Rayhan Septian Wijaya - 24060123140123

Laporan Praktikum Materi 7, 8, 9

Praktikum 7

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <iostream>
#include <cstdlib>
#include "imageloader.h"
```

Bagian ini mengimpor pustaka GLEW dan GLFW untuk membuat jendela grafis dan mengelola rendering OpenGL. imageloader.h adalah header untuk memuat file BMP.

```
const float BOX_SIZE = 7.0f;
float angle = 0.0f;
GLuint textureId;
```

Variabel BOX_SIZE menentukan ukuran objek 3D, angle untuk rotasi, dan textureId menyimpan ID tekstur yang dihasilkan dari gambar.

```
void handleKeypress(GLFWwindow* window, int key, int scancode, int action, int mods) {
    if (key == GLFW_KEY_ESCAPE && action == GLFW_PRESS)
        glfwSetWindowShouldClose(window, GLFW_TRUE);
}
```

Fungsi ini memungkinkan pengguna menutup jendela program dengan menekan tombol Escape.

```
GLuint loadTexture(Image* image) {
    ...
}
```

Fungsi ini menerima data gambar BMP, lalu mengatur properti tekstur seperti jenis filter (GL_LINEAR) agar tekstur tampil halus. Fungsi ini menghasilkan dan mengembalikan ID OpenGL untuk tekstur tersebut.

```
void initRendering() {
    ...
}
```

Di sini beberapa fitur OpenGL diaktifkan seperti depth test, lighting, dan color material. Gambar 852.bmp di-load sebagai tekstur menggunakan loadBMP() dan loadTexture().

```
void drawScene() {
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
   glTranslatef(0.0f, 0.0f, -20.0f);
   GLfloat ambientLight[] = { 0.3f, 0.3f, 0.3f, 1.0f };
   glLightModelfv(GL_LIGHT_MODEL_AMBIENT, ambientLight);
   GLfloat lightColor[] = { 0.7f, 0.7f, 0.7f, 1.0f };
   GLfloat lightPos[] = { -2 * BOX_SIZE, BOX_SIZE, 4 * BOX_SIZE, 1.0f };
   glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor);
   glLightfv(GL_LIGHT0, GL_POSITION, lightPos);
```

Pertama, kita membersihkan color buffer dan depth buffer, agar layar siap menggambar frame baru tanpa sisa dari frame sebelumnya.
Mode matriks diubah ke MODELVIEW karena kita akan mengatur transformasi objek (bukan proyeksi). glLoadIdentity() me-reset matriks ke awal. glTranslatef() memindahkan posisi kamera 20 unit ke belakang pada sumbu z agar objek bisa terlihat (karena OpenGL awalnya melihat ke sumbu z negatif).
Menambahkan pencahayaan ambient global agar semua objek punya pencahayaan dasar. Mengatur sumber cahaya utama (GL_LIGHT0), posisinya, dan warnanya. Cahaya bersifat diffuse (menyebar), bukan hanya titik terang.

```
   glBegin(GL_QUADS);
   ...
   glEnd();
```

Kode ini menggambar empat sisi kubus (atas, bawah, kiri, kanan) menggunakan warna dan glNormal3f() untuk pencahayaan. Penjelasan tiap bagian: glColor3f() memberi warna ke sisi tersebut. glNormal3f() menyatakan arah permukaan untuk pencahayaan. glVertex3f() menentukan posisi titik-titik pada sisi tersebut.

```
   glEnable(GL_TEXTURE_2D);
   glBindTexture(GL_TEXTURE_2D, textureId);
```

Mengaktifkan mode tekstur 2D. Mengikat tekstur yang telah dimuat sebelumnya (textureId).

```
   glBegin(GL_QUADS);
   ...
   glTexCoord2f(u, v); glVertex3f(...);
   glTexCoord2f(0.0f, 0.0f); glVertex3f(-BOX_SIZE/2, -BOX_SIZE/2, BOX_SIZE/2);
   glTexCoord2f(1.0f, 0.0f); glVertex3f( BOX_SIZE/2, -BOX_SIZE/2, BOX_SIZE/2);
   glTexCoord2f(1.0f, 1.0f); glVertex3f( BOX_SIZE/2,  BOX_SIZE/2, BOX_SIZE/2);
   glTexCoord2f(0.0f, 1.0f); glVertex3f(-BOX_SIZE/2,  BOX_SIZE/2, BOX_SIZE/2);
```

```
    ...
    glEnd();
```

glTexCoord2f(u, v) digunakan untuk menentukan titik pada gambar tekstur. glVertex3f(x, y, z) menyatakan posisi titik pada permukaan objek.
Koordinat (u, v) menentukan bagian dari gambar yang akan ditempel ke permukaan objek.
glDisable(GL_TEXTURE_2D); Menonaktifkan tekstur setelah dua sisi selesai digambar, agar sisi lain (kalau ada) tidak terkena tekstur juga.

