30th, Nov, 2023

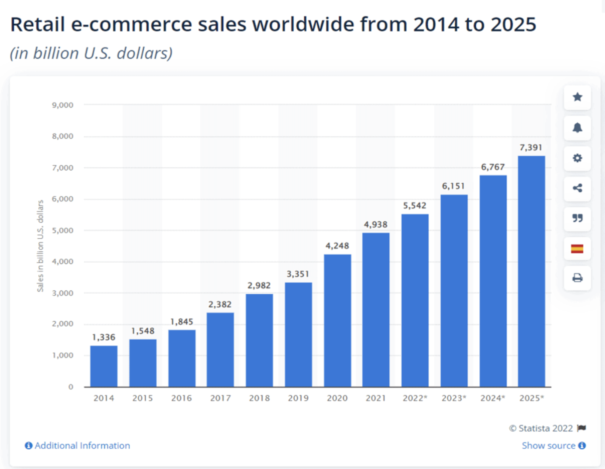
Graduation Thesis

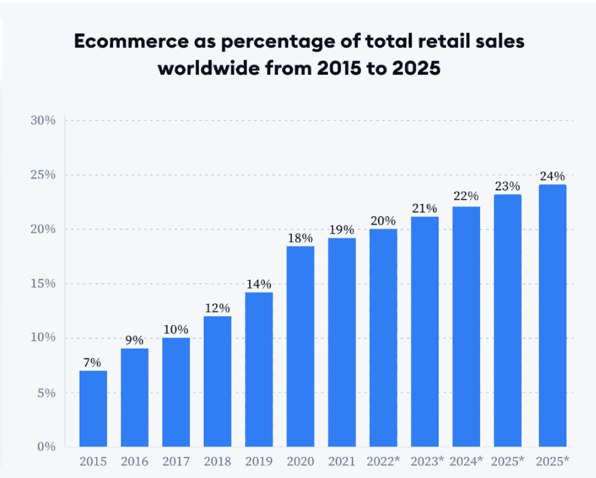
FlexiPlay : Web-based Interactive 3D Cloth Simulation

20171194 Yujin Lee

I. Introduction

In the dynamic landscape of global commerce, the paradigm shift toward e-commerce is unmistakable. Major brands are strategically establishing virtual storefronts, and an increasing number of retailers are exclusively embracing digital sales. Projections for the year 2025 indicate a staggering retail e-commerce sales worldwide market size of 7,391 billion USD (Retail E-commerce Sales worldwide from 2014 to 2025, Statista, 2022), representing a substantial 24% of the total global retail sales (E-commerce as percentage of total retail sales worldwide from 2015 to 2025, TIDIO, 2022).





While the convenience and accessibility of online shopping are undeniable, a persistent challenge looms large—its inherent incapacity to replicate the tangible, physical engagement with desired products. Navigating web interfaces, consumers are confined to visual and auditory experiences, fostering doubts and compromising confidence in online transactions. This challenge is accentuated when considering recent strides in user interaction, such as 3D model manipulation. However, even these advancements fall short, notably in capturing the authentic sensation of stretching.

This research endeavors to bridge this critical gap by introducing a sophisticated web-based system. Our objective is to empower users with an immersive and tactile experience, specifically allowing for the **intuitive simulation of 3D cloth elasticity on the Web**. In doing so, we seek to elevate the online shopping encounter, assuage concerns related to the authenticity of virtual interactions, and contribute to a more sophisticated and confidence-inspiring digital retail environment. As we navigate the evolving contours of the global retail landscape, this research serves as a timely exploration of enhancing user engagement and satisfaction in the ever-expanding domain of e-commerce.

II. Literature Review

A) Cloth Material Properties

Understanding the mechanical behavior of cloth materials involves key parameters, such as stress (σ), which denotes the force applied per unit area, measured in Pascals (Pa). In tandem, strain (ε) captures the ratio of a material's change in length to its original length, expressing the deformation experienced under stress. Calculating Young's Modulus (E) further illuminates a material's stiffness, representing the stress-to-strain ratio within the elastic region of the stress-strain curve. This fundamental metric provides insight into how a material responds to applied loads. Complementary to Young's Modulus, the Ultimate Tensile Strength (UTS) stands as a pivotal measure, representing the maximum stress a material can endure before fracture. Delving deeper, the distinctions between elastic and plastic deformation, alongside phenomena like necking, contribute to a comprehensive comprehension of a material's mechanical response under tensile conditions.

