TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

Report Aquarium

Môn: Đồ họa ứng dụng

Giáo viên hướng dẫn: Phạm Thanh Tùng
Phạm Minh Hoàng
Võ Hoài Việt

Lớp: 20TGMT

Sinh viên thực hiện: Phạm Nguyễn Anh Quốc - 19127534
Giang Gia Bảo – 20127446

Tp Hồ Chí Minh, 09 tháng 05 năm 2023

MỤC LỤC

I. THÔNG TIN SINH VIÊN

MSSV	Họ và tên	Lớp	Note
20127446	Giang Gia Bảo	20TGMT	
19127534	Phạm Nguyễn Anh Quốc	20TGMT	

II. MÔ HÌNH FISH TANK

Fish Tank là chủ đề nhóm đã chọn để thực hiện việc tạo dựng scene trong môi trường 3D. Mô hình Fish Tank được thiết kế để mô phỏng các bể cá trong thế giới thực sao cho mức độ chân thực cao nhất có thể: mô hình Fish Tank sẽ bao gồm các mô hình về các loại cá khác nhau có trong thực tế, các động - thực vật thủy sinh và các vật trang trí như là san hô, sỏi….; đồng thời, sự chuyển động của cá và những vật thể khác trong hồ cũng như sự chuyển động của mặt nước cũng sẽ được chú trọng thiết kế để mang lại cảm giác không quá khác biệt so với thế giới thực.

Mô hình Fish Tank là mô hình khá đơn giản và phổ biến trong thiết kế đồ họa 3D. Mô hình này được sử dụng khá nhiều trong các phần mềm mô phỏng, giúp cho người dùng có thể sắp xếp, hình dung được bể cá của mình nếu thêm các vật trang trí hay sinh vật thủy sinh vào thì sẽ trở thành hình dạng gì, có đáp ứng được nhu cầu về thẩm mỹ mà mình mong muốn hay không…

Fish Tank cũng có thể được sử dụng trong các game với mức độ quan trọng khác nhau: từ ít quan trọng như nằm trong một cảnh vật lớn được tạo dựng trong các game hành động, thế giới mở đến cốt lõi như là các game giả lập về nuôi cá, câu cá, bắn cá;…. Chúng ta có thể thấy được rằng, dù chỉ là một mô hình không quá phức tạp nhưng Fish Tank vẫn có thể được ứng dụng trong nhiều lĩnh vực, khía cạnh trong cuộc sống, vì thế, với những kiến thức đã tiếp thu được về đồ họa 3D, nhóm sẽ thực hiện tạo dựng scene cho mô hình Fish Tank này.

Mô tả

Thư viện	Chương trình chủ yếu sử dụng thư viện “Three.js”
Viewpoint (camera)	Xoay camera (orbit)
Vật thể	Các loại cá, sao biển, tảo biển,… Cát, sỏi đá, san hô, …
Feature	Lighting và sự di chuyển của cá trong hồ

III. FRAMEWORK

1. Tạo và hiển thị đối tượng 3 chiều

Bước 1: Tạo cấu trúc dữ liệu gồm đỉnh, cạnh và các đa giác (ví dụ: hàm đặc tả đối tượng, cấu trúc file .obj)

Bước 2: Xác định mặt phẳng chiếu (hướng nhìn, tọa độ camera)

Bước 3: Xác định phép chiếu (song song, phối cảnh)

Bước 4: Chuyển tọa độ thế giới thực về tọa độ màn hình WinToView

Bước 5: Vẽ cá, sao biển,..

2. Môi trường đồ họa

Render WebGL vào một trang hoặc một ứng dụng

Bước 1: Tạo canvas và vẽ bằng context (Khởi động môi trường đồ họa)

Bước 2: Khởi tạo viewport

Bước 3: Tạo các buffer để render (chuẩn bị dữ liệu trong buffer)

Bước 4: Tạo 1 hoặc nhiều ma trận để xác định phép biến đổi từ front buffer sang không gian màn hình (đẩy ra memory card, buffer màn hình)

Bước 5: Tạo 1 hoặc nhiều shader để thực hiện thuật toán vẽ

Bước 6: Khởi tạo shader với các tham số

Bước 7: Vẽ

IV. THƯ VIỆN

1. ThreeJS

ThreeJS là một thư viện JavaScript mã nguồn mở được sử dụng để tạo và hiển thị đồ họa 3D trên web. Nó được phát triển bởi Ricardo Cabello và được cấp phép theo giấy phép MIT.

