INF585 - Magical Popcorn Scene

Ayham Olleik
ayham.olleik@polytechnique.edu

Khalig Aghakarimov
khalig.aghakarimov@polytechnique.edu

March 19, 2021

Abstract

This small paper report discusses our computer animation project that tries to display a scene of a woody table where we have some bouncing popcorn from a black pan. The scene also contains some cups on the table that are filled with blue water fluid. These cup could fall when one of our super-magical popcorn hits it.

Keywords: Computer Animation, Smoothed Particle Hydrodynamics (SPH), Billboard, Fluid Animation, Texture Mapping, Collision Detection

1 Introduction

Computer Animation is a process used to transform or create digital scenes that could share very similar physical/movement concepts as a real scene. In the next theoretical section we will have a look at some of the theory used in simulating the popcorn scene.

2 Theoretical Background

2.1 Bouncing Trajectory

A common motion equation for a bouncing trajectory that could simulate moving particles in the space in real life is to add the main physical components available such as the initial position P_0 and the initial velocity vector v_0 with the gravity force g along a parabolic motion equation P_t that depends on time t. Equation 1 presents such an equation:

$$P(t) = \frac{1}{2} * g * t^2 + v_0 * t + P_0. \tag{1}$$

2.2 Collisions

2.2.1 Sphere-Plane Collision

Assuming the scene contains a sphere-like object with a certain velocity v, radius r_i and a center position p_i that can hit a certain plane obstacle with normal vector n and point a. Collision and velocity updates are applied in the following order using α and β as our restitution parameters:

- **Collision Condition** $(p_i - a).n \leq r_i$
- **Perpendicular Velocity** $v_T = (v.n)n$
- **Parallel Velocity** $v_{||} = v - v_T$
- **New Velocity** $v_{new} = \alpha v_{||} - \beta v_T$

2.2.2 Sphere-Sphere Collision

For the sphere collision we follow similar approach as the sphere-plane collision but with updating both of the velocities v_1 and v_2 of the spheres. This paper assumes an elastic collision that conserves the kinetic energy of the system. In order to do this, the respective masses of the spheres are given as m_1 and m_2:

- **Vector of two centers** $u = \frac{p_1 - p_2}{||p_1 - p_2||}$
- **J factor** $j = 2\frac{m_1 m_2}{m_1 + m_2}(v_2 - v_1).u$
- **Update first velocity** $v_1^{new} = v_1 + \frac{j}{m1}u$
- **Update second velocity** $v_2^{new} = v_2 + \frac{j}{m2}u$

2.2.3 Sphere-Cylinder Collision

Since our cup is simply a combination of a sphere and a cylinder, we can detect the collision of the popcorn (i.e. our particle) based on its current position and the usage of the equation of the circle. For a given cup of radius R, center (x_0, y_0) and height h, we can say that the popcorn will hit the cup in the following condition:

- if $(x_{popcorn} - x_0)^2 + (y_{popcorn} - y_0)^2 \leq R^2$ & $z_{circle} \leq z_{popcorn} \leq z_{circle} + h$

2.3 Billboard

Billboards are 2D elements encrusted in a 3D world. It is neither a 2D menu on top of everything else nor a 3D plane around which you can turn. It is something in-between, like health bars in many games. More precisely, for our simulation, a set of particles that are emitted regularly can be displayed as small images (i.e. quadrangles in our case) instead of simple spheres and then a semi-transparent texture can be mapped onto these quadrangles.
What's different with billboards is that they are positioned at a specific location, but their orientation is automatically computed so that it always faces the camera [2]. There are other extensions that can be achieved performing different things on the textures, such as sprites (the texture is animated) or impostors (the texture is adapted to the point of view). However, for the creation of smoke in our simulation, we decided to stick with the billboards.

