ACG Midterm Proposal: Realistic Multi-Material Interaction Simulation Pipeline

Yiyang Lu
luyy24@mails.tsinghua.edu.cn
IIIS, Tsinghua University
Beijing, China

Xiangchen Tian
txc23@mails.tsinghua.edu.cn
IIIS, Tsinghua University
Beijing, China

Abstract

In this project, we propose to implement a physics-based simulation pipeline for rigid, fluid, and cloth materials. We use Smoothed Particle Hydrodynamics (SPH) to simulate fluid, and spatial hashing to accelerate the neighbor search. We will implement two SPH solvers: Weakly Compressible SPH (WCSPH) and Divergence Free SPH (DFSPH). Mass-spring system is applied to simulate the cloth dynamics. Besides, we implement rigid-fluid and rigid-cloth coupling to simulate interactions between different materials. Splashsurf and Blender are used for surface reconstruction and rendering.

Our code is open source and available on Github.

Keywords

Physics-based simulation, SPH, Mass-spring system, Coupling, Rendering

ACM Reference Format:
Yiyang Lu and Xiangchen Tian. 2024. ACG Midterm Proposal: Realistic Multi-Material Interaction Simulation Pipeline.

1 Introduction

We propose to implement Collision Simulation System for rigid, fluid, and cloth materials.

Goal: The system will demonstrate realistic physical interactions and be rendered with an industrial renderer to produce high-quality visual outputs. The technical points we intend to include are:

(1) Material Simulation: Implement rigid body, fluid, and cloth simulations from scratch. In detail, use SPH for fluid simulation, Mass-Spring system for cloth simulation, and traditional rigid body dynamics for rigid body simulation.
(2) Coupling: Simulate interactions between rigid bodies, fluids, and cloth. Specifically, implement rigid-cloth coupling (e.g., cloth draping over a box), rigid-fluid coupling (e.g., buoyancy effects), as well as self-collision for three materials.
(3) Geometry: Support complex geometry for rigid bodies and fluid. Use mesh-based representations for cloth and boundary representation for fluid containers.
(4) Performance Acceleration: Implement spatial hash structures for efficient collision detection. Use advanced numerical solvers for fluid equations (e.g. WCSPH or DFSPH). Implement multi-threading for surface reconstruction and rendering.
(5) Control: Include a configuration system for setting up scenes and physical parameters. Support real-time interactive control for adding/removing objects or changing parameters during simulation.
(6) Rendering: Render the final simulation with high-quality visuals using Blender. Focus on realistic lighting, shadows, and material properties.

Schedule: We plan to follow the following schedule for the project:

(1) Week 1-2: Planning and Framework Development — Define the physical and mathematical models for rigid bodies, fluids, and cloth. Set up a modular simulation framework to allow easy addition of simulation components. Create simple test cases (e.g., falling boxes, static fluid, hanging cloth). Finish render pipeline setup.
(2) Week 3: Rigid Body Simulation — Implement rigid body dynamics, including collision detection and response. Add support for mesh based geometries, validate with test scenes.
(3) Week 4-5: Fluid Simulation — Implement different SPH for fluid dynamics. Add boundary handling and support for complex geometries. Test basic fluid behaviors (e.g., pouring, splashing).
(4) Week 6: Cloth Simulation — Implement mass-spring system for cloth dynamics. Add support for pinning and collision handling. Test basic cloth behaviors (e.g., draping, folding).
(5) Week 7-8: Coupling — Implement coupling interactions between rigid-cloth, rigid-fluid, and fluid-cloth. Test each interaction in isolation and in combination.
(6) Week 9: Performance Optimization — Profile the simulation pipeline to identify bottlenecks. Implement spatial hashing for collision detection, as well as advanced numerical solvers for fluid equations.
(7) Week 10: Scene Control and Customization — Add support for scene configuration files to set up scenarios. Implement a basic interactive GUI for real-time adjustments.
(8) Week 11-12: Final Testing and Polishing — Test the entire system for stability and correctness. Refine visual and physical fidelity. Prepare the final presentation materials and submit the project.

Advanced Computer Graphics, Beijing, China

2 Methods

We implement three kinds of materials for simulation: rigid, fluid and cloth. We will introduce the method for each kind of material.

