Praktikum 7
Tekstur dan Gambar

Penjelasan code yang ada pada modul:

Code ini menggunakan OpenGL dan GLFW untuk menampilkan kotak berputar dengan tekstur pada dua sisinya.

- **Pengaturan Awal dan Konstanta**

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

Bagian ini mendeklarasikan konstanta ukuran kotak, variabel sudut rotasi, dan ID tekstur yang akan digunakan.

- **Fungsi Penanganan Input**

```
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 menangani input keyboard. Jika tombol Escape ditekan, aplikasi akan ditutup.

- **Fungsi Untuk Tekstur**

```
GLuint loadTexture(Image* image) {
    GLuint textureId;
    glGenTextures(1, &textureId);
    glBindTexture(GL_TEXTURE_2D, textureId);
         glTexImage2D(GL_TEXTURE_2D,  0,  GL_RGB,  image->width,
image->height,
    0, GL_RGB, GL_UNSIGNED_BYTE, image->pixels);
          glTexParameteri(GL_TEXTURE_2D,  GL_TEXTURE_MIN_FILTER,
GL_LINEAR);
          glTexParameteri(GL_TEXTURE_2D,  GL_TEXTURE_MAG_FILTER,
GL_LINEAR);

    return textureId;
}
```

Fungsi ini membuat tekstur OpenGL dari objek gambar. Fungsi ini membuat ID tekstur, mengikatnya ke target GL_TEXTURE_2D, mengunggah data piksel, dan mengatur parameter filter tekstur.

- **Fungsi Inisialisasi Rendering**

```
void initRendering() {
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_NORMALIZE);
    glEnable(GL_COLOR_MATERIAL);
    Image* image = loadBMP("852.bmp");
    textureId = loadTexture(image);
    delete image;
}
```

Fungsi ini mengaktifkan fitur OpenGL yang diperlukan (depth testing, pencahayaan, normalisasi, material warna) dan memuat tekstur dari file BMP.

- **Fungsi Menggambar Scene**

```
void drawScene() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glTranslatef(0.0f, 0.0f, -20.0f);
    // ... pengaturan cahaya ...
    glRotatef(-angle, 1.0f, 1.0f, 0.0f);

    // ... kode untuk menggambar kotak ...
}
```

Fungsi ini membersihkan buffer, mengatur matriks view, posisi objek, pencahayaan, dan memutar scene. Kemudian menggambar kotak dengan warna berbeda untuk setiap sisi dan menerapkan tekstur pada sisi depan dan belakang.

- **Pengaturan Pencahayaan**

```
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);
```

Bagian ini mengatur cahaya ambient dan sumber cahaya diffuse.

- **Menggambar Sisi Kotak dengan Warna**

```
glBegin(GL_QUADS);
    // Sisi atas (kuning)
    glColor3f(1.0f, 1.0f, 0.0f);
    glNormal3f(0.0, 1.0f, 0.0f);
    glVertex3f(-BOX_SIZE / 2, BOX_SIZE / 2, -BOX_SIZE / 2);
    // ... vertex lainnya...

    // Sisi bawah (magenta)
    glColor3f(1.0f, 0.0f, 1.0f);
    glNormal3f(0.0, -1.0f, 0.0f);
    // ... vertex...

    // Sisi lainnya...
glEnd();
```

Bagian ini menggambar empat sisi kotak dengan warna berbeda (kuning, magenta, cyan, merah-hijau).

- **Menggambar Sisi dengan Tekstur**

```
glEnable(GL_TEXTURE_2D);
glBindTexture(GL_TEXTURE_2D, textureId);
glColor3f(1.0f, 1.0f, 1.0f);
glBegin(GL_QUADS);
    // Sisi depan (tekstur)
    glNormal3f(0.0, 0.0f, 1.0f);
         glTexCoord2f(0.0f,  0.0f);  glVertex3f(-BOX_SIZE  /  2,
-BOX_SIZE / 2, BOX_SIZE / 2);
    // ... vertex dan koordinat tekstur lainnya...

