**LAPORAN**

**PEMBANGUNAN OBJEK 3D MINIATUR VILLA**

**MATA KULIAH : KOMPUTER GRAFIK**

**DOSEN : Hendri Karisma, S.Kom.**



1. 10108439 R. Adzie
2. 10108431 Luthfiana A
3. 10107286 Restiana A
4. 10108445 Alfiyah

Kelas : IF - 9

**PROGRAM STUDI S1**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

**2012**

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang Masalah**

Persaingan bisnis dari masa ke masa semakin bersaing ketat terutama dalam bidang pariwisata. Salah satu fasilitas penting dalam berpariwisata adalah tempat bermalam atau penginapan. Villa menjadi pilihan banyak orang karena mamiliki fasiltas yang lengkap dan memberikan suasana nyaman seperti di rumah sendiri. Biasanya para pengunjung yang berniat menyewa sebuah villa akan datang langsung ke tempat dimana villa itu berada dan melakukan survey sebelum benar-benar menyewa tempat tersebut. Pengunjung dapat melihat seluruh fasilitas yang disediakan dan melihat keadaan bangunan secara langsung. Proses survey inilah yang terkadang memerlukan waktu dan tenaga ekstra. Semakin banyak diminati semakin banyak pula pesaingnya, para penyedia jasa perlu melakukan inovasi untuk menambah daya tarik pengunjung, sehingga kami mempunyai ide untuk untuk membuat program miniatur sebuah villa dengan objek tiga dimensi.

**1.2 Batasan Masalah**

Dalam pembangunan program miniatur villa ini, dibuat beberapa batasan masalah. Adapun batasan masalahnya adalah :

* + 1. Bahasa pemrograman yang digunakan adalah bahasa C.
    2. Komponen-komponen OpenGl yang digunakan diantaranya *cylinder*, *cube*, *sphere* dan *cone*.
    3. Fitur yang tersedia pada program miniatur villa ini adalah interaksi program dengan menggunakan *keyboard* dan *mouse*.

**BAB II**

**ISI**

**2.1 Deskripsi Komponen**

Deskripsi komponen merupakan penjelasan komponen-komponen dasar yang terdapat pada program miniatur villa. Komponen-komponen yang dipakai dalam pembangunan miniatur villa ini diantaranya adalah *cube* yang dipakai sebagian besar untuk bangunan villa, *cone* untuk pohon dan payung, *cylinder* untuk batang pohon, batang payung, tiang lampu, kaki mejadan *sphere* yang dipakai untuk lampu. Komponen-komponen tersebut disesuaikan dan dikombinasikan sehingga dapat membentuk suatu objek yang diharapkan dan dibuat menyerupai aslinya. Dilengkapi juga dengan *terrain* dan *texture* sehingga miniatur ini dikondisikan semirip mungkin dengan keadaan aslinya, tidak hanya menggambarkan bangunan fisik dan fasilitas dari villa tersebut namun keadaan sekitarnya juga.

