МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря Сікорського»
Навчально-науковий інститут атомної та теплової енергетики
Кафедра ЦТЕ

# ЗВІТ
про виконання графічно-розрахункової роботи
з дисципліни: «Візуалізація графічної та геометричної інформації»
На тему: «Операції над текстурними координатами»
Варіант №3

**Виконав:**
студент групи ТР-23мп
Бондарчук О.О.

**Перевірив:**
Демчишин А.А.

**Київ 2022**

# Постановка задачі

**Варіант**: 3 – непарний, реалізувати масштабування.

**Мета**: навчитися працювати з текстурами в WebGL, відобразити текстуру на поверхні з 2 практичного завдання та реалізувати масштабування текстурних координат.

**Вимоги**:

- Нанести текстуру на поверхню з практичного завдання №2.
- Реалізувати масштабування текстури (координати текстури) навколо визначеної користувачем точки.
- Реалізувати переміщення точки вздовж простору поверхні (u, v) за допомогою клавіатури. Клавіші **a** та **d** переміщують точку вздовж параметра **u**, а клавіші **w** та **s** переміщують точку вздовж параметра **v**.

**Звіт повинен містити**:

- титульну сторінку;
- розділ з постановкою задачі;
- розділ з теоретичними відомостями;
- розділ з описом деталей реалізації;
- розділ з інструкціями користувача зі скріншотами;
- зразок вихідного коду.

# Теоретичні відомості

Текстура (англ. Texture mapping) — це спосіб надання поверхні 3D деталі — полігону: кольору, фактури, блиску, матовості та інших фізичних властивостей (для імітації найчастіше якогось природного матеріалу, наприклад: паперу, дерева, каменю, металу тощо).

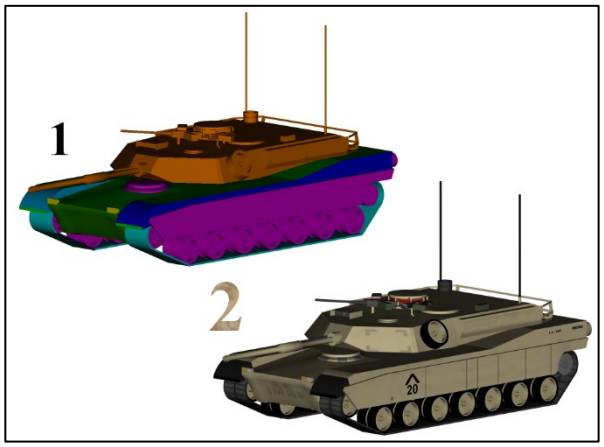

Рисунок 1. 3D-модель без текстур та 3D-модель з текстурами.

Поняття «текстура» є важливим елементом 3D-моделювання, оскільки дозволяє відтворити також малі об’єкти поверхні, створення яких полігонами виявилося б надмірно ресурсомістким. Наприклад, шрами на шкірі, складки на одязі, дрібні камені, предмети на поверхні стін і ґрунту та багато іншого. Отже, текстура використовується для заповнення поверхонь об’єктів і як шар для додання певного ефекту або зміни геометрії всьому зображенню або його частин. Якість поверхні текстури визначається кількістю пікселів на мінімальну одиницю текстури. Оскільки сама по собі текстура є зображенням, роздільність текстури і її формат відіграють велику роль, яка згодом позначається на загальному враженні від якості графіки у 3D-додатку.

Карта текстури застосовується для утворення певного параметру візуального відображення на поверхні заданої форми. Цей процес нагадує застосування візерунчастого паперу на звичайній білій коробці. Кожній вершині в 3D моделі присвоюється координати текстури (яка у разі 2D відома, як UV координата). Місця відбору зображення згодом інтерполюється по поверхні моделі з отриманням візуального результату. Найчастіше це файл растрового (рідше векторного) зображення з розширенням JPEG, PNG або PAT.

UV mapping — процес в 3D моделюванні, який полягає в накладанні двовимірного зображення на тривимірну модель. Літерами U і V позначають осі координат площини розгортки, оскільки літери X, Y і Z використовуються для позначення просторових координат.