    // Sisi belakang (tekstur)
    glNormal3f(0.0, 0.0f, -1.0f);
    // ... vertex dan koordinat tekstur...
glEnd();
glDisable(GL_TEXTURE_2D);
```

Bagian ini mengaktifkan tekstur, mengikat tekstur yang telah dimuat, dan menggambar dua sisi kotak (depan dan belakang) dengan tekstur.

- **Fungsi Main**

Fungsi main menginisialisasi GLFW, membuat jendela, mengatur proyeksi, memanggil initRendering(), dan menjalankan loop utama. Dalam loop, sudut rotasi diperbarui, scene digambar, dan buffer ditukar. Ketika loop berakhir, sumber daya dibersihkan.

Code ini menghasilkan kotak 3D berputar dengan sisi berwarna dan dua sisi bertekstur menggunakan gambar "852.bmp". Kontrol satu-satunya adalah tombol Escape untuk keluar.

Praktikum 8
Bayangan

Penjelasan code yang ada pada modul:

Code ini menampilkan kerucut 3D beserta bayangannya pada bidang datar dengan menggunakan GLFW dan GLEW

- **Variabel Global**

```
double rx = 0.0;

double ry = 0.0;

float l[] = { 0.0, 80.0, 0.0 };

float n[] = { 0.0, -40.0, 0.0 };

float e[] = { 0.0, -60.0, 0.0 };
```

Variabel ini menyimpan sudut rotasi objek (rx, ry), posisi sumber cahaya (l), vektor normal dari bidang bayangan (n), dan titik pada bidang bayangan (e).

- **Fungsi Help**

```
void help() {
    printf("Shadow projection example using a cone object\n");
    printf("Controls:\n");
    printf("W/S - Move light up/down\n");
    printf("A/D - Move light left/right\n");
```

```
    printf("Q/E - Move light forward/backward\n");
    printf("ESC - Exit\n");
}
```

Fungsi ini menampilkan petunjuk penggunaan termasuk untuk memindahkan sumber cahaya dan keluar dari aplikasi.

- **Fungsi Draw**

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

Fungsi ini menggambar objek kerucut menggunakan objek quadric OpenGL. Kerucut dibuat dengan radius bawah 20.0, radius atas 0.0, tinggi 50.0, dan detail yang cukup (40 sektor dan 50 tumpukan).

- **Fungsi Shadow Projection**

```
void glShadowProjection(float *l, float *e, float *n) {
    float d, c;
    float mat[16];
    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;
    // Pembentukan matriks proyeksi bayangan...
    glMultMatrixf(mat);
}
```

Fungsi ini menciptakan matriks proyeksi bayangan. Berdasarkan posisi sumber cahaya (l), titik pada bidang bayangan (e), dan vektor normal bidang (n), fungsi menghitung matriks transformasi yang akan memproyeksikan objek ke bidang bayangan.

- **Fungsi Render**

```
void render() {

    glClearColor(0.0, 0.6, 0.9, 0.0);
```

```
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLightfv(GL_LIGHT0, GL_POSITION, l);

    // Menggambar titik cahaya
    glDisable(GL_LIGHTING);
    glColor3f(1.0, 1.0, 0.0);
    glPointSize(5.0f);
    glBegin(GL_POINTS);
    glVertex3f(l[0], l[1], l[2]);
    glEnd();

    // Menggambar bidang datar (lantai)
    glColor3f(0.8, 0.8, 0.8);
    glBegin(GL_QUADS);
    glNormal3f(0.0, 1.0, 0.0);
    glVertex3f(-1300.0, e[1]-0.1, 1300.0);
    // ... vertex lainnya...
    glEnd();

    // Menggambar objek kerucut asli
    glPushMatrix();
    glRotatef(ry, 0, 1, 0);
    glRotatef(rx, 1, 0, 0);
```

```
    glEnable(GL_LIGHTING);
    glColor3f(0.0, 0.0, 0.8);
    draw();
    glPopMatrix();

