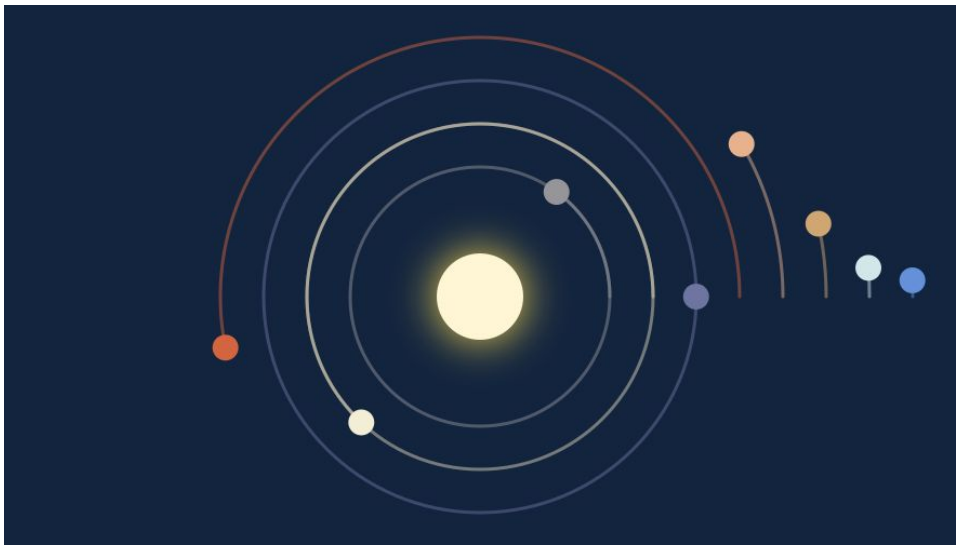
ETS Grafika Komputer

Hafizh Fauzan - 07211740000019

https://github.com/Fizdan/Grafika-Komputer/tree/main/ETS

Pada tugas Evaluasi Tengah Semester kali ini diberikan tugas yaitu membuat sebuah animasi sistem tata surya dengan menggunakan WebGL. Untuk ketentuannya sebagai berikut:

1. Buat program dalam WebGL untuk mensimulasikan animasi tata surya, minimum sampai dengan planet Bumi (nilai plus jika dapat menggambar lebih banyak)
    a. Tidak perlu menggunakan texture, tetapi beri warna untuk mebedakan satu planet dengan planet yang lain
    b. Jangan lupa menggambar bulan (wajib)
2. Buatlah laporan bagaimana Anda mengimplementasikan animasi tata surya tersebut.
3. Bonus: Jika ada penambahan camera (materi akan dipost akhir minggu)



Gambar 1. Tata Surya

Berikut merupakan laporan pengerjaan animasi tata surya menggunakan WebGL:

Menggambar Lingkaran

Untuk menggambar lingkaran pada kali ini saya menggunakan library yang ada, yaitu twgl, yang dokumentasi nya dapat diakses pada https://twgljs.org/ . Dengan menggunakan library tersebut, maka saya tidak perlu membuat vertices secara manual. Lalu kemudian dalam membuat lingkaran saya menggunakan HTML canvas arc() Method, seperti berikut:

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(x, y, 50, 0, 2 * Math.PI);
```

