Jurnal Algoritma
E-ISSN: 2302-7339 | P-ISSN: 1412-3622
INSTITUT TEKNOLOGI GARUT

PEMBUATAN OBJEK DAN PENCAHAYAAN SERTA BAYANGANNYA DENGAN LIBRARY OPENGL PADA BAHASA PEMROGRAMAN C++

Muhammad Iqbal Fathur Rohman[1], Adie Subarkah[2], Fahmi Hadi Palaha[3]

Jurnal Algoritma
Institut Teknologi Garut
Jl. Mayor Syamsu No. 1 Jayaraga Garut 44151 Indonesia
Email : jurnal@itg.ac.id

[1]2106184@itg.ac.id
[2]2106172@itg.ac.id
[3]2106056@itg.ac.id

Abstrak – OpenGL (Open Graphics Library) adalah standar API yang dapat digunakan untuk membuat aplikasi berbasis grafik, baik dua dimensi (2D) maupun tiga dimensi(3D). OpenGL ini bersifat cross-platform. Untuk membuat aplikasi menggunakan OpenGL, terlebih dahulu kita membutuhkan suatu konsepsi interfacing dalam implementasinya. Salah satu cara yang sudah umum digunakan adalah dengan membuat window-based OpenGL. Untuk dapat membuat konsep windowing pada OpenGL, kita memerlukan tool tertentu. Yang kita gunakan kali ini adalah GLUT (OpenGL Utility Toolkit). OpenGL menggunakan matrix sebagai komponen dasar untuk menghasilkan tampilan pada layar. Semua matrix ini didefinisikan untuk dapat memproses operasi-operasi dalam 3 dimensi. Untuk menghasilkan gambar yang realistis perlu memodelkan pencerminan dan pembiasan maupun memunculkan bayangan karena pengaruh dari adanya cahaya. Jurnal kaliini membahas semua itu dan mengimplementasikannya pada sebuah objek yang dirancang sendiri, yaitu model hewan anjing dari game minecraft.

Kata Kunci – 3D, C++, GLUT , OpenGL, OpenGL Utility Toolkit, Open Graphics Library, Grafika Gomputer, Praktikum Grafika Komputer.

I. PENDAHULUAN

A. Latar Belakang

OpenGL (Open Graphics Library) adalah standar API yang dapat digunakan untuk membuat aplikasi berbasis grafik, baik dua dimensi (2D) maupun tiga dimensi(3D). OpenGL ini bersifat cross-platform, artinya dapat dijalankan pada berbagai platform sistem operasi yang ada saat ini. OpenGL sendiri telah banyak dibahas oleh kalangan praktisi, akademisi, maupun industri, yang notabene berkutat dalam implementasi pemrograman grafik 2D dan 3D.

Untuk membuat aplikasi menggunakan OpenGL, terlebih dahulu kita membutuhkan suatu konsepsi interfacing dalam implementasinya. Salah satu cara yang sudah umum digunakan adalah dengan membuat window-based OpenGL. Untuk dapat membuat konsep windowing pada OpenGL, kita memerlukan tool tertentu. Yang kita gunakan kali ini adalah GLUT (OpenGL Utility Toolkit). GLUT dipilih karena di dalamnya telah terdapat banyak fungsi yang dapat dipakai untuk pembuatan application window. Disamping itu, windowing pada GLUT juga bersifat independen terhadap sistem operasi, sehingga kita tidak perlu repot-repot untuk mengubah kode program jika diterapkan pada sistem operasi yang berbeda.

OpenGL menggunakan matrix sebagai komponen dasar untuk menghasilkan tampilan pada layar. Semua matrix ini didefinisikan untuk dapat memproses operasi-operasi dalam 3 dimensi. Jika pada obyek dibuat dalam 2 dimensi, sebenarnya obyek-obyek tersebut adalah obyek 3 dimensi. Hanya saja dimensi ketiga diabaikan.