    // Menggambar bayangan objek
    glPushMatrix();
    glShadowProjection(l, e, n);
    glRotatef(ry, 0, 1, 0);
    glRotatef(rx, 1, 0, 0);
    glDisable(GL_LIGHTING);
    glColor3f(0.4, 0.4, 0.4);
    draw();
    glPopMatrix();
}
```

Fungsi ini menggambar keseluruhan scene termasuk membersihkan layar, menetapkan posisi cahaya, menggambar sumber cahaya sebagai titik kuning, menggambar bidang datar (lantai) sebagai permukaan abu-abu, menggambar kerucut biru dengan pencahayaan dan menggambar bayangan kerucut menggunakan matriks proyeksi bayangan

- **Fungsi Keypress**

```
void keypress(GLFWwindow* window, int key, int scancode, int action, int mods) {
    if (action == GLFW_PRESS || action == GLFW_REPEAT) {
        switch (key) {
            case GLFW_KEY_ESCAPE:
```

```
                glfwSetWindowShouldClose(window, GLFW_TRUE);
                break;
            case GLFW_KEY_S:
                l[1] -= 5.0;
                break;
            // ... kasus lainnya...
        }
    }
}
```

Fungsi ini menangani input keyboard. Tombol W/S menggerakkan cahaya naik/turun, A/D menggerakkan cahaya kiri/kanan, Q/E menggerakkan cahaya maju/mundur, H menampilkan bantuan, dan ESC keluar dari aplikasi.

- **Fungsi Idle**

```
void idle(GLFWwindow* window) {
    rx += 0.1;
    ry += 0.1;
    render();
    glfwSwapBuffers(window);
}
```

Fungsi ini dipanggil secara berulang dalam loop utama. Fungsi ini memperbarui sudut rotasi objek, merender scene, dan menukar buffer untuk menampilkan hasilnya.

- **Fungsi Resize**

```
void resize(GLFWwindow* window, int width, int height) {
    glViewport(0, 0, width, height);
    glMatrixMode(GL_PROJECTION);
```

```
    glLoadIdentity();

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

    glMatrixMode(GL_MODELVIEW);

}
```

Fungsi ini dipanggil ketika ukuran jendela berubah. Fungsi ini menyesuaikan viewport dan matriks proyeksi sesuai dengan rasio aspek jendela baru.

- **Fungsi Main**

Fungsi main menginisialisasi GLFW, membuat jendela, mengatur callback untuk input dan perubahan ukuran, menginisialisasi GLEW, mengatur pencahayaan, menetapkan proyeksi kamera, dan menjalankan loop utama aplikasi. Code menampilkan kerucut biru yang berputar beserta bayangannya pada lantai abu-abu, dengan sumber cahaya kuning yang dapat dikendalikan oleh pengguna.

Praktikum 9
Interaksi Antar Objek

Penjelasan code menendang bola / mempengaruhi objek:

Code ini membuat aplikasi 3D yang mensimulasikan tendangan bola menggunakan OpenGL dan GLFW. Program menampilkan sebuah kaki yang dapat bergerak dan menendang bola pada bidang grid.

- **Variabel Global dan Konstanta**

```
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 ini mengontrol tampilan kamera, rotasi objek, dan posisi kaki dan bola. kick menandai tahap animasi tendangan, roll mengontrol pergerakan bola, dan touch mendeteksi kontak kaki-bola.

- **Fungsi Init**

```
void init() {

    glEnable(GL_DEPTH_TEST);

    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);

    IDquadric = gluNewQuadric();

    gluQuadricNormals(IDquadric, GLU_SMOOTH);

    gluQuadricTexture(IDquadric, GL_TRUE);

}
```

Fungsi ini menginisialisasi OpenGL dan membuat objek quadric yang digunakan untuk menggambar bola.

- **Fungsi Reshape**

```
void reshape(GLFWwindow* window, int w1, int h1) {

    if (h1 == 0) h1 = 1;

    w = w1;

    h = h1;

    ratio = 1.0f * w / h;

    glMatrixMode(GL_PROJECTION);

    glLoadIdentity();

    glViewport(0, 0, w, h);
```

```
    gluPerspective(45, ratio, 0.1, 1000);

    glMatrixMode(GL_MODELVIEW);

    glLoadIdentity();

      gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);

}
```

Fungsi ini dipanggil saat jendela diubah ukurannya. Fungsi ini mengatur proyeksi perspective dan viewport berdasarkan rasio baru.

