DualSPHysics: past, present and future

J.M. Domínguez, A.J.C. Crespo, A. Barreiro, O. García-Feal, M. Gómez-Gesteira, Ricardo Canelas, C. Altomare,

jm Dominguez@uvigo.es

Universidade de Vigo
Outline

1. Birth of DualSPHysics
   1.1. Origin of DualSPHysics
   1.2. Why GPUs?
   1.3. DualSPHysics project

2. Stages of evolution
   2.1. Optimisations to maximize speed
   2.2. Multi-GPU to maximize the size
   2.3. Developing new capabilities

3. Future of DualSPHysics
   3.1. Next release
   3.2. Ongoing work
1.1. Origin of DualSPHysics

The DualSPHysics code was created starting from SPHysics.

**SPHysics** is a numerical model SPH developed for the study of free-surface problems. It is a code written in Fortran90 with numerous options (different kernels, several boundary conditions,…), which had already demonstrated high accuracy in several validations with experimental results… but it is too slow to apply to large domains.
1.1. Origin of DualSPHysics

The problem:

- SPH PRESENTS A HIGH COMPUTATIONAL COST THAT INCREASES WHEN INCREASING THE NUMBER OF PARTICLES
- THE SIMULATION OF REAL PROBLEMS REQUIRES A HIGH RESOLUTION WHICH IMPLIES SIMULATING MILLIONS OF PARTICLES

IT WAS NECESSARY TO INCREASE THE VELOCITY OF THE CODE A FACTOR 100x

IT WAS NECESSARY TO USE HPC TECHNIQUES

Classic options:
- **OpenMP**: Distribute the workload among all CPU cores (≈4x)
- **MPI**: Combines the power of multiple machines connected via network (high cost).

New option:
- **GPU**: Graphics cards with a high parallel computing power (cheap and accessible).
1.4. Graphics Processing Units (GPUs)

Graphics Processing Units (GPUs)
- powerful parallel processors
- designed for graphics rendering
- their computing power has increased much faster than CPUs.

**Advantages:** GPUs provide the necessary power with very low cost and without expensive infrastructures.

**Drawbacks:** An efficient and full use of the capabilities of the GPUs is not straightforward.
**1.3. DualSPHysics project**

First version in late 2009.

It includes **two implementations**:
- **CPU**: C++ and OpenMP.
- **GPU**: CUDA.

Both options optimized for the best performance of each architecture.

**Why two implementations?**

This code can be used on machines with GPU and without GPU.

It allows us to make a fair and realistic comparison between CPU and GPU.

Some algorithms are complex and it is easy to make errors difficult to detect. So they are implemented twice and we can compare results.

It is easier to understand the code in CUDA when you can see the same code in C++.

**Drawback:** It is necessary to implement and to maintain two different codes.
1.3. DualSPHysics project

DSPH project includes:

**Pre-processing tools:**
- Converts geometry into particles.
- Provides configuration for simulation.

**DualSPHysics solver:**
- Runs simulation using SPH particles.
- Obtains data simulation for time intervals.

**Post-processing tools:**
- Calculates magnitudes using particle data.
- Generates images and videos starting from SPH particles.

- Geometry (cad, 3ds, dwg, stl, vtk…)
- Configuration (parameters, motion…)

---

Iberian SPH 2015, 3-4 December 2015, Ourense (Spain)
1.5. DualSPHysics project

DEVELOPED JOINTLY BY RESEARCHERS AT

and many contributors

Dr Benedict Rogers
Dr Georgios Fourtakas
Dr Athanasios Mokos
Dr Stephen Longshaw
Abszolud Nadas
Prof. Peter Stansby

Prof. Moncho Gómez Gesteira
Dr Alejandro J. C. Crespo
Dr José Dominguez Alonso
Dr Anxo Barreiro Aller
Orlando García Feal

Prof. Rui Ferreira
Dr Ricardo Canelas

Dr Corrado Altimari
Dr Tomohiro Suzuki

Dr Renato Vacondio
Prof. Paolo Mignosa

Dr Xavier Gironella
Andrea Marzeddu

www.dual.sphysics.org
Outline

1. Birth of DualSPHysics
   1.1. Origin of DualSPHysics
   1.2. Why GPUs?
   1.3. DualSPHysics project

2. Stages of evolution
   2.1. Optimisations to maximize speed
   2.2. Multi-GPU to maximize the size
   2.3. Developing new capabilities

3. Future of DualSPHysics
   3.1. Next release
   3.2. Ongoing work
2.1. Optimisations to maximize speed

The main objectives were:
- To maximize the speedup of GPU against CPU.
- To maximize the performance on CPU and GPU.

After optimising the performance of DualSPHysics on CPU and GPU...

The most powerful GPU (**GTX Titan**) is **149 times faster** than CPU (single core execution) and **24 times faster** than the CPU using all 8 cores.


