Jelly Physics

Soft-Body Dynamics in a Virtual Environment

AMRUTHA SRIKANTH and SANDY CHENG, University of Washington

Fig. 1. A frame from our final version with multiple soft-body objects colliding in the user's field of view, available for interaction

We created a VR experience that allows the user to interact with virtual soft-body objects using head tracking through the headset and hand movement through the use of hand controllers. The main technical challenges we faced in this process were realistically simulating the physics of suspended soft-body objects and the interactions between the objects and the user. We consulted a few articles and papers to familiarize ourselves with the concept of soft-body dynamics and possible applications of soft-body objects, including the physics behind soft-body objects and examples of applications of soft-body rendering. (R.1, R.2 in References). While we were able to simulate jelly-like behavior on object meshes, it was difficult for us to adapt this experience to include hand-controllers. In the future, we would want to adapt this program to render object collisions and squishing that is more graceful and gradual, like the desired movement of the jellyfish-like woodsprites from Avatar, or adding more customized movement than the vertex 'shake' (such as having different amounts of drag on each jellyfish tentacle).

1 INTRODUCTION

While the concept of an experiential soft-body-based VR program like the one we want to create is nothing new, it is a very interesting study in programming the physics of soft-body objects and allowing the user to interact with virtual objects in various ways. In this case, we want to use this opportunity to explore the physics associated with soft-body objects in a virtual space and create an environment where users can interact with these objects and move them around in various ways. This project was initially inspired by the Pandora scene, with floating jellyfish-like woodsprites floating in the bioluminescent forest.

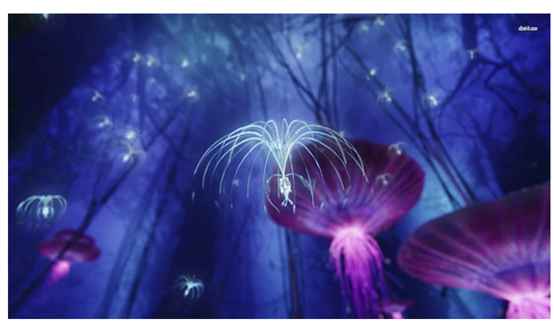

Fig. 2. A jellyfish-like woodsprite from Avatar

When building our virtual environment, we focused on the creation and collision of soft-body objects, with additional stretch goals of

Authors' address: Amrutha Srikanth, asrika@cs.washington.edu; Sandy Cheng, sandyc01@cs.washington.edu, University of Washington.

adding movement simulations for soft-body objects suspended in water (the way jellyfish would be). In order to create this virtual environment, we had to confront and deal with various technical challenges including realistically simulating the physics of soft-body objects moving about as a semi-fluid object in the air. We also had to apply the collision system for various object interactions and accurately render these objects in a virtual environment, where the user's movements have an impact on their location and orientation. We would need to take into account the user's movements and track these changes to utilize in our physics system.

The challenge with this project was mainly in getting convincing soft-body physics with realistic deformation from hand movement and simulating the movement of the objects (for example, it may rotate a little if it is suspended in fluid and the fluid itself is moved). By creating a collision system that allowed the object to deform appropriately, we would be able to create objects with jelly-like behavior. The hope of this project is to create a relaxing, atmospheric VR art piece for users, while giving us a chance to delve into and implement soft-body physics (as well as 3D modeling and possibly a bit of fluid dynamics.)

1.1 Contributions

- Writing a script to implement jelly-like behavior for the specified object mesh when it is moved or a collision occurs
- Adding hand controller-based interactions to the virtual experience to move around the jelly objects
- Building the ocean environment in Unity and adding jellyfish objects to complete the design

2 RELATED WORK

This project is a study in programming the physics of soft-body objects and allowing the user to interact with virtual objects in various ways. We consulted a few articles and papers to familiarize ourselves with the concept of soft-body dynamics and possible applications of soft-body objects, including the physics behind soft-body objects and examples of applications of soft-body rendering. (R.1, R.2 in References). These papers helped us better understand the math behind soft-body objects and the applications of soft-body dynamics in various fields. The main paper we referenced was "Physically-Based Deformable Models in Computer Graphics" by Andrew Nealen and Matthias Muller, which was a State of the Art Report describing the state of soft-body dynamics at the time.