2.1 Rigid

The simulation of rigid body is simple. Rigid body's state is defined by position and orientation. When applying force $\mathbf{f}_i$ to the rigid body at $\mathbf{p}_i$

$$\mathbf{F} = \sum_i \mathbf{f}_i \tag{1}$$

$$\mathbf{M} = \sum_i (\mathbf{p}_i - \mathbf{r}) \times f_i \tag{2}$$

where $\mathbf{r}$ is the center of mass. The linear and angular acceleration can be calculated by

$$\frac{dv}{dt} = \mathbf{a} = \frac{\mathbf{F}}{m} \tag{3}$$

$$\frac{d\omega}{dt} = \alpha = \mathbf{I}^{-1}\mathbf{M} \tag{4}$$

where $\mathbf{I}$ is the intertia tensor.

2.2 Fluid

The main idea of fluid simulation is to solve the Navier-Stokes equation. The Navier-Stokes equation is a set of nonlinear partial differential equations that describe the motion of fluid substances. The equation is as follows:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f} \tag{5}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{6}$$

where $\mathbf{u}$ is the velocity field, p is the pressure field, ρ is the density of the fluid, ν is the kinematic viscosity, and $\mathbf{f}$ is the external force field. The first equation is the momentum equation and the second equation is the continuity equation. To solve the two equations numerically, Lagrangian method and Eulerian method are two common methods. In this project, we will use Lagrangian particle-based method, specifically Smoothed Particle Hydrodynamics (SPH) method, to simulate the fluid.

The main idea of SPH is to discretize the fluid into particles and calculate the physical quantities at each particle. The core formula of SPH is kernel function $W(\mathbf{r}, h)$, which is used to interpolate the physical quantities between particles. The kernel function in our implementation is defined as follows:

$$W(\mathbf{r}, h) = \frac{8}{\pi h^3} \begin{cases} 1 - 6(\frac{r}{h})^2 + 6(\frac{r}{h})^3 & 0 \le \frac{r}{h} < \frac{1}{2} \\ 2(1 - \frac{r}{h})^3 & \frac{1}{2} \le \frac{r}{h} < 1 \\ 0 & \frac{r}{h} \ge 1 \end{cases} \tag{7}$$

where r is the distance between two particles and h is the smoothing length. Define $W_{ij} = W(\mathbf{r}_i - \mathbf{r}_j, h)$. The density at each particle can be estimated by the following equations:

$$\rho_i = \sum_j m_j W_{ij} \tag{8}$$

We implement two SPH solvers: WCSPH and DFSPH. WCSPH is a weakly compressible SPH solver, which is suitable for simulating incompressible fluid. DFSPH is a density filter SPH solver, which is suitable for simulating compressible fluid.

2.2.1 *Weakly Compressible SPH (WCSPH).*

In WCSPH [Metaxas and Popovic 2007], the pressure field is calculated by:

$$p_i = B\left(\left(\frac{\rho_i}{\rho_0}\right)^\gamma - 1\right) \tag{9}$$

where B is the bulk modulus, γ is the polytropic index, ρ_i is the density at particle i, and ρ_0 is the reference density. The acceleration at particle i consists of four parts: pressure, viscosity, gravity, and surface tension. The pressure is calculated by:

$$\mathbf{a}_{\text{pressure},i} = -\frac{1}{\rho_i}\nabla p_i = -\sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}\right)\nabla W_{ij} \tag{10}$$

The viscosity is calculated by:

$$\mathbf{a}_{\text{viscosity},i} = \nu \sum_j m_j \frac{v_{ij}^T r_{ij}}{r_{ij}^2 + \epsilon^2 h^2}\nabla W_{ij} \tag{11}$$

In practice, for stability and symmetry, we change m_j to $\frac{m_i + m_j}{2}$. The surface tension is calculated by:

$$\mathbf{a}_{\text{surface tension},i} = -\frac{\kappa}{m_i}\sum_j m_j W_{ij} \tag{12}$$

where κ is the surface tension coefficient, and $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$. Gravity is trivial, hence

$$\frac{dv}{dt} = \mathbf{a}_{\text{pressure}} + \mathbf{a}_{\text{viscosity}} + \mathbf{a}_{\text{surface tension}} + \mathbf{g} \tag{13}$$