```
int main() {
    if (!glfwInit()) {
        std::cerr << "Failed to initialize GLFW\n";
        return EXIT_FAILURE;
    }
```

glfwInit() digunakan untuk menginisialisasi library GLFW. Jika gagal, program akan mencetak error dan keluar.

```
    GLFWwindow* window = glfwCreateWindow(800, 600, "GLFW Texture Box", NULL, NULL);
    if (!window) {
        std::cerr << "Failed to create window\n";
        glfwTerminate();
        return EXIT_FAILURE;
    }
```

Membuat window dengan ukuran 800x600 piksel dan judul "GLFW Texture Box". Jika gagal, program keluar dan glfwTerminate() dipanggil untuk membersihkan resource.

```
    glewExperimental = GL_TRUE;

    if (glewInit() != GLEW_OK) {
        std::cerr << "Failed to initialize GLEW\n";
        return EXIT_FAILURE;
    }
```

Menginisialisasi GLEW untuk mengakses fungsi OpenGL terbaru. Jika gagal, program juga akan keluar.

```
    glEnable(GL_DEPTH_TEST);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45.0, 800.0 / 600.0, 1.0, 200.0);
```

Mengaktifkan depth test, agar objek yang lebih dekat ditampilkan di atas objek yang jauh.
Mengatur proyeksi perspektif (field of view 45°, aspek rasio sesuai ukuran window, dan jarak pandang antara 1.0 sampai 200.0).

```
    while (!glfwWindowShouldClose(window)) {
        angle += 0.2f;
        if (angle > 360) angle -= 360;

        drawScene();
        glfwSwapBuffers(window);
        glfwPollEvents();
    }
```

Loop ini akan terus berjalan selama window belum ditutup. angle ditambah terus-menerus agar kubus terus berputar. drawScene() dipanggil untuk menggambar kubus pada frame saat ini. glfwSwapBuffers() menukar buffer belakang ke depan (double buffering). glfwPollEvents() mengecek dan menjalankan event seperti keyboard atau mouse.

```
glfwDestroyWindow(window);
    glfwTerminate();
    return 0;
}
```

Setelah window ditutup, glfwDestroyWindow() membersihkan jendela. glfwTerminate() menutup GLFW dan membebaskan semua resource. Program berakhir dan mengembalikan 0 yang menandakan berhasil.

Praktikum 8

```
double rx = 0.0, ry = 0.0;
float l[] = { 0.0, 80.0, 0.0 }; // posisi cahaya
float n[] = { 0.0, -40.0, 0.0 }; // normal bidang proyeksi (lantai)
float e[] = { 0.0, -60.0, 0.0 }; // titik pandang proyeksi
```

l mewakili posisi sumber cahaya n adalah normal bidang tempat bayangan akan diproyeksikan (lantai) e adalah eye point atau titik pandang untuk transformasi proyeksi

```
GLUquadric* quadric = gluNewQuadric();
gluQuadricDrawStyle(quadric, GLU_FILL);
gluCylinder(quadric, 20.0, 0.0, 50.0, 40, 50);
gluDeleteQuadric(quadric);
```

Fungsi ini menggambar sebuah kerucut menggunakan gluCylinder() sebagai objek utama yang akan diberi bayangan. Ini dilakukan dengan GLUquadric, yaitu objek bantu dari GLU untuk menggambar primitif seperti bola, silinder, dll.

```
GLUquadric* quadric = gluNewQuadric();
gluQuadricDrawStyle(quadric, GLU_FILL);
gluCylinder(quadric, 20.0, 0.0, 50.0, 40, 50);
gluDeleteQuadric(quadric);
```

Fungsi ini menggambar sebuah **kerucut** dengan radius bawah 20, radius atas 0, dan tinggi 50. Detail mesh dibuat dengan 40 sektor dan 50 tumpukan.

```
float d, c, mat[16];
// Hitung nilai d dan c
d = n[0]*l[0] + n[1]*l[1] + n[2]*l[2];
c = e[0]*n[0] + e[1]*n[1] + e[2]*n[2] - d;
// Bangun matriks transformasi proyeksi bayangan
// ...
glMultMatrixf(mat);
```

Matriks ini dikalikan dengan matriks model-view aktif menggunakan glMultMatrixf(), sehingga bayangan objek dapat ditampilkan dengan proyeksi ke bidang.

Fungsi render() Fungsi ini menampilkan keseluruhan scene: Membersihkan buffer Menampilkan titik cahaya berwarna kuning dengan GL_POINTS Menggambar bidang lantai sebagai bidang datar berwarna abu-abu Menggambar objek asli (kerucut) berwarna biru Menggambar bayangan dari kerucut menggunakan matriks proyeksi Bayangan dibuat dengan cara:

```
glPushMatrix();
glShadowProjection(l, e, n); // gunakan proyeksi
glRotatef(...);              // rotasi objek
glDisable(GL_LIGHTING);      // bayangan tidak terpengaruh cahaya
glColor3f(0.4, 0.4, 0.4);    // warna abu-abu gelap
draw();                      // gambar kerucut lagi
glPopMatrix();
```

```
rx += 0.1;
ry += 0.1;
render();
glfwSwapBuffers(window);
```

Fungsi ini digunakan untuk terus-menerus **memutar objek** dan **merender ulang** scene agar terlihat animasi rotasi.