Currently, there are several VR art projects that feature relaxing scenes of jellyfish (or creating and popping bubbles), but none that we are aware of that feature other user interactions with the soft bodies' movement. The hand controller interactions in our project set our work apart from the other projects and experiences created in the past. Some art projects also involve a soundscape, such as https://www.youtube.com/watch?v=SnqubYkGlR8.

3 METHOD

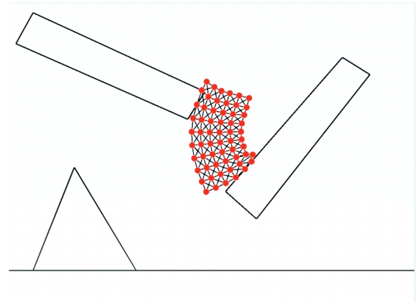

Fig. 3. A soft-body mesh deforming because of outside forces (from Youtube Video R.3 in References)

Our original goal was ambitious, to create an atmospheric experience with jellyfish floating in fluid, responding to currents caused by the user's hand motions. Through our research, we found that simulating a fluid environment was computationally intense, and this would be compounded by the limited resources of standalone VR headsets. Additionally, while the process of creating and rendering soft-body objects is well-documented, it is more difficult to find information on building a fluid environment. We decided to imitate these fluid interactions for the soft-body objects in a zero-gravity environment, where these objects would float around the user and be interactive, in order to save computational power and achieve the desired effect. We decided to break the project into a series of steps, with each step increasing in complexity, starting with rigid-body objects and escalating to soft-body collisions.

We started by experimenting with rigid bodies and rigid body collisions since rigid-body objects have very simplistic collisions. We used rigid-body objects to set up our scene and get more comfortable with working in Unity. We then replaced the single rigid-body object with a soft-body object by writing a script to make sure mesh deformation behaved as expected and made the object "jelly-like".

To imitate soft-body movement, we created a "shake" function that would shake the cubes when they collided with other objects, or moved.
The following equations were used for calculating force, velocity, and position: Force is given by Hooke's Law for springs,

$$F = k \cdot (x - x_0)$$

This worked to create simple harmonic motion in the cube, when it was moved — the further it is deformed, the further each vertex is displaced, and the more exaggerated the motion. In the equation above, x is the target position, x_0 the initial position, and k a stiffness parameter.

Assuming that all timesteps are one unit, velocity is calculated using an approximation, to reduce compute power in the headset:

$$v = (v_0 + F/m) \cdot d$$

where v is the new velocity, v_0 the velocity at the earlier step, and F/m acceleration, given by the force calculated earlier, divided by mass. d is a damping parameter, and it is less than 1, to reduce the velocity, proportionally. Position was updated similarly, as

$$x = x_0 + v$$

If velocity, acceleration, and the force falls below a threshold of 0.001, the position is set to the target.

This script was applied to a single cube (with the equations applied to each vertex), and replicated to fill a bounding box surrounding the user.

4 IMPLEMENTATION DETAILS

The hardware we used to test and view our virtual experience was the Oculus Quest 2. We used Unity and C# to implement the experience itself, with some prefabs from the XR Interactions Toolkit (namely, the starter assets set).

In order to implement our virtual jellyfish experience, we started by adding a rigidbody component to each jelly, for basic rigid-body interactions. This gave us a scene while we were familiarizing ourselves with Unity. We then added a simple softbody, using a mass-spring model that involved the eight original vertices of each cube.
After testing the game and tuning the parameters, the ones that worked best were 0.1 for stiffness, and 0.75 for damping.