2.2.2 *Divergence-Free SPH (DFSPH).*

DFSPH is a density filter SPH solver, which is suitable for simulating incompressible fluid[Bender and Koschier 2015]. The main idea of DFSPH is to filter the density field to make the density field satisfy the equation of state. We don't need to calculate the pressure field explicitly. After computing non-pressure forces using 11 and 12, correct density and $\nabla \cdot \mathbf{v}$ by iteratively

In our implementation, define

$$\alpha_i = \frac{\rho_i}{\sum_j |m_j \nabla W_{ij}|^2 + |\sum_j m_j \nabla W_{ij}|^2} \tag{14}$$

$$\kappa_i^v = \frac{1}{\Delta t}\frac{D\rho}{Dt}\alpha_i = \frac{\rho_i}{\Delta t}\alpha_i \sum_j m_j \left(v_i - v_j\right) \cdot \nabla W_{ij} \tag{15}$$

update velocity by

$$v_i^{n+1} = v_i^n - \Delta t \sum_j m_j \left(\frac{\kappa_i^v}{\rho_i} + \frac{\kappa_j^v}{\rho_j}\right)\nabla W_{ij} \tag{16}$$

until the divergence of velocity field (which is $\frac{D\rho}{Dt}$) is small enough. Similarly, define

$$\rho_i^* = \rho_i + \Delta t \sum_j m_j \left(v_i - v_j\right) \cdot \nabla W_{ij} \tag{17}$$

$$\kappa_i = \frac{\rho_i^* - \rho_0}{\Delta t^2}\alpha_i \tag{18}$$

update velocity by

$$v_i^{n+1} = v_i^n - \Delta t \sum_j m_j \left(\frac{\kappa_i}{\rho_i} + \frac{\kappa_j}{\rho_j}\right)\nabla W_{ij} \tag{19}$$

until $\rho^*_{\text{avg}} - \rho_0$ is small enough.

2.2.3 Rigid-Fluid Coupling.

Rigid-Fluid coupling is a key component in fluid simulation. The main idea is to transfer the force from fluid to rigid body and vice versa[Akinci et al. 2012]. In our implementation, when coupling fluid and rigid body, we convert rigid body from mesh to point clouds.

The contribution of j-th particle in rigid body to i-th particle in fluid is calculated by

$$\rho_{ij} = m_j W_{ij} \tag{20}$$

$$\mathbf{a}_{\text{viscosity},ij} = \nu m_j \frac{v_{ij}^T r_{ij}}{r_{ij}^2 + \epsilon^2 h^2} \nabla W_{ij} \tag{21}$$

Surface tension is not considered in rigid-fluid coupling.

In WCSPH, the pressure field is calculated by

$$\mathbf{a}_{\text{pressure},ij} = -m_j \frac{p_i}{\rho_i^2} \nabla W_{ij} \tag{22}$$

In DFSPH, we modify the

$$\alpha_i = \frac{\rho_i}{\sum_{j\in\text{fluid}} |m_j \nabla W_{ij}|^2 + |\sum_{j\in\text{fluid and rigid}} m_j \nabla W_{ij}|^2} \tag{23}$$

and

$$\frac{D\rho}{Dt} = \frac{\rho_i}{\Delta t} \alpha_i \sum_{j\in\text{fluid and rigid}} m_j \left(v_i - v_j\right) \cdot \nabla W_{ij} \tag{24}$$

The rigid body doesn't have attribute κ_i^v and κ_i in physical sense. For consistency, we set $\kappa_i^v = 0$ and $\kappa_i = 0$. The velocity update for divergence correction is

$$v_i^{n+1} = v_i^n - \Delta t \sum_{j\in\text{fluid and rigid}} m_j \left(\frac{\kappa_i^v}{\rho_i} + \frac{\kappa_j^v}{\rho_j}\right) \nabla W_{ij} \tag{25}$$

update 17 and 19 by

$$\rho_i^* = \rho_i + \Delta t \sum_{j\in\text{fluid and rigid}} m_j \left(v_i - v_j\right) \cdot \nabla W_{ij} \tag{26}$$

$$v_i^{n+1} = v_i^n - \Delta t \sum_{j\in\text{fluid and rigid}} m_j \left(\frac{\kappa_i}{\rho_i} + \frac{\kappa_j}{\rho_j}\right) \nabla W_{ij} \tag{27}$$

