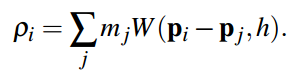
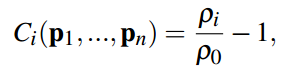
Approach

Enforcing Incompressibility

To enforce incompressibility of the water, we apply a position based constraint on each particle. First, following the principle of the SPH method, we calculate the density of each particle by applying a Poly6 kernel around its neighborhood, as shown in equation ()



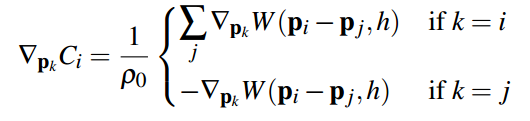
Then the density constraint is calculated as in equation()



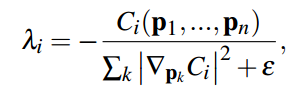
Now based on the PBD method, we aim to find the position correction that satisfies the constraint in equation ()



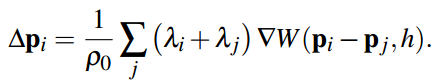
To achieve this, we first find the gradient of the constraint by looping through each particle in the scene, for each particle, we loop through itself and all of its neighbors, and apply the equation () based on whether the particle is itself or its neighbor. In this step, we use the Spiky kernel.



Then we plug this gradient into the equation () to compute the (lamda), where (epsilon) is the relaxation parameter.



Finally we use the equation () to calculate the position correction.

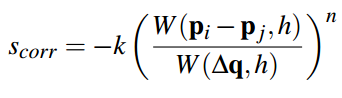


Introduction

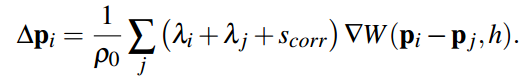
We implemented an interactive position based fluid simulation system that simulates fluid inside the Position Based Dynamics framework. The paper we are based on is the 2013 SIGGRAPH paper "Position Based Fluids" [Macklin et al. 2013]. In previous fluid simulation methods, such as the Smoothed Particle Hydrodynamics (SPH) method, enforcing incompressibility is crucial but computationally expensive, thus is impractical for real-time applications. This paper integrates the simulation process into the Position Based Dynamics (PBD) framework by solving a positional constraint that enforces constant density for each particle, and can achieve similar incompressibility and convergence to the SPH method but larger time steps that are suitable for real-time applications. In our project, we provides an interactive system implemented by following the algorithm in the paper. Also, we modified the collision detection method by applying constant position constraints on the static particles and depend fully on the PBD method for collision avoidance between the static particles and dynamic particles. The simulation system can be seen in the visual studio project, and several scenes captured in OpenGL and rendered in Maya can be found in the videos.

Tensile Instability

To prevent the problem of particle clustering or clumping caused by negative pressures when a particle has only a few neighbors and is unable to satisfy the rest density, we follow the paper's approach by adding an artificial pressure term (scorr) specified in terms of the smoothing kernel itself as in the equation ()

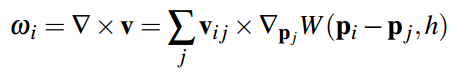


The final calculation formula then becomes



Vorticity Confinement and Viscosity

To compensate for additional damping caused by the nature of the PBD method, we use vorticity confinement to replace lost energy at each particle's location by using the estimator as in the equation ()

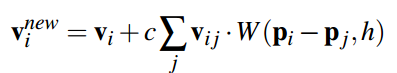


Then we calculate a corrective force using the equation ()



where 

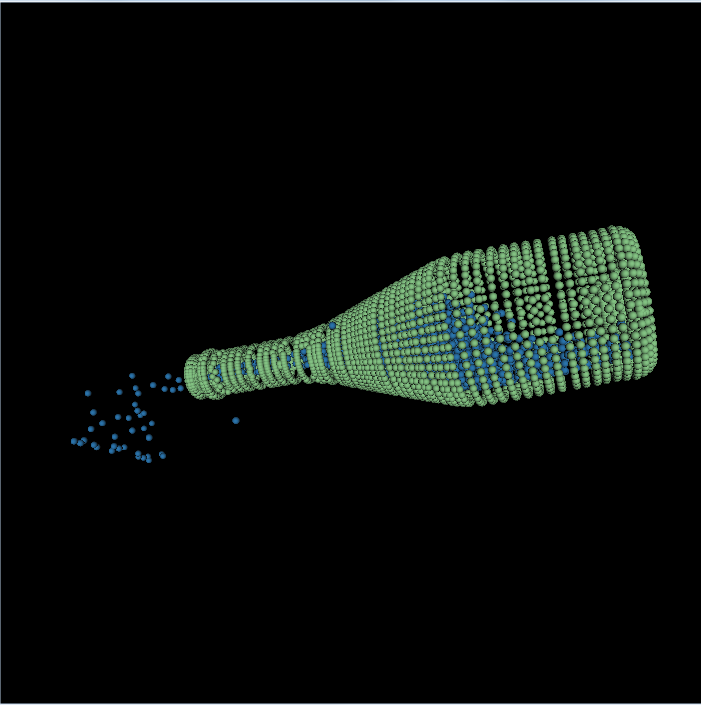
In addition, we apply XSPH viscosity as in the equation () to ensure coherent motion



