Міністерство освіти й науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра цифрових технологій в енергетиці

[Методи синтезу віртуальної реальності](https://campus.kpi.ua/student/index.php?mode=studysheet&action=view&id=125230)

Розрахунково графічна робота

Варіант - 10

**Виконав:**

Студент 1-го курсу

групи ТР-21мп НН ІАТЕ

Кедрун Володимир

**Перевірив:**

Демчишин А.А.

Київ-2023

**Завдання**

1. Використати код з практичного завдання №2.

2. Імплементувати обертання джерела звуку навколо геометричного центру площини за допомогою фізичного (матеріального) інтерфейсу (на цей раз поверхня залишається нерухомою, а джерело звуку рухається). Відтворіти улюблену пісню у форматі mp3/ogg з контролем користувача над просторовим положенням джерела звуку.

3. Візуалізувати положення джерела звуку за допомогою сфери.

4. Додати звуковий фільтр (використовуючи інтерфейс BiquadFilterNode) для відповідного варіанту (№10). Додати чекбокс(checkbox), який вмикає або вимикає фільтр. Встановити параметри фільтра на свій смак.

**Теоретична інформація**

Звуковий API (Web Audio API) - це стандартна специфікація, яка надає можливості для створення, обробки та відтворення аудіо-сигналів у веб-програмах. Він дозволяє розробникам створювати потужні аудіо-додатки, такі як аудіо редактори, синтезатори, секвенсори та інші.

Основні концепції Web Audio API включають наступне:

Аудіо контекст (AudioContext): Це головний об'єкт, який представляє аудіо-середовище. Він ініціалізується за допомогою конструктора AudioContext і служить як центральний пункт для створення, підключення та керування аудіо-вузлами. Він використовується для створення звукових джерел і підключення їх до аудіо-графу для обробки та відтворення звуків. У додатку може бути лише один екземпляр AudioContext.

Аудіо-вузли (Audio Nodes): Аудіо-вузли представляють різні елементи аудіо-графа, які обробляють аудіо-сигнали. Вони можуть бути джерелами звуку, ефектами обробки або вузлами призначення. Деякі загальні типи аудіо-вузлів включають OscillatorNode (генератор звуку), GainNode (регулятор гучності), BiquadFilterNode (фільтр) та PannerNode (позиціонування звуку).

Обробка звуку: Web Audio API надає різноманітні аудіо-вузли, які можна використовувати для обробки звукових сигналів. Це можуть бути фільтри для

Audio Sources - це елементи аудіо-графа, які представляють аудіо джерела. Вони надають аудіо-дані, які потім обробляються та відтворюються. Після обробки звук може бути підключений до аудіо-призначення, такого як AudioContext.destination, для відтворення на динаміки або запису у файл.

Panner - це вузол в Web Audio API, який використовується для просторового позиціонування звукових джерел у тривимірному просторі. Він дозволяє контролювати положення звуку в просторі шляхом панорамування (ліво-право) та нахилу (вгору-вниз).

Підключення аудіо-вузлів: Аудіо-вузли можна підключати один до одного, утворюючи аудіо-граф. Вихідний сигнал одного вузла може бути підключений до вхідного сигналу іншого вузла. Це дозволяє створювати складні аудіо-ланцюжки, де сигнал проходить через різні обробки перед тим, як потрапляє до призначення звуку (наприклад, колонок або запису).

Проходження звукового сигналу через фільтр типу bandpass призводить до приглушення або відкидання усіх частот, які знаходяться поза встановленою смугою частот, тоді як частоти, які потрапляють у цю смугу, проходять без змін. Це дозволяє виділити або підсилити компоненти звуку, які знаходяться в цій смузі частот.

Фільтр типу bandpass має два основних параметри:

Центральна частота (Center Frequency): Вона визначає центральну точку смуги частот, яку фільтр пропускає без змін. Це може бути встановлено на певне значення, яке відповідає потрібній центральній частоті.

Ширина смуги (Bandwidth): Вона визначає ширину смуги частот, яку фільтр пропускає. Ширина смуги визначає, наскільки широко фільтр дозволяє частотам проходити. Вона може бути встановлена на певне значення, яке відповідає потрібній ширині смуги.