2.3 Cloth

The main idea of cloth simulation is to discretize the cloth into two-dimensional square grid points, with the mass discretely distributed at each vertex, and the mass points connected by springs. For the sake of simplicity of calculation, and considering that the first-order approximation of many physical systems at small scales is linear, the springs are considered to be linear restoring forces, that is, they satisfy Hooke's law. The motion of each mass point follows Newton's equation of motion:

$$\mathbf{F}_{\text{external}} + \mathbf{F}_{\text{internal}} = m\mathbf{a} \tag{28}$$

where the first term is the external force and the second term is the spring force, given by

$$\mathbf{F}_{\text{internal}} = k_{\text{struct}} \Delta_{\text{struct}} \mathbf{x} + k_{\text{shear}} \Delta_{\text{shear}} \mathbf{x} + k_{\text{bend}} \Delta_{\text{bend}} \mathbf{x} \tag{29}$$

In order to connect adjacent mass points to achieve proximity constraints, first, each two nearest neighbor nodes need to be connected with a spring (structural spring) with an original length of l. The function of these springs is to perform basic proximity constraints; but if there are only these springs, there is no spring constraint for stretching along the diagonal direction of the square cloth, which is not in line with reality. Therefore, in order to ensure isotropy as much as possible, two springs (shear spring) are also added on each grid diagonal; finally, if there are only the above two springs, the "curling deformation" in three-dimensional space can be very curly, which is also not in line with actual cloth. Therefore, another spring (bend spring) is added to prevent the bending of three-dimensional space, as shown in the figure below.

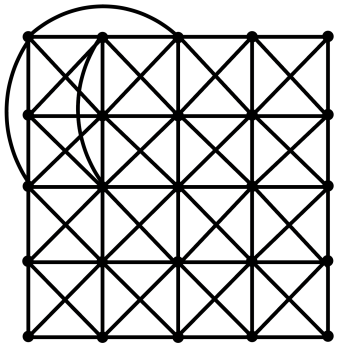

Figure 1: Mass Spring System

For coupling with rigid bodies, the algorithm is to first update the position of each mass point in the cloth normally, and then detect whether it collides with the rigid body. If there is a collision, the position of the mass point in the cloth is hard-shifted to move it out of the rigid body, the velocity change of the mass point in the cloth is calculated, and the impulse transmitted to the rigid body by this mass point of the cloth is calculated at the same time. Finally, the impulse of all mass points on the cloth to the rigid body is accumulated to obtain the total impulse to update the position of the rigid body, that is,

$$\mathbf{I}_{\text{total}} = -M\mathbf{g}\Delta t + \sum_{i\in\text{all particles in cloth}} \mathbf{I}_i \tag{30}$$

where M is the mass of the rigid.

Further, we also use an efficient hash table data structure to implement cloth self-collision: the whole space is divided into many periodic small cubes. Given the position of a particle in the cloth, its corresponding small cube can be queried from the hash table. Then, the particle can only collide with the particles in this small cube and the cubes around it in each substep, without having to traverse all the particles in the cloth again every time to detect whether a particle collides with other particles.

Note: Parameter adjustment is really critical in cloth simulation. If the time step is too large, the spring stiffness is too large, or the relative velocity loss coefficient is inappropriate, the cloth energy may dissipate and the area may become infinite after a few frames. On the contrary, if the stiffness coefficient and velocity loss parameters are too small, the rendering result will be very unrealistic and the cloth will have strange deformations.

3 Implementation

Our implementation mainly consists of three stages: configuration, simulation, and rendering.

3.1 Configuration

Users can configure the scene by specifying the materials, initial state, and external forces. Specifically, for rigid and fluid, user can specify the initial position by loading a mesh file or use our built-in geometry. Besides, any physical parameters useful for simulation can be set by the user.