2.2. Multi-GPU to maximize the size

Simulation of 1 billion SPH particles using 64 GPUs Tesla
Large wave interaction with oil rig using 10^9 particles

2.2. Multi-GPU to maximize the size

Simulation of 1 billion SPH particles using 64 GPUs Tesla
Large wave interaction with oil rig using 10^9 particles

But, is it useful?
A lot of resources are necessary
Complicated pre-processing and post-processing

2.3. Developing new capabilities

Now, the objectives are:

- Maintaining a balance between maximizing the performance and code complexity.
- Developing new capabilities to increase the usefulness of DualSPHysics on real problems.
2.3.1. New capabilities for coastal protection

The problems we are interested on:
2.3.1. New capabilities for coastal protection

Study of the run-up in an existing armour block sea breakwater


2.3.1. New capabilities for coastal protection

Wave generation
- Wave generation starting from parameters of the waves: H, T, depth, piston type…
- Wave generation using theory of 1st and 2nd order
- Regular waves and irregular waves (JONSWAP and Pierson Moskowitz spectrums)
- Analytical results to compare elevation and fluid velocity

Wave absorption
- Passive absorption: Sponge layer and beach dissipation
- Active absorption: System AWAS to avoid wave reflection on piston

Altomare C, Domínguez JM, Crespo AJC, Barreiro A, Suzuki T, Gómez-Gesteira M., Troch P. Wave Generation and absorption in SPH-based model…

IN PREPARATION
2.3.2. Coupling with other models

Coupling with wave propagation model SWASH
2.3.3. Floating objects

Buoyancy of floating objects


Iberian SPH 2015, 3-4 December 2015, Ourense (Spain)
2.3.4. Rigid objects using DEM

Interaction with rigid objects

Validation of SPH-DEM

Top left cube: Experimental (-), DualSPHysics L/Dp = 15(--), L/Dp = 10(.-.), L/Dp = 45(…)

2.3.4. Rigid objects using DEM

Simulation of debris flows

DualSPHysics simulation took 285h for 70s of physical time with close to 3 million particles with 1600 solids


UNDER REVIEW

Validation of debris flows
2.3.5. Inlet conditions

Study of runoff on real terrains
2.3.6. Moorings implementation

Moorings implementation

Barreiro A, Domínguez JM, Crespo AJC, García-Feal O, Zabala I, Gómez-Gesteira M. Quasi-Static Mooring solver implemented in SPH. Journal of Ocean Engineering and Marine Energy. UNDER REVIEW

Iberian SPH 2015, 3-4 December 2015, Ourense (Spain)
2.3.7. Multiphase flows

Dispersion in porous media with multiphase flows
2.3.8. Realistic visualization

Advanced visualization using Blender

García-Feal O, Crespo AJC, Domínguez JM, Barreiro A, Gómez-Gesteira M. Open-source software for realistic visualisation of validated fluid simulations …

IN PREPARATION

Iberian SPH 2015, 3-4 December 2015, Ourense (Spain)
2.3.9. New boundary conditions

Three main categories:
(i) Boundary repulsive forces
(ii) Fluid extensions to the solid boundary
(iii) Boundary integral representation terms preservation

Dynamic boundaries (DBC) from group (i-ii)
Local Uniform STencil (LUST) from group (ii)
Boundary Integral (INTEGRAL) from group (iii)

To be included in next releases
2.3.9. New boundary conditions

SPHERIC Benchmark Test Case #10

Dynamic boundaries (DBC)
\[ \frac{h}{dp} = 1.3 \]

Local Uniform STencil (LUST)
\[ \frac{h}{dp} = 1.3 \]

Boundary Integral (INTEGRAL)
\[ \frac{h}{dp} = 4.0 \]

Experimental and numerical pressures measured at the sensor P1
2.3.10. Variable resolution

Dynamic refinement with particles with different sizes by means of particle splitting and coalescing procedures.

Two levels of splitting/coalescing
Outline

1. Birth of DualSPHysics
   1.1. Origin of DualSPHysics
   1.2. Why GPUs?
   1.3. DualSPHysics project

2. Stages of evolution
   2.1. Optimisations to maximize speed
   2.2. Multi-GPU to maximize the size
   2.3. Developing new capabilities

3. Future of DualSPHysics
   3.1. Next release
   3.2. Ongoing work
3.1. Next release

Current formulation (DualSPHysics v3):
- Time integration scheme:
  - Verlet [Verlet, 1967]
  - Symplectic [Leimkhuler, 1996]
- Variable time step [Monaghan & Kos, 1999]
- Kernel functions:
  - Cubic Spline kernel [Monagham & Lattanzio, 1985]
  - Quintic Wendland kernel [Wendland, 1995]
- Density filter:
  - Shepard filter [Panizzo, 2004]
  - Delta-SPH formulation [Molteni & Colagrossi, 2009]
- Viscosity treatments:
  - Artificial viscosity [Monaghan, 1992]
  - Laminar viscosity + SPS turbulence model [Dalrymple & Rogers, 2006]
- Weakly compressible approach using Equation of State
- Dynamic boundary conditions [Crespo et al., 2007]
- Floating objects [Monaghan et al., 2003]
- Periodic open boundaries
- Coupling with Discrete Element Method
3.1. Next release

**DualSPHysics v4.0 (End of 2015):**
- New CPU structure that mimics the GPU threads.
- New GPU structure with to help the user to follow and modify the code.
- Double precision implementation.
- Shifting algorithm.
- Floating bodies formulation were corrected.
- New wave generation (regular, irregular by given H/Hs, T/Tp and depth).
- Source code of the coupling with Discrete Element Method (DEM).

**DualSPHysics v4.2 (Spring 2016):**
- MultiGPU code with OpenMP.
- New BCs (LUST, Integral, Ghost particles).
3.2. Ongoing work

DualSPHysics solver:
- Variable resolution.
- Advanced inlet conditions.
- Improved Multi-GPU version for huge number of GPUs.

Pre-processing tools:
- New GenCase for other boundary conditions.
- Particle generation without mesh.
- Drawing particles adapted for variable resolution.

Post-processing tools:
- Recognition of data arrays defined by user.
- New tools for other functionalities.
DualSPhysics: past, present and future

J.M. Domínguez, A.J.C. Crespo, A. Barreiro, O. García-Feal, M. Gómez-Gesteira, Ricardo Canelas, C. Altomare,

jmdominguez@uvigo.es