Fluid Simulation with Rigid Body Triangle Accuracy Collision using an Heterogeneous GPU/CPU Hardware System

Jose Ricardo da Silva Junior[*] Computation Institute - UFF
Esteban W. Clua[†] Computation Institute - UFF
Paulo A. Pagliosa[‡] DCT - UFMS
Anselmo Montenegro[§] Computation Institute - UFF

Abstract

Fluid simulation is very important for the study of natural phenomena. Most of these phenomena could not be simulated computationally some years ago and for its study the research only had available the manual calculation of its govern equation or a small physical simulation of the phenomena to be studied. With the technology advance, most of these phenomena could be simulate even in real time coming out a new research field called Computational Fluid Dynamic (DFC) only to deal with it. Many researches in DFC use Graphics Processor Unity (GPU) to simulate fluids, forgiving the existence of the CPU, which became idle most time during the simulation. Due the presented fact, we propose a method for using GPU and CPU together during the fluid simulation. Fluid simulation in this work is based on Smoothed Particle Hydrodynamics (SPH), a mesh free Newtonian method for fluid simulation. Also, the rigid body and fluid interaction is dealt at triangle level, archiving more precise results.

CR Categories: I.3.7 [COMPUTER GRAPHICS]: Three-Dimensional Graphics and Realism—Virtual reality;

Keywords: fluid simulation, SPH, particle collision

1 The Mesh's Pre-processing and Particle Stage

This paper presents a novel approach for fluid simulation with rigid body interaction, using a pre-processing step before the simulation starts. The pre-processing stage is used for extracting data from the rigid body's polygons that are more suitable for dealing with the fluid interaction during the simulation. This pre-process is composed by the following steps: first the mesh is discretized into a set of particles of variable radius. This radius variation is due the fact that some simple surfaces need fewer particles of big radius to represent it, while others need more particles of small radius. A technique called Depth Peeling [Trapp and Döllner 2008] is used to accomplish it. In this technique, a mesh is rendered multiple times, marching in depth and storing its value in a volumetric texture. In the end, this volumetric texture will be composed of various depth slices from the mesh. In a second step, this volumetric texture is used for generating the particles information of the pre-processed mesh. Differentiation of particles created from mesh and particles generated for fluids is made through group encoding in the particle's id. Moreover, this group encoding could be extended to support various types of fluids interaction in the same simulation. The whole process performed by the simulation is shown in figure 1.

2 The Simulation Stage Using SPH and the Proposed Pre-processing Approach

The simulation of fluids using SPH requires the usage of specific kernels for each type of fluid's value to be processed. All this kernels use neighbourhood particles inside a given radius, giving a $O(n^2)$ complexity. Based on the fact that only particles inside

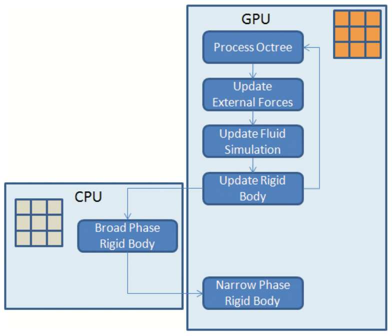

Figure 1: *Simulation Stages.*

the given compact radius will be processed, a spacial subdivision is used to avoid the $O(n^2)$ complexity. In this work, due the fact that we are using variable particle sizes, an hash function based Octree structure is being developed entirely in GPU. In this case, the complexity is given by $O(mn)$ where m is the approximate number of particles being processed. In order to avoid data transfer between GPU and CPU, each device owns its Octree structure with different kinds of information. In the GPU, a hash code is used to determine the octant of a particle. This hash code is based on the center of the position and volume of a particle, encompassing all its volume. In the CPU side, octants are calculated based on the position and bounding box volume of the rigid body encompassing all its volume. At this time, this calculations is evaluated every frame but an heuristic based method is being in study to evaluate only particles that moves from one octant to another.

In the next step, external forces are calculated. In this work, only gravity force and forces generated from the collision of particles and rigid bodies are considered, which are applied directly on the particle. Based on the fact that rigid body is also a set of particles, when collision occur these force are applied directly on the point of contact, which could be very expensive and complex if triangles were used.

When all the forces are in place, particles are updated considering only its own octant and deeper octants. Fluid particles compute values like viscosity, pressure and velocity using the SPH kernels. In relation to rigid body particles, a minimal set of information about forces applied to them by fluid particles are sent to the CPU. It's important to say here that after this information is sent to the CPU, the GPU goes to the next frame of simulation.

After that, the CPU does the broad phase collision detection to verify if rigid bodies have collided. In a positive case, another kernel is called to do the narrow phase collision in the GPU side. We remark here that the data do not need to be resent, since all necessary information for processing the narrow phase collision are already in the GPU memory in a form of particles.

References

TRAPP, M., AND DÖLLNER, J. 2008. Real-time volumetric tests using layered depth images. In *Eurographics 2008*, The Eurographics Association, K. Mania and E. Reinhard, Eds., Eurographics, 235–238.

[*]e-mail: jricardo@ic.uff.br
[†]e-mail:esteban@ic.uff.br
[‡]e-mail:pagliosa@dct.ufms.br
[§]e-mail:anselmo@ic.uff.br