**2.2 Fungsi-Fungsi Pembentuk Miniatur Villa**

1. Villa

|  |
| --- |
| glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.0, 4.22, 12.5);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1,1.3,0.8); // villa utama  glColor3f(1.5,0.5,1.5);  glutSolidCube(6.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.0, 2.1, 10);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(-1.3,1.8,0.05); // Jendela kanan Border bawah  glColor3f(0,0,0);  glutSolidCube(1.7);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.0, 2.1, 10);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1,1.4,0.08);  glColor3f(0.3, 0.2, 0.1); // Jendela kanan bawah  glutSolidCube(1.9);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.3, 2, 11.4);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1,1.3,0.1);  glColor3f(0.7, 0.7, 1); // Jendela kiri bawah  glutSolidCube(1.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.3, 2, 11.5);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(-1.3,1.8,0.1);  glColor3f(0,0,0); // Jendela kiri Border  glutSolidCube(1.3);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.3, 6, 11.4);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,1.0,0.1);  glColor3f(0.7, 0.7, 1); // Jendela kiri Atas  glutSolidCube(1.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.3, 6, 11.7);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1,1.0,0.3);  glColor3f(0,0,0); // Jendela Border kiri Atas  glutSolidCube(1.8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.1, 6, 10);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,1.0,0.03);  glColor3f(0.7, 0.7, 1); // Jendela kanan Atas  glutSolidCube(1.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.1, 6, 10.1);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,1.0,0.05);  glColor3f(0,0,0); // Jendela Border kanan Atas  glutSolidCube(1.8);  glPopMatrix();  glPopMatrix();      glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.35, 4.3, 13.24);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,2.2,0.95); // villa kiri  glColor3f(1.5,1.5,0.5);  glutSolidCube(3.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-2.2, 2.8, 8.0);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.5,1.8,0.2); // villa penyangga depan kiri  glColor3f(0,1,1);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.95, 6.5, 10.9);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.2,1.0,0.5); // villa penyangga samping kiri atas  glColor3f(0.5,0.5,1.5);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.95, 2.5, 10.9);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.2,1.37,0.5); // villa penyangga samping kiri bawah  glColor3f(0.5,0.5,1.5);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(2.2, 2.8, 8.0);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.5,1.8,0.2); // villa penyangga depan kanan  glColor3f(0,1,1);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.0, 5.3, 8.0);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(2.0,0.7,0.2); // villa penyangga depan atas  glColor3f(0,1,1);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(0.0, 4.8,9.2);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(2.0,0.3,0.6); // villa atap depan  glColor3f(0.5,0.5,0.5);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.6, 7.7,10.9);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,0.3,0.5); // villa atap kiri atas  glColor3f(0.5,0.5,1.5);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.6, 4.8,10.9);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(1.1,0.3,0.5); // villa atap kiri bawah  glColor3f(0.5,0.5,1.5);  glutSolidCube(3.0);  glPopMatrix();  glPopMatrix(); |

