**WebGL Fundamentals**

Video: <https://youtu.be/YUlNR9bMEyw>

**Modul I: How It Works**

Source: <https://webglfundamentals.org/webgl/lessons/webgl-how-it-works.html>

Pada modul ini, terdapat dua eksperimen yang dilakukan, yaitu menggambar bentuk segitiga yang nilai atributnya dapat diubah dengan *slider*, dan membuat bentuk dua segitiga.

1. Eksperimen I: Single Triangle

Pertama-tama, pada file index.html, akan diinisialisasi vertex shader dan fragment shader. Inisialisasi vertex shader ini dilakukan dengan mengisi variable gl\_Position dan v\_color yang nilainya bergantung relatif terhadap position clip space. Kemudian inisialisasi fragment shader diisi dengan pengisian gl\_FragColor dengan nilai v\_color yang berisi nilai hasil konversi clip space sebelumnya agar valid dengan nilai di color space.

Untuk membuat bentuk triangle, akan dilakukan penggambaran 3 titik untuk menjadi bentuk segitiga. Pembuatan triangle ini dilakukan dengan mengisi buffer (sekumpulan data yang diupload ke GPU) dengan nilai-nilai yang menggambarkan segitiga. Kemudian, menggambar titik dengan fungsi drawArrays(gl.TRIANGLES, 0, 3). Selanjutnya, vertex shader akan mem-passing data yaitu v\_color ke fragment shader.

Hasilnya, akan tampak sebuah segitiga dengan warna yang berbeda-beda relatif terhadap koordinat di clip space. Berikut ini adalah hasil dari eksperimen I.

Shape

Description automatically generated

1. Eksperimen II: Double Triangle

Pada eksperimen I, vertex shader mem-passing data color untuk fragment shader untuk satu segitiga. Pada eksperimen ini, vertex shader menggunakan satu atribut lagi yaitu a\_color yang nilainya akan diisi dengan data color untuk dua segitiga. Untuk melakukan ini, perlu dibuat pula buffer untuk color yang kemudian di bind dengan nilai r g b untuk kedua segitiga. Pada eksperimen ini, nilai r g b untuk kedua segitiga di-set random untuk segitiga 1 dan segitiga 2.

Hasil dari eksperimen ini adalah sebagai berikut. Dihasilkan dua segitiga dengan warna yang berbeda berdasarkan nilai random yang di-assign untuk masing-masing r g b.

Icon

Description automatically generated with low confidence

**Modul II: Image Processing**

Source: <https://webglfundamentals.org/webgl/lessons/webgl-image-processing.html>

Berbeda dengan pemrosesan titik dan garis, selanjutnya untuk melakukan pemrosesan gambar dalam WebGL, dilakukan dengan melakukan texture. Jika sebelumnya menggunakan koordinat clip space, untuk gambar yang digunakna adalah koordinat dari texture.

Pada eksperimen di modul II ini, vertex shader akan mem-passing nilai koordinat texture (textCoord) ke fragment shrader yang kemudian di fragment shader, akan dilakukan pewarnaan pixel-pixel pada canvas ke dalam image yang di-assign berdasarkan koordinat texture yang di-passing oleh vertex shader.

Proses render image dilakukan dengagn pengisian buffer untuk texture coordinate yang prosedurnya mirip dengan yang dilakukan pada modul I. Buffer diisi dengan bufferData berupa texture coordinates, dilanjutkan dnegan membuat texture, dan terakhir upload image ke texture.

A picture containing text, tree, plant, maple

Description automatically generated

Pada WebGL, kita juga dapat memanipulasi gambar tersebut seperti menukar susunan warna rgb dengan melakukan modifikasi pada fragment shader di variable gl\_FragColor = texture2D(u\_image, v\_textCoord).brga akan menghasilkan gambar seperti di bawah ini.

Graphical user interface

Description automatically generated