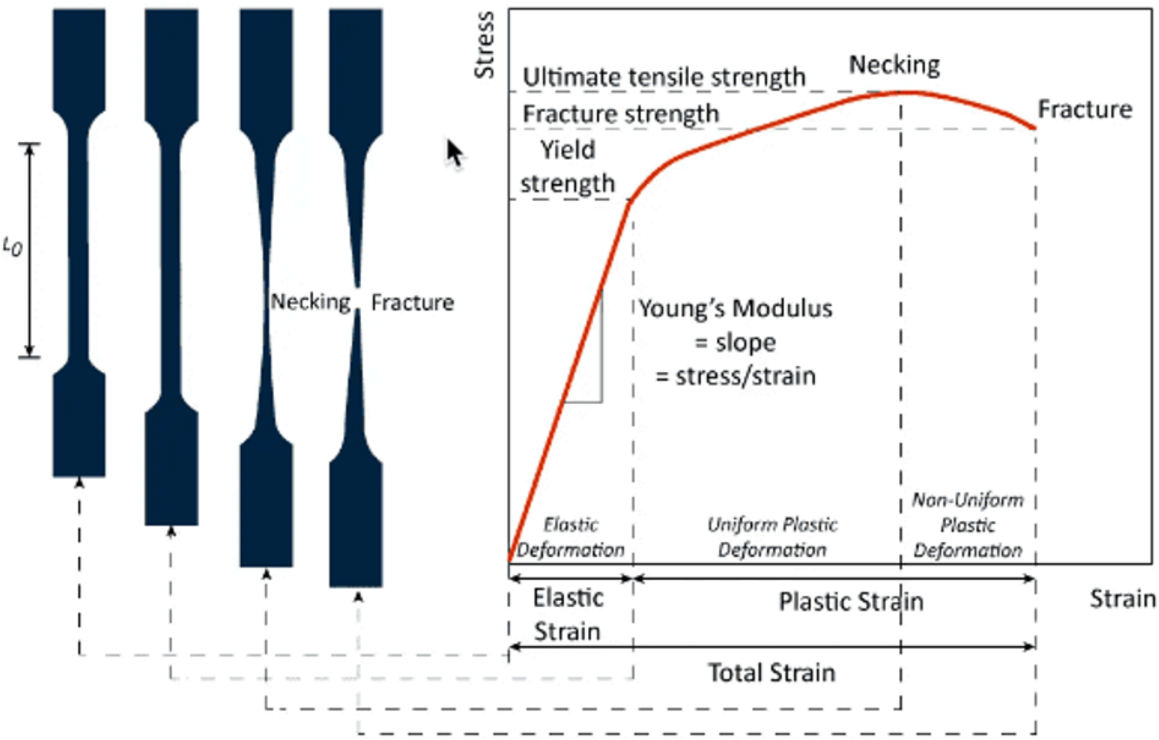


Figure I. Tensile properties of Textile Fibers
(Retrieved from textilestudycenter.com, 2023)

- **Stress (σ)**: Stress is the force applied per unit area, measured in Pascals (Pa). In tensile testing, stress is calculated as the applied force divided by the original cross-sectional area of the specimen.
- **Strain (ε)**: Strain is the ratio of the change in length of a material to its original length. It represents the deformation of a material under stress and is often expressed as a percentage.
- **Young's Modulus (E)**: Young's Modulus is a measure of a material's stiffness and is

calculated as the ratio of stress to strain within the elastic (linear) region of the stress-strain curve. It quantifies how much a material will deform under a given load.

- **Ultimate Tensile Strength (UTS)**: UTS is the maximum stress a material can withstand without breaking. It is a critical parameter indicating the material's overall strength.
- **Elastic Deformation vs. Plastic Deformation**: In the elastic region, materials return to their original shape after the applied force is removed. Beyond the elastic limit, plastic deformation occurs, resulting in permanent changes to the material's shape.
- **Necking**: Necking is a localized narrowing of a material during tensile deformation, typically occurring just before fracture. It is a phenomenon observed in ductile materials.

B) Examples of Products with Elastic Cloth

Elasticity visualization can enhance the shopping experience for a variety of e-commerce products, but it is particularly **valuable for products where fabric and material characteristics,** as well as fit and comfort, are essential considerations. Some types of e-commerce products for which elasticity visualization can make a significant difference are :

- **Footwear** : Shoes and sneakers, especially those with stretchable materials or designs that affect comfort and fit.

- **Undergarments and Hosiery** : Bras, underwear, and shapewear, as elasticity plays a significant role in support and fit. Tights, stockings, and socks, where elasticity determines how well they stay in place and conform to the legs.

- **Home Textiles** : Bedding, blankets, and towels, where customers may want to understand the feel and durability of the fabric.

