# LAPORAN PRATIKUM GRAFIK KOMPUTER

Diajukan untuk memenuhi Tugas mata kuliah Pratikum Grafik Komputer

## IMPLEMENTASI RUANGAN DALAM OPENGL

Dosen Pengampu : Sri Rahayu, M.Kom

Instruktur Pratikum : Arul Budi Kalimat, S.Kom


**Kelompok 2**

| | |
|---|---|
| Rizky Bagja Nugraha | 2306075 |
| Andi Muhamad Ramdani | 2306085 |

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN ILMU KOMPUTER**

**INSTITUT TEKNOLOGI GARUT**


**2025**

# KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT atas segala rahmat dan karunia-Nya sehingga kami dapat menyelesaikan Laporan Praktikum Grafik Komputer ini. Laporan ini dibuat sebagai salah satu tugas dari mata kuliah Grafik Komputer, dengan tujuan untuk memberikan pemahaman yang lebih baik tentang OpenGl.

Kami mengucapkan terima kasih kepada dosen pengampu Sri Rahayu M.Kom, instruktur praktikum Arul Budi Kalimat S.Kom, serta semua pihak yang telah memberikan dukungan dalam penyusunan laporan ini.

Kami menyadari bahwa laporan ini masih memiliki kekurangan, untuk itu kami mengharapkan kritik dan saran yang membangun demi perbaikan di masa yang akan datang.

Garut,15 Januari 2025

Kelompok 2

# DAFTAR ISI

# DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Grafika Komputer adalah teknik-teknik dalam ilmu komputer dan matematika untuk merepresentasikan dan memanipulasi data gambar menggunakan komputer. Dengan bahasa lain, istilah grafika komputer juga dapat diartikan segala sesuatu selain teks atau suara. Seiring dengan perkembangan teknologi, gambar-gambar yang dihasilkan dan ditampilkan pada komputer menjadi bagian kehidupan sehari-hari yang dapat ditemui misalnya pada televisi, koran dan majalah yang fungsinya untuk menampilkan hasil yang lebih komunikatif dan realistis. Selain itu juga grafika komputer ditemukan pada bidang- bidang kedokteran, geologi dan tak terkecuali dalam bidang pendidikan untuk pengajaran dan penulisan karya-karya ilmiah. Salah satu aplikasi yang nyata dari grafika komputer adalah untuk visualisasi data dalam bentuk grafis 2D atau 3D dilengkapi dengan animasi. Walaupun bentuk grafis 3D lebih realistis, namun bentuk 2D masih banyak dipergunakan. Grafika komputer muncul sebagai bagian ilmu komputer yang mempelajari metode-metode sintesa dan manipulasi konten visual secara dijital. Visualisasi informasi dan sains telah menjadi fokus penelitian terutama yang berkaitan dengan fenomena-fenomena 3D dalam bidang arsitektur, meterorologi, kedokteran, biologi dan sebagainya.[1]

## 1.2 Rumusan Masalah

1. Bagaimana cara menggunakan OpenGL untuk menggambar objek 3D secara efisien dan akurat?
2. Bagaimana OpenGL menangani transformasi seperti rotasi, translasi, dan skala pada objek 3D?
3. Bagaimana langkah-langkah untuk membuat dan menampilkan model ruangan kamar menggunakan OpenGL?

## 1.3 Tujuan

1. Memahami dan menerapkan penggunaan OpenGL untuk menggambar objek 3D secara efisien dan akurat.
2. Mempelajari cara OpenGL menangani transformasi seperti rotasi, translasi, dan skala pada objek 3D.
3. Merancang dan membuat model ruangan kamar menggunakan OpenGL, termasuk elemen-elemen interior seperti dinding, lantai, dan perabotan.