- **Fungsi Orientasi dan Pergerakan Kamera**

```
void orientMe(float ang) {

    lx = sin(ang/10);

    lz = -cos(ang/10);

    glLoadIdentity();

      gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);

}

void moveMeFlat(int i) {

    x = x + i*(lx)*0.1f;

    z = z + i*(lz)*0.1f;

    glLoadIdentity();

      gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);

}
```

Fungsi orientMe mengubah orientasi kamera berdasarkan sudut, sedangkan moveMeFlat memindahkan kamera maju atau mundur sesuai arah pandangan.

- **Fungsi Callback Keyboard**

```
void keyCallback(GLFWwindow* window, int key, int scancode, int
action, int mods) {

    if (action == GLFW_PRESS || action == GLFW_REPEAT) {

        switch (key) {

              case GLFW_KEY_W: rotAngleX += 2; break;  // Rotasi
X+

              case GLFW_KEY_S: rotAngleX -= 2; break;  // Rotasi
X-

              case GLFW_KEY_A: rotAngleY += 2; break;  // Rotasi
Y+

              case GLFW_KEY_D: rotAngleY -= 2; break;  // Rotasi
Y-

              case GLFW_KEY_Q: rotAngleZ += 2; break;  // Rotasi
Z+

              case GLFW_KEY_E: rotAngleZ -= 2; break;  // Rotasi
Z-

            case GLFW_KEY_O: // Gerakkan kaki dan bola ke kiri

                posXKaki -= 1;

                if (posXBola < -2.9f) posXBola += 1;

                break;

            case GLFW_KEY_P: // Gerakkan kaki dan bola ke kanan

                posXKaki += 1;

                posXBola -= 1;

                break;

             case GLFW_KEY_K: kick = 1; break;  // Mulai animasi
tendangan
```

```
            case GLFW_KEY_SPACE: // Reset posisi dan rotasi
                rotAngleX = rotAngleY = rotAngleZ = 0;
                   posXKaki = 10; posXBola = -10; posYKaki = 6;
posYBola = -5;
                rotKaki = kick = roll = 0;
                break;
                                        case   GLFW_KEY_ESCAPE:
glfwSetWindowShouldClose(window, GLFW_TRUE); break;
        }
    }
}
```

Fungsi ini menangani input keyboard untuk merotasi scene (W, A, S, D, Q, E), menggerakkan kaki dan bola (O, P), memulai tendangan (K), mereset scene (Space), dan keluar (Escape).

- **Fungsi Lighting**

```
void lighting() {
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LESS);
    glEnable(GL_LIGHT0);
    glEnable(GL_NORMALIZE);
    glEnable(GL_COLOR_MATERIAL);
    glEnable(GL_LIGHTING);
    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
    glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

```
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
}
```

Fungsi ini mengatur pencahayaan 3D, termasuk properti material dan sumber cahaya.

- **Fungsi Grid**

```
void Grid() {
    double i;
    const float Z_MIN = -50, Z_MAX = 50;
    const float X_MIN = -50, X_MAX = 50;
    const float gap = 2;
    glColor3f(0.5f, 0.5f, 0.5f);
    glBegin(GL_LINES);
    for (i = Z_MIN; i < Z_MAX; i += gap) {
        glVertex3f(i, 0, Z_MIN);
        glVertex3f(i, 0, Z_MAX);
    }
    for (i = X_MIN; i < X_MAX; i += gap) {
        glVertex3f(X_MIN, 0, i);
        glVertex3f(X_MAX, 0, i);
    }
    glEnd();
```

```
}
```

Fungsi ini menggambar grid pada bidang XZ untuk membantu visualisasi ruang 3D.

- **Fungsi Balok**

```
void Balok(float panjang, float lebar, float tinggi) {
    glPushMatrix();
    float p = panjang/2;
    float l = lebar/2;
    float t = tinggi/2;
     // Front face, Back face, Top face, Bottom face, Right face,
Left face
    // ...
    glPopMatrix();
}
```

Fungsi ini menggambar objek balok 3D dengan dimensi yang diberikan. Balok dibuat dari enam sisi yang mewakili kaki penendang.

- **Fungsi Pergerakan Kaki**