- **Maternity and Baby Products** : Maternity wear and baby clothing, where comfort and flexibility are vital for both expectant mothers and infants.

- **Outdoor Gear** : Tents, sleeping bags, and camping gear, where elasticity can impact durability and comfort.

- **Medical Products** : Compression garments and orthopedic braces, where proper fit and support are critical for health and comfort.

- **Clothing and Apparel** : Clothing items such as dresses, jeans, shirts, and lingerie, where the fit, drape, and stretch of the fabric are crucial for customer satisfaction. Sportswear and activewear, where elasticity and flexibility affect performance and comfort.

- **Furniture Upholstery** : Sofas, chairs, and cushions, where customers may want to visualize how the fabric stretches and conforms to their body.

- **Smart Clothing and Wearables** : Wearable technology integrated into clothing, where elasticity affects the placement and comfort of sensors and devices.

III. Methodology

A) System Architecture

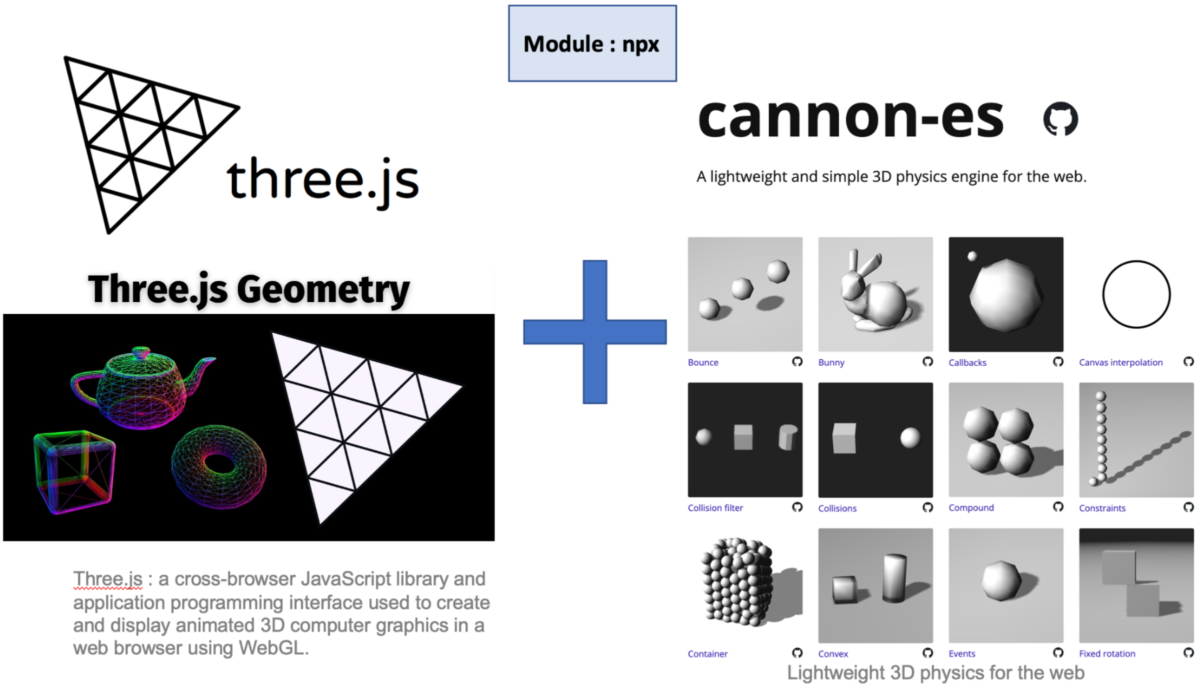


Figure II. System Architecture

The system architecture is designed with a robust and efficient framework, utilizing key technologies to ensure seamless 3D cloth interaction simulation. At its core, the system relies on the powerful **Three.js library**, providing a comprehensive set of tools for creating and rendering 3D content in web browsers. This choice facilitates a rich and dynamic visual experience, enhancing user engagement. The simulation is optimized through the use of **npx**, a package runner tool that simplifies the execution of scripts and commands. Leveraging npx ensures streamlined development processes and efficient management of dependencies, contributing to the overall agility of the system. For physics-based cloth simulation, the system integrates **cannon-es**, a lightweight physics engine designed for use in web applications. This engine enables realistic simulating of cloth behavior, including elasticity and deformation, along with realistic settings such as gravity field..

In summary, these technologies form a cohesive system architecture, with **Three.js** handling the visual rendering, **npx** streamlining development workflows, and **cannon-es** providing the underlying physics engine for accurate cloth simulation. This integration ensures a powerful, responsive, and immersive platform for users to intuitively interact with 3D cloth simulations on the web.