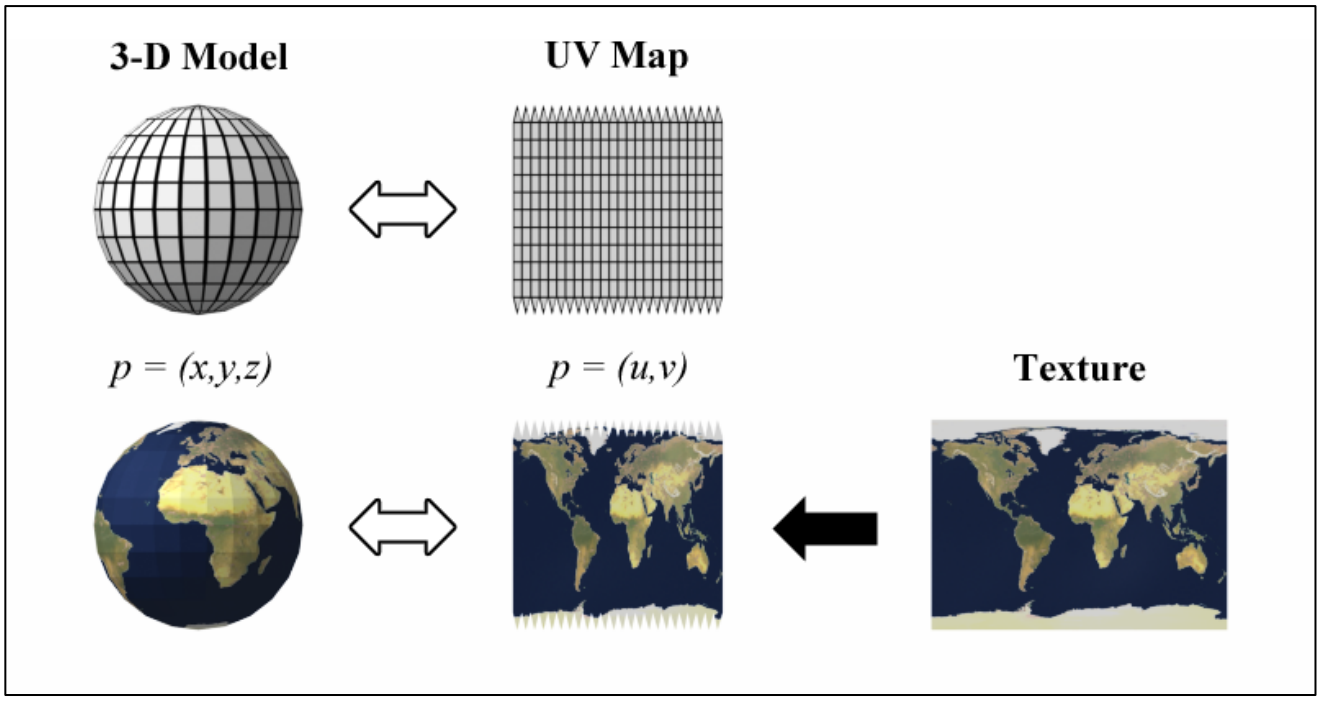


Рисунок 2. Приклад накладання двовимірної текстури (з координатами U, V) на 3D модель глобуса (з координатами X, Y, Z).

UV mapping (UV-розгортка) — відповідність між координатами на поверхні тривимірного об'єкту (X, Y, Z) і координатами на текстурі (U, V). Значення U і V зазвичай змінюються від 0 до 1. Розгортка може будуватися як вручну, так і автоматично.

Сучасне тривимірне апаратне забезпечення вважає, що UV-перетворення в межах одного трикутника є афінним — тому достатньо задати U і V для кожної вершини кожного з трикутників. Втім, як саме стикувати трикутники один з одним, вибирає 3D-моделер, і вміння будувати вдалу розгортку — один з показників його класу.

Існує кілька показників якості розгортки, які можуть суперечити один одному:

- Максимально повне використання площі текстури. Втім, в залежності від розриву між «мінімальними» і «максимальними» системними вимогами, по краях розгорнення текстурі потрібен певний «допуск» на генерацію текстур меншого розміру.
- Відсутність областей з недостатньою або надлишковою деталізацією текстури.
- Відсутність областей з зайвими геометричними спотвореннями.
- Подібність із стандартними ракурсами, з яких зазвичай малюється або фотографується об’єкт — спрощує роботу художника по текстурам.
- Вдало розташовані «шви» — лінії, відповідні одному ребру, але розташовані в різних місцях текстури. Шви бажані, якщо є природний «розрив» поверхні (шви одягу, кромки, зчленування і т. д.), і небажані, якщо таких немає.
- Для частково симетричних об’єктів: вдале поєднання симетричних і асиметричних ділянок розгорнення. Симетрія підвищує деталізацію текстури і спрощує роботу художника по текстурам; асиметричні деталі «оживляють» об’єкт.

