

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ІМПЛЕМЕНТАЦІЯ ТЕСТУ ЗІТКНЕНЬ API WEBXR

МЕТОДИЧНІ ВКАЗІВКИ

**до виконання лабораторної роботи № 2
з дисципліни «Віртуальна реальність»
для студентів першого (бакалаврського) рівня вищої освіти спеціальності
121 "Інженерія програмного забезпечення"**

Львів -- 2025

Імплементация тесту зіткнень API WebXR: методичні вказівки до виконання лабораторної роботи №2 з дисципліни "Віртуальна реальність" для студентів першого (бакалаврського) рівня вищої освіти спеціальності 121 "Інженерія програмного забезпечення" . Укл.: О.Є. Бауск. -- Львів: Видавництво Національного університету "Львівська політехніка", 2025. -- 16 с.

Укладач: Бауск О.Є., к.т.н., асистент кафедри ПЗ

Відповідальний за випуск: Федасюк Д.В., доктор техн. наук, професор

Рецензенти: Федасюк Д.В., доктор техн. наук, професор

Задорожний І.М., асистент кафедри ПЗ

Тема роботи: Імплементація тесту зіткнень (hit test) у WebXR додатку.

Мета роботи: Розширити функціональність WebXR додатку, створеного в попередній лабораторній роботі, додавши можливість тесту зіткнень для розміщення 3D об'єктів на реальних поверхнях.

Теоретичні відомості

Тест зіткнень (Hit Test) у WebXR

Тест зіткнень (hit test) у контексті доповненої реальності - це процес визначення, де промінь, що виходить з певної точки у певному напрямку, перетинається з реальними об'єктами у фізичному світі. У WebXR API цей функціонал дозволяє визначити, де віртуальні об'єкти можуть бути розміщені на реальних поверхнях.

Основні компоненти тесту зіткнень у WebXR:

1. **XRSession.requestHitTestSource()** - метод для створення джерела тесту зіткнень
2. **XRRHitTestSource** - об'єкт, що представляє джерело тесту зіткнень
3. **XRFrame.getHitTestResults()** - метод для отримання результатів тесту зіткнень
4. **XRRHitTestResult** - об'єкт, що містить інформацію про результат тесту зіткнень

Принцип роботи тесту зіткнень

1. Створюється джерело тесту зіткнень, яке визначає, звідки буде виходити промінь
2. У кожному кадрі анімації отримуються результати тесту зіткнень
3. Результати містять інформацію про точки перетину променя з реальними поверхнями
4. На основі цієї інформації можна розміщувати віртуальні об'єкти на реальних поверхнях

Типи джерел тесту зіткнень

WebXR підтримує два типи джерел тесту зіткнень:

1. **Транзйєнтне джерело (Transient Hit Test Source)** - промінь виходить з певної точки у просторі відстеження, наприклад, з центру екрану
2. **Постійне джерело (Persistent Hit Test Source)** - промінь виходить з певної точки у просторі відліку, наприклад, з контролера// додайте: У цій лабораторній роботі ми будемо використовувати транзйєнтне джерело тесту зіткнень, щоб визначити, де користувач "вказує" на реальні поверхні.

Реалізація тесту зіткнень у WebXR

Для реалізації тесту зіткнень у WebXR необхідно:

1. Перевірити підтримку функціоналу тесту зіткнень у браузері
2. Запросити необхідні функції при створенні сесії WebXR
3. Створити джерело тесту зіткнень
4. Обробляти результати тесту зіткнень у кожному кадрі анімації
5. Використовувати отримані результати для розміщення віртуальних об'єктів

Хід роботи

1. Підготовка проекту

1.1. Використайте код, реалізований у першій лабораторній роботі, як основу для цієї роботи.

Додавання тесту зіткнень

Поширеним способом взаємодії зі світом доповненої реальності є тест зіткнень, який знаходить перетин між променем та геометрією реального світу. У нашому WebXR додатку ми будемо використовувати тест зіткнень для розміщення віртуального об'єкта (соняшника) у реальному світі.

Видалення демонстраційного куба

Видаліть непідсвічений куб і замініть його сценою, що включає освітлення, або перевірте наявність освітлення у вашому фрагменті коду.

```
const scene = new THREE.Scene();
// ...Інший код...
const directionalLight = new THREE.DirectionalLight(0xffffff, 0.3);
directionalLight.position.set(10, 15, 10);
scene.add(directionalLight);
```

Використання функції тесту зіткнень

Для ініціалізації функціональності тесту зіткнень, запитайте сесію з функцією hit-test. Знайдіть попередній фрагмент requestSession() та додайте до нього `hit-test` або перевірте наявність `hit-test` у вашому фрагменті коду.

```
const session = await navigator.xr.requestSession("immersive-ar",
  {requiredFeatures: ['local', 'hit-test']}
);
```

Додавання модельного завантажувача

Наразі сцена містить лише кольоровий куб. Щоб зробити досвід більш цікавим, додайте модельний завантажувач, який дозволяє завантажувати GLTF-моделі.