```
void pergerakanKaki() {
    if (kick == 1) {  // Fase ayunan ke belakang
        if (rotKaki <= 45) rotKaki += 0.03f;
        if (rotKaki > 44.9f) kick = 2;
    }
    if (posXBola > -2.9f) touch = 1;
    else if (posXBola < -12) touch = 0;
    if (kick == 2) {  // Fase menendang
```

```
        if (rotKaki >= -90) {
            rotKaki -= 0.2f;
            if (rotKaki < 1 && touch == 1) roll = 1;
        }
        if (rotKaki < -90) kick = 3;
    }
    if (kick == 3) {  // Fase kembali ke posisi awal
        if (rotKaki <= 0) rotKaki += 0.05f;
        if (rotKaki > -1) kick = 0;
    }
}
```

Fungsi ini mengatur animasi tendangan kaki yaitu ayunan ke belakang, dorongan ke depan, kembali ke posisi awal dan juga mendeteksi kontak kaki dengan bola.

- **Fungsi Pergerakan Bola**

```
void pergerakanBola() {
    if (roll == 1) {
        if (jarak > 0) {
            posXBola -= 0.03f;
            jarak -= 0.01f;
        }
        if (jarak < 0) {
            roll = 0;
            jarak = 1;
        }
```

```
    }
}
```

Fungsi ini mengatur pergerakan bola setelah ditendang, dengan variabel jarak menentukan seberapa jauh bola bergerak.

- **Fungsi Object**

```
void Object() {
    glPushMatrix();
    glTranslatef(posXKaki, posYKaki, 0);

    glPushMatrix();
    pergerakanKaki();
    glRotatef(rotKaki, 0, 0, 1);
    glColor3f(1, 1, 1);
    Balok(2, 3, 6);  // Gambar kaki
    glPopMatrix();

    glPushMatrix();
    pergerakanBola();
    glColor3f(0.8f, 0.4f, 0.0f);
    glTranslatef(posXBola, posYBola, 0);
    gluSphere(IDquadric, 1.0, 20, 20);  // Gambar bola
    glPopMatrix();

    glPopMatrix();
```

```
}
```

Fungsi ini menggambar objek utama yaitu kaki penendang (balok putih) dan bola (sphere berwarna oranye), serta mengaktifkan fungsi pergerakan untuk keduanya.

- **Fungsi Display**

```
void display(GLFWwindow* window) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    if (deltaMove) moveMeFlat(deltaMove);
    if (deltaAngle) {
        angle += deltaAngle;
        orientMe(angle);
    }
    glPushMatrix();
    glRotated(rotAngleX, 1, 0, 0);
    glRotated(rotAngleY, 0, 1, 0);
    glRotated(rotAngleZ, 0, 0, 1);
    Grid();
    Object();
    glPopMatrix();
    glfwSwapBuffers(window);
}
```

Fungsi ini merender scene. Fungsi ini memperbarui posisi kamera jika diperlukan, merotasi seluruh scene berdasarkan kontrol pengguna, menggambar grid dan objek, lalu menukar buffer untuk menampilkan hasilnya.

- **Fungsi Main**

Fungsi main menginisialisasi GLFW dan GLEW, membuat jendela, mengatur callback, mengaktifkan pencahayaan, dan menjalankan loop utama aplikasi. Pada loop ada render scene dan memeriksa input pengguna.

Penjelasan code memegang objek:

Code ini adalah program OpenGL menggunakan GLFW dan GLEW untuk membuat simulasi 3D sederhana dari sebuah robot yang dapat bergerak dan mengambil objek kotak.

- **Inisialisasi dan Variabel Global**

```
float angle = 0.0f, deltaAngle = 0.0f, ratio;
float x = 5.0f, y = 10.0f, 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 posXBadan = 10, posXKotak = 0, posYBadan = 7, posYKotak =
6;
float rotTangan1 = 0.0f, rotTangan2 = 0.0f, rotTangan3 = 0.0f,
rotTangan4 = 0.0f;
int kick = 0, roll = 0, hit = 0, gerakTangan = 0, drop = 0,
bring = 0, grab = 0, tabrak = 0;
```

Bagian ini mendefinisikan variabel global untuk posisi kamera (x, y, z), arah pandang (lx, ly, lz), rotasi model (rotAngleX, Y, Z), posisi robot dan kotak (posXBadan, posXKotak, dll), dan variabel status untuk animasi (kick, roll, hit, dll).

- **Fungsi Inisialisasi**