ThreeJS sử dụng WebGL, một API đồ họa 3D được tích hợp sẵn trong các trình duyệt web hiện đại, để hiển thị các đối tượng 3D trên trang web. ThreeJS cung cấp các công cụ và tính năng để dễ dàng tạo ra các đối tượng 3D, cũng như hiển thị chúng trên một bề mặt web và tương tác với chúng.

Với ThreeJS, bạn có thể tạo ra các đối tượng 3D phức tạp và đa dạng, bao gồm các đối tượng hình học cơ bản (như hình cầu, hình trụ, hình hộp), các mesh, ánh sáng, môi trường, vật liệu và các hiệu ứng hình ảnh đặc biệt.

ThreeJS là một trong những thư viện phổ biến nhất để phát triển ứng dụng đồ họa 3D trên web và được sử dụng trong nhiều ứng dụng khác nhau, bao gồm các trò chơi, trình trình diễn sản phẩm và các ứng dụng giáo dục.

Trong project này, ThreeJS sẽ đóng vai trò cốt lõi trong việc tạo dựng mô hình Fish Tank và hiển thị nó trong không gian 3D.

2. Ưu điểm

Một số ưu điểm của ThreeJS là:

- Nó ẩn các chi tiết khác nhau của WebGL, giúp làm việc với đồ họa 3D trên web dễ dàng hơn.
- Nó được hỗ trợ bởi hầu hết tất cả các trình duyệt web, vì vậy bạn không phải lo lắng về vấn đề tương thích.
- Nó không yêu cầu bất kỳ ứng dụng khách hoặc plugin nào để thêm đồ họa 3D vào các trang web của bạn, điều này làm cho nó dễ truy cập và thân thiện hơn với người dùng.
- Nó cung cấp quyền truy cập vào sức mạnh đồ họa của máy tính, cho phép bạn tạo hoạt ảnh mượt mà và cảnh 3D phức tạp.
- Nó có một API nhỏ, dễ học và dễ sử dụng, đồng thời nó có thể được tích hợp với các thư viện và khung JavaScript khác.

- Nó có một cộng đồng rộng lớn và tích cực, với nhiều tài nguyên trực tuyến có sẵn để tìm hiểu và khắc phục sự cố.

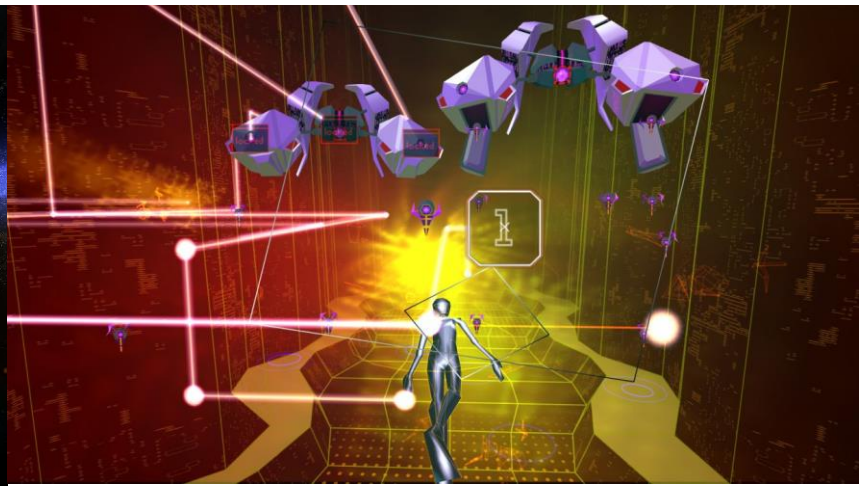
3. Các trò chơi nổi tiếng sử dụng threeJS

Crossy road

Clash of Avatars

Run 3

Rez Infinite



4. Cài đặt Threejs

https://github.com/mrdoob/three.js/

```
<script type="importmap">
{
  "imports": {
    "three": "https://unpkg.com/three@0.129.0/build/three.module.js",
    "three/addons/": "https://unpkg.com/three@0.129.0/examples/jsm/"
  }
}
</script>
```

```
import * as THREE from 'three';
```

V. MÃ NGUỒN

Những bước thiết lập buffers, thiết lập trạng thái render, clear viewports, thiết lập shader, thiết lập texture,… được thực hiện thông qua:

```
RENDERER.RENDER(SCENE, CAMERA)
```

Render WebGL:

```
let canvas = document.querySelector("#gl-canvas");
CONST RENDERER = NEW THREE.WEBGLRENDERER({ CANVAS, ANTIALIAS: TRUE });
```