Untuk menghasilkan gambar yang realistis perlu memodelkan pencerminan dan pembiasan maupun memunculkan bayangan karena pengaruh dari adanya cahaya. Dengan memodelkan pencerminan untuk benda yang reflektif seperti cermin akan dihasilkan pantulan ataupun bayangan benda. Dan efek pembiasan cahaya dapat dimodelkan pada benda yang transparan untuk menghasilkan penampakan obyek lain yang berada di belakang obyek transparan tersebut serta efek pengumpulan cahaya bias. Efek bayangan ini sangat penting karena dengan adanya efek tersebut seolah-olah benda tersebut tampak nyata. Bayangan sebuah obyek benda harus disesuaikan dengan bentuk benda aslinya dan asal sumber cahaya tersebut berada dan banyaknya sumber cahaya.

Salah satu tujuan dari grafika komputer adalah menghasilkan tampilan yang senyata mungkin, dan karena pengaruh cahaya sangat besar terhadap hasil nyata maka dalam membuat tampilan akhir faktor pencahayaan harus diperhitungkan pula. Tetapi mengingat bahwa grafika komputer adalah model matematika dari kehidupan nyata maka pencahayaan juga harus diubah menjadi model matematika. Model matematika itu harus memenuhi persyaratan sebagai berikut :

1) Menghasilkan efek seperti cahaya sungguhan
2) Dapat dihitung dengan cepat

Model pencahayaan tiga dimensi yang realistik menyangkut dua elemen penting yang sangat berkaitan erat dengan shading model, yaitu

1) Keakuratan dalam menggambarkan objek.
2) Teknik pencahayaan yang baik.

Saat cahaya menimpa permukaan benda maka sebagian cahaya akan dipantulkan dan sebagian lain diserap. Bergantung kepada frekuensi atau panjang gelombang yang dipantulkan dan diserap maka kita akan melihat warna. Mata kita selain sensitif terhadap warna juga sensitif terhadap intensitas cahaya (brightness). Secara awam kita menyebut intensitas cahaya sebagai kecerahan. Bergantung kepada materi penyusun benda maka ada tiga kemungkinan arah pantulan cahaya ketika cahaya menimpa permukaan benda : diffuse, specular, translucent.

Pencahayaan Ambient adalah cahaya yang sudah berserakan begitu banyak disebabkan oleh lingkungan dan arahnya tidak mungkin ditentukan atau tampaknya datang dari segala penjuru. Backlighting pada sebuah ruangan memiliki komponen ambient besar, karena sebagian besar cahaya yang mencapai mata yang memantul dari banyak permukaan. Sebuah lampu sorot kecil di luar rumah memiliki komponen ambient, sebagian besar cahaya dalam arah yang sama, dan karena diluar, sangat sedikit cahaya mencapai mata setelah memantul dari benda-benda lain. Ketika cahaya ambient menyerang permukaan, maka akan tersebar merata di segala penjuru.

Komponen cahaya diffuse adalah komponen yang berasal dari satu arah, jadi akan terang kalau hal tersebut terjadi tepat diatas sebuah permukaan dibandingkan jika hampir tidak terjadi di atas permukaan. Setelah mengenai permukaan, akan tersebar merata di segala penjuru, sehingga tampak sama-sama terang, tak peduli di mana mata berada. Setiap cahaya yang datang dari posisi atau arah tertentu mungkin memiliki komponen diffuse.

Cahaya specular datang dari arah tertentu, dan cenderung terpental pada permukaan dalam arah yang diinginkan. sinar laser berkualitas tinggi memantul pada cermin dan menghasilkan hampir 100 persen refleksi specular. Logam atau plastik mengilap memiliki komponen specular tinggi, dan kapur atau karpet telah hampir tidak ada. Specularity dapat juga dianggap sebagai shininess. Meskipun sumber cahaya memberikan satu distribusi frekuensi, komponen ambient, diffuse, dan specular mungkin berbeda.

B. Rumusan Masalah

Adapun rumusan masalah dari penulisan jurnal ini adalah sebagai berikut:
1) Bagaimana cara membuat objek dari sebuah model dan menerapkan Pencahayaan serta bayangannya Terhadap Objeknya di Bahasa pemrograman C++ menggunakan library OpenGL;
2) Apa saja Syntax yang digunakan dalam menerapkan Pencahayaan dan bayangan Terhadap Objek di Bahasa pemrograman C++ menggunakan library OpenGL.