Аби реалізувати масштабування навколо довільної точки, потрібно створити матрицю переміщення (**T**), яка переміщує об’єкт з початкової точки в точку масштабування, та матрицю масштабування (**R**), яка виконує масштабування об’єкта навколо початкової точки. Тепер для масштабування навколо довільної точки спершу необхідно перемістити точку на місце початкової точки за допомогою інвертування матриці переміщення (**T**), записаної як **T_1**. Потім, ми масштабуємо об’єкт відповідно до початкової точки, за допомогою матриці масштабування (**R**), а потім застосовуємо матрицю переміщення (**T**) для переміщення точки масштабування до свого вихідного положення.

# Опис деталей реалізації

Для відображення текстури на поверхні з комп'ютера завантажується зображення чи відео використовуючи функції JavaScript. Налаштування параметрів текстури було встановлено на такі значення:

- TEXTURE_WRAP_S:CLAMP_TO_EDGE;
- TEXTURE_WRAP_T:CLAMP_TO_EDGE;
- TEXTURE_MIN_FILTER — LINEAR.

Обрахунок текстурних координат відповідно кожного вертекса реалізовано в основному циклі і здійснюється за допомогою нормалізації параметрів **u** та **v**, за допомогою яких будується поверхня.

Також, були створені глобальні параметри, які зберігають координати точки масштабування та значення масштабування. Для взаємодії з веб-додатком реалізовано інтерфейс який включає в себе наступні клавіші та поля:

- (**a** та **d**) – переміщення точки масштабування вздовж параметра u;
- (**w** та **s**) – переміщення точки масштабування вздовж параметра v;
- (**+** та **-**) – зміна значення масштабування;
- Поля вводу типу *file* для завантаження фото чи відео;
- Поля вводу типу *text* для переміщення в просторі;
- Поле вводу типу *checkbox* для запуску анімації освітлення за параболою;
- Поля вводу типу *color* для задання кольорів;
- Поля вводу типу *range* для масштабування текстури та поверхні і зміни внутрішньої та зовнішньої межі освітлення.

Також, у вертексному шейдері було створено функцію, що реалізує масштабування текстурних координат відповідно заданій точці на поверхні.

## Інструкція користувача

Для масштабування вздовж параметра u використовується (**a** та **d**). Для масштабування вздовж параметра v використовується (**w** та **s**). (+ та -) використовуються для зміни значення масштабування. Поля вводу типу *file* використовуються для завантаження фото чи відео. Поля вводу типу *text* використовуються для переміщення в просторі. Поле вводу типу *checkbox* використовуються для запуску анімації освітлення за параболою. Поля вводу типу *color* використовуються для задання кольорів. Поля вводу типу *range* використовуються для масштабування текстури та поверхні і зміни внутрішньої та зовнішньої межі освітлення.

При завантаженні сторінки, за замовчуванням задано фото, відео та наступні параметри (рисунок 3):

