Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

Розрахунково-графічна робота

З дисципліни «Методи синтезу віртуальної реальності»

Варіант 1

Виконав: Бабій І.О.

Студент групи ТР-22мп

Київ 2023

# Завдання

**Тема роботи**:

Просторовий звук. Реалізація просторового звуку за допомогою WebAudio HTML5 API.

**Вимоги**:

* Перевикористати код з практичної роботи №2.
* Імплементувати обертання джерела звуку навколо геометричного центру поверхні за допомогою матеріального інтерфейсу. Програвати улюблену пісню у форматі mp3/ogg, змінюючи розташування джерела звуку відповідно до введення користувача.
* Візуалізувати джерело звуку у вигляді сфери.
* Додати звуковий фільтр низьких частот. Додати «галочку», яка вмикає і вимикає фільтр. Задати параметри фільтру за смаком.

# Теоретичні відомості

Web Audio API є потужним інструментом для розробників веб-додатків, який надає широкі можливості з маніпулювання та синтезу звуку в реальному часі. Це API включає набір інтерфейсів і об'єктів, що дозволяють створювати, змінювати та маршрутизувати аудіосигнали у веб-додатках.

Один з ключових аспектів Web Audio API - це його здатність працювати з аудіо за допомогою модульного підходу. Це означає, що розробники можуть створювати складні конвеєри обробки звуку, використовуючи різні об'єкти і вузли.

Основним об'єктом в Web Audio API є AudioContext. Він представляє граф обробки аудіо і є центральним елементом для створення та підключення аудіо-вузлів. AudioContext надає доступ до широкого спектру методів і властивостей, які дозволяють керувати відтворенням аудіо, маршрутизацією та ефектами.

У фрагменті коду "context = new AudioContext();" створюється екземпляр об'єкта AudioContext, який є основою для обробки звуку.

Ще одним важливим об'єктом є MediaElementSourceNode, який використовується для отримання аудіоданих з медіа-елементів HTML, наприклад, елементів <audio> або <video>. Це джерело аудіо можна підключити до інших аудіо-вузлів для подальшої обробки і маршрутизації.

У фрагменті коду "source = context.createMediaElementSource(audio);" створюється об'єкт MediaElementSourceNode, де змінна audio вказує на елемент HTML <audio>. Це дозволяє обробляти аудіодані з вказаного медіа-елемента за допомогою Web Audio API.

PannerNode відповідає за просторове позиціонування та панорамування звуку. Він дозволяє розробникам створювати тривимірний звук, контролюючи положення, орієнтацію та швидкість звукових джерел у віртуальному 3D-просторі.

У фрагменті коду "panner = context.createPanner();" створюється об'єкт PannerNode, який можна підключити до аудіо-графіка. PannerNode використовується для керування положенням та рухом аудіоджерела, що дозволяє динамічно розподіляти звук у просторі.

BiquadFilterNode використовується для реалізації різних типів цифрових фільтрів, наприклад, фільтрів низьких частот, високочастотних, смугових або пікових фільтрів. Це дозволяє розробникам змінювати частотну характеристику аудіосигналу і застосовувати різноманітні ефекти, такі як еквалайзер або резонанс.

У фрагменті коду "biquadFilter = context.createBiquadFilter();" створюється об'єкт BiquadFilterNode. Після підключення до аудіографа, цей вузол можна використовувати для застосування ефектів фільтрації до аудіосигналу, змінюючи або покращуючи його спектральні характеристики.

Застосовуючи Web Audio API та різноманітні об'єкти, розробники можуть створювати захоплюючі та інтерактивні аудіо-додатки в Інтернеті. Вони можуть маніпулювати звуком в реальному часі, створювати тривимірний звуковий простір, застосовувати різноманітні ефекти та фільтри для досягнення бажаного звукового досвіду.

# Особливості виконання завдання

Під час другої лабораторної роботи була реалізована можливість обертання поверхні під назвою "Virich Cyclic Surface" у стереоформаті за допомогою акселерометра смартфона. Повертаючи телефон, фігура також оберталась. Демонстрацію роботи програми можна побачити на рисунках 3.1 та 3.2.