1. Tempat Berjemur

|  |
| --- |
| glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.7, 0.6, 4.0);  glRotatef(-0.5, 0.0, 1.0, 0.0);  glScalef(0.8,0.05,1.5);  glColor3f(0.2,0.2,0.2); // tempat berjemur  glutSolidCube(3.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.2, 1, 4.5);  glRotatef(-0.5, 0.0, 1.0, 0.0);  glScalef(0.8,0.05,1.5);  glColor3f(0,0.2,0.2); // kursi 1 kanan  glutSolidCube(0.6);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.2, 1, 5.0);  glRotatef(-40, 1, 0, 0);  glScalef(0.8,0.05,1.5);  glColor3f(0.5,0,0.2); // kursi sandaran kanan  glutSolidCube(0.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.2, 1, 5.0);  glRotatef(-40, 1, 0, 0);  glScalef(0.8,0.05,1.5);  glColor3f(0.5,0,0.2); // kursi sandaran kiri  glutSolidCube(0.5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.2, 1, 4.5);  glRotatef(-0.5, 0.0, 1.0, 0.0);  glScalef(0.8,0.05,1.5);  glColor3f(0,0.2,0.2); // kursi 2 kiri  glutSolidCube(0.6);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.7, 1.8, 5.0); // payung  glScalef(0.9,0.5,0.5);  glRotated(-90,1,0,0);  glColor3f(0.0,0,1);  glutSolidCone(0.8, 1.0, 10, 5);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.7, 1.8, 5.0);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.3,0.2,0.2); // Tiang Payung  qobj = gluNewQuadric();  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0,0,0);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.6, 1, 2.8);  glRotatef(-0.5, 0.0, 1.0, 0.0);  glScalef(0.8,0.05,1.3);  glColor3f(0.3, 0.2, 0.1); // meja  glutSolidCube(0.6);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.8, 1.0, 3.1);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki meja kiri bagian belakang sebelah kiri  qobj = gluNewQuadric();  glColor3f(0.3, 0.2, 0.1);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.8, 1.0, 2.5);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki meja kiri bagian depan sebelah kiri  qobj = gluNewQuadric();  glColor3f(0.3, 0.2, 0.1);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.5, 1.0, 3.1);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki meja kanan bagian belakang sebelah kanan  qobj = gluNewQuadric();  glColor3f(0.3, 0.2, 0.1);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.5, 1.0, 2.5);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki meja kanan bagian depan sebelah kanan  qobj = gluNewQuadric();  glColor3f(0.3, 0.2, 0.1);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.0, 1.0, 4.2);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian depan sebelah kanan  qobj = gluNewQuadric();  glColor3f(0, 0, 0);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.4, 1.0, 4.2);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian depan sebelah kiri  qobj = gluNewQuadric();  glColor3f(0, 0, 0);  gluQuadricDrawStyle(qobj,GLU\_LINE);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.0, 1.0, 5.0);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian belakang sebelah kanan  qobj = gluNewQuadric();  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0, 0, 0);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-5.4, 1.0, 5.0);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian belakang sebelah kiri  qobj = gluNewQuadric();  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0, 0, 0);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.0, 1.0, 4.2);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian depan sebelah kanan  qobj = gluNewQuadric();  glColor3f(0, 0, 0);  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0.3, 0.2, 0.1);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.4, 1.0, 4.2);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian depan sebelah kiri  qobj = gluNewQuadric();  glColor3f(0, 0, 0);  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0.3, 0.2, 0.1);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.0, 1.0, 5.0);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2);// kaki kursi kanan bagian belakng sebelah kanan  qobj = gluNewQuadric();  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0, 0, 0);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-4.4, 1.0, 5.0);  glRotatef(90, 1.0, 0.0, 0.0);  glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian belakng sebelah kiri  qobj = gluNewQuadric();  gluQuadricDrawStyle(qobj,GLU\_LINE);  glColor3f(0, 0, 0);  gluCylinder(qobj,0.2,0.2,5.0,50,8);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(-1.8, 0.6, 4.0);  glRotatef(-0.5, 0.0, 1.0, 0.0);  glScalef(2.2,0.02,1.5);  glColor3f(0.5, 0.5, 0.5); // alas  glutSolidCube(4.3);  glPopMatrix();  glPopMatrix(); |

1. Garasi

|  |
| --- |
| glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(7.8, 2.0, 8.0);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.6,1.8,0.2);  glColor3f(0.1, 0.3, 0.3);// villa penyangga depan kanan garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(7.8, 2.0, 15.2);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.6,1.8,0.2);  glColor3f(0.1, 0.3, 0.3);// villa penyangga belakang kanan garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(4.1, 2.0, 8.0);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.5,1.8,0.2);  glColor3f(0.1, 0.3, 0.3);// villa penyangga samping kiri garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(4.1, 2.0, 15.1);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.5,1.8,0.2);  glColor3f(0.1, 0.3, 0.3);// villa penyangga belakang kiri garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(6.0, 2.0, 15.1);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(0.5,1.8,0.2);  glColor3f(0.1, 0.3, 0.3);// villa penutup belakang garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glScalef(skala,skala,skala);  glPushMatrix();  glTranslatef(6.0, 4.0, 11.6);  glRotatef(0, 0.0, 1.0, 0.0);  glScalef(2.4,0.5,3.8); // villa penyangga depan atas garasi  glutSolidCube(2.0);  glPopMatrix();  glPopMatrix(); |