3.2 Simulation

All simulations are implemented in Python, especially Taichi, a high performance acceleration library.

In fluid simulation, naive SPH is slow since each step requires $O(n^2)$ computation. To speed up the simulation, we use spatial hashing to accelerate the neighbor search. Specifically, we divide the space into cells and store the particles in each cell. After each step, we update the cell index of each particle. When searching neighbors, we only need to search the particles in the same cell and its adjacent cells. Using spatial hashing, the complexity of neighbor search is reduced to $O(n)$.

3.3 Rendering

For all three materials, we store the mesh as simulation results.

The simulation of fluid is implemented in point cloud. After simulation, we use splashsurf [Löschner et al. 2023] to reconstruct the surface, followed by Blender to render the scene. To speed up surface reconstruction and rendering, we use multi-threading to parallelize the process.

4 Experiment

In this section, we present several compelling results generated using our pipeline. For all simulations, we consistently use the same hyper-parameter settings: particle radius $r = 0.01$, supporting radius $h = 4r$, rest density $\rho_0 = 1000\text{kg/m}^3$, and time step $\Delta t = 0.0001$. The rendering results is done in Blender at 60 frames per second.

4.1 Rigid-Fluid Coupling

In this chapter, we present the results of rigid-fluid coupling, including fluid coupling with both fixed and movable rigid bodies. Specifically, we simulate a water droplet falling on a fixed bunny using both WCSPH (Figure 2) and DFSPH (Figure 3). The number of fluid particles is 400,000 and the number of rigid particles is 100,000.

Besides, two-way rigid-fluid coupling is presented by simulating a rubber duck falling into a pool of water using WCSPH (Figure 4). The number of fluid particles is 700,000 and the number of rigid particles is 100,000.

4.2 Rigid-Cloth Coupling

In this chapter, we present the results of rigid-cloth coupling, including cloth coupling with both fixed and movable rigid bodies. Specifically, we simulate a cloth draping over a fixed rigid body (Figure 5)