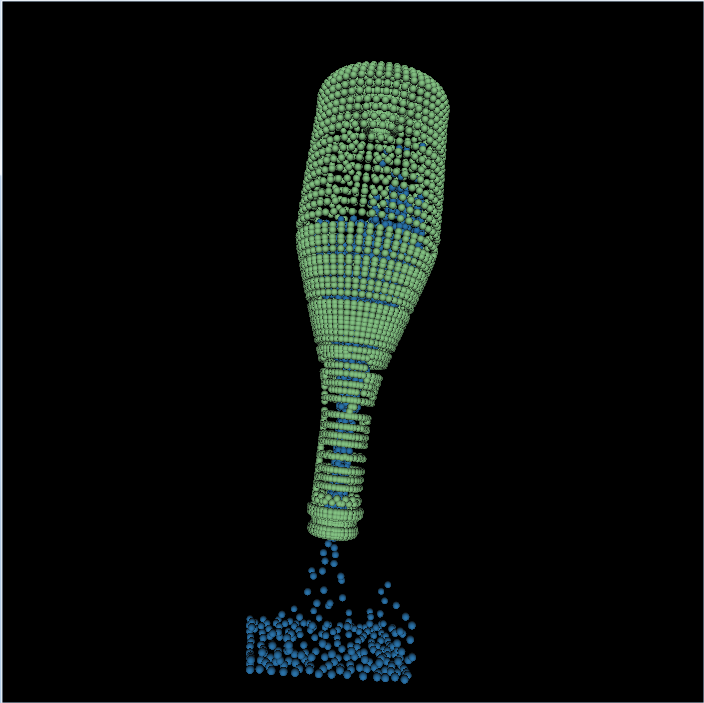
Implementation & Results

User Interactivity

Our simulation system allows user to use mouse to control the rotation of the scene. By clicking and dragging the mouse, users can rotate the objects as well as the direction of the gravity force. Also, pressing 'g' or 'G' allows user to unlock the static particles, which will fall and become dynamic particles once they come into contact with the bounding box.



