Міністерство освіти й науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра цифрових технологій в енергетиці

Розрахунково-графічна робота

З дисципліни: «Методи синтезу віртуальної реальності»

Варіант 9

Виконав:

Студент 1-го курсу

групи ТР-32мп

Журавель В.І.

Перевірив:

Демчишин А.А.

Київ-2024

**Завдання**

* Реалізувати обертання джерела звуку навколо геометричного центру ділянки поверхні круговим способом протягом певного часу (цього разу поверхня залишається нерухомою, а джерело звуку рухається). Відтворити свою улюблену пісню у форматі mp3/ogg, маючи просторове положення джерела звуку, кероване користувачем
* Візуалізувати положення джерела звуку за допомогою сфери.
* Додати звуковий фільтр (використати інтерфейс BiquadFilterNode). Додати чекбокс, який вмикає або вимикає фільтр. Встановити параметри фільтра на свій смак.

**Теоритичні відомості**

**Web Audio API**

Web Audio API – це потужний і гнучкий інтерфейс програмування, який дозволяє створювати, обробляти та контролювати аудіо в веб-браузерах. Цей API надає розробникам можливості для складних аудіообробок без необхідності використовувати додаткові плагіни чи сторонні бібліотеки.

Інструмент включає в себе обробку аудіооперацій всередині аудіоконтексту. AudioContext є центральним об'єктом в Web Audio API. Він створює аудіограф, який складається з різних вузлів, що можуть бути джерелами звуку, вузлами обробки або вузлами призначення. AudioContext також відповідає за управління часом у аудіографі, що є важливим для синхронізації аудіообробки.

Основні аудіооперації виконуються за допомогою аудіовузлів, які пов’язані між собою, щоб утворити граф маршрутизації аудіо. Кілька джерел — із різними типами компонування каналів — підтримуються навіть в одному контексті. Така модульна конструкція забезпечує гнучкість для створення складних функцій з динамічними ефектами.

Web Audio API може здатися складним для тих, хто не знайомий із аудіо чи музичними термінами, і оскільки він включає в себе велику кількість функціональних можливостей, може виявитися складним почати працювати. Проте, завдяки наявності численних ресурсів, навчальних матеріалів і прикладів, освоєння цього інструменту стає значно простішим. Залучення до вивчення та використання Web Audio API може не лише розширити технічні навички, але й дозволити створювати унікальні аудіо проекти.

**BiquadFilterNode**

BiquadFilterNode є частиною Web Audio API і використовується для створення різноманітних фільтрів, які можуть змінювати частотний спектр аудіосигналу. Цей вузол підтримує декілька типів фільтрів, таких як низькочастотні, високочастотні, смугові та багато інших. Він є важливим інструментом для аудіообробки, дозволяючи створювати ефекти, покращувати якість звуку та налаштовувати його відповідно до потреб користувача.

BiquadFilterNode може представляти різні типи фільтрів, еквалайзерів та регуляторів тембру. Він завжди має один вхід і один вихід. BiquadFilterNode можна використовувати в різних сценаріях, наприклад, для створення ефекту еквалайзера, ревербу або для обробки аудіосигналу в реальному часі.

BiquadFilterNode має декілька ключових параметрів, які можна налаштувати для створення різних фільтрів, а саме

* Тип фільра. Кожен тип фільтра виконує специфічну функцію, що дозволяє досягати різних ефектів і результатів при обробці звуку. Наприклад, в мене за варіантом – це фільтра високих частот (highpass). Цей тип фільтра пропускає частоти вище встановленої змінної і обрізає частоти нижче неї.
* Частота. Цей параметр визначає точку зрізу, на якій фільтр починає обрізати сигнали в залежності від свого типу.
* Q-фактор. Визначає ширину смуги частот, які будуть підсилені або обрізані. Вищі значення Q роблять фільтр більш "вузьким".
* Підсилення. Визначає, наскільки частоти будуть підсилені або послаблені. Надмірне використання цього параметру може призвести до спотворення звуку. Тому рекомендується використовувати gain обережно та експериментувати з різними значеннями, щоб знайти те, що найкраще підходить для звуку.

