Interactive Particle Simulation

06/30/2024

1 Introduction

Particle simulations have become an increasingly popular tool in various fields, from computer graphics and visualization to scientific research. These simulations allow for the realistic modeling of fluid dynamics, granular materials, and other complex systems. In this research paper, we present an interactive particle simulation that allows users to adjust various parameters to create different visual effects.

2 Methodology

The interactive particle simulation was developed using HTML5 canvas and JavaScript. The simulation consists of several key components:

2.1 Particle Properties

Each particle in the simulation has the following properties:

- Position (x, y)
- Velocity (vx, vy)
- Initial position (initialX, initialY)
- Color (color, subColor)

These properties are stored in an array of particle objects, which is used to update and render the particles throughout the simulation.

2.2 Particle Interaction

The particles in the simulation interact with each other and the mouse cursor based on the following rules:

1. Smooth return to initial position: When the mouse cursor is not interacting with the particles, the particles smoothly return to their initial positions. This is achieved by applying a return force proportional to the distance between the particle's current position and its initial position.

2. Repulsion: When the mouse cursor enters the repulsion radius of a particle, the particle is repelled from the cursor with a force proportional to the distance between the particle and the cursor. This creates a dynamic and interactive simulation, where the particles respond to the user's mouse movements.

3. Boundary wrapping: If a particle goes beyond the canvas boundaries, it wraps around to the opposite side of the canvas. This ensures that the particles remain within the simulation area and creates a continuous, looping effect.

The particle interaction logic is implemented in the 'updateParticles()' function, which is called in a loop using 'requestAnimationFrame()' to update the particle positions and render the simulation.

2.3 User Interface

The user interface of the simulation includes a settings bar that allows the user to adjust the following parameters:

- Particle count
- Particle size
- Particle speed
- Repulsion radius
- Repulsion force
- Return force

The settings bar can be toggled on and off using a button in the top-left corner of the canvas. When the settings bar is visible, the user can adjust the various parameters using the provided sliders. The changes are immediately reflected in the simulation, allowing the user to experiment and create different visual effects.

2.4 Code Flow

The flow of the interactive particle simulation code can be summarized as follows:

1. Initialize the canvas and particle properties: The code starts by getting the canvas element, setting its size to the window size, and creating an array of particle objects with random initial positions and properties.

2. Define particle interaction functions: The code includes functions to draw the 5-pointed star particles, update the particle positions based on the mouse interaction, and limit the particle velocities to prevent erratic movement.

3. Set up event listeners: Event listeners are added to the document to track the mouse position and interaction state (whether the mouse is interacting with the particles or not).

4. Implement the settings bar: The code sets up the settings bar, including the buttons and sliders for adjusting the various simulation parameters. Event listeners are added to the settings inputs to update the corresponding parameters.

5. Update the particle simulation: The 'updateParticles()' function is called in a loop using 'requestAnimationFrame()'. This function updates the particle positions based on the current mouse interaction and simulation parameters, and then renders the particles on the canvas.

6. Toggle the settings bar: Event listeners are added to the settings button and close button to show or hide the settings bar when clicked.

This flow of the code ensures that the interactive particle simulation is responsive to user input, visually appealing, and easy to customize through the settings bar.

3 Results and Discussion

The interactive particle simulation allows users to create a variety of visual effects by adjusting the various parameters. For example, increasing the particle count and repulsion force can create a more chaotic and dynamic simulation, while decreasing the return force can result in the particles staying in their new positions after being repelled by the mouse cursor.

The simulation also features colorful 5-pointed star particles with a sub-color effect, which adds to the visual appeal of the simulation. The smooth return to initial position and the boundary wrapping create a captivating and seamless visual experience for the user.

4 Conclusion

In this research paper, we have presented an interactive particle simulation that allows users to explore and create different visual effects by adjusting various parameters. The simulation was developed using HTML5 canvas and JavaScript, and features smooth particle interactions, boundary wrapping, and a user-friendly settings interface.

The code flow of the simulation ensures that the particles respond dynamically to user input, while the settings bar provides a convenient way for users to customize the simulation to their liking. The visual effects created by the simulation can be used in a variety of applications, from data visualization to artistic and creative projects.

Future work on this project could include the addition of more advanced features, such as particle collisions, fluid dynamics, and the ability to save and load custom simulation configurations.

5 References

1. Experiments with Google: Particles
2. Particle Simulation - an overview — ScienceDirect Topics
3. Particle Simulation Omniverse Extensions latest documentation
4. Particle Physics Simulator - Apps on Google Play
5. 3D particles - Experiments with Google