Fluid Simulation using Smoothed Particle Hydrodynamics on GPU

Mohammad Adil, Sugeerth Murugesan

March 6, 2014

1 Introduction

Fluids are a common occurrence in real world applications, and considerable work has been done on simulating fluids. The two major class of algorithms for simulating fluid flows are grid-based (Eulerian grids) and particle-based (Smoothed Particle Hydrodynamics). SPH has certain benefits over traditional grid-based methods and is being increasingly used for fluid simulation. There is a great deal of interest in porting SPH to GPUs to achieve real-time fluid simulation. We will implement one such method and analyse its performance on different GPUs under various settings.

2 Smoothed Particle Hydrodynamics

Smoothed-particle hydrodynamics (SPH) was developed in 1977 by Gingold and Monaghan for computing fluid flows and is used in many fields including astrophysics, ballistics and oceanography. It Is a mesh-free Langrangian method which works by decomposing the fluid into a set of arbitrarily distributed particles. The hydrodynamical equations for the domain of the fluid are represented as an integral over these particles. A kernel function weights the effect of neighbouring particles. In this manner, each particle is "smoothed" over a finite volume of fixed mass and, therefore, SPH is naturally adaptive to density. This process is inherently conservative of mass, momentum and energy and the absence of a mesh allows it to handle arbitrary boundaries elegantly [6].

3 Real-Time Implementation

The biggest challenge in implementing SPH on GPUs is the neighbour search required in the kernel interpolant. This contribution is critical to the algorithm and can't be avoided. The problem can be solved by using a spatial data structure like kd-tree [3]. The construction of such a structure is time-consuming, but the lookup is very efficient. Alternative methods for neighbour search include neighbor texture maps [5], 3d buckets encoded as multiple textures [1] and virtual indexing grids [4]. We intend to use a uniform grid covering the bounding box for a naive implementation on the CPU. Later, this can be extended to a kd-tree and a GPU implementation maybe used to further accelerate the neighbor search. A Each thread on the GPU can then solve the motion equations for one particle while the spatial structure resides in shared memory. For real-time implementation, we impose the restriction that the computational space is fixed during one generation. This means that particles can't split or merge as this will disturb our parallel setup. As hardware support for geometry creation on GPU is added, this restriction can be removed.

4 Visualization

Our final problem is converting these particles into a mesh that can be rendered on the GPU. Popular surface creation techniques include meatballs, marching cubes, point splatting [6] or volume rendering [2]. Since the rendering is not a core part of our algorithm, we will choose meatballs for visualization. In case we have more time, point splatting can be considered for high quality output as meatballs suffers from bumpy surfaces [2].

5 Performance Analysis

Traditionally, SPH has not been used for fluid simulation because it requires a high number of particles for visually appealing results. Additionally, the nearest neighbour search is inefficient to implement on a GPU. Therefore, we will conduct a thorough study of the performance of our algorithm on different GPUs with different input parameters (density, etc.). We will also propose theoretical improvements to our algorithm that may enhance its efficiency. We will compare our results to other variants [2],[6],[5].

6 Timeline

During the first week of our project, we plan to learn OpenCL, and the fundamental mathametics behind particle hydrodynamics. In the second week, we perform basic prototyping by implementing important aspects of the prototype. During the third week, we record performance data by varying the different parameters pertaining to the application. During the last weeks, we benchmark our application and communicate our results by presenting our prototype to the class.

7 Deliverables

We plan to deliver a particle-simulation, results about performance on different GPUs with varying parameters such as viscosity and density. At the end of the project, the major deliverables are: Simulation, Performance data, a detailed report on algorithm and a brief write-up on effect of GPU in SPH.

References

[1] Takahiro Harada, Seiichi Koshizuka, Yoichiri Kawaguchi *Smoothed Particle Hydrodynamics on GPUs* Tokyo, Japan, 2007

[2] Keenan Crane, Ignacio Llamas, Sarah Tariq *Real-time simulation and Rendering of 3D Fluids* GPU GEMS, 2007

[3] Bart Adams, Mark Pauly, Richard Keiser, Leonidas Guibas *Adaptively Sampled Particle Fluids* SIGGRAPH 2007

[4] Prashant Goswami, Phillip Schleget, Barbara Solenthalar, Renato Pajarola *Interactive SPH Simulation and Rendering on GPU* ACM SIGGRAPH Symposium on Computer Animation, 2010

[5] Takashi Amada, Masataka Imura, Yoshihiro Yasumuro, Yoshitsugu Manabe *Particle-based fluid simulation on GPU* Nara Institute of Science and Technology, 2003

[6] Mathias Muller, Davis Charypar, Murkas Gross *Particle-based fluid simulation for interactive applications* Eurographics/SIGGRAPH Symposium on Computer Animation, 2003