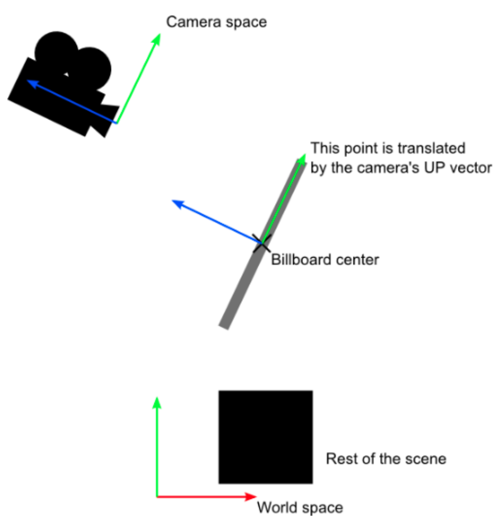


Figure 1: Billboard mechanism

2.4 Smoothed Particle Hydrodynamics (SPH) and Fluids

In the SPH theory we follow the main forces that play a role in controlling the flow of the fluid simulation. At each iteration i we compute the following:

- **Density** $\rho_i = \sum_j m_j * W_h^{poly6}(||p_i - p_j||)$
- **Pressure** $p_i = (\rho_i - \rho_j)$
- **Weight Force** $F_{weight} = m_i * g$
- **Pressure Force**$F_{pressure} = -\frac{m_i}{\rho_i} \sum_j^{j \neq i} m_j \frac{p_i + p_j}{2\rho_j} \nabla W_h^{spiky}(||p_i - p_j||)$
- **Viscosity Force** $F_{viscosity} = m_i * v \sum_j^{j \neq i} m_j \frac{v_j - v_i}{\rho_j} \triangle W_h^{spiky}(||p_i - p_j||)$
- **Time integration**: $v_i^{k+1} = v_i^k + \triangle t \frac{(F_{weight} + F_{pressure} + F_{viscosity})}{m_i}$

The mentioned function above controls the motion of the SPH particles of the fluid.

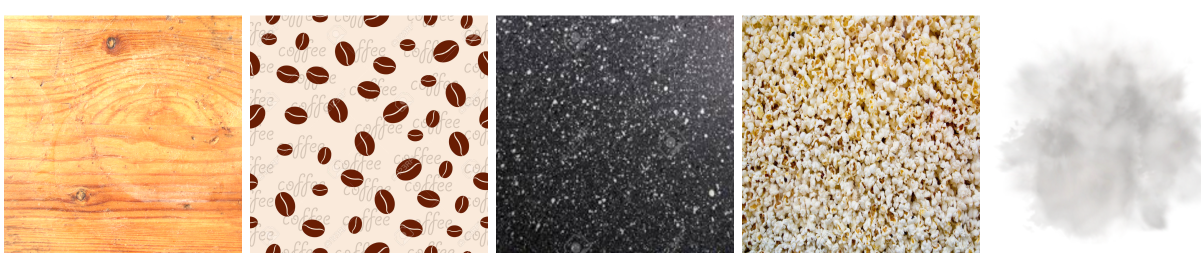
3 Algorithm Implementation and Development

3.1 Objects Used

There are several object elements that are used in the scene. These mainly include a table, a pan, 2 cups, popcorn, smoke and fluid. Some of them are drawn in terms of their mesh structures found online on the Internet (pan and table) whereas others are manually constructed (cup, for instance). Besides, to be more realistic, we decided to map an appropriate texture for each object. You can see the objects and their associated textures used in the project in the figures below:

(a) Table (b) Cup (c) Pan (d) Popcorn (e) Smoke

Figure 2: Objects



(a) woody texture (b) cup-like texture (c) pan texture (d) popcorn texture (e) smoke texture

Figure 3: Textures

3.2 Popcorn

3.2.1 Bouncing popcorn

After explaining the theory in **Section 2** we build our bouncing popcorn scene using the stone object [1] with a popcorn texture as shown in **Section 3.1** fig d. These popcorn particles follow the bouncing trajectory explained in **Section 2.2** while simulating the sphere-sphere and sphere-plane collision where the table is considered as a fixed plane obstacle at a certain z position

3.2.2 Vibrating popcorn

Vibrating popcorn are the ones that can be found inside the pan on the table (*Fig 4.c*). To be more realistic, we decided to have a slight "movement" of popcorn inside in the pan. Their structure is exactly the same as the bouncing popcorn, but different due to the action they are performing. For each displayed scene, we are trying to produce 70 popcorn in the random positions, but in the surface restricted to the pan. This way we achieve the vibration effect for the popcorn like if they are made inside the pan under heat.

3.3 Smoke

To create the smoke effect (*Fig 4.c*), we have used the billboard for which the theory is described in **Section 2.3**. Our billboard is assigned a 3D position and a certain time step. At each step of the new frame of our scene, we are randomly creating a new billboard, assigning it to the quadrangle of a predefined size and rotating them so that they face the camera orientation. The quadrangles are mapped with the smoke-like texture that you can see in *Fig 3.c*. We also do not forget to update the positions of the active billboards at each step depending on the current time. After a certain time, we end up with the removal of some billboards who have reached their lifespan. This disappearance ensures a realistic scenery for the smoke.