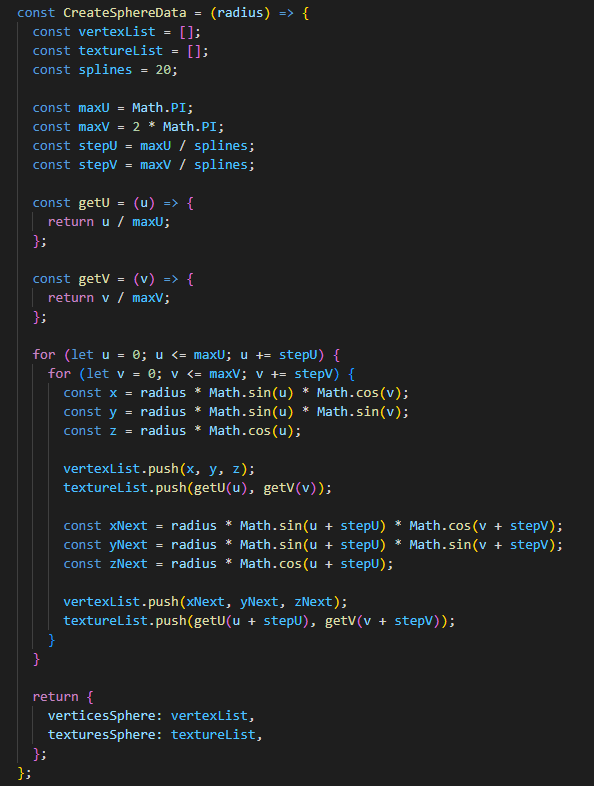
Опис фільтра типу bandpass:

Проходження звукового сигналу через фільтр типу bandpass призводить до приглушення або відкидання усіх частот, які знаходяться поза встановленою смугою частот. Центральна частота визначає центральну точку смуги частот, яку фільтр пропускає без змін, тоді як ширина смуги визначає ширину цієї смуги. Це дозволяє виділити або підсилити компоненти звуку, які знаходяться в цій смузі частот, що призводить до більш прозорого або свіжого звучання.

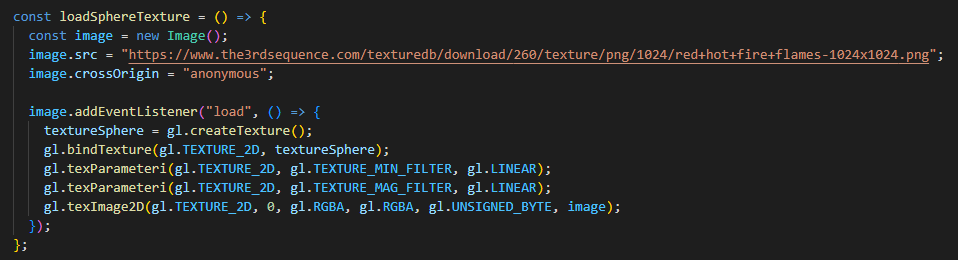
**Виконання роботи**

1. **Система контролю версій**Перед початком роботи було створено гілку в GIT з назвою CGW, для зручного зберігання та контролю версій програми.
2. **Створення** **сфери**

Створюємо сферу для візуалізації джерела аудіо, аналогічно створенню основної фігури.



Накладання текстури на сферу, яка обгортає її поверхню аналогічно тому, як текстура накладається на фігуру.



1. **Створення аудіо та прив’язка його до сфери**.

Ми створюємо аудіо-контекст і визначаємо початкові елементи, такі як AudioSource та PannerNode. Для об'єкту audioPanner ми встановлюємо модель панування на HRTF (Head-Related Transfer Function), яка дозволяє враховувати індивідуальні особливості слухувача. Ми також використовуємо модель відстані, яка забезпечує лінійне зменшення гучності звуку пропорційно до відстані.

Для об'єкту audioFilter ми встановлюємо тип фільтру відповідно до варіанту.

