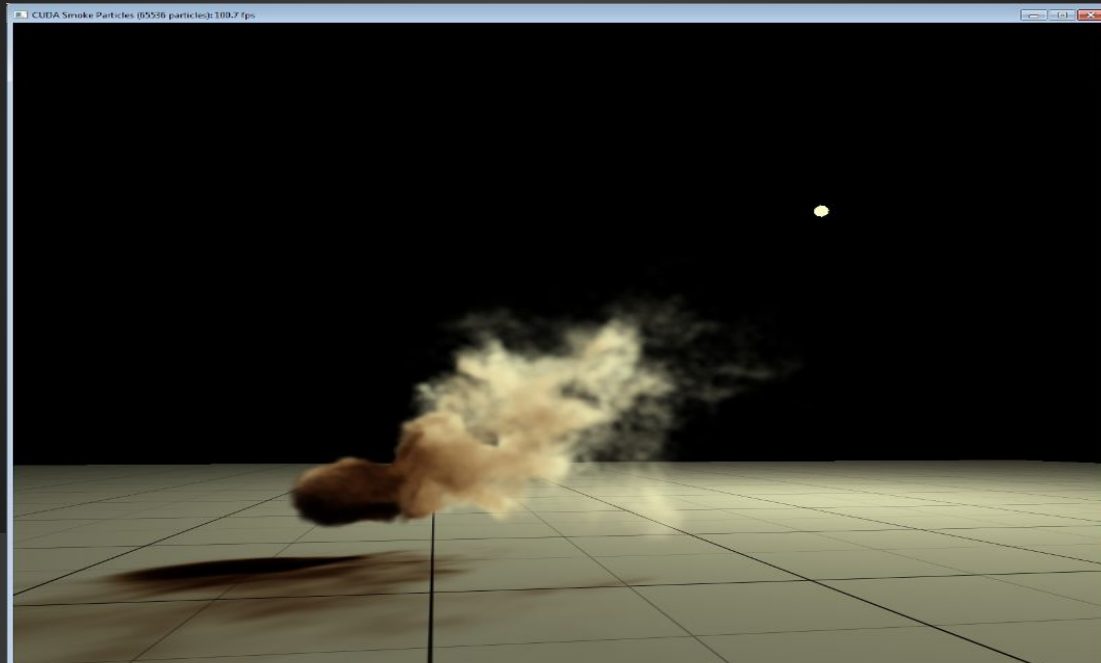
Herramientas

- Fluid OpenGL



Herramientas

- Smoke OpenGL



Herramientas

- Debugging
 - No es posible depurar el código con herramientas tradicionales
 - `printf` : compute capabilities ≥ 2.0
 - CUDA-GDB -> Linux based system
 - Ver el estado de CUDA -> GPU + capabilities
 - Breakpoint CUDA C
 - GPU memory : all global and shared memory
 - Single stepping a warp of threads
 - `cuda-memcheck`
 - violación de acceso



Herramientas

- Debugging - Nsight 4.1
- Mismas funciones que GDB
- Visual Studio - Windows
- Eclipse - Mac + Linux

 System	Attribute	GeForce GTX 670 Compute Capability: 3.0 Driver Model: WDDM
 Display Devices	ASYNC_ENGINE_COUNT	1
 GPU Devices	CAN_MAP_HOST_MEMORY	1
	CAN_TEX2D_GATHER	1
	CLOCK_RATE	1058500
 CUDA Devices	COMPUTE_CAPABILITY_MAJOR	3
	COMPUTE_CAPABILITY_MINOR	0
	COMPUTE_MODE	0
 OpenCL Devices	CONCURRENT_KERNELS	1
	DISPLAY_NAME	GeForce GTX 670
	ECC_ENABLED	0
	GLOBAL_L1_CACHE_SUPPORTED	0
	GLOBAL_MEMORY_BUS_WIDTH	256
	GPU_OVERLAP	1
	GPU_PCI_DEVICE_ID	294195422
	GPU_PCI_EXT_DEVICE_ID	4489
	GPU_PCI_EXT_DOWNSTREAM_LINK_RATE	5000
	GPU_PCI_EXT_DOWNSTREAM_LINK_WIDTH	16
	GPU_PCI_EXT_GEN	1
	GPU_PCI_EXT_GPU_GEN	2
	GPU_PCI_EXT_GPU_LINK_RATE	5000
	GPU_PCI_EXT_GPU_LINK_WIDTH	16
	GPU_PCI_REVISION_ID	161
	GPU_PCI_SUB_SYSTEM_ID	893523032
	INTEGRATED	0
	KERNEL_EXEC_TIMEOUT	1

Herramientas

- Debugging - Nsight 4.1

The screenshot displays the Microsoft Visual Studio (Administrator) interface with the Nsight 4.1 debugging extension. The main window is titled 'voxelpipe_demo_vc10 (Debugging) - Microsoft Visual Studio (Administrator)'. The menu bar includes File, Edit, View, Project, Build, Debug, Team, Nsight, Data, Tools, Test, Analyze, Window, and Help. The toolbar shows various debugging tools. The 'Process' dropdown is set to '[1840] voxelpipe_demo.exe' and the 'Thread' is '[2874912] <No Name>'. The 'Stack Frame' is 'CUmodule 05508f0 - [2] trace - Line 148'. The 'Connections' dropdown is set to 'localhost'.

