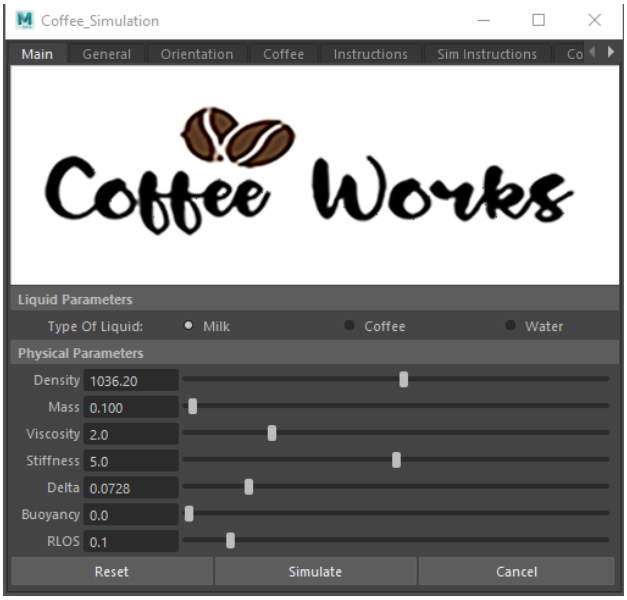
Coffee simulation:

User manual:



- Select a default liquid or select physical parameters for each particle in main tab
- Choose either a uniform or random distribution for the general simulation from the general tab
- If the bounding box uniform selection is used the simulation will be 20 minutes.
- Within general also select particle colour and particle properties such as size, and other physical properties such as gravity and initial position.
- From the orientation tab select an orientation type to spawn the liquid. Options are cylindrical and bounding box.

From the coffee tab, choose a machine and cup type to collide with the particles. Other options include roast which will change particle colour.

If you require more help, you can find instructions in the UI window. The simulation only works if a **filepath** is specified to the folder containing the assets

Algorithms used:

The algorithm used for this simulation is the **SPH method**. It is a **Langragian** based fluid simulation method that treats individual particles within the system and finds their corresponding neighbours. If a neighbour is within the vicinity of another particle, it will classify that particle within its cluster group and calculate physical properties on neighbours corresponding to that particle. The general implementation involves the use of Kernel functions that approximate mass density, viscosity and pressure field vector values for particles within a neighbourhood of a particle.

Algorithm 1 SPH with state equation.

for all *particle i* **do**
  find neighbors j
for all *particle i* **do**
  $\rho_i = \sum_j m_j W_{ij}$
  compute p_i using ρ_i
for all *particle i* **do**
  $\mathbf{F}_i^{pressure} = -\frac{m_i}{\rho_i}\nabla p_i$
  $\mathbf{F}_i^{viscosity} = m_i \nu \nabla^2 \mathbf{v}_i$
  $\mathbf{F}_i^{other} = m_i \mathbf{g}$
  $\mathbf{F}_i(t) = \mathbf{F}_i^{pressure} + \mathbf{F}_i^{viscosity} + \mathbf{F}_i^{other}$
for all *particle i* **do**
  $\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \Delta t \mathbf{F}_i(t)/m_i$
  $\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t) + \Delta t \mathbf{v}_i(t+\Delta t)$

General implementation:

For each frame of the simulation, the discrete sum of all physical forces acting on particles within a cluster group needs to be calculated. This value is then used to approximate the velocities and positions of particles in subsequent frames of the simulation.

A kernel function is a gaussian function that, when applied to the discrete sum of all forces acting on particles, will smooth their continuous nature to simulate a fluid. See **figure 1**.