```
ctx.stroke();
```

dimana nanti untuk melakukan translasi saya hanya perlu merubah variable X dan variable Y dan melakukan render ulang.

Pop dan Stack

Pada bagian ini menjelaskan bagaimana implementasi Matrix Stack. Dibuat fungsi untuk mengalikan matriks atas tumpukan menggunakan baik dengan matriks terjemahan, rotasi, atau skala. Ketika ingin melakukan translasi atau rotasi pada matriks yang diinginkan, maka pertama melakukan Stack(push) terlebih dahulu. Jika matriks sudah selesai diubah dan ingin melakukan perubahan pada matriks lainnya, melakukan Pop. Jika ingin ada passing parameter dari matriks sebelumnya, maka melakukan Stack didalam Stack.

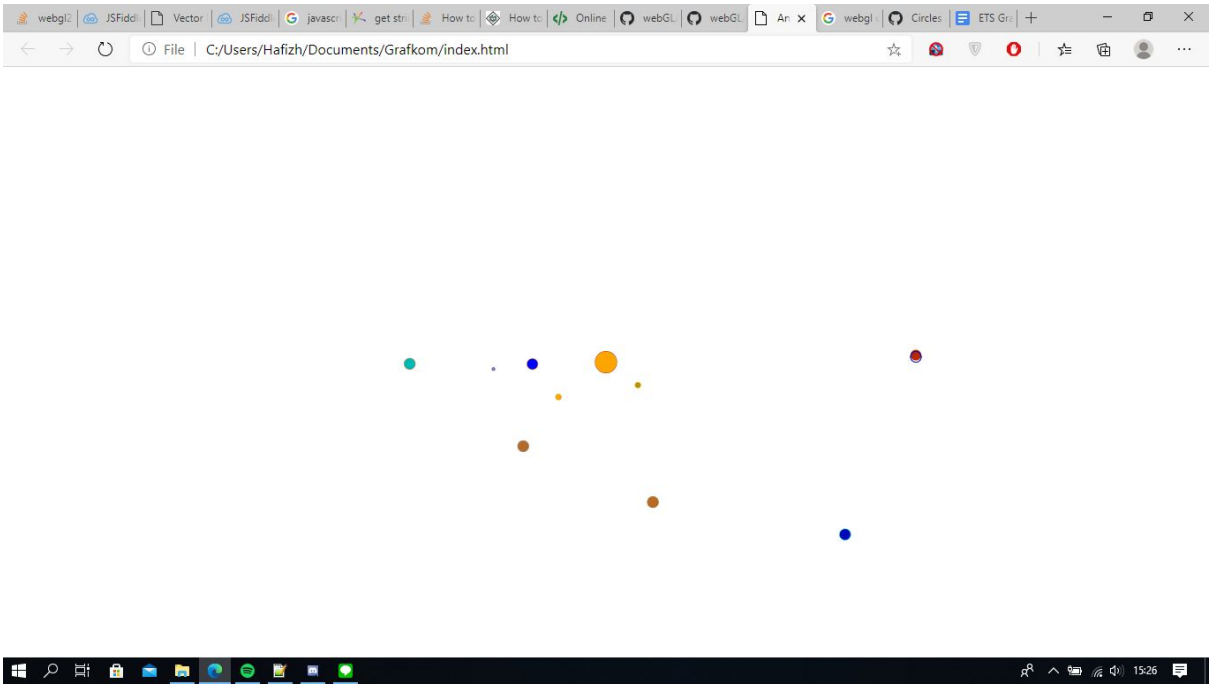
```
identity();
    // translate to center of solar system
    translate([width / 2, height / 2, 0]);
        sunMat = current();
      rotate(zAxis, time / 10) // push(mercury orbit rotation)
                           translate([height / 15, 0, 0]);
//push(mercury translation)
           rotate(zAxis, time * 8); // push(mercury rotation)
                mercuryMat = current();
           pop(); // mercury rotation
           pop(); // mercury translation
     pop(); // mercury rotation
      rotate(zAxis, time / 15) // push(venus orbit rotation)
                           translate([height / 10, 0, 0]);
//push(venus translation)
           rotate(zAxis, time * 8); // push(venus rotation)
                venusMat = current();
           pop(); // venus rotation
           pop(); // venus translation
     pop(); // venus rotation
      rotate(zAxis, time / 2) // push(earth orbit rotation)
        translate([height / 8, 0, 0]); //push(earth translation)
          rotate(zAxis, time * 8); // push(earth rotation)
            earthMat = current();  //
          pop(); // earth rotation
          rotate(zAxis, time * 2); //push(moon orbit rotation)
            translate([height / 15, 0, 0]); // push(moon
translation)
              rotate(zAxis, time * 16); //push(moon rotation)
                moonMat = current();
              pop();  // moon rotation
            pop();  // moon translation
```