# BAB II
# TINJAUAN PUSTAKA

## 2.1 OpenGL

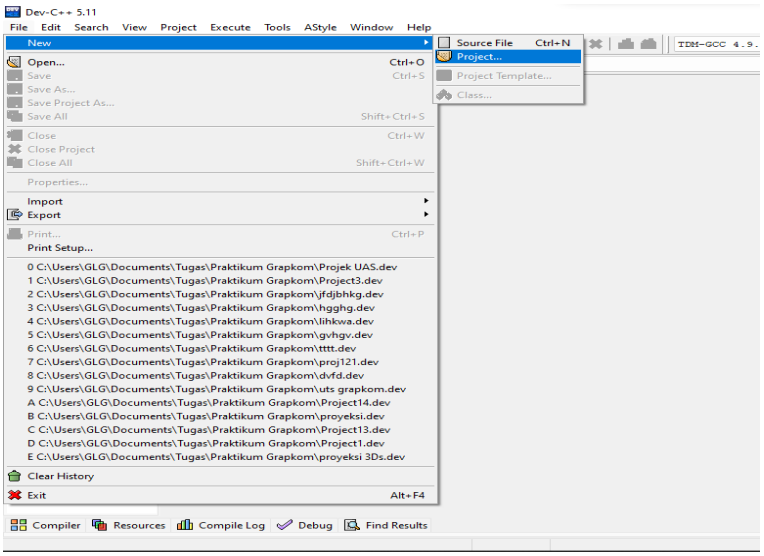
OpenGL (Open Graphic Library) merupakan library yang terdiri dari berbagai macam fungsi dan biasanya digunakan untuk menggambar sebuah atau beberapa objek 2 dimensi dan 3 dimensi. Library-library ini mendefinisikan sebuah cross-bahasa, cross-platform API (antarmuka pemrograman aplikasi) untuk menulis aplikasi yang menghasilkan komputer 2D dan 3D grafis. Bahasa pemrograman yang digunakan pada umumnya adalah pemrograman C/C++.. OpenGL merupakan library yang digunakan untuk melakukan pemrograman grafik; Graphic Programming. Untuk mempelajari pemrograman grafik ini, diharapkan kita dapat menguasai persamaan matematika, terutama operasi matriks. Karena, di dalam melakukan pemrograman grafik, akan dihadapkan mengenai pembuatan shading, shape, transform (rotate, translation, scala).[3]