3.4 Cups

The two cups in our scene (*Fig 4.b*) have a very simple structure: we are using the basic built-in shapes of OpenGL such as cylinder for the body and circle for the seat of the cup. You can find the texture we have used for the cups in *Fig 3*. The peculiarity about our cups is their falling effect when either of them is hit by a popcorn. The falling procedure is implemented in 2 steps:

- Detection of the collision between popcorn and cup based on the idea described in **Section 2.2.3**
- Rotation and translation of the cups in case of the detection

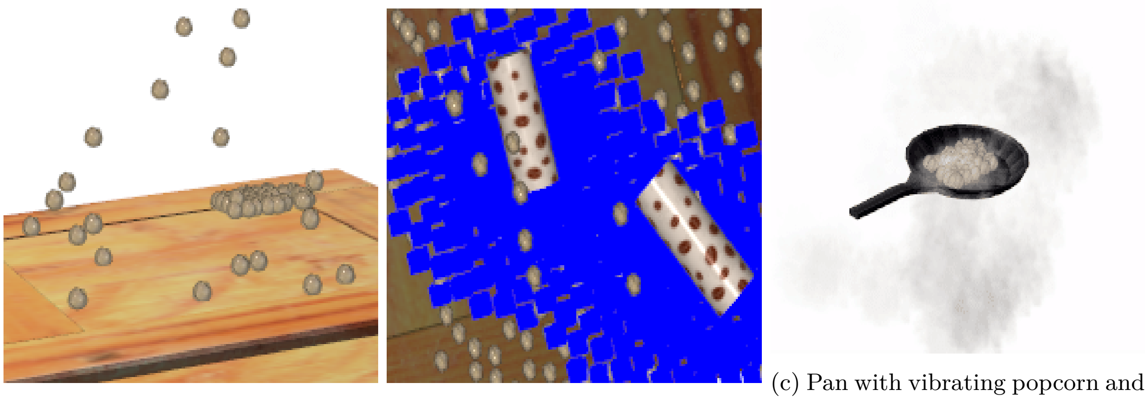
Besides, they contain blue fluid inside, similar to water, that will flow when the cup falls. This process is purely handled by the notion of SPH explained in the previous section.

3.5 Water fluid

We simulate our fluids as a grid of small cubes with blue texture that follow the SPH formula and movement forces (*Fig 4.b*). This event is only triggered in the case of a falling cup in our scene. As explained before, our magical popcorn grains can in fact push and make a glass of water fall. That's the beauty of computer animation! you create your own scene!!

4 Computational Results

For the test of our scene we run our code on a Intel Core i7-6820HQ 2.80GHz / 16Go RAM DDR3 / Nvidia Quadro M1000M and the code runs smoothly with no lag. Below you can see some results of different subscenes. In *Fig 4.a* you can see our bouncing popcorn on the woody table, in *Fig 4.b* we also have the fallen cup with water fluid that surrounds them and at the end we have *Fig 4.c* that shows the vibrating popcorn with smoke on the black pan.



(a) Bouncing popcorn (b) Fluid flowing from the cups (c) Pan with vibrating popcorn and smoke

Figure 4: Subscenes

5 Future Work

The scene we currently have can be improved to simulate a more realistic environment by adding the following possible extensions:

- Better collision formula that adds randomness popcorn collisions
- Better skins and objects for cups and popcorn
- Deal with popcorn that could hit the cup after it falls
- Add dynamic velocity update for popcorn especially when they hit the water surface on the table
- Better falling mechanism for water coming from the cup for more resolution and accuracy
- Add a fire source for the pan on the table

Many other improvements can also be added, but due to the limit of time most of these improvements were not applied to our scene.

6 Code, Demo & Mentions

Our code and demo video are available public on the following Github link. Special thanks to our great Professor Damien Rohmer for his support and guidance.

References

[1] *3D objects*. URL: https://sketchfab.com/tags/popcorn.

[2] *OpenGL tutorial, Billboards*. URL: http://www.opengl-tutorial.org/intermediate-tutorials/billboards-particles/billboards/#:~:text=Billboards.