```
void init(void) {
    glEnable(GL_DEPTH_TEST);
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
    IDquadric = gluNewQuadric();
    gluQuadricNormals(IDquadric, GLU_SMOOTH);
    gluQuadricTexture(IDquadric, GL_TRUE);
}
```

Fungsi ini mengaktifkan depth testing untuk menampilkan objek 3D dengan benar, mengatur mode polygon fill, dan membuat objek quadric untuk menggambar bentuk bola atau silinder.

- **Fungsi Reshape**

```
void reshape(GLFWwindow* window, int w1, int h1) {
```

```
    if (h1 == 0) h1 = 1;
    w = w1;
    h = h1;
    ratio = 1.0f * w / h;
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glViewport(0, 0, w, h);
    gluPerspective(45, ratio, 0.1, 1000);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);
}
```

Fungsi ini dipanggil saat ukuran jendela berubah. Fungsi ini mengatur viewport, matriks proyeksi dengan sudut pandang 45 derajat, dan matriks model-view untuk menentukan posisi dan arah kamera.

- **Fungsi Pergerakan Kamera**

```
void orientMe(float ang) {
    lx = sin(ang/10);
    lz = -cos(ang/10);
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);
}

void moveMeFlat(int i) {
    x = x + i*(lx)*0.1f;
    z = z + i*(lz)*0.1f;
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx, y + ly, z + lz, 0.0f, 1.0f,
0.0f);
}
```

Fungsi orientMe mengubah arah pandang kamera berdasarkan sudut, sedangkan moveMeFlat memindahkan kamera ke depan atau belakang sesuai arah pandang.

- **Fungsi Callback Keyboard**

```
void keyCallback(GLFWwindow* window, int key, int scancode, int
action, int mods) {
    if (action == GLFW_PRESS || action == GLFW_REPEAT) {
        switch (key) {
            case GLFW_KEY_W: rotAngleX += 2; break;
            case GLFW_KEY_S: rotAngleX -= 2; break;
            case GLFW_KEY_A: rotAngleY += 2; break;
            // ...lebih banyak kasus
        }
    }
}
```

Fungsi ini menangani input keyboard untuk memutar model (W, A, S, D, Q, E), menggerakkan robot (O, P), dan mengaktifkan animasi pengambilan objek (G untuk gerakan tangan, T untuk menjatuhkan). Tombol SPACE mereset semua ke posisi awal.

- **Fungsi Callback Tombol Khusus**

```
void specialKeyCallback(GLFWwindow* window, int key, int
scancode, int action, int mods) {
    if (action == GLFW_PRESS) {
        switch (key) {
            case GLFW_KEY_UP: deltaMove = 1; break;
            case GLFW_KEY_DOWN: deltaMove = -1; break;
            case GLFW_KEY_LEFT: deltaAngle = -0.01f; break;
            case GLFW_KEY_RIGHT: deltaAngle = 0.01f; break;
        }
    }
    // ...kode untuk GLFW_RELEASE
}
```

Fungsi ini menangani tombol panah untuk mengontrol pergerakan dan rotasi kamera.

- **Fungsi Pencahayaan**

```
void lighting() {
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LESS);
    glEnable(GL_LIGHT0);
    glEnable(GL_NORMALIZE);
    glEnable(GL_COLOR_MATERIAL);
    glEnable(GL_LIGHTING);
    // ...pengaturan properti cahaya dan material
```

```
}
```

Fungsi ini mengaktifkan pencahayaan dan mengatur properti cahaya seperti ambient, diffuse, specular, dan juga properti material.

- **Fungsi Menggambar Grid**