Наприкінці налаштувань ми з'єднуємо всі елементи за допомогою функцій connect, щоб встановити потрібні зв'язки між ними.

Для роботи з фільтром був створений спеціальний чекбокс. При перемиканні чекбоксу в положення "увімкнено", просторовий вузол відключається, а замість нього підключається фільтрувальний вузол, який в свою чергу підключається до вихідного призначення (звукового пристрою або вихідного вузла). При перемиканні чекбоксу в положення "вимкнено" відбувається зворотний процес - фільтрувальний вузол відключається, а просторовий вузол знову підключається.

Також була додана кнопка для відтворення пісні, що дозволяє почати програвання аудіо.

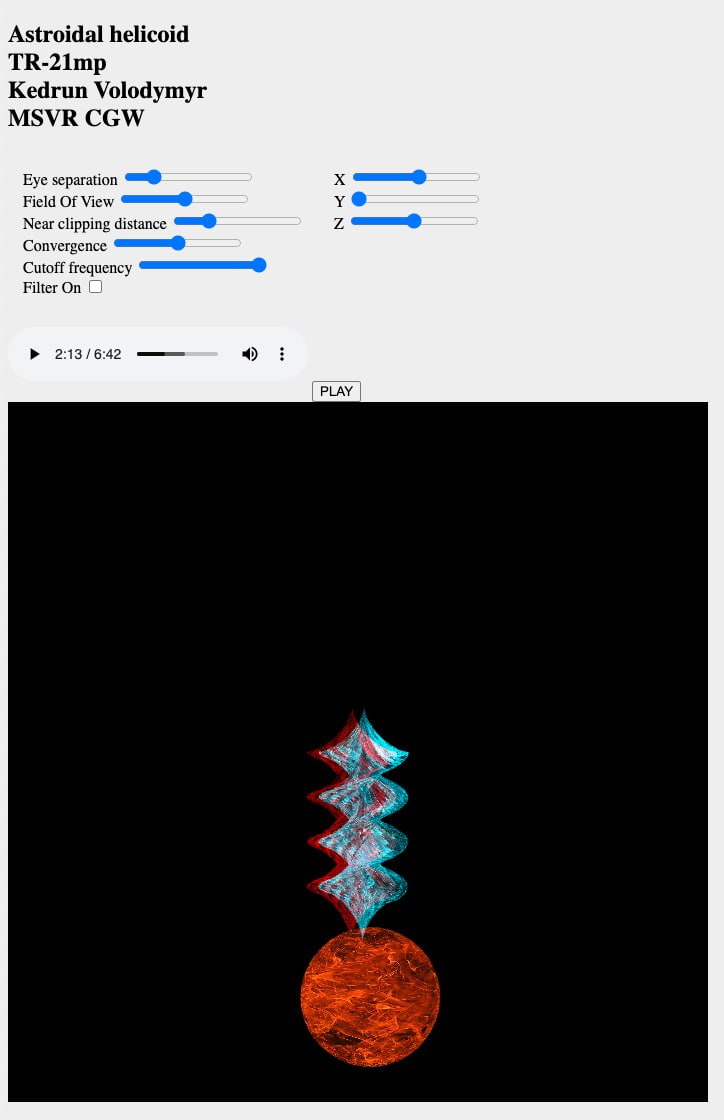


**Інструкція користувача**

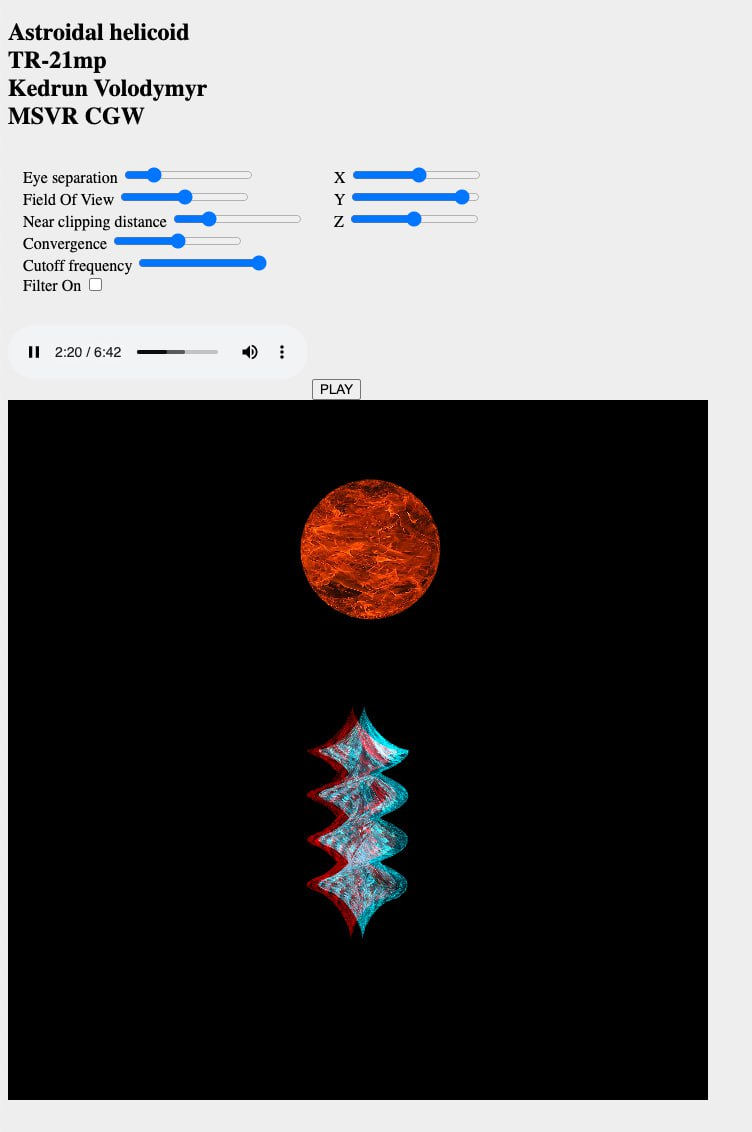
У новій версії було додано декілька нових елементів: флаг для фільтру, аудіодоріжка і повзунки для сфери та частоти зрізу.

Флаг фільтру дозволяє вмикати або вимикати фільтр. За допомогою аудіодоріжки можна взаємодіяти з треком, змінюючи час відтворення, гучність або завантажуючи новий файл. Повзунки з координатами використовуються для переміщення сфери у просторі. Повзунок для зміни частоти використовується для зміни частоти зрізу.