BiquadFilterNode є важливим інструментом у Web Audio API, що дозволяє створювати і налаштовувати різні типи фільтрів для обробки аудіосигналів. Він забезпечує гнучкість і потужність, необхідні для створення високоякісних звукових ефектів та аудіопроектів, роблячи його незамінним інструментом для веб-розробників та аудіоінженерів

**Деталі розробки**

У ході виконання попередніх двох робіт було розроблено програму, що створює анагліфічне стереозображення моделі, а також дозволяє захоплювати та транслювати відео на фоні моделі.

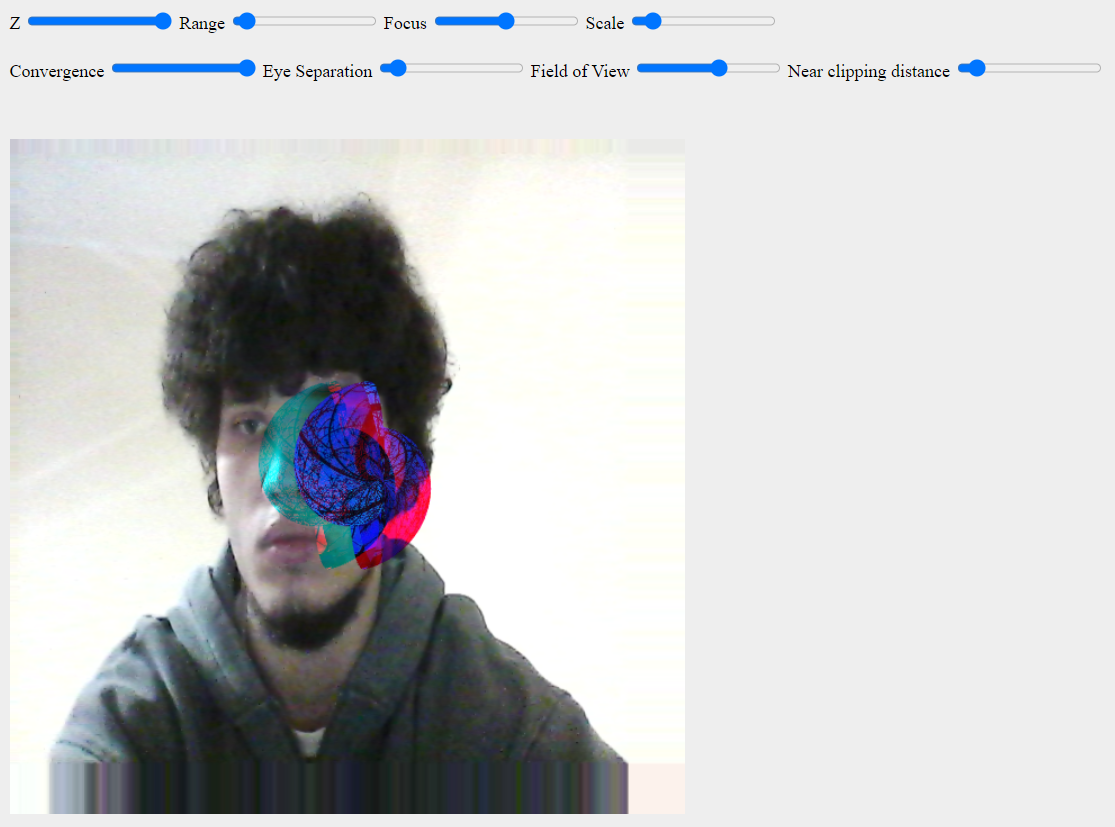


Рис. 1 – Стереозображення моделі

У ході розрахунково-графічної роботи було використано Web Audio API та BiquadFIlterNode. До існуючої програми було додано функціонал обертання джерела звуку навколо геометричного центру ділянки поверхні круговим способом протягом певного часу. Джерело звуку зображено на рисунку 2 у вигляді сфери.

За варіантом (9) було обрано фільтр високих частот (highpass). Такий тип фільтра пропускає частоти вище встановленої змінної і обрізає частоти нижче неї.