B) Cloth Simulation Algorithm

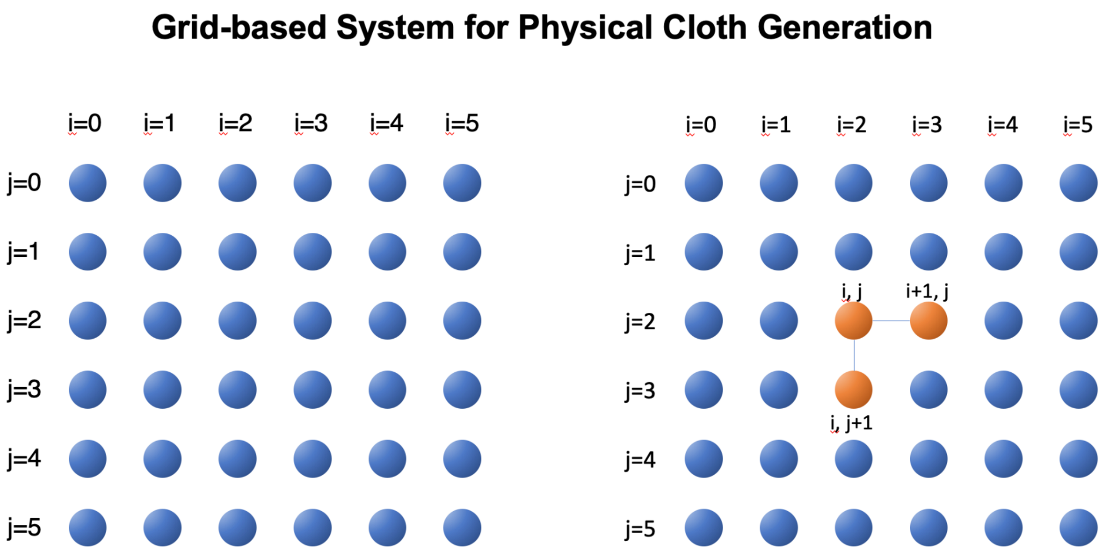


Figure III. Grid-based Physical Cloth Generation

The Cloth Simulation Algorithm employed by the system leverages a **grid-based approach**, where planes are constructed with particles arranged in a matrix, defined by the two parameters, i columns and j rows. Each individual particle within this grid is strategically connected to its adjacent neighbors, fostering a cohesive structure that emulates the interconnected nature of fabric. Specifically, the $(i,\ j)$th particle is linked to its neighboring $(i\ +\ 1,\ j)$th and $(i,\ j\ +\ 1)$th particles.

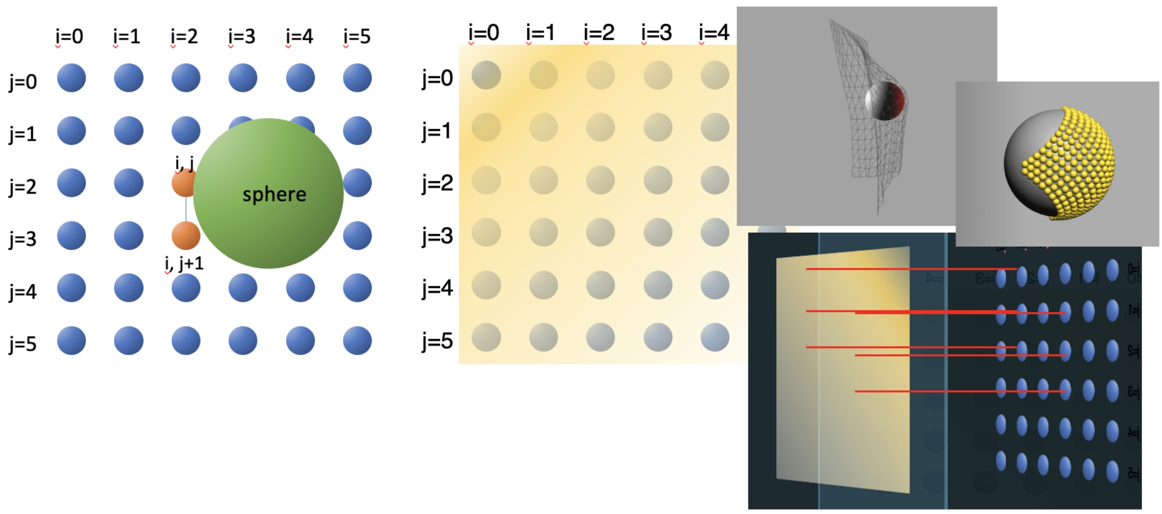


Figure IV. Grid-based Physical Cloth Generation

To emulate the physical behavior of cloth, each particle is assigned a **weight**, and these particles possess the capacity for dynamic interaction with other objects within the simulation space. This interaction is governed by the implementation of a **distance constraint**, effectively limiting the elongation or compression of the material. The distance constraint serves as a crucial parameter, determining the allowable length between adjacent particles and thus regulating the overall structural integrity of the simulated cloth.