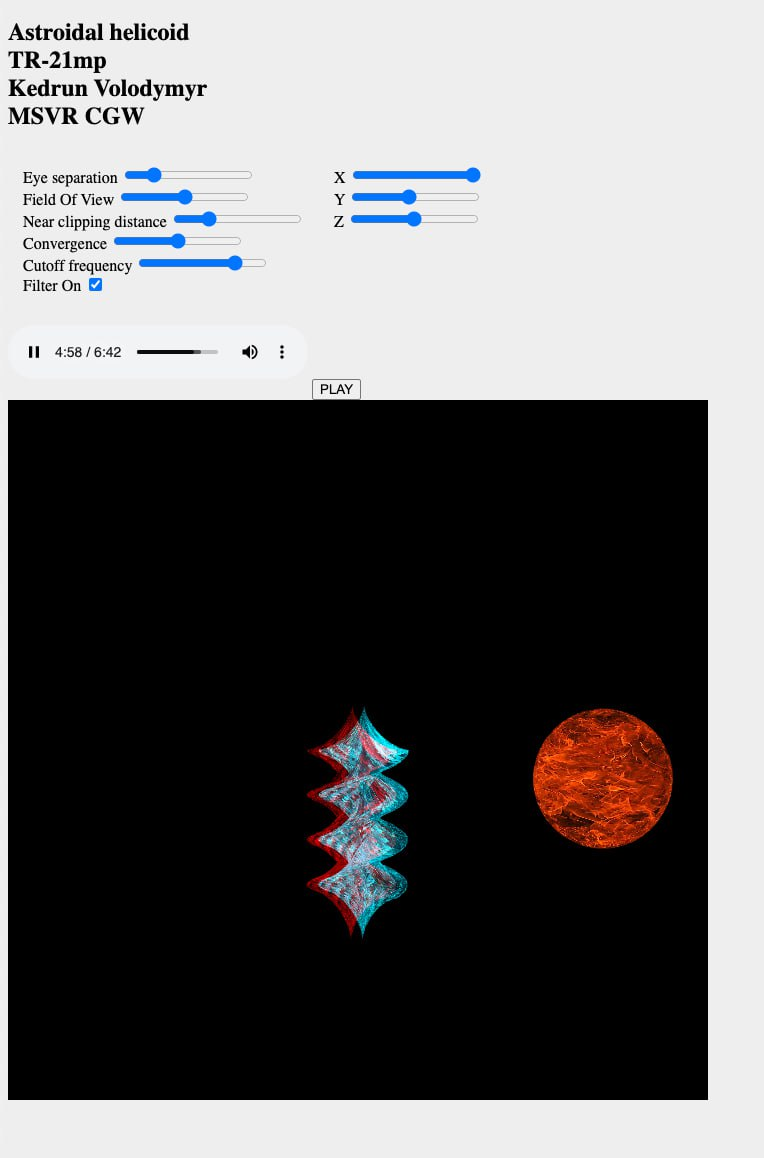
На зображенні показано фігуру та сферу, а елементи інтерфейсу є результатом виконання розрахунково-графічної роботи. Сфера виступає як джерело звуку і може бути керована за допомогою повзунків "вперед" та "назад". Коли сфера знаходиться ближче до користувача (тобто знизу), звук лунає гучніше з основних динаміків.



сфера знаходиться далі від користувача (тобто зверху) – звук лунає гучніше з розмовного динаміку.



Увімкнений фільтр частоти та звук праворуч



**Лістинг**

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

if (!audioContext) {

audioContext = new (window.AudioContext || window.webkitAudioContext)();

audioSource = audioContext.createMediaElementSource(audio);

audioPanner = audioContext.createPanner();

audioFilter = audioContext.createBiquadFilter();

audioPanner.panningModel = "HRTF";

audioPanner.distanceModel = "linear";

audioFilter.type = "bandpass";

audioFilter.frequency.value = cutoffFrequencyInput.value;

audioSource.connect(audioPanner);

audioPanner.connect(audioFilter);

audioFilter.connect(audioContext.destination);

audioContext.resume();

}

});

let filter = document.getElementById("filterCheckbox");

filter.addEventListener("change", function () {

if (filter.checked) {

audioPanner.disconnect();

audioPanner.connect(audioFilter);

audioFilter.connect(audioContext.destination);

} else {

audioPanner.disconnect();

audioPanner.connect(audioContext.destination);

}

});

let audiobutton = document.getElementById("audioButton");

audiobutton.addEventListener("click", function () {

audio.play();

});

const loadSphereTexture = () => {

const image = new Image();

image.src = "https://www.the3rdsequence.com/texturedb/download/260/texture/png/1024/red+hot+fire+flames-1024x1024.png";

image.crossOrigin = "anonymous";

image.addEventListener("load", () => {

textureSphere = gl.createTexture();

gl.bindTexture(gl.TEXTURE\_2D, textureSphere);

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_MIN\_FILTER, gl.LINEAR);

gl.texParameteri(gl.TEXTURE\_2D, gl.TEXTURE\_MAG\_FILTER, gl.LINEAR);

gl.texImage2D(gl.TEXTURE\_2D, 0, gl.RGBA, gl.RGBA, gl.UNSIGNED\_BYTE, image);

});

};

index.html

<audio

id="audio"

style="margin-top: 15px; margin-bottom: 15px"

controls

loop

>

<source src="scorpions.mp3" type="audio/mpeg" />

Ваш браузер не підтримує mp3.

</audio>

<button id="audioButton">PLAY</button>

<div id="canvas-holder">

<canvas

width="700"

height="700"

id="webglcanvas"

style="background-color: red;"

></canvas>

</div>