## 2.2 Konfigurasi OpenGL pada Dev C++ atau VSCode

1. Pertama Buka aplikasi Dev C++

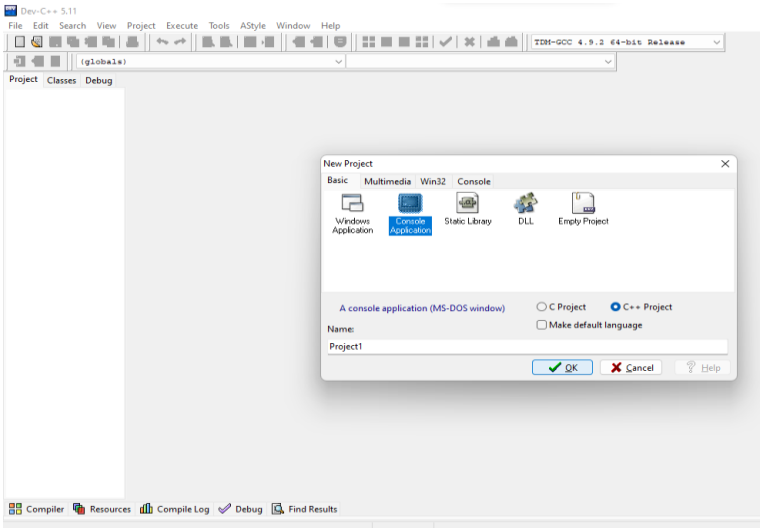

*Gambar 1 tampilan software*

2. Setelah dibuka Buka file dan click File, New dan pilih poject.



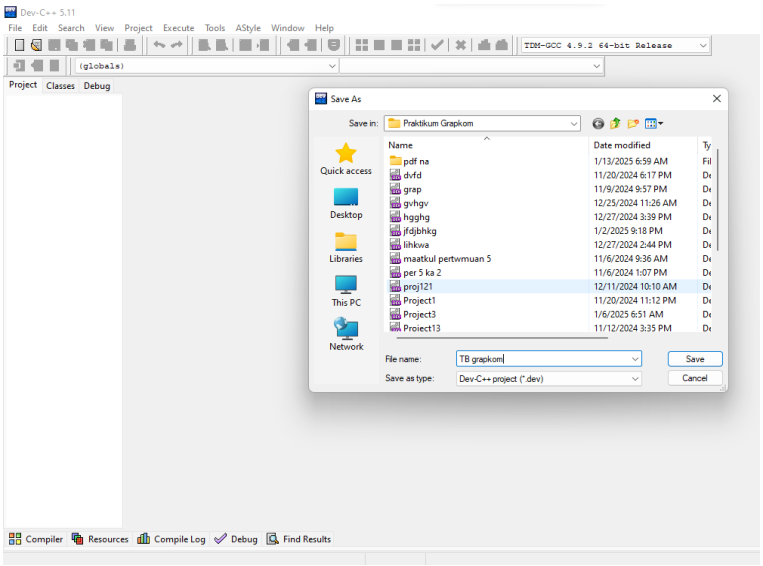

*Gambar 2 add project*

3. Jika sudah di New Project Kemudian Click Console Application, kemudian clik yang C++ Project , Kemudian OK.



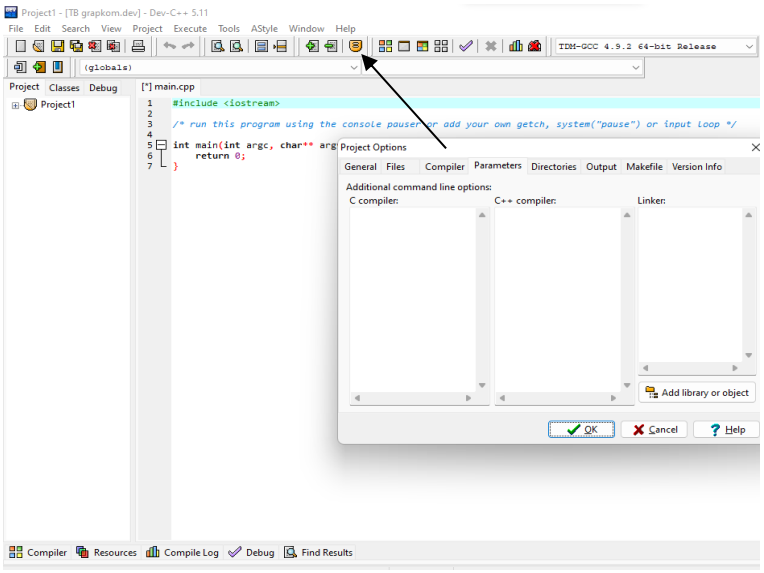

*Gambar 3 setelan project*

4. Selanjutnya Simpan (save) Project.



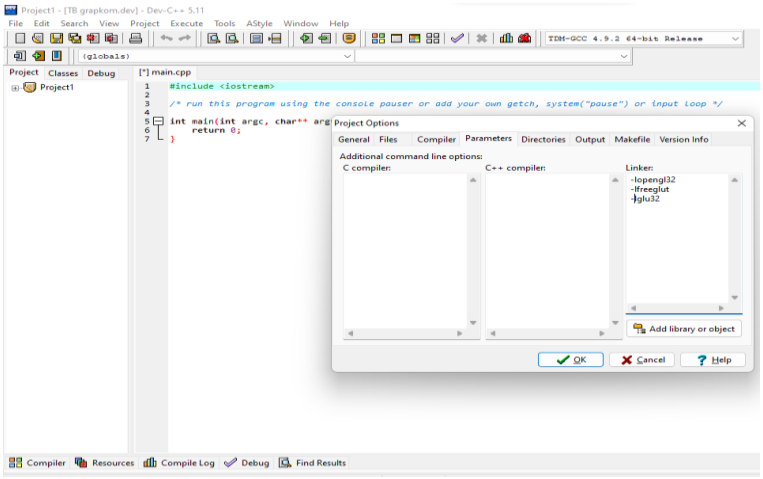

*Gambar 4 save project*

5. klik untuk masuk ke project option




*Gambar 5 project option*

6. Selanjutnya dibagian Linker masukan -lopengl32, -lfreeglut dan -lglu32.




*Gambar 6 konfigurasi*

# BAB III
# HASIL

## 3.1 Source Code

```
#include <stdio.h>
#include <GL/glut.h>

float angle = 0.0;
float scale = 1.0;
bool hidden = false;

GLfloat lightPos[] = {-0.8, 0.9, 0.4, 1.0};
bool lightingEnabled = false;

void dindingkiri();
void dindingkanan();
void ubin();
void drawKasur();
void meja();
void lampu();
void stand();
void drawcartecius();

//andi
void dindingkiri()
{
    // Dinding Kiri
    glPushMatrix();
    glColor3ub(60, 61, 55);
    glTranslatef(-1.1, 0.98, 0.0);
    glScalef(0.2, 2.0, 2.0);
    glutSolidCube(1.0);
    glPopMatrix();
}

//andi
void dindingkanan()
```

```
{
    // Dinding kanan
    glPushMatrix();