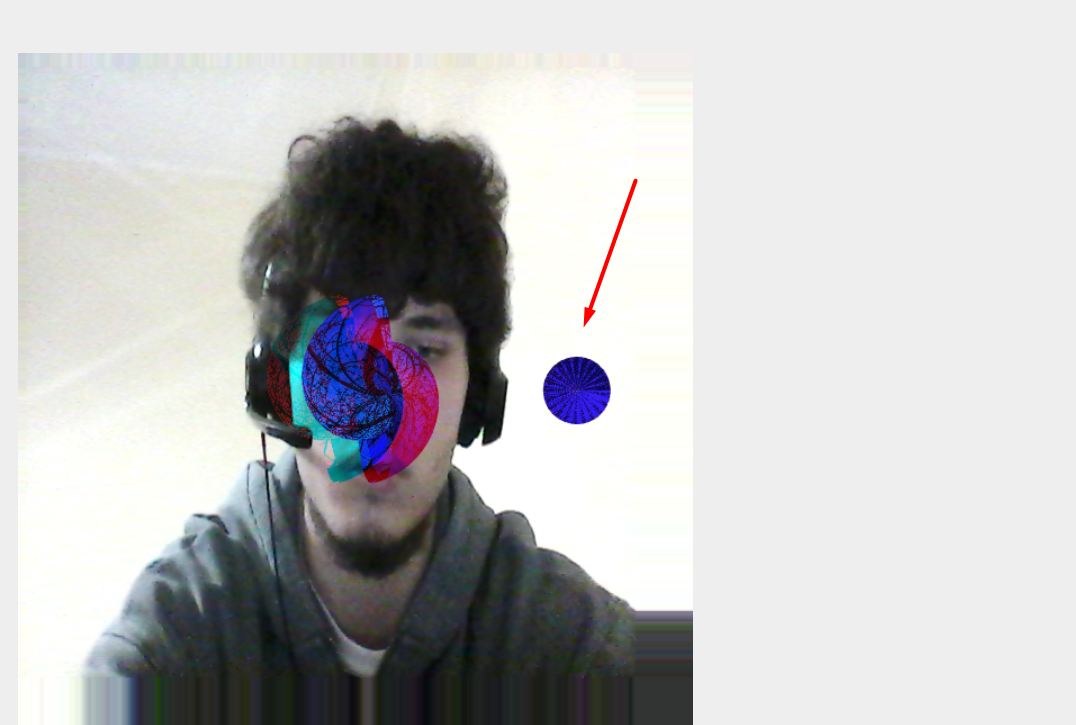


Рис. 2 – Візуалізоване джерело звуку у вигляді сфери.

Також, в ході роботи були вказані довільні параметри для фільтру частот, а саме frequency(1000) та Q(5) та оброблено сценарій паузи аудіо.

**Вказівки користувачу**

Користувач може керувати налаштуваннями стереозображення використовуючи елементи графічного інтерфейсу. Ці параметри впливають на виляд самого зображення, що можна побачити на рисунку 3. Також є можливість надання доступу до камери, який можна надати, або відхилити прямо в браузері.