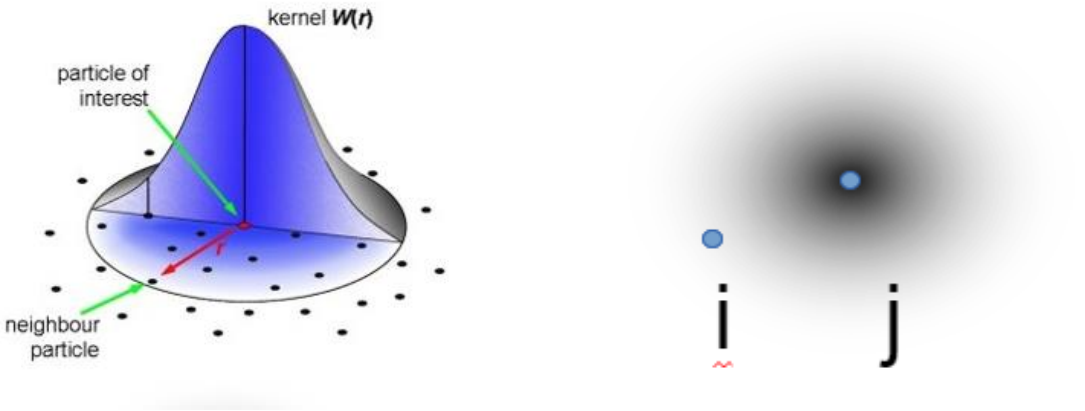


Figure 1: Kernel function. Works by finding neighbour particles j within the vicinity of a given particle i. The kernel functions used for this program are the **polygonal 6, spiky and viscosity kernels**.

$$W_{poly6}(\mathbf{r},h) = \frac{315}{64\pi h^9}\begin{cases}(h^2 - \|\mathbf{r}\|^2)^3, & 0 \le \|\mathbf{r}\| \le h \\ 0, & \|\mathbf{r}\| > h\end{cases}$$

$$W_{spiky}(\mathbf{r},h) = \frac{15}{\pi h^6}\begin{cases}(h - \|\mathbf{r}\|)^3, & 0 \le \|\mathbf{r}\| \le h \\ 0, & \|\mathbf{r}\| > h\end{cases}$$

$$W_{viscosity}(\mathbf{r},h) = \frac{15}{2\pi h^3}\begin{cases}-\frac{\|\mathbf{r}\|^3}{2h^3} + \frac{\|\mathbf{r}\|^2}{h^2} + \frac{h}{2\|\mathbf{r}\|} - 1, & 0 \le \|\mathbf{r}\| \le h \\ 0, & \|\mathbf{r}\| > h\end{cases}$$

Where **h** is the **cluster radius** and **r** is the position vector of a particle **i**.

Finding particle neighbourhoods:

This program uses a **function-based approach** to calculate forces and find neighbours within a cluster radius of a particle. To find particles within the cluster radius of any given particle, we assume that particles are perfect spheres and apply the algorithm used to find sphere-sphere intersection:

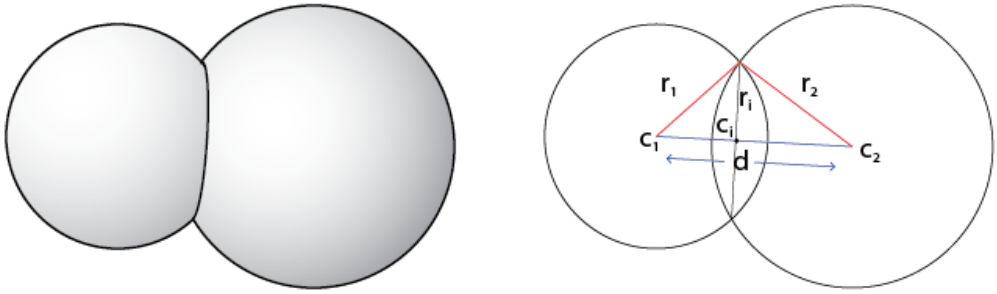


Figure 2: a sphere-sphere collision is detected if the magnitude of the vector distance between c_1 and c_2 minus the sum of radius r_1 and r_2 is less than or equal to zero.