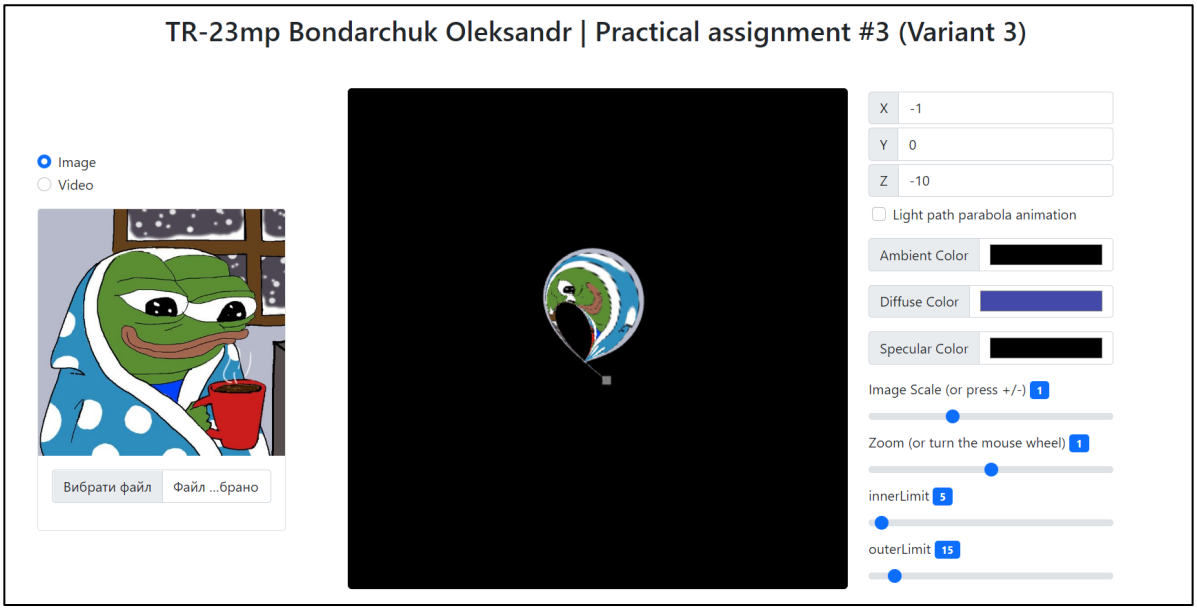


Рисунок 3. Сторінка веб-додатку.

Натиснувши на відповідні кнопки, користувач має змогу завантажити зі свого комп'ютера фотографію чи відео яке він хотів би нанести на поверхню.

На рисунку 4, зображено відрисовку відео на поверхні. Для переконання ви можете перейти за наступним посиланням (https://bondar4uk.github.io/VGGI) та завантажити своє відео.

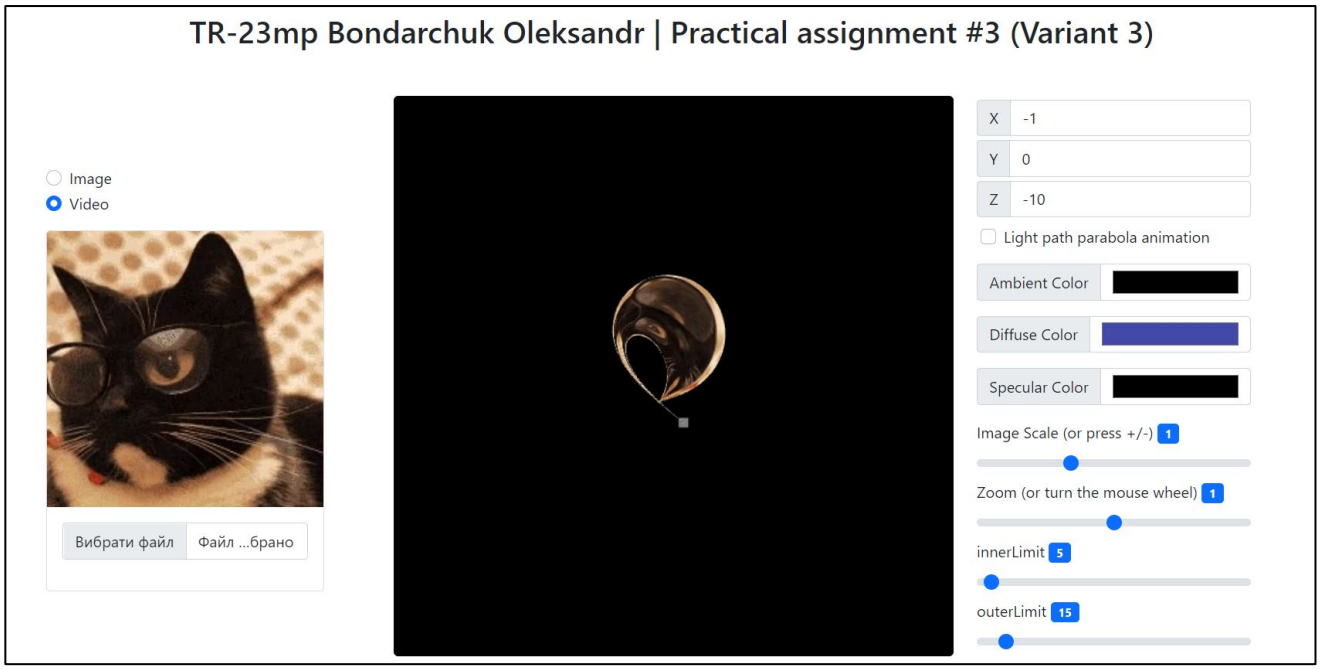


Рисунок 4. Демонстрація накладання відео на поверхню.

На наступному рисунку, зображено масштабування текстури.