.js file

```
<CANVAS ID="GL-CANVAS">
        OOPS! IT HAS ERROR!
  </CANVAS>
```

.html file trong thẻ body

1. Point, vector và mesh

Load đối tượng .obj file

```
    IMPORT { OBJLOADER } FROM
"HTTPS://THREEJS.ORG/EXAMPLES/JSM/LOADERS/OBJLOADER.JS";


    CONST LOADER = NEW OBJLOADER();
    LOADER.LOAD( 'HTTPS://CDN.GLITCH.GLOBAL/D934A862-1D4B-4035-B04D-
64826046BC27/FISHTANK.OBJ?V=1672188948843', FUNCTION ( OBJ ) {
          OBJECT = OBJ;
          OBJECT.POSITION.Y = 30;
          SCENE.ADD( OBJECT );
    }
```

2. Color và Material

Load texture vào đối tượng .obj file bằng MTLLoader và OBJLoader

```
      const onProgress = function ( hr ) {
        if ( hr.lengthComputable ) {
          const percentComplete = hr.loaded / hr.total * 100;
          console.log( Math.round( percentComplete, 2 )+'% downloaded' );
        }
      };
      new MTLLoader().load('https://cdn.glitch.global/d934a862-1d4b-4035-b04d-
64826046bc27/fishTank.mtl?v=1672189681410', function ( materials ) {
      materials.preload();


      const loader = new OBJLoader();
      loader.setMaterials( materials );
      loader.load( 'https://cdn.glitch.global/d934a862-1d4b-4035-b04d-
64826046bc27/fishTank.obj?v=1672188948843', function ( obj ) {


        object = obj;
        // object.scale.multiplyScalar( 0.8 );
        object.position.y = 30;
        scene.add( object );


        },onProgress );
```

```
    });
```

3. Camera

Phép chiếu phối cảnh

```
   CAMERA = NEW THREE.PERSPECTIVECAMERA( 50, WINDOW.INNERWIDTH / WINDOW.INNERHEIGHT, 1, 1000 );
    CAMERA.POSITION.SET(0, 0, 100);
```

Xoay camera bằng chuột

```
   CONTROLS = NEW ORBITCONTROLS( CAMERA, RENDERER.DOMELEMENT );
    CONTROLS.UPDATE();
```

```
   CONTROLS.UPDATE()
     RENDERER.RENDER( SCENE, CAMERA );
```

4. Light

Point light

```
CONST LIGHT = NEW THREE.POINTLIGHT(COLOR, INTENSITY, DISTANCE);
```

Directional light

```
LIGHT = NEW THREE.DIRECTIONALLIGHT(COLOR, INTENSITY);
```

5. Texture và Reflection

Hiệu ứng dòng nước với thuật toán Flow map

```
IMPORT { WATER } FROM "HTTPS://THREEJS.ORG/EXAMPLES/JSM/OBJECTS/WATER.JS";
const waterGeometry = new THREE.PlaneGeometry( 30.5, 11 );
  water = new Water(
    waterGeometry,
    {
      textureWidth: 512,
      textureHeight: 512,
```

```
      waterNormals: new
THREE.TextureLoader().load( 'https://cdn.glitch.global/d934a862-1d4b-4035-b04d-
64826046bc27/Water_1_M_Normal.jpg?v=1672657598625', function ( texture ) {


        texture.wrapS = texture.wrapT = THREE.RepeatWrapping;


      } ),
      sunDirection: new THREE.Vector3(),
      sunColor: 0xffffff,
      waterColor: 0x001e0f,
      distortionScale: 3.7,
      fog: scene.fog !== undefined
    }
  );

  // water.rotation.x = -Math.PI /2;
  water.position.x = 0
  water.position.y = 0
  water.position.z = 15
  scene.add( water );
```

VI. LÍ THUYẾT

1. Point, vector và mesh

1.1. Cấu trúc file .obj

```
# List of geometric vertices, with (x, y, z, [w]) coordinates, w tùy chọn,
mặc định là 1.0
v 0.123 0.234 0.345 1.0
v ...
...
# List các tọa độ texture, in (u, [v, w]) coordinates, khoảng giá trị từ 0
tới 1. v, w are tùy chọn and mặc định 0
vt 0.500 1 [0]
vt ...
...
# List of vertex normals in (x,y,z) form; normals might not be vector đơn
vị.
vn 0.707 0.000 0.707
vn ...
...
# Parameter space vertices in (u, [v, w]) form; free form geometry
statement
vp 0.310000 3.210000 2.100000
vp ...
```

```
...
# Polygonal face element
f 1 2 3
f 3/1 4/2 5/3
f 6/4/1 3/5/3 7/6/5
f 7//1 8//2 9//3
f ...
...
# Line element
l 5 8 1 2 4 9
```

