Welcome!

Optimization & Vectorization

D. Alexandridis, J. Bikker - Lecture 10: “GPGPU-3”
Today's Agenda:

GPGPU programming
Introduction

Today we will practice GPGPU programming

We will convert a CPU-fluid dynamics implementation to a GPU implementation

The method is based on the Navier-Stokes Equations for incompressible flow

We will not go into the mathematical details of the simulation today

If you want to learn more about this method, please have a look at:

Introduction

The fluid dynamics simulation uses a grid of velocities and pressure values to implement the simulation.

In multiple passes, we update the velocity field values according to 4 global steps:

1. Advection (move values along the velocity-values in the grid)
2. Viscous diffusion (resistance to flow resulting in the dissipation of velocity/values)
3. External forces (e.g., user input)
4. Pressure (fluid against itself)
Introduction

In the actual algorithm we perform the following steps for each grid-cell:

- **Update velocities**
  - Boundaries
  - Advection
  - Diffusion
- **Compute divergence**
- **Apply pressure**
  - Compute
  - Boundaries
  - Subtract
- **Update colors**
  - Boundaries
  - Advection
Introduction

Each of these steps iterate over all the grid-cells

Update the values in the corresponding grid/field (e.g., velocity, pressure, dye color)

Pass these values to the next iteration step

```plaintext
foreach (cell ∈ grid):
    Update velocity boundary(cell);
```

```plaintext
foreach (cell ∈ grid):
    Advect velocity(cell);
```

```plaintext
foreach (cell ∈ grid):
    ...
```

Let's have a look at the actual code

Introduction

Several important properties:

- Simulation steps loop over all grid-cells (*massive parallelism*)
- Boundary steps are somewhat different (*loop only over the boundaries*)
- Each step uses input from the previous (*we use input and output buffers*)
Introduction

Today we will convert this CPU code to GPGPU:

- Start with a one-to-one conversion
- Apply some simple but powerful optimizations
Introduction

Code along!

Pull the project repository from: https://github.com/Dionysi/FluidDynamics

Setting-up the project:

Set the working directory: right click on the Sandbox project file → Debugging → Working Directory → set to:

$(SolutionDir)bin\$(Platform)\$(Configuration)\n
The code is based on a template I am preparing for next year, therefore we will use a different OpenCL template for the programming.
Introduction

Final note:

- The git-repository contains 6 branches: master, gpgpu, and gpgpu-opt1 through gpgpu-opt4
  - master: cpu-code
  - gpgpu: straight-forward gpu conversion

The other branches contain some optimizations that build upon each other. There is obviously quite some more we could do that we may discuss at the end of the lecture.