C. Tujuan

Adapun tujuan kami dari penulisan makalah ini adalah sebagai berikut:
1) Mahasiswa mampu menjelaskan tentang konsep Pencahayaan dan bayangan Terhadap Objek di Bahasa pemrograman C++ menggunakan library OpenGL.;
2) Mahasiswa mampu menerapkan pembuatan objek dari sebuah model dan menerapkan pencahayaan serta bayangannya Terhadap Objek di Bahasa pemrograman C++ menggunakan library OpenGL.

D. Objek list

Model yang dibuat pada penelitian ini merukapan hewan yang berasal dari sebuah game Minecraft yaitu anjing.

E. Objek Pembangun

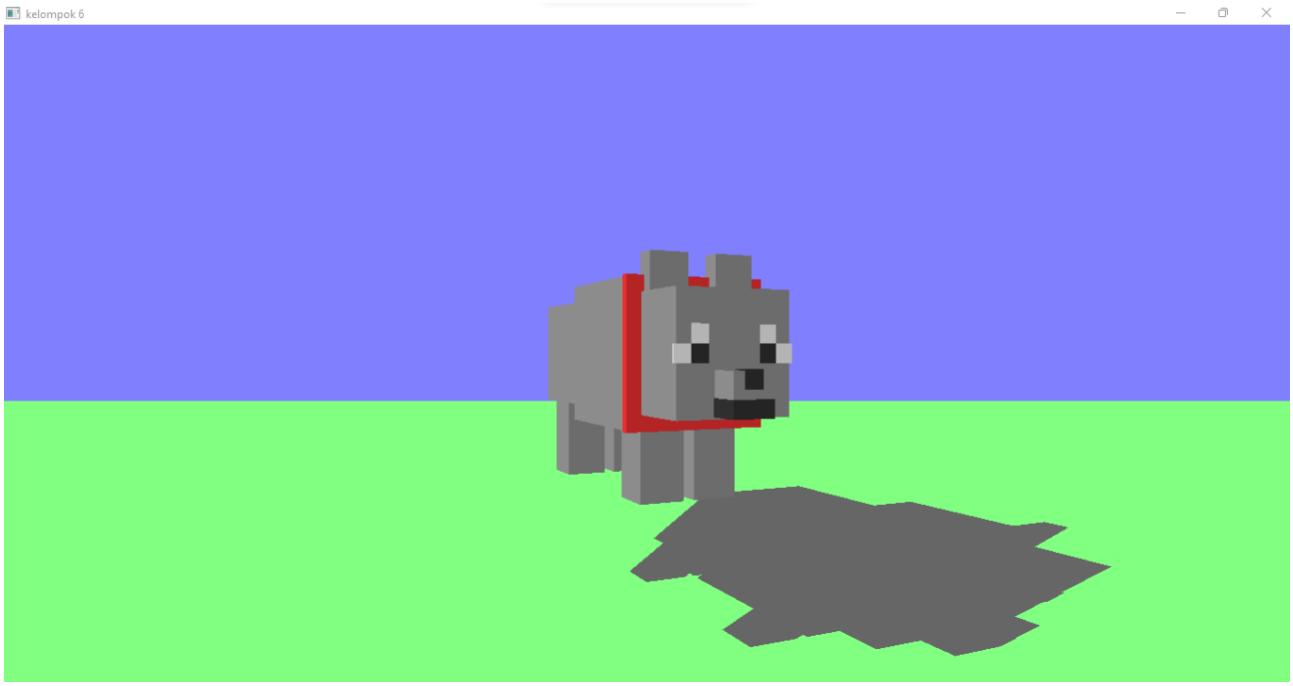
Objek yang membangun model hewan anjing dari game minecraft terdiri dari beberapa bagian, yaitu:
1) Kepala:
   a. 3 buah solid cube sebagai kerangka awal kepala
   b. Aksesoris kepala:
      i. 1 buah solid cube untuk Kalung berwarna merah
      ii. 1 buah solid cube untuk hidung
      iii. 6 buah solid cube yang di gepengkan untuk sepasang mata
2) Badan:
   a. 1 buah solid cube yang di rentangkan
3) Telinga:
   a. 2 buah solid cube yang di gepengkan
4) Ekor:
   a. 1 buah solid cube yang di rentangkan
5) Kaki:
   a. 4 buah solid cube yang di rentangkan

F. Batasan Masalah

Dalam pembahasan penelitian ini dibatasi sebagai berikut :
1) Bahasa yang digunakan bahasa pemrograman C++;
2) IDE yang digunakan DevC++ dengan openGL, freeglut;
3) Program ini hanya menampilkan model anjing dan pencahayaan serta bayangannya;
4) Model yang dibuat bisa diputar, diperbesar/perkecil, digeser;
5) Pencahayaan serta bayangannya dapat diatur posisi nya.