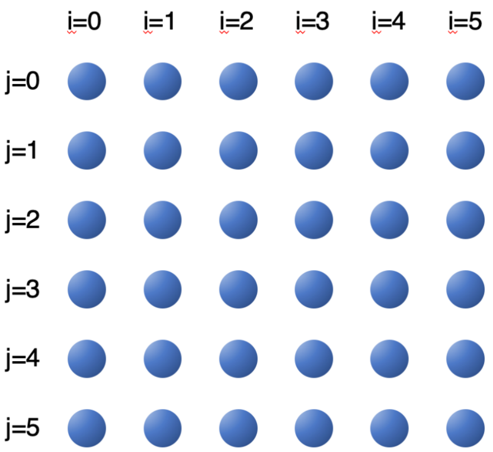
The user interaction is implemented by creating a 3d sphere and letting the user control the z-position of the sphere. And finally the cloth is visually rendered in a form of 3d cloth (not a group of particles) by creating a new geometry using each particles' position.

This algorithmic foundation, rooted in **particle-grid connectivity and distance constraints**, facilitates the realistic representation of cloth elasticity and deformation. As particles respond to external forces and influences, the system dynamically simulates the intricate interplay between individual particles, thereby offering an immersive and visually authentic 3D cloth interaction experience on the web.

C) Implementation

The code implementation step by step is presented below with visual aids. Full code is implemented in https://github.com/yuujinleee/FlexiPlay .

- Step 1 : Grid generation (Cloth particles)



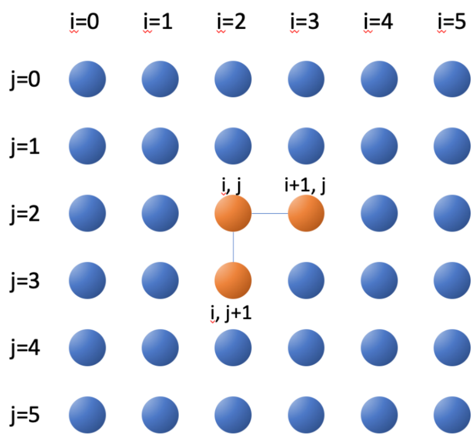

```
const particles = [];
// each particle : physical body (CANNON.Body)
```

```
const particle = new CANNON.Body({
  // mass: j == rows || (j == 0) | (i == cols) || i == 0 ? 0 : mass,
  mass: j == rows ? 0 : mass,
  shape: new CANNON.Particle(),
  position: new CANNON.Vec3(
    (i - cols * 0.5) * dist,
    (j - rows * 0.5) * dist,
    0
  ),
  material: particlePhysMat,
});
```

```
                                                              scripts.js:191
(16) [Array(16), Array(16), Array(16), Array(16), Array(16), Array
(16), Array(16), Array(16), Array(16), Array(16), Array(16), Array
(16), Array(16), Array(16), Array(16), Array(16)] i
 ▶ 0: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 1: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 2: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 3: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 4: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 5: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 6: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 7: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 8: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 9: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bod
 ▶ 10: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
 ▶ 11: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
 ▶ 12: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
 ▶ 13: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
 ▶ 14: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
 ▶ 15: (16) [Body, Body, Body, Body, Body, Body, Body, Body, Body, Bo
   length: 16
 ▶ [[Prototype]]: Array(0)
```