```
          pop();  // moon orbit rotation
        pop();  // earth translation
      pop();  // earth orbit rotation
           rotate(zAxis, time / 8) // push(mars orbit rotation)
                          translate([height / 5, 0, 0]);
//push(mars translation)
           rotate(zAxis, time * 8); // push(mars rotation)
                marsMat = current();
           pop(); // mars rotation
           pop(); // mars translation
     pop(); // mars rotation
      rotate(zAxis, time / 5) // push(jupiter orbit rotation)
                          translate([height / 4, 0, 0]);
//push(jupiter translation)
           rotate(zAxis, time * 8); // push(jupiter rotation)
                jupiterMat = current();
           pop(); // jupiter rotation
           pop(); // jupiter translation
     pop(); // jupiter rotation
      rotate(zAxis, time / 6) // push(saturn orbit rotation)
                          translate([height / 3, 0, 0]);
//push(saturn translation)
           rotate(zAxis, time * 8); // push(saturn rotation)
                saturnMat = current();
           pop(); // saturn rotation
           pop(); // saturn translation
     pop(); // saturn rotation
      rotate(zAxis, time / 10) // push(uranus orbit rotation)
                          translate([height / 2, 0, 0]);
//push(uranus translation)
           rotate(zAxis, time * 8); // push(uranus rotation)
                uranusMat = current();
           pop(); // uranus rotation
           pop(); // uranus translation
     pop(); // uranus rotation
      rotate(zAxis, time / 3) // push(neptune orbit rotation)
                          translate([height / 1.9, 0, 0]);
//push(neptune translation)
           rotate(zAxis, time * 8); // push(neptune rotation)
                neptuneMat = current();
           pop(); // neptune rotation
           pop(); // neptune translation
     pop(); // neptune rotation
    pop();  // center of solar system
  pop();  // identity
```

Render

Setelah semua matriks set dan posisi lingkaran juga telah di set berdasarkan matriks tersebut, hanya tinggal melakukan render. Pemanggilan render dilakukan berdasarkan satuan waktu, dimana dia melakukan looping. Oleh karena itu setiap loop kita melakukan translasi matriks dan redraw circle berdasarkan matriks tersebut.

berikut merupakan hasil dari animasi tata surya



Gambar 2. Animasi Tata Surya

Source Code

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Animated circles</title>

<style>
      body {
  margin: 0;
}

canvas {
  display: block;
  width: 100vw;
  height: 100vh;
}
```

```
#ui {
  position: absolute;
  left: 10px;
  top: 10px;
  z-index: 2;
  background: rgba(255, 255, 255, 0.9);
  padding: .5em;
}

</style>

<script src="https://twgljs.org/dist/4.x/twgl-full.js"></script>

</head>

<body>
<canvas id="canvas"></canvas>
<script>
'use strict';
const ctx = document.querySelector('canvas').getContext('2d');
const m4 = twgl.m4;

const stack = [];
const current = () => stack[stack.length - 1];
const pop = () => stack.pop();
const identity = () => stack.push(m4.identity());
const translate = (t) => stack.push(m4.translate(current(), t));
const rotate = (axis, r) => stack.push(m4.axisRotate(current(),
axis, r));
const zAxis = [0, 0, 1];

function render(time) {
  time *= 0.001; // convert to seconds

  twgl.resizeCanvasToDisplaySize(ctx.canvas);
  const {width, height} = ctx.canvas;
  ctx.setTransform(1, 0, 0, 1, 0, 0);
  ctx.clearRect(0, 0, width, height);

  let sunMat;
  let mercuryMat;
  let venusMat;
  let earthMat;
  let moonMat;
  let marsMat;
  let jupiterMat;
  let saturnMat;
```

```
  let uranusMat;
  let neptuneMat;


  identity();
    // translate to center of solar system
    translate([width / 2, height / 2, 0]);
        sunMat = current();
      rotate(zAxis, time / 10) // push(mercury orbit rotation)
                              translate([height / 15, 0, 0]);
