Dokumentasi Grafika Komputer
Kelompok BukaGL

Anggota Kelompok:

- Fersandi Pratama (1506724505) /50%
- Ariel Miki Abraham (1606836383) /50%

Link Github: https://github.com/fersandiP/grafkom-uas

1. Cara Menggunakan

Web ini menggunakan beberapa asset untuk texture dan wavefront object. Oleh karena itu, diperlukan web server. Untuk mempermudah penggunaan, kami juga menyediakan versi live dari program ini di http://128.199.161.1:8000/ (tidak tersedia HTTPS, jadi gunakan HTTP).

Berikut kontrol-kontrol yang tersedia:

1) **Ubah view kamera : [SPACE]**
   Anda bisa mengubah view dari global view menjadi Point of View dari object yang berupa wanita berwana merah.
2) **Mengubah cahaya**
   Cahaya default adalah directional lighting dan point lighting. Namun anda juga bisa merubah tipe cahaya menjadi spot lighting dengan menggunakan checkbox yang tersedia.
3) **Menggerakan objek wanita berwarna merah : [W/A/S/D]**
   Anda bisa menggerakan wanita ke arah depan, belakang, kiri, dan kanan. Ini juga sekaligus untuk mengontrol sumber cahaya saat mode PoV dan Spot Lighting. (Sumber cahaya spot light adalah kamera).
4) **Menggerakan spot light direction: [←/→/ ↑ / ↓]**
   Anda hanya bisa menggunakan ini pada mode Point of View objek dan Spot Lighting. Anda bisa mengubah lighting direction.
5) **Menggerakan spot light source : [W/A/S/D]**
   Anda hanya bisa menggunakan ini pada mode Point of View dan Spot Lighting. Anda bisa menggerakan source dari spot light ke kiri, kanan, depan, dan belakang.
6) **Menggerakan kamera target: [←/→/ ↑ / ↓]**
   Anda hanya bisa menggunakan ini pada mode Point of View objek. Anda bisa mengubah view target dari kamera.
7) **Menggerakan objek monyet berwarna coklat : [</>]**
   Monyet berwarna coklat bisa digerakan ke kiri dan ke kanan.
8) **Menggerakan objek robot berwarna hijau : [Z/X]**

Robot berwarna hijau bisa digerakan ke kiri dan ke kanan.

9) **Toggle Wireframe : [Q]**
Anda bisa mengubah mode drawing dari shading ke wireframe.

2. Penjelasan Kode

a. Dependensi

Tugas ini menggunakan dua buah library untuk membantu mempermudah pembuatan kode. Kedua buah library tersebut adalah TWGL dan webgl-obj-loader,

i. TWGL

TWGL adalah *helper library* untuk WebGL. Pada tugas ini TWGL digunakan untuk melakukan *wrapper* pada *API call WebGL,* sehingga kode yang dibuat tidak terlalu *verbose* dan berantakan. Penggunaan TWGL digunakan pada saat inisialisasi WebGL, pembuatan shader, binding variabel ke GLSL, dan pembuatan objek-objek dasar.

ii. Webgl-obj-loader

Webgl-obj-loader adalah library untuk melakukan *parsing* terhadap file dengan format obj sehingga model dari obj dapat di gambar dengan mudah. Library ini akan melakukan parsing terhadap file obj dan mengembalikan objek yang berisi array yang dibutuhkan seperti texture, normal, vertices, dan indices. Penginputan ke dalam buffer array tetap menggunakan TWGL.

b. Struktur Kode

i. Objek dan Hirarki

Terdapat sebuah file object.js yang mendefinisikan objek-objek yang akan digambar pada canvas. Kode dibuat sedemikian rupa sehingga tiap objek berbentuk JSON. Format json yang digunakan adalah sebagai berikut :

```
{
    Obj1 : {
        JSON_definisi_objek
    },
    Obj2 : {
        JSON_definisi_objek
    }, ...
    hierarchy : {
        JSON_definisi_hirarki
    }
}
```

Dimana tiap Obj mendefinisikan satu buah objek. JSON definisi objek memiliki atribut sebagai berikut :

Nama atribut	**Informasi Atribut**	**Tipe data**	**Keterangan**
translation	Translasi objek terhadap titik origin/ terhadap parent objeknya.	Float vec3 / Function	Required
rotateX	Rotasi terhadap sumbu x pada objek	Float (degree) / Function	Optional
rotateY	Rotasi terhadap sumbu y pada objek	Float (degree) / Function	Optional
rotateZ	Rotasi terhadap sumbu z pada objek	Float (degree) / Function	Optional
scale	Scaling objek.	Float vec3	Required
color	Warna pada objek.	Float vec3	Optional, default hijau([0.2, 0.8, 0.2])
texture	Nama texture yang digunakan sesuai dengan nama yang didefinisikan pada method twgl.createTextures	String	Optional
objName	Nama objek yang akan di render. Jika tidak di definiskan maka default membuat kubus/cube.	String	Optional, Defalut ‘cube’
function	Integer berisi atribut mana saja yang merupakan fungsi. Integer disimpan dengan menggunakan skema bitmask. Sebagai contoh jika atribut	Int	Optional, Nilai konstan : TRANSLATE, ROTATE_X, ROTATE_Y, ROTATE_Z

	translation dan rotateX merupakan fungsi maka nilai atribut ini adalah = TRANSLATE | ROTATE_X		

Function digunakan apabilai atribut tersebut dapat berubah pada tiap proses render. Terdapat objek parameter yang akan menyimpan tiap perubahan nilai atribut-atribut tersebut. Dan di objek parameter ini juga nilai atribut diubah-ubah.