II. HASIL DAN PEMBAHASAN

A. Hasil



Gambar 1: Rendering Objek Anjing 3D

Tabel 1: Source Code

Source Code
Link Download https://github.com/miqbalfr12/Grafika-Komputer-C-OpenGL-3D-Objek-Anjing

```
#include <math.h>
#include <stdio.h>
#include <GL/glut.h>
double rx = 0.0;      // deklarasi rotasi sumbu x statik
double ry = 0.0;      // sumbu y
double sc = 1;        // scaling
float light_position[] = { 0.0, 80.0, 0.0 }; // koordinat sumber cahaya
float n[] = { 0.0, -1.0, 0.0 };
float e[] = { 0.0, -60.0, 0.0 };
bool warna;                // boolean untuk aktifasi warna
int a,b,c;            // untuk variabel statik tranlasi

int screen_width=400;
int screen_height=400;


int anjing(bool warna){
    // membuat objek anjing

      glPushMatrix(); //kepala
           if(warna)glColor3f(0.6,0.6,0.6);
           glTranslatef(0, 0, 7);
```

```
        glutSolidCube(25);
        glTranslatef(0, 0, 15);
        glutSolidCube(20);
        glTranslatef(0, -6, 12);
        glutSolidCube(7);

        //aksesoris kepala
        glPushMatrix(); //kalung merah
             if (warna)glColor3f(1,0.2,0.2);
             glTranslatef(0, 6, -15);
             glScaled(1,1,0.05);
             glutSolidCube(25.5);
        glPopMatrix();

        glPushMatrix(); // mulut
          if (warna) glColor3f(0.2,0.2,0.2);
             glTranslatef(0,-2.2, 0);
             glScaled(1,0.4,1);
             glutSolidCube(7.2);
        glPopMatrix();

        glPushMatrix(); // hidung
             glTranslatef(0,2.2, 2.2);
             glutSolidCube(3);
        glPopMatrix();

          glPushMatrix(); // mata
               glPushMatrix(); // mata kiri
                    glTranslatef(-6, 9, -2);
                   glScaled(1,1,-0.1);
                   if(warna)glColor3f(1,1,1);
                   glutSolidCube(3);
                   glTranslatef(-3,-3, 0);
                   glutSolidCube(3);
                   glTranslatef(3,0, 0);
                   if(warna)glColor3f(0.2,0.2,0.2);
                   glutSolidCube(3);
               glPopMatrix();
               glPushMatrix(); // mata kanan
                    glTranslatef(6, 9, -2);
                   glScaled(1,1,-0.1);
                   if(warna)glColor3f(1,1,1);
                   glutSolidCube(3);
                   glTranslatef(3,-3, 0);
                   glutSolidCube(3);
                   glTranslatef(-3,0, 0);
                   if(warna)glColor3f(0.2,0.2,0.2);
                   glutSolidCube(3);
               glPopMatrix();
        glPopMatrix();
    glPopMatrix();

    glPushMatrix(); // badan
      if (warna)glColor3f(0.6,0.6,0.6);
```

```
            glScaled(1,1,3);
            glutSolidCube(20);
    glPopMatrix();

    glPushMatrix(); // Telinga
            glTranslatef(6, 13, 20);
            glScaled(1,1,-0.5);
            glutSolidCube(7);
            glTranslatef(-12, 0, 0);
            glutSolidCube(7);