2. CAMERA

Phép chiếu phối cảnh

Projection Reference Point: $C(c_x, c_y, c_z)$

View Plane: $(R_0, \vec{N}), R_0(x_0, y_0, z_0), \vec{N}(n_1, n_2, n_3)$

$$P' = Per(C, (R_0, \vec{N})).P$$

$$\begin{bmatrix} x_H \\ y_H \\ z_H \\ H \end{bmatrix} = \begin{bmatrix} d + c_x.n_1 & c_x.n_2 & c_x.n_3 & -c_x.d_0 \\ c_y.n_1 & d + c_y.n_2 & c_y.n_3 & -c_y.d_0 \\ c_z.n_1 & c_z.n_2 & d + c_z.n_3 & -c_z.d_0 \\ n_1 & n_2 & n_3 & -d_1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$d_0 = n_1 x_0 + n_2 y_0 + n_3 z_0 \quad d_0 = n_1.c_x + n_2.c_y + n_3.c_z$$

$$d = d_0 - d_1$$

Với:

$$Per(C, (R_0, \vec{N})) = T^{-1}(\vec{CO}).Per(O, (R_0, \vec{N})).T(\vec{CO})$$

$$= \begin{bmatrix} 1 & 0 & 0 & c_x \\ 0 & 1 & 0 & c_y \\ 0 & 0 & 1 & c_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 1 & 0 \\ 0 & 0 & d & 0 \\ n_1 & n_2 & n_3 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. PHÉP BIẾN ĐỔI VÀ MA TRẬN

Phép xoay

Chuyển tọa độ thế giới thực về tọa độ màn hình WinToView

$$P(x,y) \rightarrow P'(x',y')$$
$$P' = WTV.P$$
$$s_x = \frac{v_{max}.x - v_{min}.x}{w_{max}.x - w_{min}.x}$$
$$s_y = \frac{v_{max}.y - v_{min}.y}{w_{max}.y - w_{min}.y}$$
$$t_x = v_{min}.x - w_{min}.x$$
$$t_y = v_{min}.y - w_{min}.y \Rightarrow WTV = S(v_{min}, s_x, s_y).T(t_x, t_y)$$
$$\Rightarrow WTV = \begin{pmatrix} s_x & 0 & v_{min}.x - s_x.v_{min}.x \\ 0 & s_y & v_{min}.y - s_y.v_{min}.y \\ 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 1 & 0 & v_{min}.x - w_{min}.x \\ 0 & 1 & v_{min}.y - w_{min}.y \\ 0 & 0 & 1 \end{pmatrix}$$
$$\Rightarrow \begin{pmatrix} s_x & 0 & v_{min}.x - s_x.w_{min}.x \\ 0 & s_y & v_{min}.y - s_y.w_{min}.y \\ 0 & 0 & 1 \end{pmatrix}$$

4. ÁNH SÁNG

Hemisphere light: ánh sáng cho bề mặt trần và sàn.