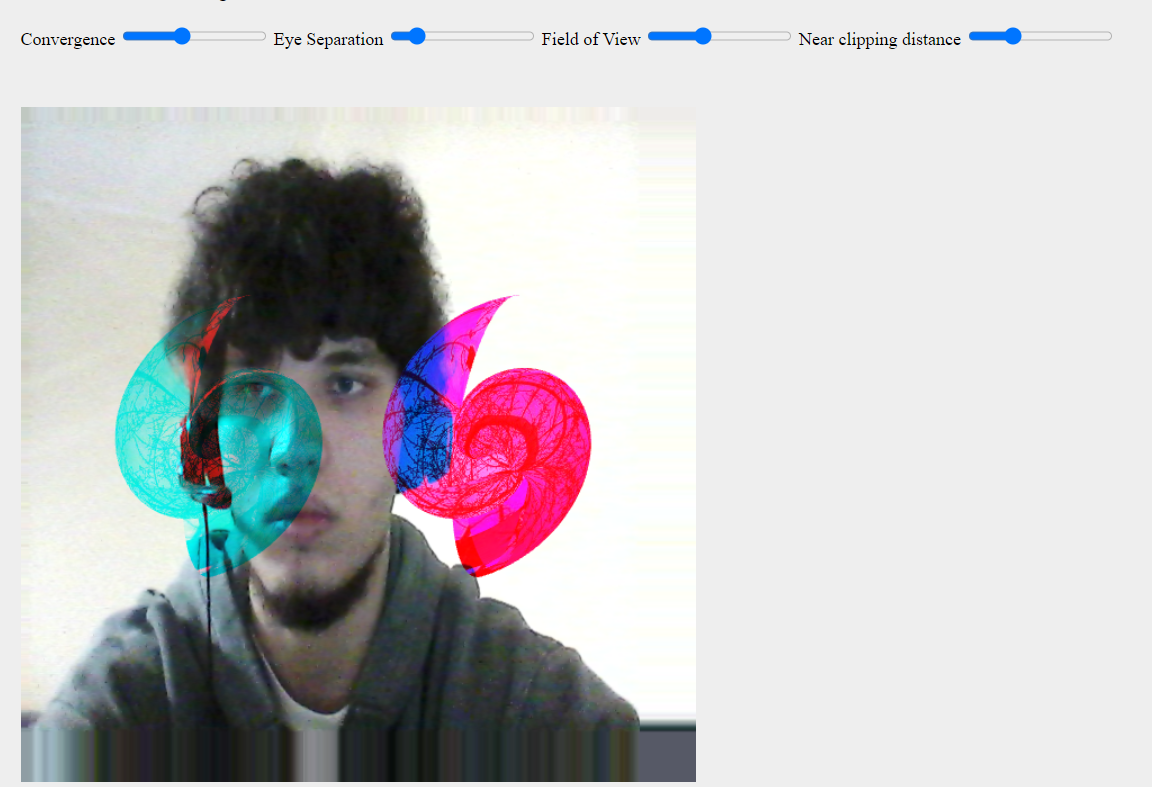


Рис. 3 – Елементи графічного інтерфейсу для керування стереозображенням.

Стосовно аудіо, у користувача є можливість ввімкнути та ввимкнути аудіозапис, керувати гучністю, а також перемотувати трек на потрібне місце. Ці всі можливості надані HTML-тегом <audio> який підтримується у всіх браузерах. Також є спеціальний елемент, за допомогою якого можна ввімкнути та вимкнути фільтр високих частот. Обидва елементи наведені на риснуку 4.

При ввімкнені чекбоксу, відбувається приглушення низьких частот. В такому випадку практично не чутно ударні інструменти, звук стає більш різким, та майже не чутно басу.

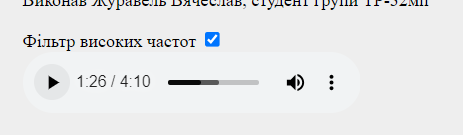


Рис. 4 – Елементи керування аудіо та фільтром частот.

Також, важливим аспектом є сфера, яка відповідає за центр джерела звуку та його місце в просторі. Користувач може явно побачити візуалізацію того, яким чином та наскільки звучання аудіозапису міняється в залежності від позиції сфери.

**Лістинг частини програми**

if (panner) {

        const s = 0.001;

        const time = Date.now()

        const x = Math.sin(time \* s)

        const z = Math.cos(time \* s)

        panner.setPosition(x, 0, z)

        modelViewProjection = m4.multiply(m4.identity(), m4.translation(x, 0, z));

        gl.uniformMatrix4fv(shProgram.iModelViewProjectionMatrix, false, modelViewProjection);

        lightSphere.Draw();

        gl.clear(gl.DEPTH\_BUFFER\_BIT);

    }

function initAudio() {

    audio = document.getElementById('audio');

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

        if (!context) {

            context = new AudioContext();

            audioSrc = context.createMediaElementSource(audio);

            panner = context.createPanner();

            biquadFilter = context.createBiquadFilter();

            audioSrc.connect(panner);

            panner.connect(biquadFilter);

            biquadFilter.connect(context.destination);

            biquadFilter.type = 'highpass';

            biquadFilter.Q.value = 5;

            biquadFilter.frequency.value = 1000;

            context.resume();

        }

    })

    audio.addEventListener('pause', () =>   context.resume())

    checkbox = document.getElementById('check-filter');

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

        if (checkbox.checked) {

            panner.disconnect();

            panner.connect(biquadFilter);

            biquadFilter.connect(context.destination);

        } else {

            panner.disconnect();

            panner.connect(context.destination);

        }

    });

    audio.play();

}