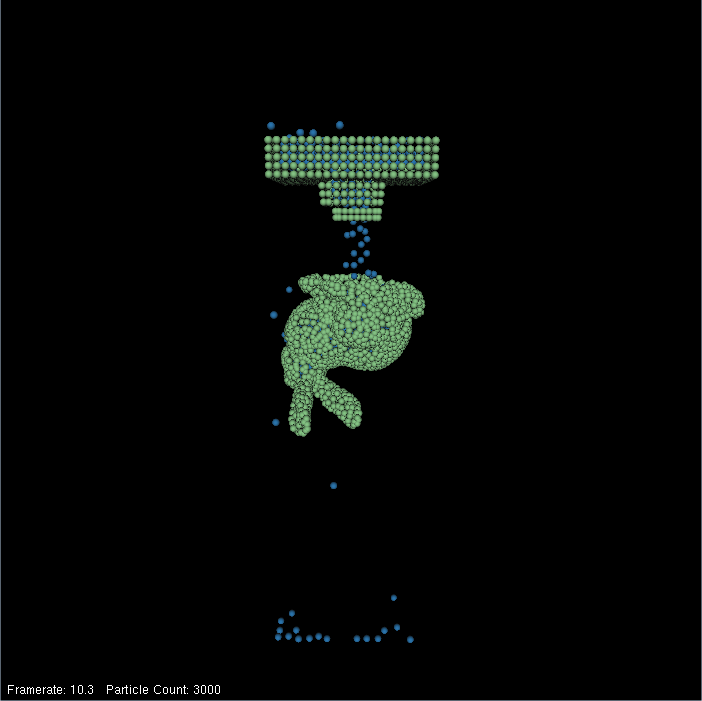
User Interaction 1



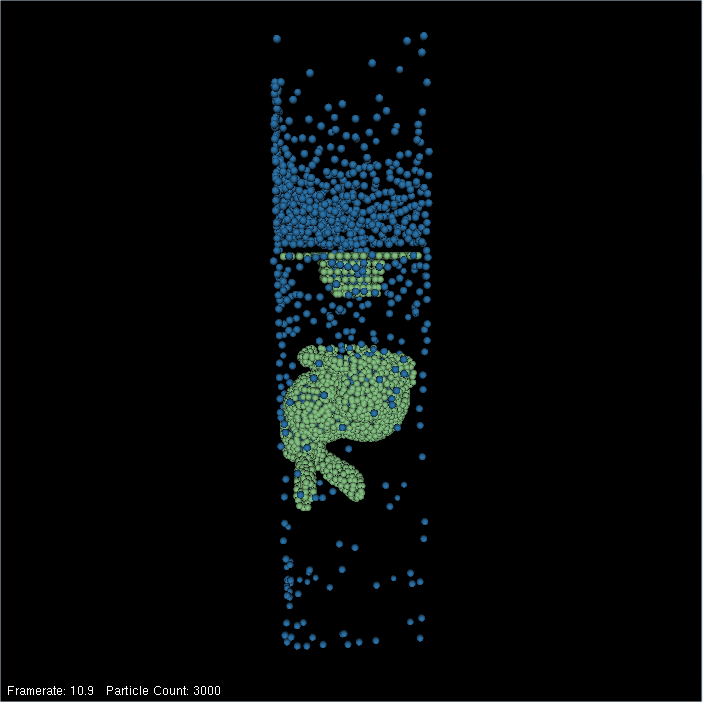
User Interaction 2

Real-time Performance

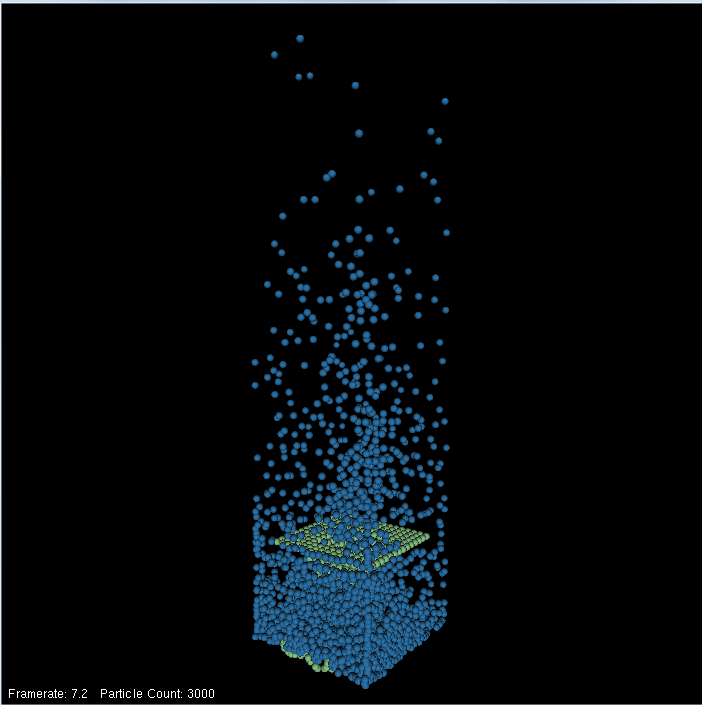
Our simulation system is based on CPU programming, and the typical number of particles it is able to support in order to provide real-time performance is 3K, with a FPS of approximately 10.



Bunny and funnel stay in the air



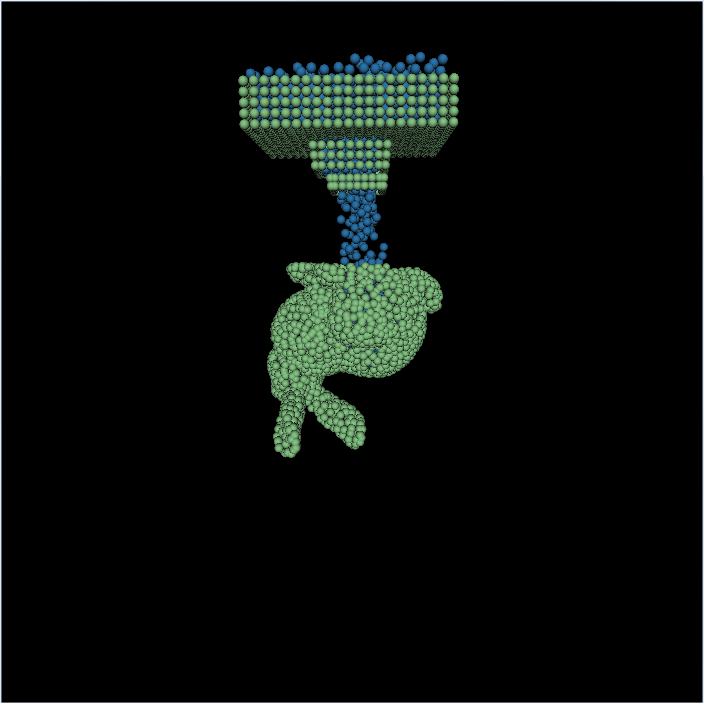
After pressing 'g' bunny and funnel fall with gravity



Bunny and funnel merge into the dynamic particles

Obj File Loader

Our simulation system implemented an Obj file loader to import multiple objects to define the shape of the container.



Bunny and Funnel Container



Dragon Container



Bottle Container

Position Based Collision Detection

For collision detection between the static particles and the dynamic particles, we makes use of the PBD method. By defining a larger rest density for the static particles and calculating the position correction for the dynamic particles in terms of the static particles, we ensure that the container made of static particles will hold the dynamic particles inside. For the collision detection between particles and the bounding box (not drawn in the scene), we still use geometric collision detection with the box.