Figure V. Grid Generation and regarding code

- Step 2 : Connecting neighbor particles and setting the distance constraint

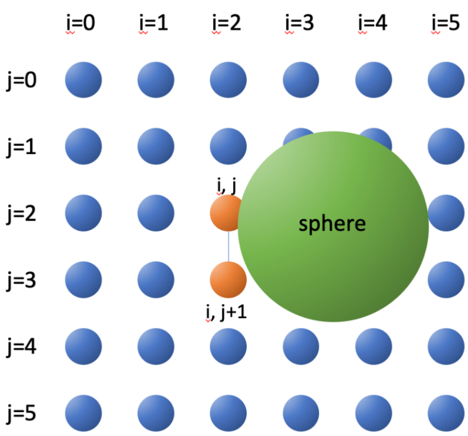


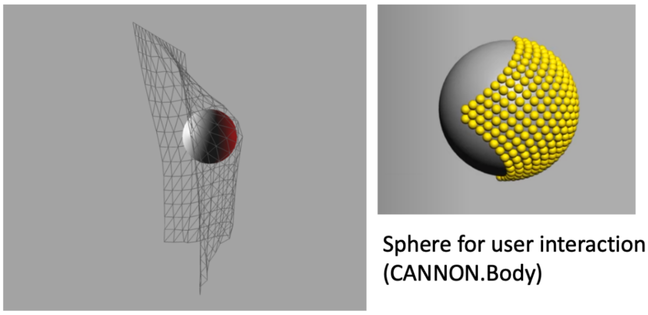

```
function connect(i1, j1, i2, j2) {
  world.addConstraint(
    new CANNON.DistanceConstraint(
      particles[i1][j1],
      particles[i2][j2],
      dist * distOffest
    )
  );
}

for (let i = 0; i < cols + 1; i++) {
  for (let j = 0; j < rows + 1; j++) {
    if (i < cols) connect(i, j, i + 1, j);
    if (j < rows) connect(i, j, i, j + 1);
  }
}
```

const distanceConstraint = new CANNON.DistatnceConstraint(ParticlaA, ParticleB, restDistance)

Figure VI. Particle Connection and regarding code

- Step 3 : Creating new 3d sphere for user interaction and collision





Sphere for user interaction (CANNON.Body)

```
// Sphere Geometry (Ball)
const sphereGeometry = new THREE.SphereGeometry(sphereSize)
const sphereMat = new THREE.MeshPhongMaterial();
sphereMesh = new THREE.Mesh(sphereGeometry, sphereMat);
scene.add(sphereMesh);

const spherePhysMat = new CANNON.Material();
const sphereShape = new CANNON.Sphere(sphereSize * 1.3); //
sphereBody = new CANNON.Body({
  mass: 0,
  shape: sphereShape,
  material: spherePhysMat,
});
world.addBody(sphereBody);
```