    glColor3ub(105, 117, 101);
    glTranslatef(0.65, 0.95, -1.05);
    glScalef(0.7, 1.9, 0.2);
    glutSolidCube(1.0);
    glPopMatrix();

    // dinding kiri
    glPushMatrix();
    glColor3ub(105, 117, 101);
    glTranslatef(-0.75, 0.95, -1.05);
    glScalef(0.9, 1.9, 0.2);
    glutSolidCube(1.0f);
    glPopMatrix();

    // dinding bawah
    glPushMatrix();
    glColor3ub(105, 117, 101);
    glTranslatef(0.0, 0.25, -1.05);
    glScalef(0.6, 0.5, 0.2);
    glutSolidCube(1.0f);
    glPopMatrix();

    // dinding atas
    glPushMatrix();
    glColor3ub(105, 117, 101);
    glTranslatef(0.0, 1.65, -1.05);
    glScalef(0.6, 0.5, 0.2);
    glutSolidCube(1.0);
    glPopMatrix();
}

//bj
void ubin()
{
    // Ubin
```

```
    glPushMatrix();
    glColor3ub(184, 0, 31);
    glScalef(2.0, 0.1, 2.0);
    glutSolidCube(1.0);
    glPopMatrix();
}

//andi
void drawCube(float x, float y, float z)
{
    glPushMatrix();
    glScalef(x, y, z);
    glutSolidCube(1.0);
    glPopMatrix();
}

//andi
void drawKasur()
{
    glPushMatrix();
    glTranslatef(-0.8, 0.0, -0.8);

    // matras
    glColor3f(0.9, 0.9, 0.7);
    glPushMatrix();
    glTranslatef(0.5, 0.15, 0.4);
    drawCube(1.4, 0.2, 0.6);
    glPopMatrix();

    // selimut
    glColor3f(0.0, 0.0, 0.8);
    glPushMatrix();
    glTranslatef(0.7, 0.3, 0.4);
    drawCube(1.0, 0.04, 0.6);
    glPopMatrix();

    // bantal
