Computer Graphics Lab 10: introduction for WebXR

A-Frame WebXR Lab - Interactive Animations with Components

Objective

By the end of this lab, students will:

- Understand how to create **VR scenes** using **A-Frame**.
- Implement **custom animations** using `AFRAME.registerComponent`.
- Add **interactive elements** such as color changes and sound effects.
- Explore **basic physics-like motion** (bouncing, rotating, scaling).
- Develop an interactive **WebXR experience**.

Prerequisites

✔ Basic knowledge of **HTML & JavaScript**
✔ A **WebXR-compatible browser** (Chrome, Edge, Oculus Browser)
✔ (Optional) A **VR headset** (Oculus Quest, HTC Vive, etc.)

Step 1: Setting Up the Environment

1. Open a **code editor** (e.g., VS Code, Sublime, or Notepad++).
2. Create a **new HTML file** (`index.html`).
3. Copy and paste the **A-Frame WebXR code** (provided in the next section).

Step 2: Understanding the Scene

The following objects are included:

- **Sky & Ground** → For immersion

- **Bouncing Sphere** (`bouncing`) → Moves up & down
- **Rotating Torus** (`rotating`) → Spins continuously
- **Pulsating Cylinder** (`pulsating`) → Scales in & out
- **Interactive Color Change** (`change-color`) → Click objects to change colors
- **Sound Effect on Click** (`sound-effect`) → Clicking the cylinder plays a sound

Step 3: Implementing the Code

Copy and paste the following code into your `index.html` file:

```html
<!DOCTYPE html>
<html>
 <head>
   <meta charset="utf-8" />
   <title>A-Frame WebXR - Custom Animation Components</title>
   <script src="https://aframe.io/releases/1.6.0/aframe.min.js"></script>
 </head>
 <body>
   <a-scene>
     <!-- Sky with Texture -->
     <a-sky src="./floor.jpg"></a-sky>

     <!-- Ground with Texture -->
     <a-plane
       position="0 0 0"
       rotation="-90 0 0"
       width="20"
       height="20"
       src="./floor.jpg"
       repeat="10 10"
       change-color
     >
     </a-plane>
     <!-- Static Box -->
     <a-box
       position="-2 1 -3"
       rotation="0 45 0"
       width="1"
       height="1"
       depth="1"
       src="./mable.png"
       pulsating
       sound-effect
       change-color
       bouncing
     >
     </a-box>
```

```
  <!-- Cylinder -->
  <a-cylinder
    id="soundCylinder"
    position="3 1 -3"
    radius="0.5"
    height="1.5"
    color="#4CC3D9"
    pulsating
    sound-effect
    change-color
    rotating
  >
  </a-cylinder>

  <!-- Sound Asset -->
  <a-assets>
    <audio id="clickSound" src="./mouse-click.mp3"></audio>
  </a-assets>

  <!-- Camera & Cursor -->
  <a-entity position="0 1.6 0">
    <a-camera>
      <a-cursor color="#c1ff31"></a-cursor>
    </a-camera>
  </a-entity>
</a-scene>

<script>
  // Change Color on Click
  AFRAME.registerComponent("change-color", {
    init: function () {
      this.el.addEventListener("click", () => {
        this.el.setAttribute("color", getRandomColor());
      });
    },
  });
```

```javascript
// Sound Effect on Click
AFRAME.registerComponent("sound-effect", {
  init: function () {
    this.el.addEventListener("click", () => {
      let sound = document.createElement("a-sound");
      sound.setAttribute("src", "#clickSound");
      sound.setAttribute("autoplay", "true");
      this.el.appendChild(sound);
    });
  },
});

// Bouncing Animation Component
AFRAME.registerComponent("bouncing", {
  schema: { speed: { type: "number", default: 1000 } },
  tick: function (time, deltaTime) {
    let y = Math.sin(time / this.data.speed) * 0.5 + 1;
    this.el.setAttribute("position", `0 ${y} -3`);
  },
});

// Rotating Animation Component
AFRAME.registerComponent("rotating", {
  schema: { speed: { type: "number", default: 3000 } },
  tick: function (time, deltaTime) {
    let rotation = this.el.getAttribute("rotation");
    this.el.setAttribute("rotation", {
      x: rotation.x,
      y: (rotation.y + deltaTime * 0.1) % 360,
      z: rotation.z,
    });
  },
});

// Pulsating Animation Component
AFRAME.registerComponent("pulsating", {
  schema: { speed: { type: "number", default: 1000 } },
  tick: function (time, deltaTime) {
    let scale = Math.sin(time / this.data.speed) * 0.2 + 1;
    this.el.setAttribute("scale", `${scale} ${scale} ${scale}`);
  },
});
```

```
    // Generate a Random Color
    function getRandomColor() {
      return "#" + Math.floor(Math.random() * 16777215).toString(16);
    }
  </script>
 </body>
</html>
```

Step 4: Running the Scene

1. **Save the file** (`index.html`).
2. Open the file in **Google Chrome** or any **WebXR-compatible browser**.
3. **Click on the objects** to see the interactions!
4. **Enter VR mode** by clicking the **VR button** (bottom-right corner).
5. If using a **VR headset**, view the scene in **immersive VR**.

Step 5: Challenges for Students

1. ✅ Modify the **speed** of the animations (e.g., make the torus spin faster).
2. ✅ Change the **bounce height** of the sphere.
3. ✅ Replace colors with **textures** (use `src="your-texture.jpg"`).
4. 🚀 Stop the **rotation** when clicking the torus.
5. 🚀 Make the sphere **move forward and backward** instead of up and down.
6. 🚀 Add a **new shape** (e.g., a cone) and animate it.
7. 🔥 Add a **floating effect** to the box.
8. 🔥 Implement **user movement** (teleportation).
9. 🔥 Make objects **change size dynamically on click**.