The 'CUDA Info 1' window shows a table of warp information. The table has columns: Current, blockidx, Warp Index, PC, Active Mask, Status, Exception, File Name, Source Lin, and Lanes. The table shows several warps, with the current warp (Warp Index 0) highlighted. The 'CUDA WarpWatch 1' window shows a table of warp data. The table has columns: Name, Type, and Value. The table shows several warps, with the current warp (Warp Index 0) highlighted.

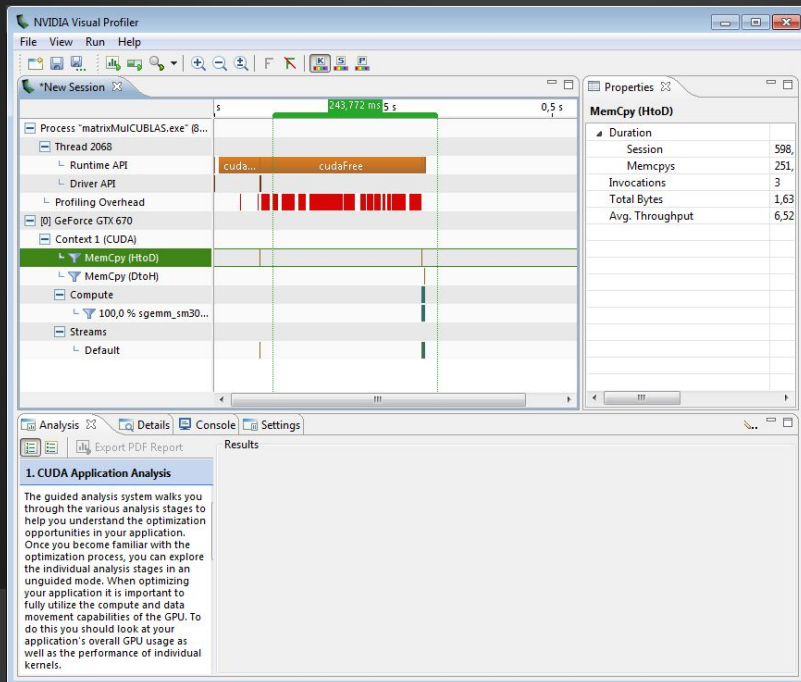
The 'Disassembly' window shows the assembly code for the current thread. The code is for the 'rt_render.cu' file, line 148. The code is as follows:

```
143 node_index = node.get_index(); // jump to child
144
145 else
146 {
147 // leaf intersection
148 const uint32 leaf_index = node.get_index();
149 const Bvh_leaf leaf = geometry.m_bvh_leaves[ leaf_index ];
150 const uint32 leaf_end = leaf.get_index() + leaf.get_size();
151 const uint32 leaf_begin = leaf.get_index();
152
153 for (uint32 tri_index = leaf_begin; tri_index < leaf_end; ++tri_index)
```

The 'Locals' window shows the local variables for the current thread. The variables are: leaf, leaf_index, leaf_end, leaf_begin, node, T21669, ray_inv, and node_index. The 'Call Stack' window shows the call stack for the current thread. The stack frames are: CUmodule 05508f0 - [2] trace - Line 148, CUmodule 05508f0 - [1] render_pixel - Line 409, and CUmodule 05508f0 - [0] rt_trace_primary_kernel - Line 493.

Herramientas

- CUDA Visual Profiler



Documentación y lectura

- Documentación y lectura
- Nvidia cuda Programming guide
 - <http://docs.nvidia.com/cuda/cuda-c-programming-guide/#abstract>
- Programming Massively Parallel Processors: A Hands-on Approach
 - David Kirk y Wen-mei W. Hwu
- Cuda by example (2010)
 - Jason Sanders y Edward Kandrot
- Professional CUDA C programming (2014)
 - John Cheng, Max Grossman, Ty McKercher
- Cuda Education and Training
 - materiales : cursos, ejercicios, etc.
 - <https://developer.nvidia.com/cuda-education-training>