```
glViewport(0, 0, width, height);
gluPerspective(60.0f, (float)width/(float)height, 1.0f, 400.0f);
```

Menyesuaikan viewport dan proyeksi perspektif saat ukuran jendela berubah. cpp Copy Edit

- Fungsi main()
Fungsi utama program: Inisialisasi GLFW dan GLEW Membuat jendela OpenGL Mengatur callback untuk input (keypress) dan resize Mengaktifkan pencahayaan, normalisasi, dan depth test Menetapkan properti pencahayaan seperti glLightfv(GL_LIGHT0, ...) Menetapkan proyeksi dan posisi kamera Memanggil help() untuk menampilkan kontrol Menjalankan loop utama dengan pemanggilan idle()

Praktikum 9

```
float angle = 0.0f, deltaAngle = 0.0f, ratio;
float x = -5.0f, y = 12.0f, z = 40.0f;
float lx = 0.0f, ly = 0.0f, lz = -1.0f;
int deltaMove = 0, w, h;
static int rotAngleX = 0, rotAngleY = 0, rotAngleZ = 0;
float posXKaki = 10, posXBola = -10, posYKaki = 6, posYBola = -5;
float rotKaki = 0.0f;
int kick = 0, roll = 0, touch = 0;
float jarak = 1;
```
Variabel-variabel ini mengatur: posisi dan arah pandang kamera (x, y, z, lx, dll.) rotasi kaki dan scene (rotAngleX, dst.) posisi kaki dan bola (posXKaki, posXBola) status animasi seperti kick (menendang), roll (bola bergulir), touch (kontak) serta jarak untuk menentukan seberapa jauh bola bisa bergulir.

```
glEnable(GL_DEPTH_TEST);
glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
IDquadric = gluNewQuadric();
gluQuadricNormals(IDquadric, GLU_SMOOTH);
gluQuadricTexture(IDquadric, GL_TRUE);
```
Fungsi ini menginisialisasi pengujian kedalaman, mode polygon, dan membuat quadric object yang digunakan untuk menggambar bola dengan fungsi gluSphere().

```
glViewport(0, 0, w, h);
gluPerspective(45, ratio, 0.1, 1000);
gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f, 0.0f);
```
Fungsi ini menyesuaikan viewport dan kamera ketika ukuran jendela diubah.

- Fungsi Kamera: orientMe() dan moveMeFlat() orientMe(float ang): mengubah arah pandang kamera berdasarkan sudut. moveMeFlat(int i): memindahkan kamera maju/mundur berdasarkan arah pandang.

- Fungsi keyCallback() Menangani berbagai input keyboard: Rotasi: WASD, Q, E Gerak kaki/bola: O, P Tendang: K Reset: SPACE Keluar: ESC

```
glEnable(GL_LIGHT0);
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
// dll.
```

Mengatur pencahayaan dan material agar objek terlihat 3D

```
for (i = Z_MIN; i < Z_MAX; i += gap) {

glVertex3f(i, 0, Z_MIN);

   glVertex3f(i, 0, Z_MAX);

}
```

Menggambar grid pada lantai untuk orientasi spasial

- Fungsi Balok() Membuat balok 3D berdasarkan panjang, lebar, dan tinggi. Fungsi ini digunakan untuk menggambar kaki penendang.

- Fungsi pergerakanKaki() Mengatur tiga fase animasi menendang: Ayunan ke belakang Dorongan ke depan Kembali ke posisi awal Juga mengecek apakah bola tersentuh dan mulai menggelinding jika kontak terjadi.

```
if (roll == 1) {

   if (jarak > 0) {

      posXBola -= 0.03f;

      jarak -= 0.01f;

   }

}
```

Mengatur bola untuk bergerak ke kiri saat disentuh kaki, lalu berhenti ketika jarak <= 0

- Fungsi Object() Menggambar objek utama: Kaki putih sebagai penendang (Balok) Bola oranye dengan gluSphere Panggil pergerakanKaki() dan pergerakanBola() di sini
- Fungsi display() Menampilkan scene: Membersihkan layar Memproses input kamera Menerapkan rotasi scene Menampilkan grid dan objek (kaki + bola) Swap buffer untuk efek animasi
- Fungsi main() Inisialisasi GLFW & GLEW Buat window Daftarkan callback (input, resize) Aktifkan pencahayaan Jalankan loop utama while (!glfwWindowShouldClose(window)) yang terus memanggil display() dan merespon input