1. Lampu

|  |
| --- |
| float rbolalampu =1.0 ,gbolalampu = 1.0 ,bbolalampu =0.0;  void lampuvila(){  glPushMatrix();  glPushMatrix();  glColor3f(0.3, 0.2, 0.1); //glColor3f(0.4, 0.8, 0.1);-->warna hijau  glTranslatef(0.1, 0.8, 0.0);  glScalef(0.1, 2.40, 0.1);  glutSolidCube(0.5);  glPopMatrix();  //++++++++++++++++dasar lampu++++++++++++++  glPushMatrix();  glColor3f(180, 180, 179);  glTranslatef(0.1, 0.1, 0.0);  glScalef(0.3, 0.04, 0.3);  glutSolidSphere(0.5, 30, 30);  glPopMatrix();  //++++++++++++++++penyangga lampu++++++++++++++  glPushMatrix();  glColor3f(0.1, 0.1, 0.1);  glTranslatef(0.1, 0.35, 0.0);  glScalef(0.2, 0.99, 0.2);  glutSolidCube(0.5);  glPopMatrix();  //++++++++++++++++penahan lampu1++++++++++++++  glPushMatrix();  glColor3f(0.3, 0.2, 0.1);  glTranslatef(0.35, 1.28, 0.0);  glScalef(1.5, 0.1, 0.1);  glutSolidCube(0.5);  glPopMatrix();  //++++++++++++++++penahan lampu2++++++++++++++  glPushMatrix();  glColor3f(0.3, 0.2, 0.1);  glTranslatef(0.20, 1.15, 0.0);  glScalef(0.7, 0.1, 0.1);  glutSolidCube(0.5);  glPopMatrix();  //++++++++++++++++penahan lampu3++++++++++++++  glPushMatrix();  glColor3f(0.3, 0.2, 0.1);  glTranslatef(0.20, 1.20, 0.0);  glScalef(0.1, 0.5, 0.1);  glutSolidCube(0.5);  glPopMatrix();  //++++++++++++++++atap lampu++++++++++++++  glPushMatrix();  glColor3f(0.5, 0.3, 0.2);  glTranslatef(0.75, 1.23, 0.0);  glScalef(0.3, 0.04, 0.3);  glutSolidSphere(0.8, 210, 10);  glPopMatrix();  //++++++++++++++paku penahan lampu+++++++++++  glPushMatrix();  glColor3f(180, 180, 179);  glTranslatef(0.75, 1.28, 0.0);  glScalef(0.3, 0.3, 0.3);  glutSolidSphere(0.10, 10, 10);  glPopMatrix();  //++++++++++++++bola lampu+++++++++++  glPushMatrix();  glColor3f(rbolalampu, gbolalampu, bbolalampu);  glTranslatef(0.75, 1.15, 0.0);  glScalef(0.8, 0.8, 0.8);  glutSolidSphere(0.10, 20, 20);  glPopMatrix();  glPopMatrix();  } |

1. Pohon dan Pepohonan

|  |
| --- |
| void pohon(){  glPushMatrix();  glPushMatrix();  glRotatef(-90,1.0,0.0,0.0);  glColor3f(0.5, 0.3, 0.2);  glutSolidCone(4,20,100,100);  glPopMatrix();  glPushMatrix();  glTranslatef(0.0,10.0,0.0);  glPushMatrix();  glRotatef(-90,1.0,0.0,0.0);  glColor3f(0.6, 1, 0.6);  glutSolidCone(9,20,100,10);  glPopMatrix();  glPopMatrix();    glPushMatrix();  glTranslatef(0.0,15.0,0.0);  glPushMatrix();  glRotatef(-90,1.0,0.0,0.0);  glColor3f(0.5, 1, 0.5);  glutSolidCone(8,20,100,10);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0,20.0,0.0);  glPushMatrix();  glRotatef(-90,1.0,0.0,0.0);  glColor3f(0.4, 1, 0.4);  glutSolidCone(7,20,100,10);  glPopMatrix();  glPopMatrix();  glPopMatrix();  }  void pepohonan(){  glPushMatrix();  glTranslatef(100,-5,-100);  glScalef(2,2,2);  pohon();  glPopMatrix();  glPushMatrix();  glTranslatef(65,-5,-20);  glScalef(1.5,1.5,1.5);  pohon();  glPopMatrix();  glPushMatrix();  glTranslatef(135,-5,10);  glScalef(1.5,1.5,1.5);  pohon();  glPopMatrix();  glPushMatrix();  glTranslatef(-110,-5,-80);  glScalef(1.7,1.7,1.7);  pohon();  glPopMatrix();  glPushMatrix();  glTranslatef(-120,-5,-20);  glScalef(1.4,1.4,1.4);  pohon();  glPopMatrix();  glPushMatrix();  glTranslatef(-130,-5,20);  glScalef(1.4,1.4,1.4);  pohon();  glPopMatrix();  } |