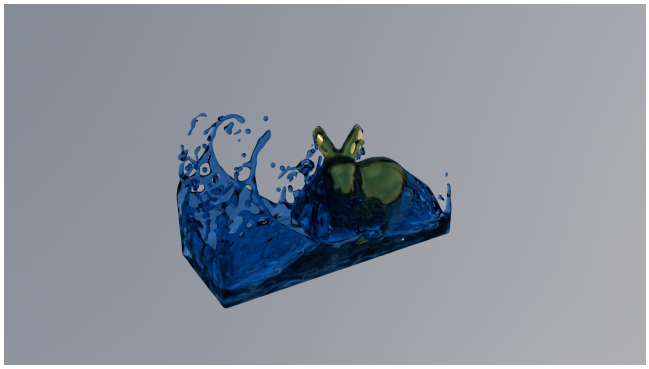

Figure 2: Rigid-Fluid Coupling in WCSPH

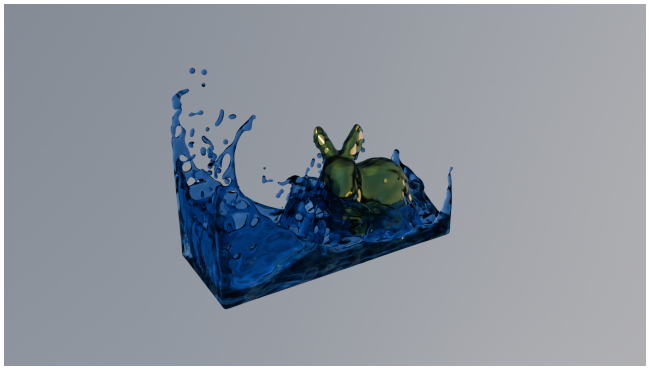

Figure 3: Rigid-Fluid Coupling in DFSPH

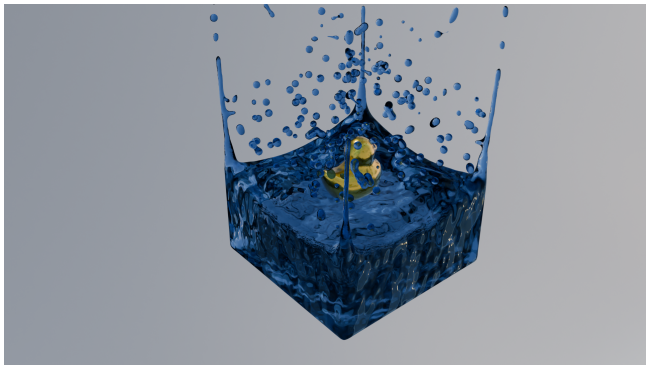

Figure 4: Two-way Rigid-Fluid Coupling

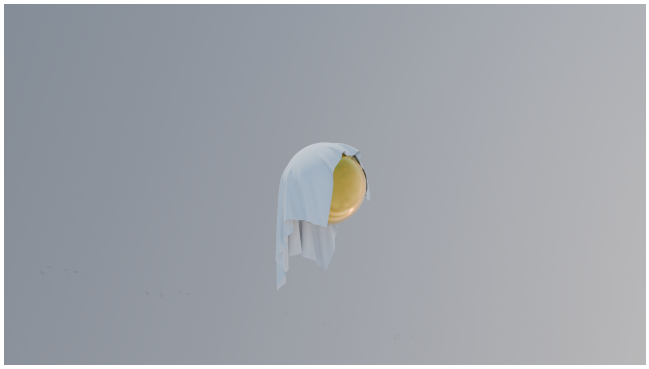

Figure 5: One-way Rigid-Cloth Coupling

and a ball falling into a cloth fixed at four corners (Figure 6) (With custom background[1]):

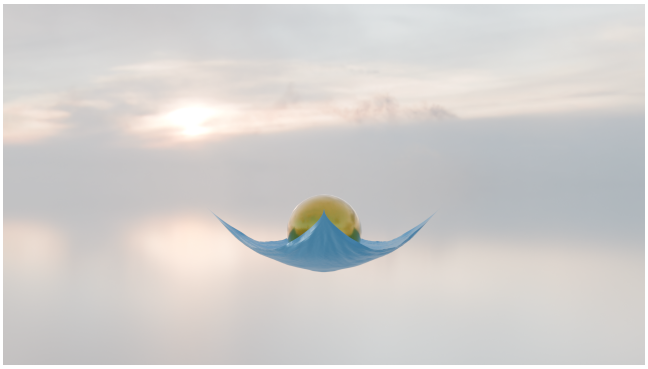

Figure 6: Two-way Rigid-Cloth Coupling

4.3 Other results

Besides, we also simulate the scene with single material, for more details, please refer to our Github. We will keep updating the results.

5 Future Work

In the future, we plan to extend our simulation pipeline in the following ways:

(1) Implement more SPH solvers, such as PCISPH, to achieve more stable and accurate fluid simulation.
(2) Implement more advanced cloth solvers, such as FEM, to achieve more realistic cloth deformation.
(3) Implement more advanced coupling methods, such as fluid-cloth coupling, to simulate more complex interactions between different materials.
(4) Optimize the performance of our pipeline to achieve faster real-time simulation, as well as add support for higher resolution rendering. Interactive scene control and customization will also be considered.
(5) Implement more complex scenes, such as cloth draping over a moving object, to demonstrate the capabilities of the simulation pipeline.

References

Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. 2012. Versatile rigid-fluid coupling for incompressible SPH. *ACM Trans. Graph.* 31, 4, Article 62 (July 2012), 8 pages. https://doi.org/10.1145/2185520.2185558

Jan Bender and Dan Koschier. 2015. Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics symposium on computer animation*. 147–155.

Fabian Löschner, Timna Böttcher, Stefan Rhys Jeske, and Jan Bender. 2023. Weighted Laplacian Smoothing for Surface Reconstruction of Particle-based Fluids. In *Vision, Modeling, and Visualization*. The Eurographics Association. https://doi.org/10.2312/vmv.20231245

D Metaxas and J Popovic. 2007. Weakly compressible SPH for free surface flows. (2007).

[1]Aside: Blender defaults to the positive z direction for the background upwards. In our implementation, the default direction of gravity acceleration is the -y direction. We need to transform the coordinate axes during rendering to achieve the correct background image display.