Лабораторна робота №5: Інтерактивність (Raycaster) та занурення у VR

1. Мета роботи

- Опанувати механізм вибору об'єктів у 3D-просторі за допомогою класу `THREE.Raycaster`.
- Навчитися перетворювати 2D-координати курсора у 3D-промені через нормалізацію.
- Реалізувати базове занурення у віртуальну реальність (VR) через WebXR Device API.

2. Теоретична частина

Принцип Raycasting

Математично Raycasting базується на рівнянні прямої у просторі:

де:

- — початок променя (позиція камери або контролера).
- — одиничний вектор напрямку.
- — параметр відстані.

Нормалізація координат

Для Raycaster координати миші мають бути у діапазоні (NDC):

-

3. Завдання для самостійного виконання (Деталізація)

Нижче наведено технічні алгоритми для реалізації додаткових функцій проєкту.

Завдання 3.1: Реалізація "Cursor Ring" (Кільце-курсор)

Мета: Створити візуальний фідбек, який показує точну точку контакту променя з поверхнею.

- **Математична суть:** Ви повинні не лише змінити позицію (`position`) кільця на `intersects[0].point`, а й зорієнтувати його площину паралельно грані об'єкта.
- **Алгоритм:**
    1. Створіть `THREE.TorusGeometry` з дуже малим внутрішнім радіусом (наприклад, 0.05).
    2. У циклі анімації, якщо знайдено перетин:

- Зробіть кільце видимим.
- Встановіть `ring.position.copy(intersects[0].point)`.
- Використовуйте метод `ring.lookAt(intersects[0].point.clone().add(intersects[0].face.normal))`. Це змусить "обличчя" кільця дивитися у напрямку нормалі грані.

3. Якщо перетину немає — приховайте кільце (`visible = false`).

Завдання 3.2: Система вибору для мобільних пристроїв (Touch Support)

Мета: Забезпечити працездатність інтерфейсу на смартфонах.

- **Технічна особливість:** Подія `mousemove` не працює на сенсорних екранах. Потрібно використовувати `touchstart`.
- **Алгоритм:**
  1. Додайте слухач: `window.addEventListener('touchstart', onTouchStart, { passive: false })`.
  2. У функції `onTouchStart` отримайте координати першого дотику:

```
const touch = event.touches[0];
mouse.x = (touch.clientX / window.innerWidth) * 2 - 1;
mouse.y = -(touch.clientY / window.innerHeight) * 2 + 1;
```

  3. Викличте вашу логіку Raycasting одразу після оновлення координат.

Завдання 3.3: Творче завдання (Звуковий ефект)

Мета: Додати аудіо-фідбек для покращення імерсивності.

- **Обмеження браузерів:** Відтворення звуку можливе лише після взаємодії користувача зі сторінкою (кліку).
- **Алгоритм:**
  1. Виберіть короткий звук (wav/mp3) тривалістю до 1 секунди.
  2. Створіть об'єкт: `const clickSound = new Audio('path_to_sound.mp3')`.

3. У функції `onClick`, якщо знайдено перетин з об'єктом, викличте `clickSound.play()`.
4. **Бонус:** Спробуйте змінювати висоту звуку (`playbackRate`) залежно від висоти об'єкта у просторі.

4. Контрольні запитання

1. Чим відрізняється метод `.intersectObject()` від `.intersectObjects()`?
2. Як параметр `recursive` у Raycaster впливає на пошук перетинів у вкладених групах?
3. Чому при роботі у VR ми не можемо використовувати стандартні координати миші для Raycasting?
4. Яку роль відіграє функція `renderer.setAnimationLoop()` замість стандартного `requestAnimationFrame()`?