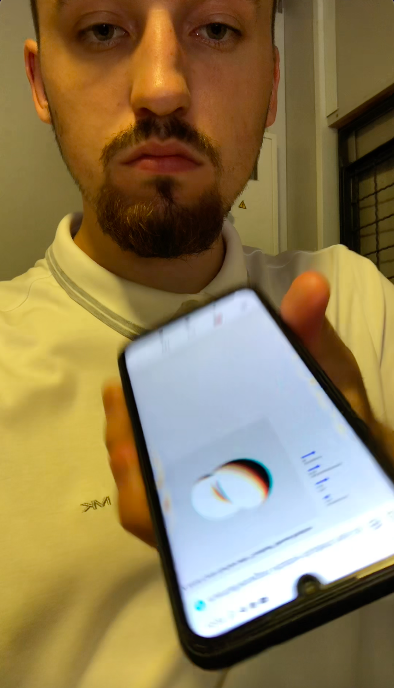


Рис 3.1 та Рис 3.2 Демонстрація застосування програми, отриманої в ході другої лабораторної

За допомогою Web Audio API, зокрема документації на сторінці https://webaudio.github.io/web-audio-api/, була реалізована основна частина розрахунково-графічної роботи.

Під час проведення лабораторної роботи було створено об'єкт аудіоконтексту для використання Web Audio API. Вибір аудіофайлу у форматі mp3 та його відображення на веб-сторінці було здійснено за допомогою HTML-елемента <audio>. Джерело аудіо було створено шляхом передачі аудіоелемента в конструктор. Об'єкт panner було створено в контексті для подальшого керування звуком, включаючи зміну його положення відповідно до повороту телефону (джерело звуку розташовувалося на відстані 2 від центру в напрямку повороту телефону в просторі).

Важливим етапом завдання було впровадження фільтра на вихідний звук. Згідно з варіантом, було виконано низькочастотний фільтр з параметрами, вказаними в розділі 5. Потім було здійснено з'єднання відповідних об'єктів. Було додано eventListener, який контролює зупинку та відновлення відтворення аудіофайлу. Було створено поле для включення та виключення фільтра, а також додано додатковий eventListener для переключення фільтра відповідно до стану цього поля. Оновлення положення звуку шляхом переміщення об'єкта panner було виконано в основній функції "draw".

# Вказівки користувачу

Управління рухом віртуальної сфери, що відображає приблизне місце знаходження джерела звуку, може здійснюватись за допомогою користувача. Цей процес можна спостерігати на рисунках 4.1, 4.2, 4.3.