```
void Grid() {
    double i;
    const float Z_MIN = -50, Z_MAX = 50;
    const float X_MIN = -50, X_MAX = 50;
    const float gap = 2;
    glColor3f(0.5f, 0.5f, 0.5f);
    glBegin(GL_LINES);
    // ...kode untuk menggambar grid garis
    glEnd();
}

void Grid2() {
    glColor3f(1.0f, 1.0f, 1.0f);
    glBegin(GL_QUADS);
    // ...kode untuk menggambar dasar putih
    glEnd();
}
```

Grid() menggambar grid garis abu-abu di lantai, sementara Grid2() menggambar permukaan lantai putih.

- **Fungsi Balok**

```
void Balok(float panjang, float lebar, float tinggi) {
    glPushMatrix();
    float p = panjang/2;
    float l = lebar/2;
    float t = tinggi/2;
    // ...kode untuk menggambar 6 sisi balok
    glPopMatrix();
}
```

Fungsi ini menggambar objek balok dengan dimensi tertentu menggunakan GL_QUADS untuk setiap sisi.

- **Fungsi Perubahan Kotak**

```
void perubahKotak() {
    if (drop == 1 && grab == 1) {
        if (posYKotak >= 3) {
            posYKotak -= 0.01f;
        }
        if (posYKotak < 3) {
            bring = 0;
            hit = 0;
            grab = 0;
        }
    }
}
```

Fungsi ini mengatur animasi jatuhnya kotak ketika dilepaskan oleh robot.

- **Fungsi Pengubah Tangan**

```
void pengubahTangan() {
    if (posXBadan != 4) {
        hit = 0;
    } else {
        hit = 1;
    }
    if (hit == 1 && grab == 1) {
        bring = 1;
    }
    // ...kondisi untuk animasi rotasi tangan
}
```

Fungsi ini menangani animasi pergerakan tangan robot, mengecek apakah robot mengenai kotak, dan mengatur status pengambilan/pembawaan objek.

- **Fungsi Objek**

```
void Object() {
    // Menggambar platform hitam
    glPushMatrix();
    glColor3f(0.1f, 0.1f, 0.2f);
    glTranslatef(0, 3, 0);
    Balok(5, 5, 3);
    glPopMatrix();

    // Menggambar kotak merah
```

```
    glPushMatrix();
    perubahKotak();
    glColor3f(0.8f, 0.3f, 0.3f);
    glTranslatef(posXKotak, posYKotak, 0);
    Balok(3, 3, 3);
    glPopMatrix();

    // Menggambar robot (badan biru, tangan hijau)
    glPushMatrix();
    pengubahTangan();
    glColor3f(0.3f, 0.3f, 0.8f);
    glTranslatef(posXBadan, posYBadan, 0);
    Balok(3, 3, 7);
    // ...kode untuk menggambar tangan
    glPopMatrix();
}
```

Fungsi ini merupakan fungsi utama untuk menggambar objek-objek dalam simulasi seperti platform hitam, kotak merah yang dapat diambil, dan robot biru dengan tangan hijau.

- **Fungsi Display**

```
void display(GLFWwindow* window) {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    if (deltaMove) moveMeFlat(deltaMove);
    if (deltaAngle) {
        angle += deltaAngle;
        orientMe(angle);
    }
    glPushMatrix();
    glRotated(rotAngleX + 10, 1, 0, 0);
    glRotated(rotAngleY, 0, 1, 0);
    glRotated(rotAngleZ, 0, 0, 1);
    Grid();
    Grid2();
    Object();
    glPopMatrix();
    glfwSwapBuffers(window);
}
```

Fungsi ini dipanggil setiap frame untuk membersihkan buffer, memperbarui kamera, menerapkan rotasi, menggambar grid dan objek, serta menukar buffer.

- **Fungsi Main**

Fungsi ini membuat jendela, menyiapkan callback, menginisialisasi GLEW dan pencahayaan, kemudian melakukan loop utama untuk rendering dan input. Code ini merupakan simulasi robot yang dapat digerakkan ke arah objek kotak untuk mengambilnya, membawanya, dan menjatuhkannya. Interaksi dikendalikan melalui keyboard, dan pengguna dapat memutar kamera untuk melihat simulasi dari sudut yang berbeda.