Додайте в вашого HTML файлу GLTFLoader (ТІЛЬКИ якщо ви використовуєте unpkg для three.js)^

```
<!-- three.js -->

<script src="https://unpkg.com/three@0.126.0/examples/js/loaders/GLTFLoader.js"></script>
```

Та імпоруйте GLTFLoader у ваш проект:

```
import { GLTFLoader } from 'three/addons/loaders/GLTFLoader.js';
```

Завантаження GLTF-моделей

Використовуйте модельний завантажувач з попереднього кроку для завантаження цільового прицілу та соняшника з веб-сторінки.

Додайте цей код **вище onXRFrame**:

```
const loader = new THREE.GLTFLoader();
let reticle: any;
loader.load("
  https://immersive-web.github.io/webxr-samples/media/gltf/reticle/reticle.gltf",
  function(gltf) {
    reticle = gltf.scene;
    reticle.visible = false;
    scene.add(reticle);
  })

let flower: any;
loader.load(
  "https://immersive-web.github.io/webxr-samples/media/gltf/sunflower/sunflower.gltf",
  function(gltf) {
    flower = gltf.scene;
  });

// далі лишається наступний код:
// Create a render loop that allows us to draw on the AR view.
const onXRFrame = (time, frame) => {
```

Примітка: Обробка помилок виключена з цього підручника для стислості. Дивіться `GLTFLoader.load` для обробки `onProgress` та `onError`.

Створення джерела тесту зіткнень

Щоб обчислити перетини з реальними об'єктами, створіть `XRHitTestSource` за допомогою `XRSession.requestHitTestSource()`. Промінь, який використовується для тесту зіткнень, має простір відліку "viewer" як початок, що означає, що тест зіткнень проводиться з центру вікна.

Щоб створити джерело тесту зіткнень, додайте цей код після створення локального простору відліку. Замініть перебор варіантів просторів на тільки єдиний 'local':

```
const referenceSpaceTypes: XRReferenceSpaceType[] = [
  'local',
];
```

```
// Поряд з:
let referenceSpace: XRReferenceSpace | null = null;
// Додайте:
let hitTestSource: XRHitTestSource | undefined = undefined;

// Простір відліку 'local' має нативне походження, яке розташоване
// поблизу позиції глядача на момент створення сесії.
// referenceSpace = ...
```

```

// Створіть інший XRReferenceSpace, який має глядача як точку відліку.
const viewerSpace = await session.requestReferenceSpace('viewer');

// Виконуємо тест зіткнень, використовуючи глядача як точку відліку.
// Зверніть увагу, як забезпечується безпека типів.
const viewerSpace = await session.requestReferenceSpace('viewer');
let hitTestSource: XRHitTestSource | undefined = undefined;
if (session.requestHitTestSource) {
  hitTestSource = await session.requestHitTestSource(
    { space: viewerSpace }
  );
}

```

Відображення прицілу наведення

Щоб чітко показати, де буде розміщено соняшник, додайте прицільний маркер до сцени. Цей прицільний маркер буде прикріплюватися до поверхонь реального світу, вказуючи, де буде закріплено соняшник.

`XRRFrame.getHitTestResults` повертає масив `XRHitTestResult` та показує перетини з геометрією реального світу. Використовуйте ці перетини для позиціонування прицільного маркера на кожному кадрі.

```

// перед:
camera.projectionMatrix.fromArray(view.projectionMatrix);
camera.updateMatrixWorld(true);

// додайте:
const hitTestResults = frame.getHitTestResults(hitTestSource);
if (hitTestResults.length > 0 && reticle) {
  const hitPose = hitTestResults[0].getPose(referenceSpace);
  reticle.visible = true;
  reticle.position.set(
    hitPose?.transform.position.x,
    hitPose?.transform.position.y,
    hitPose?.transform.position.z
  )
  reticle.updateMatrixWorld(true);
}

```

Додавання взаємодій при натисканні

`XRSession` отримує події `select`, коли користувач завершує основний дію. У сесії AR це відповідає натисканню на екран.

Щоб зробити соняшник з'являтися при натисканні на екран, додайте цей код під час ініціалізації:

```

// перед:
let flower: any;

loader.load("
https://immersive-web.github.io/webxr-samples/media/gltf/sunflower/sunflower.gltf",

```

```
function(gltf) {  
  flower = gltf.scene;  
});  
  
// додайте:  
session.addEventListener("select", (event) => {  
  if (flower) {  
    const clone = flower.clone();  
    clone.position.copy(reticle.position);  
    scene.add(clone);  
  }  
});
```

Тестування визначення перетинів

Використайте ваш мобільний пристрій для переходу на сторінку. Після того, як WebXR побудує розуміння навколишнього середовища, прицільний маркер повинен з'явитися на поверхнях реального світу. Торкніться екрану, щоб розмістити соняшник, який можна буде оглянути з усіх боків.