Atribut hirarki didefinisikan sebagai sebuah array yang berisi atribut objek hirarki sebagai berikut:

Nama atribut	**Informasi Atribut**	**Tipe data**	**Keterangan**
name	Nama objek yang akan digambar	String	Required
hasChild	Flag menandakan apakah objek memililki anak (berhirarki)	Boolean	Required
childs	Array berisi objek hirarki (rekursif) yang merupakan anak dari objek sekarang.	Array	Required jika hasChild = true

Dengan menggunakan JSON objek di atas, pembuatan objek menjadi lebih mudah. Cukup dengan mendefinisikan objek JSON-nya saja. Proses render objek dapat dilakukan dengan memanggil fungsi drawObject(nama_objek) di dalam fungsi render.

ii. Tekstur

Tekstur dibuat dengan menggunakan helper library twgl. Fungsi yang digunakan twgl.createTextures(obj_textures) dimana obj_textures berisi array yang mendefinisikan tekstur-tekstur yang di *load*. Sebagai contoh kode di bawah

```
textures = twgl.createTextures(gl, {
      wall: {
         src: 'textures/texture2.jpeg',
      })
```

Akan membuat texture bernama wall dengan gambar texture adalah textures/texture2.jpeg. Nama texture dapat digunakan pada objek JSON yang didefinisakan di atas.

iii. Shader

a. Vertex shader

Di dalam vertex shader dilakukan operasi perkalian matrix transformasi model, view, dan projection. Selain itu dilakukan juga pemetaan posisi lighting ke posisi lighting pada world dan perhitungan jarak vertex ke lighting. Selebihnya dilakukan *passing* beberapa variable ke fragment shader menggunakan varying attribute.

b. Fragment shader

Di dalam fragment shader dilakukan perhitungan untuk semua lighting. Untuk membuat specular dari point light dan spot light digunakan half vektor dari vektor objek ke kamera dengan vektor objek ke lighting. Di dalam fragment shader juga dilakukan penggambaran texture.

iv. Lighting

a. Directional light

Directional lighting diatur menggunakan uniform variable yaitu u_lightDirection,u_worldInverseMatrix dan juga normal vektor. Dalam program ini directional lighting tidak diubah-ubah. WorldInverseMatrix digunakan untuk membuat lighting tetap benar saat objek dirotasi. Directional lighting hanya diubah pada line 16. Nilai default lighting directionnya adalah [0.0, 0.5, -0.5]. Perhitungan directional lighting dilakukan di fragment shader dengan memanfaatkan dot product dengan vektor normal.

b. Point light

Point lighting menggunakan uniform variable untuk lighting position dan vektor normal. Dalam program ini lighting position tidak diubah-ubah yaitu pada [0.0, 0.0, -10.0]. Untuk perhitungan, pertama-tama dilakukan perhitungan jarak objek ke kamera dan jarak objek ke sumber cahaya di vertex shader. Selanjutnya nilai-nilai itu di*passing* ke fragment shader. Di dalam fragment shader dilakukan perhitungan intensitas cahaya dengan dot product. Dilakukan juga perhitungan specular dengan memanfaatkan half vektor.

c. Spot light

Perhitungan spot lighting kurang lebihnya sama dengan point light. Perbedaan utamanya adalah spot light menggunakan light direction, yaitu arah dari sumber cahaya dan menggunakan limit untuk threshold dari

jangkauan cahaya. Perhitungan juga dilakukan fragment shader. Light position dari spotlight merupakan posisi kamera dan posisi light direction diambil dari matrix lookAt dengan target yang sama dengan target kamera.

v. Kamera

Dalam pemetaan view (matrix view) digunakan method **lookAt.** Sedangkan untuk pemetaan projection (matrix projection) digunakan method **perspective.** Penggunaan perspective agar terlihat perbedaan jarak objek. Target dari kamera adalah sumbu **z** positive. Semua method perhitungan kamera dilakukan pada method **setCameraAndSpotLight().**

a. World view

World view dilakukan dengan meletakan kamera pada coordinate [0,0,-20].

b. Point of view

Point of view dilakukan degan meletekan kamera persis pada posisi objek wanita berwarna merah. kamera akan bergerak bersamaan dengan bergeraknya objek wanita.