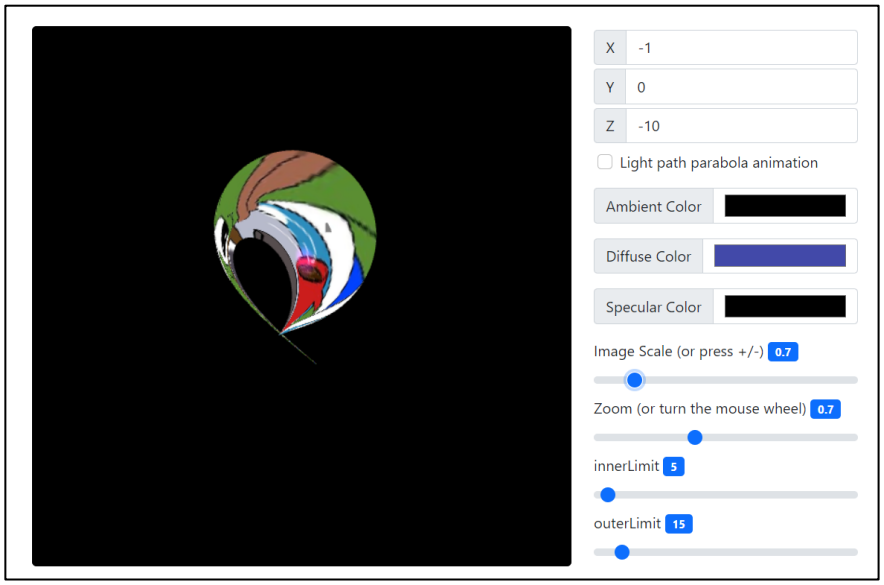


Рисунок 5. Демонстрація масштабування текстури.

## Зразок вихідного коду

Код, що відповідає за завантаження зображення, його створення та ініціалізацію, для подальшого відображення на поверхні:

```
function loadTexture(gl, url) {
    var texture = gl.createTexture();
    gl.bindTexture(gl.TEXTURE_2D, texture);

    gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 1, 1, 0, gl.RGBA, gl.UNSIGNED_BYTE,
        new Uint8Array([0, 0, 255, 255]));

    var image = new Image();
    image.crossOrigin = "anonymous"
    image.onload = () => {
        gl.bindTexture(gl.TEXTURE_2D, texture);
        gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, gl.RGBA, gl.UNSIGNED_BYTE,
image);

        if (isPowerOf2(image.width) && isPowerOf2(image.height)) {
            gl.generateMipmap(gl.TEXTURE_2D);
        } else {
            gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_S, gl.CLAMP_TO_EDGE);
            gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_T, gl.CLAMP_TO_EDGE);
            gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);
        }
        draw();
    };
    image.src = url;

    return texture;
}

function isPowerOf2(value) {
    return (value & (value - 1)) === 0;
}

var texture = loadTexture(gl, image_src);
```

Код, що відповідає за завантаження відео, його створення та ініціалізацію, для подальшого відображення на поверхні:

```
function initTexture(gl) {
    var texture = gl.createTexture();
    gl.bindTexture(gl.TEXTURE_2D, texture);

    gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 1, 1, 0, gl.RGBA, gl.UNSIGNED_BYTE,
        new Uint8Array([0, 0, 255, 255]));

    gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_S, gl.CLAMP_TO_EDGE);
    gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_T, gl.CLAMP_TO_EDGE);
    gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

    return texture;
}

function updateTexture(gl, texture, video) {
    gl.bindTexture(gl.TEXTURE_2D, texture);
    gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, gl.RGBA, gl.UNSIGNED_BYTE, video);
    draw();
}
```

```
function setupVideo(url) {
    var video = document.createElement("video");
    video.crossOrigin = "anonymous"

    let playing = false;
    let timeupdate = false;

    video.playsInline = true;
    video.muted = true;
    video.loop = true;

    video.addEventListener(
        "playing",
        () => {
            playing = true;
            checkReady();
        },
        true
    );

    video.addEventListener(
        "timeupdate",
        () => {
            timeupdate = true;
            checkReady();
        },
        true
    );

    video.src = url;
    video.play();

    function checkReady() {
        if (playing && timeupdate) {
            copyVideo = true;
        }
    }

    return video;
}

var texture = initTexture(gl);
var video = setupVideo(video_src);
var then = 0;

function render(now) {
    now *= 0.001;
    var deltaTime = now - then;
    then = now;
    if (copyVideo) {
        updateTexture(gl, texture, video);
    }
    if (texture_type != "image") {
        requestAnimationFrame(render);
    }
}
requestAnimationFrame(render);
```