//push(mercury translation)
           rotate(zAxis, time * 8); // push(mercury rotation)
                mercuryMat = current();
           pop(); // mercury rotation
           pop(); // mercury translation
     pop(); // mercury rotation
      rotate(zAxis, time / 15) // push(venus orbit rotation)
                              translate([height / 10, 0, 0]);
//push(venus translation)
           rotate(zAxis, time * 8); // push(venus rotation)
                venusMat = current();
           pop(); // venus rotation
           pop(); // venus translation
     pop(); // venus rotation
      rotate(zAxis, time / 2) // push(earth orbit rotation)
        translate([height / 8, 0, 0]); //push(earth translation)
          rotate(zAxis, time * 8); // push(earth rotation)
            earthMat = current();  //
          pop(); // earth rotation
          rotate(zAxis, time * 2); //push(moon orbit rotation)
            translate([height / 15, 0, 0]); // push(moon
translation)
              rotate(zAxis, time * 16); //push(moon rotation)
                moonMat = current();
              pop();  // moon rotation
            pop();  // moon translation
          pop();  // moon orbit rotation
        pop();  // earth translation
      pop();  // earth orbit rotation
           rotate(zAxis, time / 8) // push(mars orbit rotation)
                              translate([height / 5, 0, 0]);
//push(mars translation)
           rotate(zAxis, time * 8); // push(mars rotation)
                marsMat = current();
           pop(); // mars rotation
           pop(); // mars translation
     pop(); // mars rotation
      rotate(zAxis, time / 5) // push(jupiter orbit rotation)
```

```
                        translate([height / 4, 0, 0]);
//push(jupiter translation)
          rotate(zAxis, time * 8); // push(jupiter rotation)
               jupiterMat = current();
          pop(); // jupiter rotation
          pop(); // jupiter translation
     pop(); // jupiter rotation
      rotate(zAxis, time / 6) // push(saturn orbit rotation)
                        translate([height / 3, 0, 0]);
//push(saturn translation)
          rotate(zAxis, time * 8); // push(saturn rotation)
               saturnMat = current();
          pop(); // saturn rotation
          pop(); // saturn translation
     pop(); // saturn rotation
      rotate(zAxis, time / 10) // push(uranus orbit rotation)
                        translate([height / 2, 0, 0]);
//push(uranus translation)
          rotate(zAxis, time * 8); // push(uranus rotation)
               uranusMat = current();
          pop(); // uranus rotation
          pop(); // uranus translation
     pop(); // uranus rotation
      rotate(zAxis, time / 3) // push(neptune orbit rotation)
                        translate([height / 1.9, 0, 0]);
//push(neptune translation)
          rotate(zAxis, time * 8); // push(neptune rotation)
               neptuneMat = current();
          pop(); // neptune rotation
          pop(); // neptune translation
     pop(); // neptune rotation
    pop();  // center of solar system
  pop();  // identity

  drawSquare(sunMat, 40, '#ba2800');
  drawSquare(mercuryMat, 10, 'orange');
  drawSquare(venusMat, 10, '#ba9500');
  drawSquare(earthMat, 20, 'orange');
  drawSquare(moonMat, 5, 'blue');
  drawSquare(marsMat, 20, 'gray');
  drawSquare(jupiterMat, 20, '#b56c28');
  drawSquare(saturnMat, 20, '#b56c28');
  drawSquare(uranusMat, 20, '#00bab7');
  drawSquare(neptuneMat, 20, '#0009ba');


  requestAnimationFrame(render);
}
```

```
requestAnimationFrame(render);

function drawSquare(mat, size, color) {
  const {width, height} = ctx.canvas;
  //ctx.setTransform(mat[0], mat[1], mat[4], mat[5], mat[12],
mat[13]);
  ctx.strokeStyle = color;
  ctx.fillStyle = color;
  ctx.fill();
      ctx.beginPath();
      ctx.arc(mat[12],  mat[13], size * 0.3, 0, 2 * Math.PI);
      ctx.stroke();
}

</script>
</body>
</html>
```