    glColor3f(0.6, 0.3, 0.2);
```

```
    glPushMatrix();
    glTranslatef(-0.05, 0.35, 0.4);
    drawCube(0.25, 0.15, 0.6);
    glPopMatrix();
    glPopMatrix();
}

//andi
void meja()
{
    glPushMatrix();
    glTranslatef(-1.2, 0.0, 0.1);

    // meja
    glColor3f(0.6, 0.4, 0.2);
    glPushMatrix();
    glTranslatef(0.4, 0.3, 0.3);
    drawCube(0.4, 0.5, 0.6);
    glPopMatrix();

    glPopMatrix();

    // laci
    glColor3f(0.82, 0.71, 0.55);
    glPushMatrix();
    glTranslatef(-0.6, 0.35, 0.4);
    drawCube(0.02, 0.18, 0.6);
    glPopMatrix();
}

//bj
void lampu()
{
    // lampu
    glColor3ub(255, 241, 0);
    glPushMatrix();
    glTranslatef(-0.8, 0.9, 0.4);
    glutSolidSphere(0.15, 20, 20);
```

```
    glPopMatrix();
}

//bj
void stand()
{
    // stand
    glColor3ub(255, 130, 37);
    glPushMatrix();
    glTranslatef(-0.8, 0.7, 0.4);
    drawCube(0.04, 0.4, 0.04);
    glPopMatrix();
}

//bj
void drawcartecius()
{
    glLineWidth(2.0);

    // Sumbu X
    glColor3f(1.0, 0.0, 0.0);
    glBegin(GL_LINES);
    glVertex3f(-50.0, 0.0, 0.0);
    glVertex3f(50.0, 0.0, 0.0);
    glEnd();

    // Sumbu Y
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_LINES);
    glVertex3f(0.0, -50.0, 0.0);
    glVertex3f(0.0, 50.0, 0.0);
    glEnd();

    // Sumbu Z
    glColor3f(0.0, 0.0, 1.0);
    glBegin(GL_LINES);
    glVertex3f(0.0, 0.0, -50.0);
    glVertex3f(0.0, 0.0, 50.0);
```