    glPopMatrix();

    glPushMatrix(); // Ekor
            glTranslatef(0, 0, -32);
            glRotatef(130,130,1,1);
            glScaled(1,1,3);
            glutSolidCube(5);

    glPopMatrix();

    glPushMatrix(); //Kaki
      glPushMatrix(); //Kaki depan
                glTranslatef(-5, -13, 13);
                glScaled(1,3,1);
                glutSolidCube(8);
                  glTranslatef(10, 0, 0);
                glutSolidCube(8);
            glPopMatrix();
           glPushMatrix(); //Kaki belakang
                glTranslatef(-5, -13, -23);
                glScaled(1,3,1);
                glutSolidCube(8);
                  glTranslatef(10, 0, 0);
                glutSolidCube(8);
      glPopMatrix();
    glPopMatrix();
}

// membuat proyeksi bayangan
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;
    mat[0] = l[0]*n[0]+c; // membuat matrik.
    mat[4] = n[1]*l[0];
    mat[8] = n[2]*l[0];
    mat[12] = -l[0]*c-l[0]*d;
    mat[1] = n[0]*l[1];
    mat[5] = l[1]*n[1]+c;
    mat[9] = n[2]*l[1];
    mat[13] = -l[1]*c-l[1]*d;
    mat[2] = n[0]*l[2];
    mat[6] = n[1]*l[2];
```

```
    mat[10] = l[2]*n[2]+c;
    mat[14] = -l[2]*c-l[2]*d;
    mat[3] = n[0];
    mat[7] = n[1];
    mat[11] = n[2];
    mat[15] = -d;
    glMultMatrixf(mat); // kalikan matrik
}

void lantai(){
    glColor3f(0.5,1,0.5); //Lantai
    glBegin(GL_QUADS);
         glNormal3f(0.0,1.0,0.0);
         glVertex3f(-1300.0,e[1]-0.1, 1300.0);
         glVertex3f( 1300.0,e[1]-0.1, 1300.0);
         glVertex3f( 1300.0,e[1]-0.1,-1300.0);
         glVertex3f(-1300.0,e[1]-0.1,-1300.0);
    glEnd();
}

void objek(){
    // untuk objek
    glMatrixMode(GL_MODELVIEW);
    glPushMatrix();
     glTranslatef(a,b,c);
         glPushMatrix();
              glRotatef(ry,0,1,0);
              glRotatef(rx,1,0,0);
              glScalef(sc,sc,sc);
              glEnable(GL_LIGHTING);
              glColor3f(0,0,1);
              anjing(1);
         glPopMatrix();
     glPopMatrix();
    glFlush();
}

void bayangan(){
    // bayangan yang muncul
    glPushMatrix();
         glPushMatrix();
              glShadowProjection(light_position,e,n);
              glTranslatef(a,b,c);
              glRotatef(ry,0,1,0);
              glRotatef(rx,1,0,0);
              glScalef(sc,sc,sc);
              glDisable(GL_LIGHTING);
              glColor3f(0.4,0.4,0.4);
              anjing(0);
         glPopMatrix();
    glPopMatrix();
}

void render(){
```

```
      glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
    glDisable(GL_CULL_FACE);
    glDisable(GL_LIGHTING);
    glClearColor(0.5,0.5,1,0.0); // latar belakang
    glColor3f(1.0,1.0,0.0);
    glBegin(GL_POINTS);
    glVertex3f(light_position[0],light_position[1],light_position[2]);
    glEnd();

    lantai();
    objek();
    bayangan();

    glutSwapBuffers();
}

void rotX(){
    rx+=0.1;
    render();
}
void rotY(){
    ry+=0.1;
    render();
