Dynamic Fluid Particle Simulator

**Project Description:**

The finished product will be an interactive 3D fluid particle simulator that will allow the user to apply a force to the particles with the mouse. The final product will also allow the user to change the different properties of the fluid. The simulator will mimic the viscosity and movement of the fluid. The simulator can also be used by the user to make the fluid move towards a desired destination.

**Competitive Analysis:**

Based on other particle simulators I’ve seen online, most tend to be 2D and simulate water movement through rippling effect. My simulator will be in 3D and allow the users to visualize the properties from different angles. Other particle simulators also often use other applications to handle the physics and has very limited user interaction. My simulator will allow the user to manipulate the different properties of the fluid such as size,color, speed, etc. My program will also be different from other 3D fluid particle simulations because my model will allow for more user interaction such as movement of water with the mouse and movement of the water by rotating the closed system to move.

My particle simulator is similar to other particle simulations since it will still try to retain fluid properties and try to imitate the physical properties of fluids. My model, however, will be less accurate since it will not use a physics engine. Beyond the MVP phase of my program, I might also include interaction with a leap motion so that the user can manipulate the water with their hands. Another feature would include mixing different types of fluids with different properties.

**Structural Plan**:

The outline for the project will include creating a particle class, which will incorporate the properties of a fluid particle. It will also include a system class, which will hold all the properties of the closed system. Some of the methods that will be incorporated is a method to update the frame and everything inside the closed system. There will also be a method that loops over until the user presses quit and redraws the frame and handles events. There will also be an obstacle component that allows the user to move the fluid through obstacles to a desired location.

**Algorithmic Plan:**

The trickiest part of the project is figuring out the physics for the particle movement and making it as accurate as possible. Another issue is designing the structure to maximize efficiency since the program will be run on my personal mac and the computational power is limited. Working in 3D will also increase the computational work needed for the program. I will try to limit any intensive computations by reducing redundant calculations. I will also be utilizing the FLIP/HIC method that will allow me to create particles efficiently and more accurately.

Timeline Plan:

I will finish the MVP and make the physics aspects of the program more accurate by the time Thanksgiving break is over. This final MVP will include the movement of water in a closed system and allowing the water to move around an obstacle. The MVP will also allow interaction with the mouse and being able to apply forces. I will also try to add simple features such as sliders that allow the users to manipulate the properties of the fluid before Thanksgiving break is over.

The next feature that I will work on leading up to the TP3 meeting is the incorporation of a leap motion to allow movement of the water with the users hands. Another alternative feature is adding interaction between two or more fluids with different properties.

**Version Control Plan:**

I will be using github for version control and backing up my code.

**Module List:**

The MVP will only use the modules pygame and pyOpenGL. When adding additional features, the leap motion might be used.

**TP2 Update:**

Due to unforeseen complications with computational inefficiencies, the 3D simulation of water can’t meet MVP status since the additional aspects require very computationally intensive calculations that my Mac can not handle.

Therefore, I have created a 2D simulation of water in a “Build Mode,” in which the user can alternate between using ground and water tiles to simulate water. Instructions have not been added yet in the interface but pressing “s” allows the user to alternate between ground and water tiles. Pressing “p” allows the user to unpause the scene and allow water to move. Using the mouse allows the user to create the tile they are using. The user can make tiles regardless of whether the scene is paused or unpaused. The water has some properties of cohesion since if water particles hit other water particles then it takes the particle with it. The movement is realistic to a degree since it will flow down a slope if the user draws a staircase ground. Displacement has not yet been implemented so the water does not rise as more pressure is added downwards. However, there is a gravitational force and there is dispersion if the user were to spawn multiple water particles in the air. There is also a semi-working force function where if the user presses f they can apply a force to the water and make the water move away from the mouse, but there are a few bugs with how the force is implemented so the water disappears over time.

I also added a UI , which is accessible through the launch.py file, which allows the user to access build mode along with the 3D model that I spent the first few weeks working on. The 3D model only shows dispersion and has little interactions. The user is able to rotate the camera angle of the closed system but they are not able to do much beyond that. Anything additional with the 3D model would have to be a feature and not computationally intensive. I included the 3D model to show that work was done to figure out how a 3D model could potentially function.

Lastly, I included a game mode that currently has one built in stage that allows the user to erase ground blocks to move the water towards a bucket. I have currently made a proof of concept model of the game since you can erase an ‘x’ number of boxes and you can see how many water tiles remain on the screen. However, the functionality of the bucket and a score system has not been implemented yet. The game could have included features such as additional levels (likely no more than 3 or perhaps just one very comprehensive level), obstacles, and target items that the user should acquire. The bucket could also be modified so that it shows the bucket being filled up as water enters The functionalities of the build mode are also removed in game mode so that the user does not have as much freedom.

The UI systems handles transfers between the programs but does not have a “go backwards button” so if you want to operate across the different modes you need to rerun the program. The “go backwards“ button will likely be an added feature in the final version.

TP3 Update:

The main feature that has been added since TP2 is the inclusion of a stage builder feature within the 2D mode after you press m. In the stage builder mode you have access to more tiles such as ones that can’t be destroyed and ones that can absorb water. You also have the ability to erase tiles in stage builder, which allows the user to modify the level. The user may also place down collectible items and a bucket. The user must place at least one bucket and three collectibles before entering game mode. The user only needs to press spacebar to enter game mode.

The other feature that was added was the force feature, which was slightly worked on in TP2 but had to be implemented in TP3. The force essentially pushes particles away from the cursor and moves it into the ground making the water disappear. Another feature that was added was instructions for both the game mode and the build mode.

I also made it so that it is easier for the user to go back from the different modes back to the game page. If you click the close window button you are sent back to the launch page.