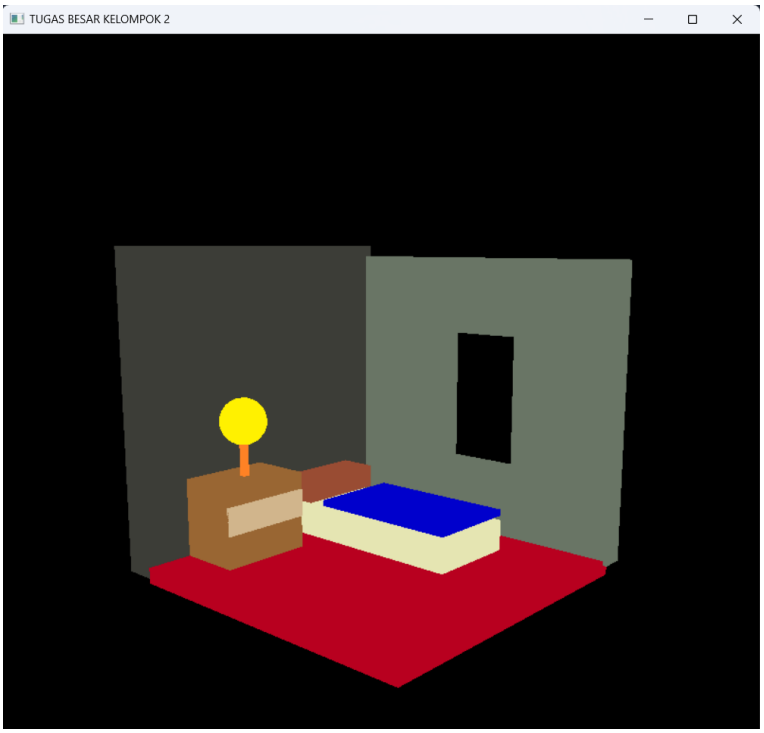
```
    glEnd();
}

//bj
void display()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    gluLookAt(2.0, 2.0, 5.0,  // Posisi kamera
              0.0, 1.0, 0.0,  // Titik yang dilihat
              0.0, 1.0, 0.0); // Arah atas kamera

    if (lightingEnabled)
    {
        glEnable(GL_LIGHTING);
        glLightfv(GL_LIGHT0, GL_POSITION, lightPos);
        glEnable(GL_LIGHT0);
        glEnable(GL_COLOR_MATERIAL);
    }
    else
    {
        glDisable(GL_LIGHT0);
        glDisable(GL_LIGHTING);
    }

    glPushMatrix();
    glTranslatef(0.0, 0.0, 0.0);
    glRotatef(angle, 0.0, 1.0, 0.0);
    glScalef(scale, scale, scale);

    dindingkiri();
    dindingkanan();
    ubin();
    drawKasur();
    meja();
    lampu();
    stand();
    glPopMatrix();
```

```
    if (hidden)
{
    drawcartecius();
}

    glutSwapBuffers();
}

//bj
void myKeyboard(unsigned char key, int x, int y)
{
    switch (key)
    {
    case 'c':
        hidden = !hidden;
        break;
    case 'a':
        angle += 10.0;
        break;
    case 'd':
        angle -= 10.0;
        break;
    case '1':
        scale += 0.1;
        break;
    case '2':
        scale -= 0.1;
        break;
    case 'o':
        lightingEnabled = !lightingEnabled;
        if (lightingEnabled)
        {
            glEnable(GL_LIGHTING);
        }
        else
        {
            glDisable(GL_LIGHTING);
```

```
        }
        glutPostRedisplay();
        break;
    }
    glutPostRedisplay();
}

//bj
int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(800, 800);
    glutInitWindowPosition(400, 50);
    glutCreateWindow("TUGAS BESAR KELOMPOK 2");
    glEnable(GL_DEPTH_TEST);
    glutDisplayFunc(display);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(45.0, 1.0, 1.0, 100.0);
    glMatrixMode(GL_MODELVIEW);
    glutKeyboardFunc(myKeyboard);
    glutMainLoop();
    return 0;
}
```