We then populated the scene with more soft-body objects and tested it multiple times to ensure that the soft-body objects interacted and collided realistically with each other. Finally, we added hand-controller interactions so the user could interact with the soft-body objects and move them around. This way, the user would also be able to knock into the soft-body objects and push them around the scene. We used the Unity XR Interaction Toolkit to implement the hand controller interactions in our project.

The scene was originally built and test on a computer, so some adjustments had to be made when building to the Oculus:

- **The jelly objects would float away**, in the gravity-free environment. To adjust, we created a bounding-box, to keep them from floating too far. (Increasing drag was not enough to stop the continued dispersion of the objects.)
- **The model scale was too large**. The bounding-box was shrunk until it allowed for a 1.5-m radius around the user, to allow for the illusion of being in open space while keeping the jelly-like objects close enough to reach.
- **The headset had limited computing power**. From building and testing, 70 cubes caused the headset visuals to lag significantly, so we increased the sizes of each jelly object and found that 15 or fewer still populated the space well, without significant lags.

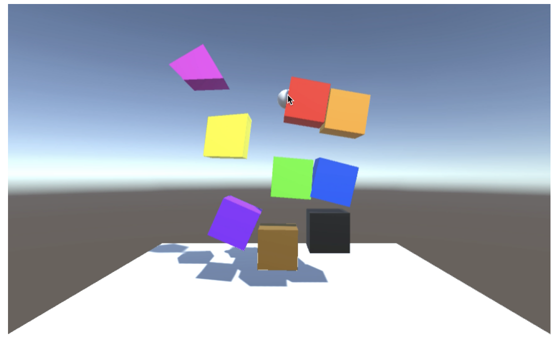

Fig. 4. Multiple soft-body cubes colliding and squishing in a no-gravity environment

5 EVALUATION OF RESULTS

While we had to simplify our goal to meet with the time constraints, we succeeded in building a virtual environment filled with floating objects with jelly-like behavior ('jellies') that users could interact with to move around. We evaluated the work qualitatively, in seeing whether the behaviors appeared realistic as a user. The cuts in the number of jellies was successful in making the program run smoothly with significantly less lag, and the user is able to reach the jellies, after we altered the bounding box to be closer to (but still slightly larger than) the user's arm range.

Limitations include that populating a scene with many such "jelly" objects will be computation-heavy, especially when the objects have more complex meshes with more vertices (70 cubes was overwhelming, but we realized that 15 was still quite enough to populate the space.)

Usage of simple physics equations to calculate mesh deformation may not work for more complex meshes where vertices can collide.

6 CONCLUSION

The current behavior of the "Jelly" Objects is similar to that of Jello. In the future, we would want to adapt this program to render object collisions and squishing that is more graceful and gradual, like the desired movement of the jellyfish-like woodsprites from Avatar, or adding more customized movement than the vertex 'shake' (such as having different amounts of drag on each jellyfish tentacle). Our main goal with this virtual experience was to create an interactive environment that would engage and calm users.

Additionally, having a better grasp on hand-controller interactions would elevate our project and allow it to be interactive in a 3D space. While we have a good grasp on the soft-body physics required for the calm floating "Jelly" objects, decorating the rest of the environment with other ocean elements and adding calming audio elements would help fulfill our initial goal.

ACKNOWLEDGMENTS

Thanks to Douglas Lanman and the CSE 493V TAs for their guidance and support throughout the course and this project.

Thanks to John Akers and UW Reality Lab for supplying us with the Oculus Quest 2.
The jellyfish model we used is Simple Jellyfish by Rick Stikkelorum, containing 557 vertices, and made up of four main structures (the cap, and bones for each of the three tentacles)

REFERENCES

R1. "Physically Based Deformable Models in Computer Graphics", Andrew Nealen, Matthias Müller, et.al, 2005.
R2. "Modeling And Simulation Of Soft Bodies", Jaruwan Mesit, 2010.
R3. "But How DO Soft Body Simulations Work?" (Youtube Video), Gonkee, 2021.
R4. "Discontinuous Fluids", Jeong-Mo Hong, Chang-Hun Kim, 2005.