Figure VII. 3d Sphere generation and regarding code

- Step 4 : Visually rendering the 3d cloth by creating new geometry

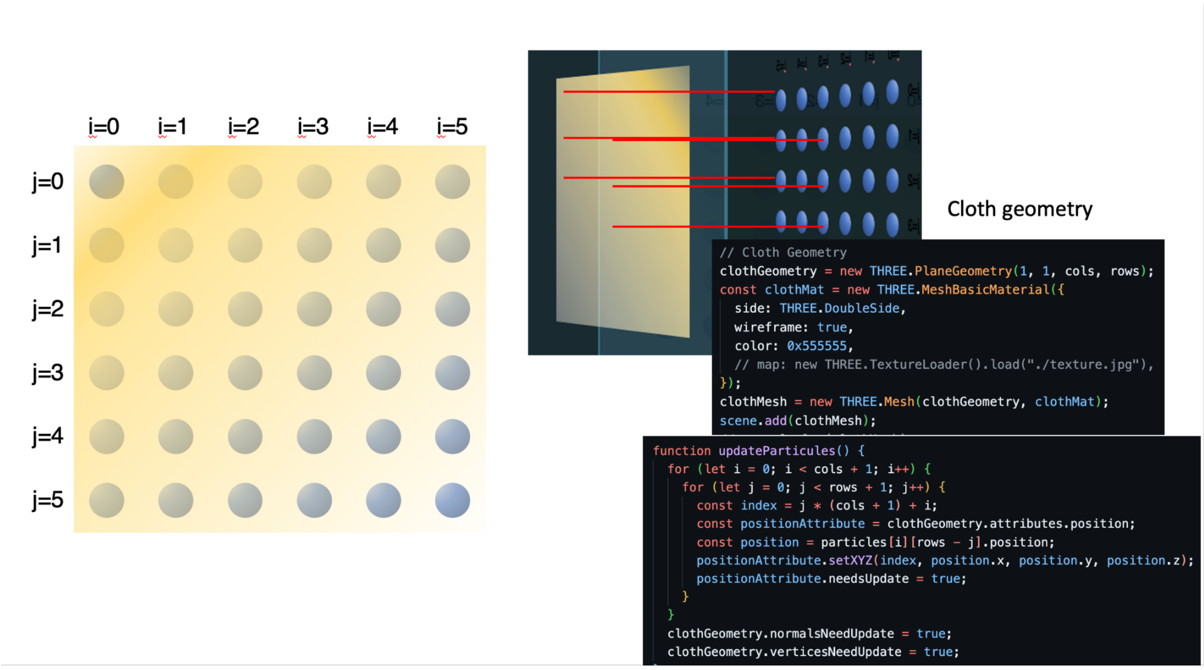


Figure VIII. 3d cloth generation and regarding code

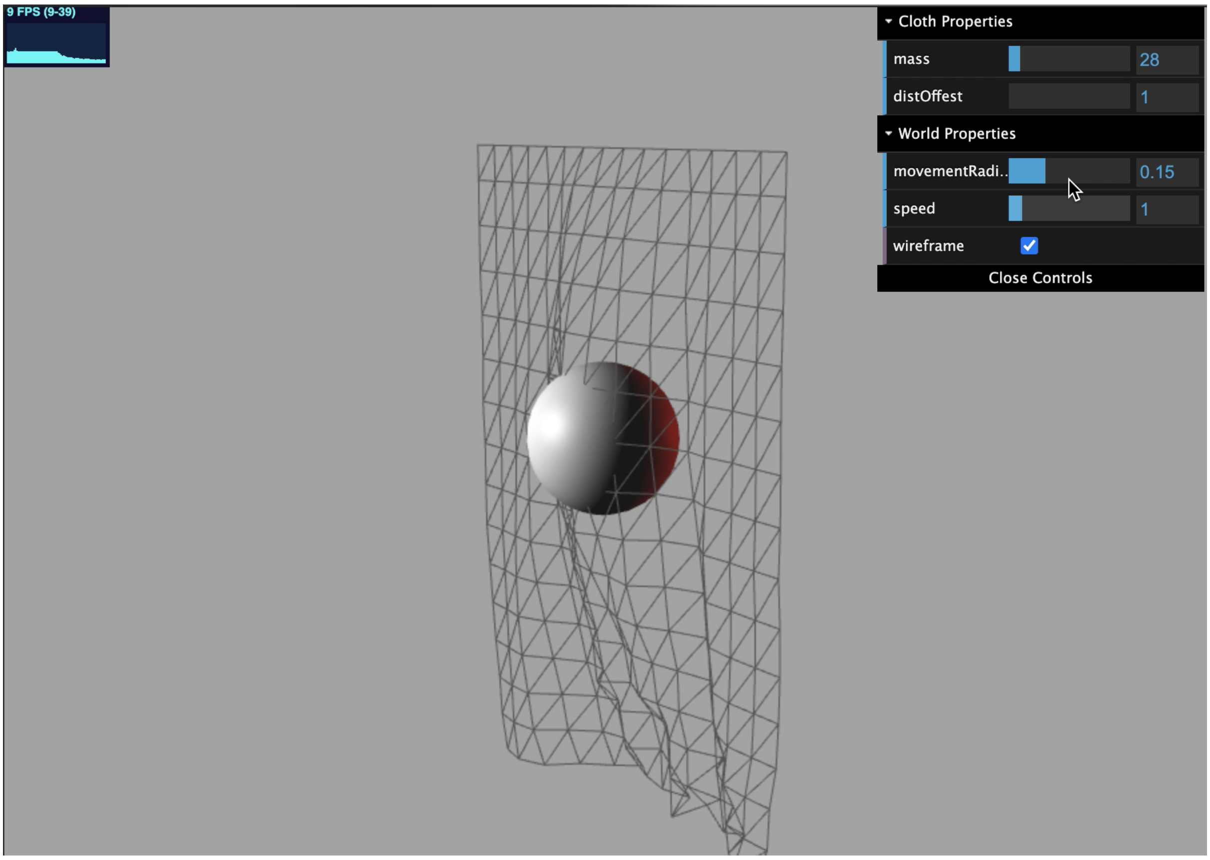


Figure IX. Screenshot of developed system

IV. Results

In the evaluation of the implemented web-based 3D cloth interaction simulation system, a multifaceted analysis was conducted to comprehensively assess its performance and capabilities. The assessment spanned performance testing to evaluate the system's responsiveness and speed, as well as a comparative analysis to measure the developed system against existing solutions in the domain.

- **Performance Testing**

Performance testing was executed to scrutinize the system's responsiveness under varying conditions. The simulation consistently exhibited real-time responsiveness with less than 100fps, even under increased user interactions and dynamic adjustments of cloth parameters. This robust performance underscores the efficiency of the implemented algorithm and the seamless integration of Three.js, npx, and cannon-es. The system's ability to maintain a high level of responsiveness contributes to a fluid and immersive user experience.