Spotlight: nguồn sáng từ 1 điểm ra hình nón

Point light: nguồn sáng từ 1 điểm toả ra mọi hướng

5. INTERACTION VÀ ANIMATION

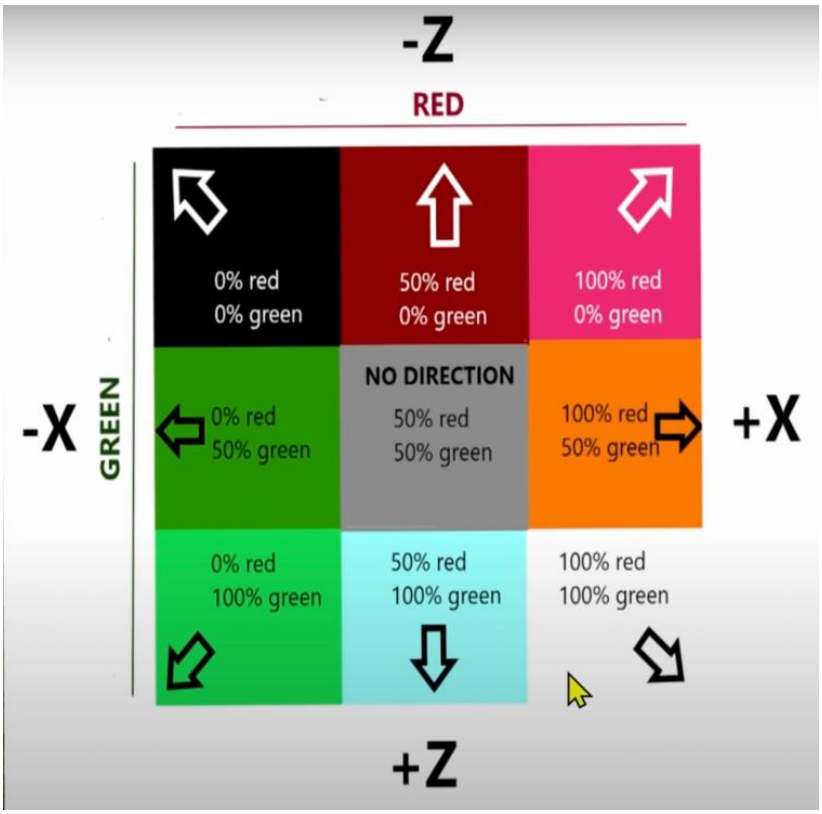
5.1. Texture và reflection

Hiệu ứng dòng nước:

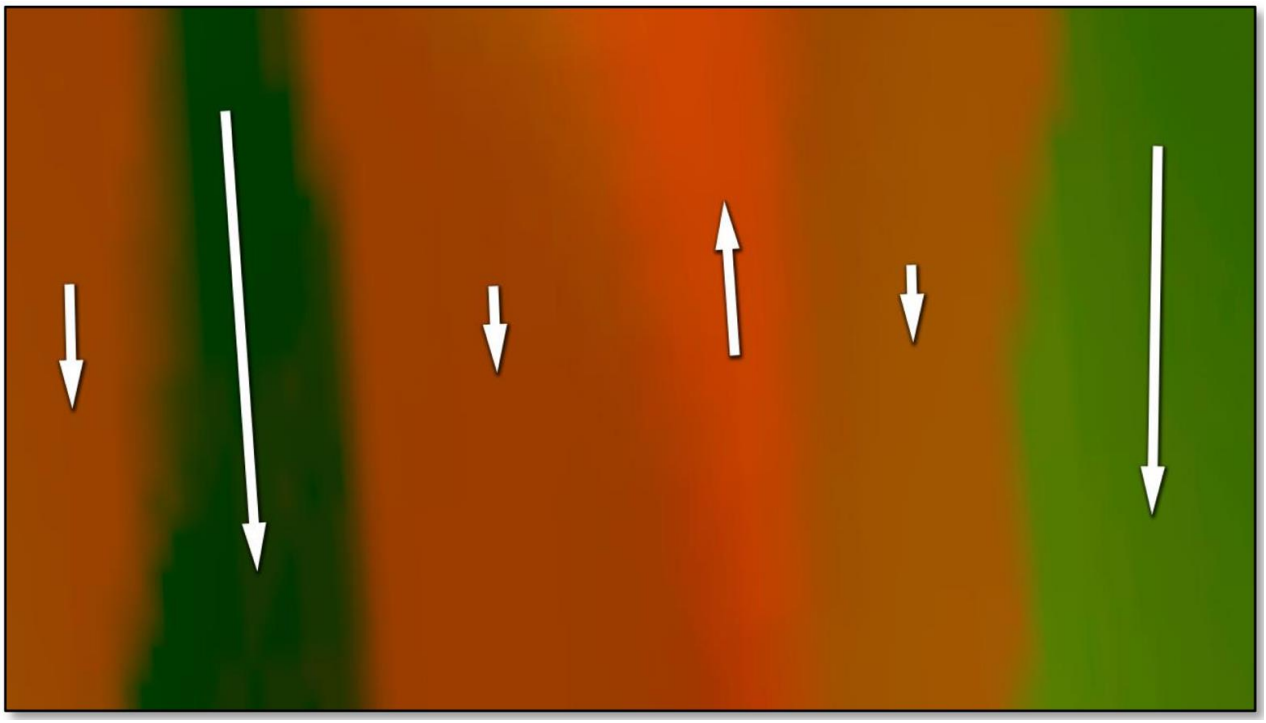
Dòng chảy là các vector 2D.

5.2. Nguyên lí Flow map

Hướng dòng chảy?



Mỗi pixel có 1 giá trị, quy định Red Green khoảng giá trị 0 – 100 (%)
Red > 50% thì di chuyển nhiều hơn về bên phải
Red = 50% không di chuyển
Red < 50% di chuyển nhiều hơn về bên trái
Green > 50% di chuyển xuống dưới
Green = 50% không di chuyển
Green < 50% di chuyển lên trên



VII. REFERENCE

[1] Mã nguồn mở github

[2] Hiệu ứng dòng chảy

[3] Cấu trúc .obj

[4] Collision threejs

[5] How to detect collisions three.js

[6] Kiểm tra va chạm (trang 42)

[7] Three.js – webgl ocean