}
void zoomIn(){
    sc+=0.001;
    render();
}
void zoomOut(){
    sc-=0.001;
    render();
}
void stop(){
    rx+=0;
    ry+=0;
    sc+=0;
    render();
}

void keypress(unsigned char c, int a, int b){
    if ( c==27 ) exit(0);
    else if ( c=='s' ) light_position[1]-=5.0;
    else if ( c=='w' ) light_position[1]+=5.0;
    else if ( c=='a' ) light_position[0]-=5.0;
    else if ( c=='d' ) light_position[0]+=5.0;
    else if ( c=='q' ) light_position[2]-=5.0;
    else if ( c=='e' ) light_position[2]+=5.0;
    else if ( c=='g' ) glutIdleFunc(rotY);
    else if ( c=='h' ) glutIdleFunc(rotX);
    else if ( c=='j' ) glutIdleFunc(stop);
    else if ( c=='k' ) glutIdleFunc(zoomIn);
    else if ( c=='l' ) glutIdleFunc(zoomOut);
}
```

```
void SpecialInput(int key, int x, int y){
      switch(key){
            case GLUT_KEY_UP:
                  c = c + 1;
                  glutPostRedisplay();
                  break;
            case GLUT_KEY_DOWN:
                  c = c - 1;
                  glutPostRedisplay();
                  break;
            case GLUT_KEY_LEFT:
                  a = a + 1;
                  glutPostRedisplay();
                  break;
            case GLUT_KEY_RIGHT:
                  a = a - 1;
                  glutPostRedisplay();
                  break;
            case GLUT_KEY_PAGE_UP:
                  b = b + 1;
                  glutPostRedisplay();
                  break;
            case GLUT_KEY_PAGE_DOWN:
                  b = b - 1;
                  glutPostRedisplay();
                  break;
      }
}

void resize(int width,int height)
{
       screen_width=width;
       screen_height=height;

       glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
       glViewport(0,0,screen_width,screen_height);
       glMatrixMode(GL_PROJECTION);
       glLoadIdentity();

gluPerspective(45.0f,(GLfloat)screen_width/(GLfloat)screen_height,1.0f,
1000.0f);

       glutPostRedisplay();
}

// Setup Pencahayaan
void SetupPencahayaan(){
    GLfloat mat_specular[] = {1.0, 1.0, 1.0, 1.0};
    GLfloat mat_shininess[] = {50.0};
    GLfloat light_diffuse[] ={1.0, 1.0, 1.0, 1.0};


    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
```

```
    glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_NORMALIZE);
    glEnable(GL_COLOR_MATERIAL);
    glEnable(GL_TEXTURE_2D);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60.0f, 1.0, 1.0, 400.0);
     glMatrixMode(GL_MODELVIEW);
     // Reset koordinat sebelum dimodifikasi/diubah
    glLoadIdentity();
     glTranslatef(0.0, 0.0, -150.0);
}

int main(int argc, char * argv[]){
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutCreateWindow("kelompok 6 ");
    glutReshapeFunc(resize);
    glutReshapeWindow(700,700);
    glutKeyboardFunc(keypress);
    glutSpecialFunc(SpecialInput);
    SetupPencahayaan();
    glutDisplayFunc(render);
    glutMainLoop();
    return 0;
}
```