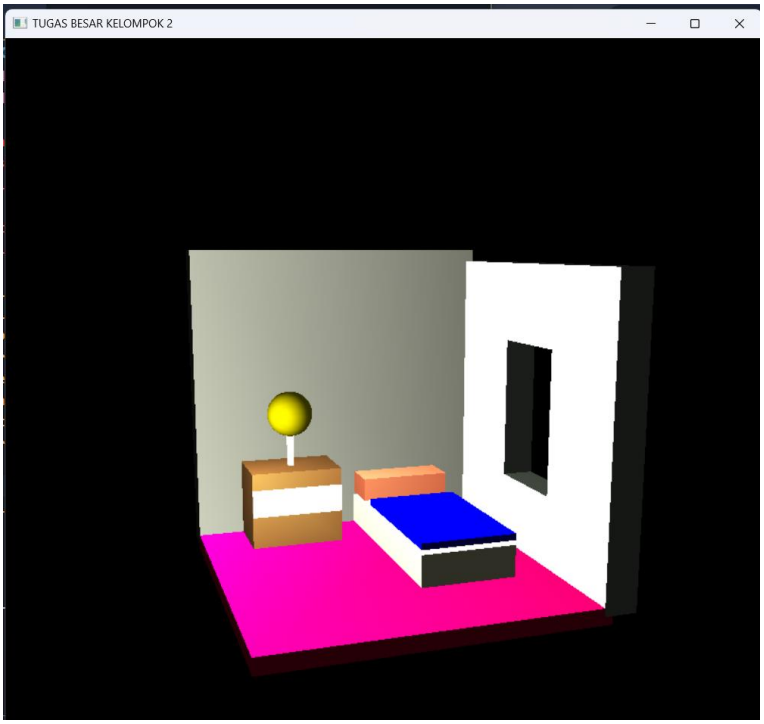
## 3.2 Output

1. Tampilan pada saat awal dijalankan.



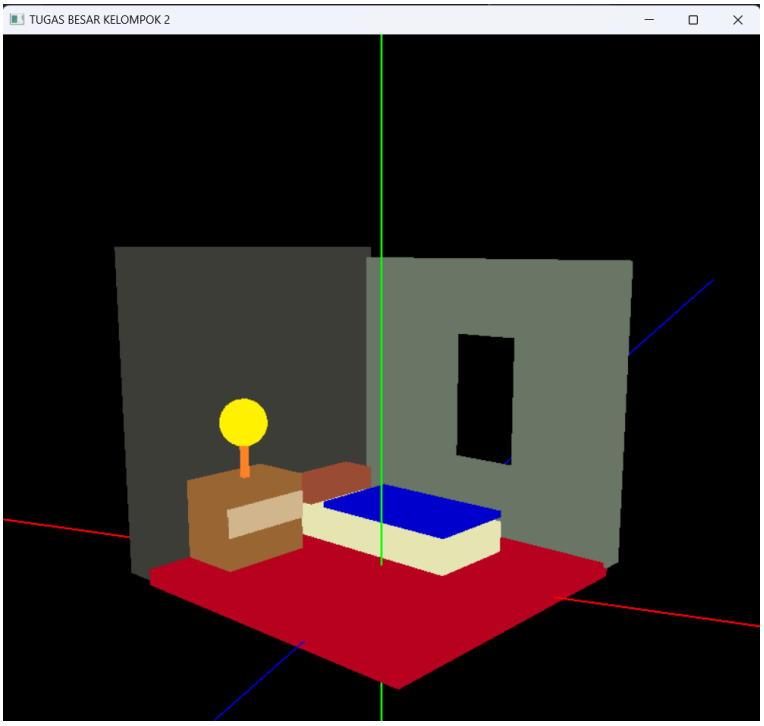

*Gambar 7 Tampilan awal*

2. Tampilan pada saat menggunakan pencahayaan.




*Gambar 8 tampilan pencahayaan*

3. Tampilan pada saat menggunakan garis cartecius.




*Gambar 9 tampilan garis cartesius*

### 3.3 Penjelasan

Source code ini adalah program OpenGL sederhana yang membuat sebuah ruangan 3D dengan berbagai elemen seperti dinding, kasur, meja, lampu, dan stand. Program ini menggunakan GLUT untuk menangani rendering, interaksi pengguna, dan kontrol kamera.

Penjelasan Mengapa Output Terlihat Seperti Itu:

1. **Transformasi Geometri**

   Setiap objek diposisikan dan diskalakan menggunakan fungsi seperti glTranslatef(), glScalef(), dan glutSolidCube(). Kombinasi transformasi ini menentukan bentuk, posisi, dan ukuran masing-masing objek.

2. **Pencahayaan**

   Jika pencahayaan diaktifkan dengan tombol o, efek pencahayaan dari glLightfv() menciptakan highlight dan bayangan pada objek.

3. **Interaksi Keyboard**

Transformasi global (rotasi dan skala) diterapkan pada seluruh ruangan berdasarkan input keyboard.

4. **Pipeline Rendering**

Semua elemen digambar dalam urutan tertentu pada pipeline rendering, sehingga terlihat sebagai ruangan lengkap dengan objek-objeknya.

5. **Koordinat Kamera**

Kamera ditentukan dengan gluLookAt(), yang memberikan sudut pandang 3D dan titik fokus pada ruangan.

# BAB IV

## 4.1. Kesimpulan

Kesimpulan dari praktikum ini adalah bahwa penggunaan OpenGL memungkinkan pembuatan model ruangan 3D secara efisien melalui pengaplikasian transformasi geometris seperti translasi, rotasi, dan skala. Dengan konfigurasi yang tepat, berbagai elemen interior seperti dinding, kasur, meja, dan lampu dapat dibuat dengan presisi. Implementasi pencahayaan menambahkan dimensi realistis pada visualisasi, sementara koordinat kamera dan interaksi keyboard memberikan fleksibilitas dalam pengamatan dan manipulasi model. Secara keseluruhan, praktikum ini berhasil mencapai tujuannya untuk memahami dasar-dasar OpenGL dalam visualisasi objek 3D serta memberikan wawasan tentang pipeline rendering, efek pencahayaan, dan peran transformasi geometris dalam grafika komputer.

## DAFTAR PUSTAKA


[1] bpmbkm.uma, “Pengertian Grafika Komputer, Sejarah”, [Online]. Available: https://bpmbkm.uma.ac.id/2022/01/28/pengertian-grafika-komputer-sejarah-aplikasi-model-dasar/

[2] D. Kamaltz, “Pengertian Grafika Komputer Dan Contohnya Serta Pemanfaatannya”, [Online]. Available: https://id.scribd.com/document/685175243/Pengertian-Grafika-Komputer-dan-Contohnya-serta-Pemanfaatannya

[3] dufatan, “Pengertian OpenGL”, [Online]. Available: https://www.dufatancom.id/2019/08/pengertian-opengl-dan-cara-kerjanya.html