- **Comparative Analysis**

In comparing the developed web-based 3D cloth interaction simulation system with Blender's cloth simulation capabilities, we could find several key distinctions and strengths.

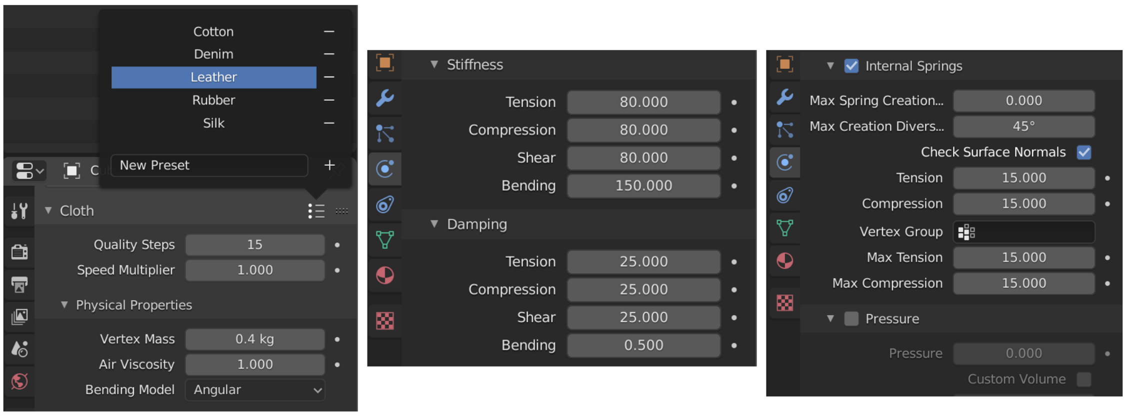


Figure X. Cloth Simulation options of Blender

1) Physical Property Customization

Blender's cloth simulation system has a comprehensive set of physical property customization options, allowing users to fine-tune parameters such as tension, compression, shear, bending, and internal springs. This various control factors over cloth behavior provides a sophisticated level of realism, helping the diverse needs of users engaged in virtual cloth simulation. Contrastingly, the developed web-based system offers a simple GUI with 4 options without an exhaustive array of physical property customization.

2) Material Presets

Blender elevates user convenience through the provision of material presets, offering predefined settings for various cloth types such as Cotton, Denim, Leather, Silk, and Rubber. These presets facilitate rapid prototyping and experimentation, saving users valuable time in achieving specific material aesthetics. In contrast, the developed web-based system currently lacks predefined material presets.

V. Discussion

The development of the web-based 3D cloth interaction simulation system, while exhibiting notable strengths, also presents certain limitations that warrant consideration for future enhancements and refinements.

1) Limited Cloth Property Customization Options:

One primary limitation is the constrained range of options available for customizing cloth properties compared to sophisticated tools like Blender. Unlike Blender's extensive suite of parameters encompassing tension, compression, shear, bending, and internal springs, the current system offers a more streamlined approach. This limitation arises from a gap between theoretical physics properties and the method developed system used to generate a 3d cloth.

2) Absence of Material Presets:

Another noteworthy limitation is the absence of material presets in the developed system. Blender's cloth simulation system offers a valuable feature in predefined material presets such as Cotton, Denim, Leather, Silk, and Rubber. These presets expedite the cloth design process, allowing users to achieve specific material characteristics efficiently. The current system, as of now, does not incorporate this feature, which might impact users' ability to rapidly prototype and experiment with various cloth types.

VI. Conclusion

In conclusion, this research has introduced a grid-based approach to 3D cloth generation in a web-based simulation system. By strategically connecting particles within a matrix and implementing distance constraints, the system authentically simulates cloth elasticity and deformation. The algorithmic foundation leverages Three.js, npx, and cannon-es, offering users an intuitive and interactive experience without the need for local software installation.

While the system excels in speed and accessibility, it acknowledges limitations in comparison to more intricate tools like Blender. The simplified customization options and absence of material presets highlight opportunities for future improvement.

Understanding the limitations, future iterations of the system could explore avenues to incorporate more advanced cloth property customization options, aligning with the sophisticated controls offered by industry-standard tools like Blender. Additionally, the introduction of material presets could further enhance user convenience, enabling quicker iterations in achieving desired cloth aesthetics.

As the system evolves, the balance between simplicity and advanced features will be key, ensuring a versatile tool that caters to a broad range of users. By engaging with user feedback and iteratively refining the system, the potential for enhanced 3D cloth interaction simulations on the web remains promising.

VII. Appendices

- Github repository (Full code) : https://github.com/yuujinleee/FlexiPlay