$||c_2-c_1||$ - r_1-r_2 <= 0, collision detected

If a sphere-sphere collision is detected, it will add a particle j neighbour to particle i's cluster group. After adding all the particles to their associated cluster groups, the kernel functions are used to find forces acting on particles. **Figure 3** demonstrates implementation of algorithm into python code.

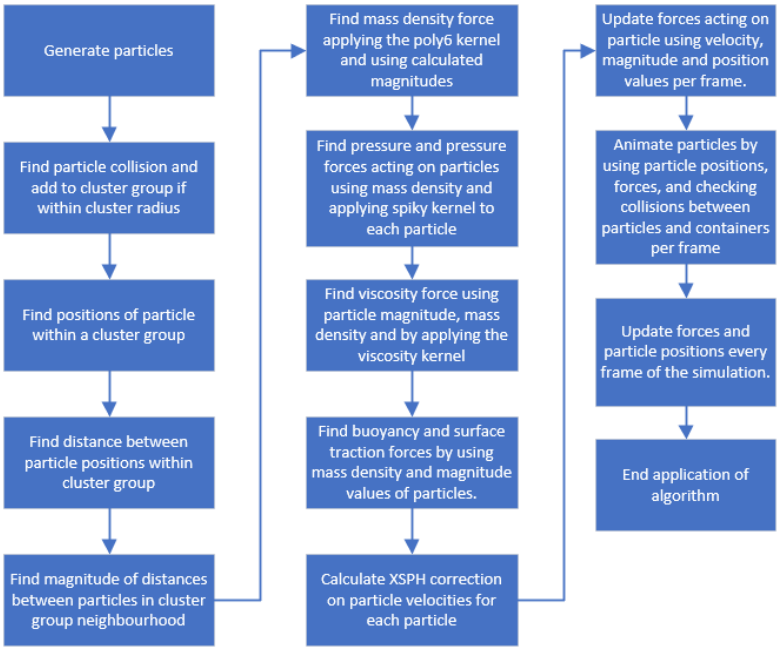


Figure 3: general control flow of SPH implementation in python

Python implementation:

- To classify particles i within the neighbourhood of a particle j, a dictionary was used. Whenever a neighbour particle j satisfied the condition of being within the cluster radius of a particle i, the particles instance name would be added to the dictionary.
- This dictionary was used to find the distances between particles within particle neighbourhoods and applied in force calculations to define subsequent positions of particles.
- All forces and vectors fields were handled using an embedded list of the individual vector coordinates of every particle in the system.

Results:

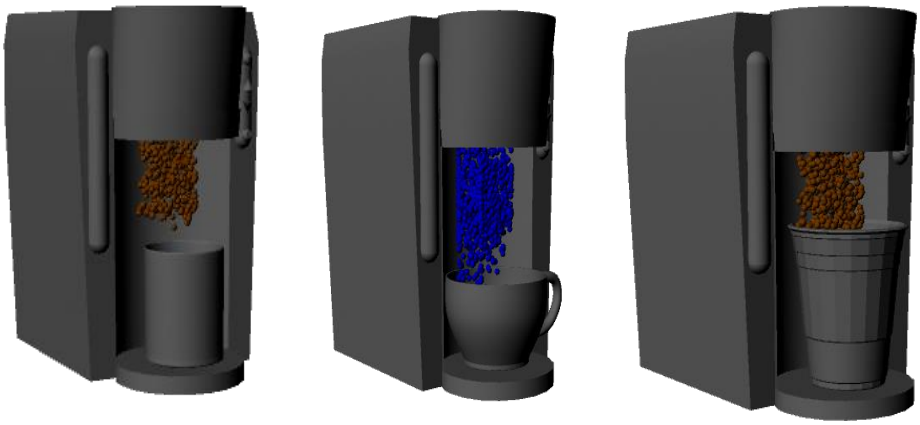


Figure 4:

general control flow of SPH implementation in python

Figure 4 shows some of the resulting liquids and cup types that can be created using this machine.

References

Strantzi, A., 2016. *Fluid Simulation Using Smoothed Particle ,* Bournemouth: s.n.