Link download https://github.com/miqbalfr12/Grafika-Komputer-C-OpenGL-3D-Objek-Anjing

Fitur-fitur yang dapat di gunakan antara lainnya:

Tabel 2: Fitur

Tombol	Penjelasan
q	Memindahkan sumber cahaya kearah utara
e	Memindahkan sumber cahaya kearah selatan
a	Memindahkan sumber cahaya kearah barat
d	Memindahkan sumber cahaya kearah timur
w	Menaikan sumber cahaya ke atas
s	Menurunkan sumber cahaya ke bawah
g	Memutar objek terhadap sumbu Y
h	Memutar objek terhadap sumbu X
k	Memperbesar objek
l	Memperkecil objek
j	Menghentikan pergerakan objek
→	Menggeser Objek kearah kanan perspektif layar/objeck
←	Menggeser Objek kearah kiri perspektif layar/objeck
↑	Menggeser Objek depan kiri perspektif layar/objeck
↓	Menggeser Objek belakang kiri perspektif layar/objeck
PgUp	Menaikan Objek Keatas
PgDn	Menurunkan Objek Kebawah

B. Pembahasan

Yang membedakan dari penelitian sebelumnya antara lain:

1) Penerapan objek 3D menggunakan pencahayaan serta bayangannya itu menggunakan objek bawaan dari freeglut, seperti bola pejal, kotak, kerucut, teapot dan lain lain. Pada penelitian sekarang kami menggunakan objek yang dibuat sendiri dari kubus kubus yang disatukan dan di atur sedemikian rupa sehingga menjadi seperti anjing dalam game Minecraft.
2) Penggunaan fitur yang menggerakan objek dan cahaya serta bayangannya, pada penelitian sebelumnya, tombol hanya terbatas hanya mengatur posisi cahaya serta bayangannya. Pada penelitian sekarang kami menambahkan dan membuat fitur baru seperti pada Tabel 2.
3) Fungsi reshape, pada penelitian sebelumnya fungsi ini tidak digunakan maksimal. Pada penelitian sekarang kami menggunakan fungsi ini beserta kami modifikasi sehingga program saat dijalankan tidak usah khawatir apabila ukuran window di ubah-ubah, objek akan tetap berbentuk.
4) Pewarnaan objek, pada penelitian sebelumnya model bola pejal tidak memiliki warna, melainkan cahayanya yang diberi warna, pada penelitian sekarang objek diberikan warna.
   Pada bagian pewarnaan objek, digunakan fungsi percabangan, karena objek dibuat dalam sebuah fungsi yang dimana digunakan juga untuk bayangan, sehingga untuk memperbaiki bayangan yang berwarna, kami menggunakan fungsi percabangan ini yang hanya memutuhkan variable Boolean untuk mengaktifkan atau menonaktifkan warna. Pada saat model dipanggil sebagai objek, warna diaktifkan. Saat menjadi bayangan, warna dinonaktifkan.

III. KESIMPULAN

A. Kesimpulan

Adapun kesimpulan dari penulisan jurnal ini adalah sebagai berikut:
1) Kita dapat membuat objek dari sebuah model dan menerapkan Pencahayaan serta bayangannya Terhadap Objeknya di Bahasa pemrograman C++ menggunakan library OpenGL;
2) Kita dapat mengetahui Syntax yang digunakan dalam membuat objek dari sebuah model dan menerapkan Pencahayaan serta bayangannya Terhadap Objeknya di Bahasa pemrograman C++ menggunakan library OpenGL.

B. Saran

Adapun saran dari penulisan jurnal ini adalah sebagai berikut:
1) Objek masih bisa di tambah lagi dalam satu program;
2) Lantai dan background dapat ditambahkan objek seperti rerumputan, pohon, rumah, pegunungan dan lain-lain supaya lebih menarik.

UCAPAN TERIMA KASIH

“Penulis M.I. mengucapkan terima kasih kepada Dosen Pengampu Grafika Komputer I.T.J yang telah memberikan dukungan materi melalui Perkulihan Grafika Komputer di Kampus Institut Teknologi Garut, semoga beliau diberikan selalu kesehatan dan kelancaran segala urusannya oleh Allah S.W.T, Aamiin”.

DAFTAR PUSTAKA

[1] “Modul Praktikum 1_Praktikum Grafika Komputer,” Institut Teknologi Garut, Garut, 2022.
[2] microsoft, “fungsi glPushMatrix,” 6 Nov 2022. [Online]. Available: https://learn.microsoft.com/id-id/windows/win32/opengl/glpushmatrix.
[3] manual-inkscape.blogspot.com, “Render - 3D Polyhedrons,” 06 2016. [Online]. Available: http://manual-inkscape.blogspot.com/2016/06/render-3d-polyhedrons.html.
[4] Bruce Mechtly, Eric Rooker, and Konrad Mast. 2001. 3D rendering with C++ and openGL in undergraduate projects. J. Comput. Sci. Coll. 17, 1 (October 2001), 168–177.
[5] D. Hearn and M. P. Baker, “Computer Graphics 1 Instructor Information Books Required Textbook Title: Computer Graphics with OpenGL 3/E Course Goals and Outcomes”, Accessed: Jan. 25, 2023. [Online]. Available: www.cs.rit.edu/~jdb/teachingPhilosophy.html
[6] “Computer Graphics Programming in OpenGL with C++ - V. Scott Gordon, PhD, John L. Clevenger, PhD - Google Buku.” https://books.google.co.id/books?hl=id&lr=&id=0W8REAAAQBAJ&oi=fnd&pg=PT11&dq=C%2B%2B+opengl&ots=tvMsNGoZjL&sig=7Xii_U791oWvvzKAlsLyG6KpAmc&redir_esc=y#v=onepage&q&f=false (accessed Jan. 25, 2023).
[7] if102008.wordpress.com, “kumpulan Artikel tugas besar grafkom kelas if-10,” 17 Juli 2012. [Online]. Available: https://if102008.wordpress.com/category/tugas-besar-grafkom/.