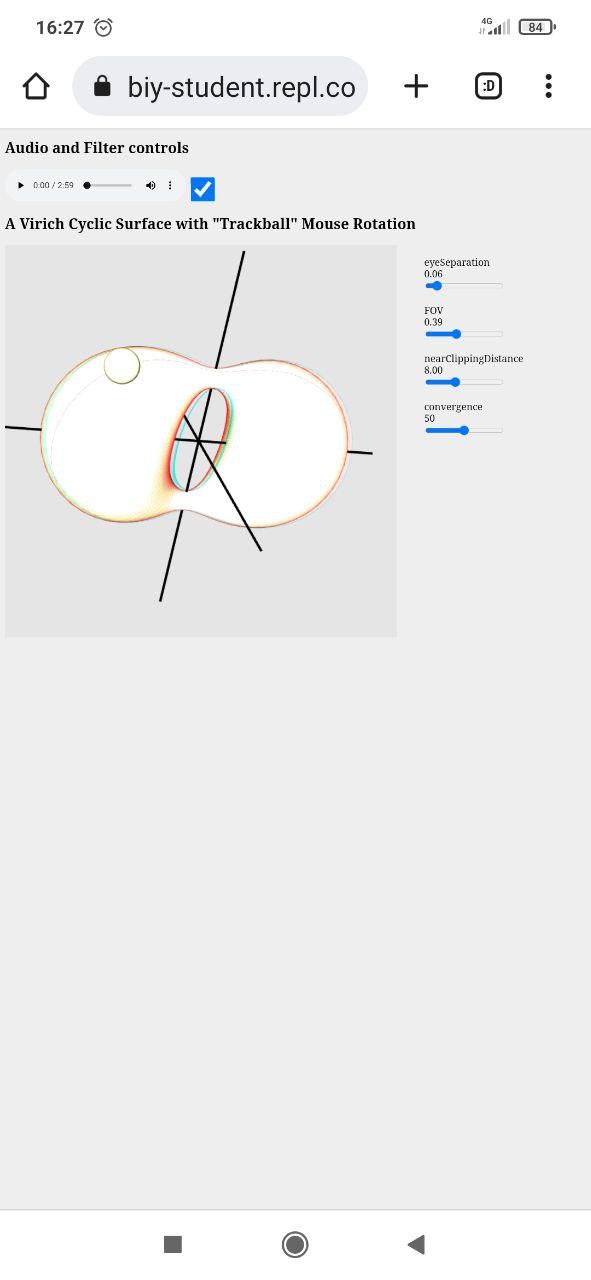


Рис. 4.1, 4.2 Матеріальний інтерфейс повернуто верхнього лівого кута



Рис. 4.3 Матеріальний інтерфейс повернуто в лівий нижній кут

При повороті телефону відбувається переміщення сфери навколо об'єкта, створюючи ілюзію руху джерела звуку, яка найбільш відчутна через навушники або стереосистеми.

На веб-сторінці присутні елементи інтерфейсу для регулювання параметрів стерео-зображення, включаючи значення eye separation, field of view, near clipping distance та convergence. Ці контролери зображені на рисунку 4.4.

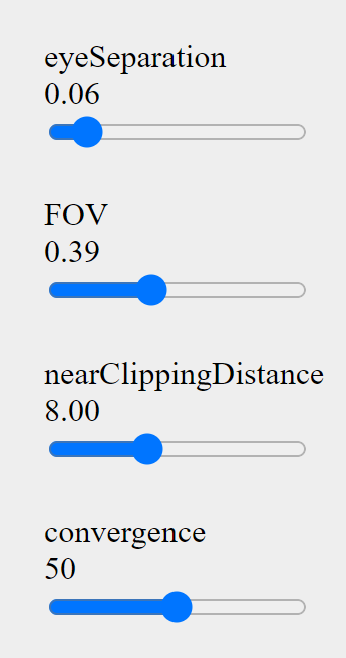


Рис. 4.4 Контролери для регулювання параметрів стерео-зображення

Елементи управління аудіофайлом, такі як перемотування, пауза, відтворення та регулювання гучності, також відображені на сторінці. Їх можна побачити на рисунку 4.5.

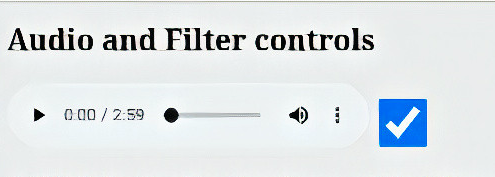


Рис. 4.5 Елемент управління аудіо-файлом

Для активізації та деактивізації фільтра також було створено "чекбокс", який можна побачити на рисунку 4.6.

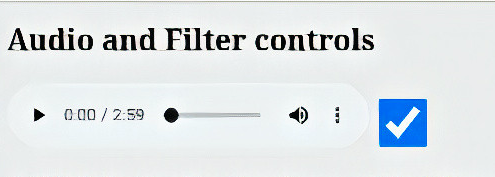


Рис. 4.6 Чекбокс перемикання стану фільтру

# Деякі зразки коду

Код функцій для налаштування та ініціалізації аудіо:

let audio = null;

let audioContext;

let source;

let panner;

let filter;

function AudioSetup() {

audio = document.getElementById('audio');

audio.addEventListener('play', () => {

console.log('play');

if (!audioContext) {

audioContext = new AudioContext();

source = audioContext.createMediaElementSource(audio);

panner = audioContext.createPanner();

filter = audioContext.createBiquadFilter();

// Connect audio nodes

source.connect(panner);

panner.connect(filter);

filter.connect(audioContext.destination);

// highshelf filter parameters

filter.type = 'lowpass';

filter.Q.value = 1;

filter.frequency.value = 800;

filter.gain.value = 13;

audioContext.resume();

}

})

audio.addEventListener('pause', () => {

console.log('pause');

audioContext.resume();

})

}

function startAudio() {

AudioSetup();

let filterCheckbox = document.getElementById('filterCheckbox');

filterCheckbox.addEventListener('change', function() {

if (filterCheckbox.checked) {

// Connect filter when checkbox is checked

panner.disconnect();

panner.connect(filter);

filter.connect(audioContext.destination);

} else {

// Disconnect filter when checkbox is unchecked

panner.disconnect();

panner.connect(audioContext.destination);

}

});

audio.play();

}