1. *Terrain* dan *Texture*

|  |
| --- |
| class Terrain {  private:  int w; //Width  int l; //Length  float\*\* hs; //Heights  Vec3f\*\* normals;  bool computedNormals; //Whether normals is up-to-date  public:  Terrain(int w2, int l2) {  w = w2;  l = l2;  hs = new float\*[l];  for (int i = 0; i < l; i++) {  hs[i] = new float[w];  }  normals = new Vec3f\*[l];  for (int i = 0; i < l; i++) {  normals[i] = new Vec3f[w];  }  computedNormals = false;  }  ~Terrain() {  for (int i = 0; i < l; i++) {  delete[] hs[i];  }  delete[] hs;  for (int i = 0; i < l; i++) {  delete[] normals[i];  }  delete[] normals;  }  int width() {  return w;  }  int length() {  return l;  }  //Sets the height at (x, z) to y  void setHeight(int x, int z, float y) {  hs[z][x] = y;  computedNormals = false;  }  //Returns the height at (x, z)  float getHeight(int x, int z) {  return hs[z][x];  }  //Computes the normals, if they haven't been computed yet  void computeNormals() {  if (computedNormals) {  return;  }  //Compute the rough version of the normals  Vec3f\*\* normals2 = new Vec3f\*[l];  for (int i = 0; i < l; i++) {  normals2[i] = new Vec3f[w];  }  for (int z = 0; z < l; z++) {  for (int x = 0; x < w; x++) {  Vec3f sum(0.0f, 0.0f, 0.0f);  Vec3f out;  if (z > 0) {  out = Vec3f(0.0f, hs[z - 1][x] - hs[z][x], -1.0f);  }  Vec3f in;  if (z < l - 1) {  in = Vec3f(0.0f, hs[z + 1][x] - hs[z][x], 1.0f);  }  Vec3f left;  if (x > 0) {  left = Vec3f(-1.0f, hs[z][x - 1] - hs[z][x], 0.0f);  }  Vec3f right;  if (x < w - 1) {  right = Vec3f(1.0f, hs[z][x + 1] - hs[z][x], 0.0f);  }  if (x > 0 && z > 0) {  sum += out.cross(left).normalize();  }  if (x > 0 && z < l - 1) {  sum += left.cross(in).normalize();  }  if (x < w - 1 && z < l - 1) {  sum += in.cross(right).normalize();  }  if (x < w - 1 && z > 0) {  sum += right.cross(out).normalize();  }  normals2[z][x] = sum;  }  }  //Smooth out the normals  const float FALLOUT\_RATIO = 0.5f;  for (int z = 0; z < l; z++) {  for (int x = 0; x < w; x++) {  Vec3f sum = normals2[z][x];  if (x > 0) {  sum += normals2[z][x - 1] \* FALLOUT\_RATIO;  }  if (x < w - 1) {  sum += normals2[z][x + 1] \* FALLOUT\_RATIO;  }  if (z > 0) {  sum += normals2[z - 1][x] \* FALLOUT\_RATIO;  }  if (z < l - 1) {  sum += normals2[z + 1][x] \* FALLOUT\_RATIO;  }  if (sum.magnitude() == 0) {  sum = Vec3f(0.0f, 1.0f, 0.0f);  }  normals[z][x] = sum;  }  }  for (int i = 0; i < l; i++) {  delete[] normals2[i];  }  delete[] normals2;  computedNormals = true;  }  //Returns the normal at (x, z)  Vec3f getNormal(int x, int z) {  if (!computedNormals) {  computeNormals();  }  return normals[z][x];  }  };  Terrain\* loadTerrain(const char\* filename, float height) {  Image\* image = loadBMP(filename);  Terrain\* t = new Terrain(image->width, image->height);  for (int y = 0; y < image->height; y++) {  for (int x = 0; x < image->width; x++) {  unsigned char color = (unsigned char) image->pixels[3 \* (y  \* image->width + x)];  float h = height \* ((color / 255.0f) - 0.5f);  t->setHeight(x, y, h);  }  }  delete image;  t->computeNormals();  return t;  }  Terrain\* \_terrainRumput;  Terrain\* \_terrainTanah;  Terrain\* \_terrainAir;  void light\_select(GLenum which)  {  glEnable(which);  glLightfv(which, GL\_DIFFUSE, cahaya);  glutPostRedisplay();  }  void initRendering() {  glEnable(GL\_DEPTH\_TEST);  glEnable(GL\_COLOR\_MATERIAL);  glEnable(GL\_LIGHTING);  glEnable(GL\_NORMALIZE);  glShadeModel(GL\_SMOOTH);  }  void drawSceneTerra(Terrain \*terrain, GLfloat r, GLfloat g, GLfloat b) {  float scale = 500.0f / max(terrain->width() - 1, terrain->length() - 1);  glScalef(scale, scale, scale);  glTranslatef(-(float) (terrain->width() - 1) / 2, 0.0f,  -(float) (terrain->length() - 1) / 2);  glColor3f(r, g, b);  for (int z = 0; z < terrain->length() - 1; z++) {  //Makes OpenGL draw a triangle at every three consecutive vertices  glBegin(GL\_TRIANGLE\_STRIP);  for (int x = 0; x < terrain->width(); x++) {  Vec3f normal = terrain->getNormal(x, z);  glNormal3f(normal[0], normal[1], normal[2]);  glVertex3f(x, terrain->getHeight(x, z), z);  normal = terrain->getNormal(x, z + 1);  glNormal3f(normal[0], normal[1], normal[2]);  glVertex3f(x, terrain->getHeight(x, z + 1), z + 1);  }  glEnd();  }  }  //Load texture BMP  int load\_texture(char \*file\_name, unsigned int width, unsigned int height){  GLubyte \*imgbitmap;  FILE \*file;  unsigned short int depth=3;  if ((file = fopen(file\_name, "rb"))==NULL){  printf("File tidak ditemukan: %s!\n",file\_name);  exit(1);  }  imgbitmap = (GLubyte \*) malloc (width \* height \* depth \* (sizeof(GLubyte)));  if (imgbitmap == NULL){  printf("Alokasi textures ke Memory gagal!\n");  fclose(file);  exit(1);  }  fread(imgbitmap, width \* height \* depth, 1, file);  fclose(file);  glGenTextures(1, &texture\_id);  glBindTexture(GL\_TEXTURE\_2D, texture\_id);  glPixelStorei(GL\_UNPACK\_ALIGNMENT,1);  //menyesuaikan ukuran textur ketika image lebih besar dari texture  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MAG\_FILTER, GL\_NEAREST); //  //menyesuaikan ukuran textur ketika image lebih kecil dari texture  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MIN\_FILTER, GL\_NEAREST); //  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_S, GL\_REPEAT);  glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_T, GL\_REPEAT);  glTexEnvi(GL\_TEXTURE\_ENV, GL\_TEXTURE\_ENV\_MODE, GL\_REPLACE);  gluBuild2DMipmaps(GL\_TEXTURE\_2D, GL\_RGB8, width, height, GL\_BGR, GL\_UNSIGNED\_BYTE, imgbitmap);  glTexImage2D(GL\_TEXTURE\_2D, 0, GL\_RGB8, width, height, 0, GL\_BGR, GL\_UNSIGNED\_BYTE, imgbitmap);  free(imgbitmap);  return texture\_id;  } |

## 2.3 Fitur-fitur yang Terdapat Dalam Program

Dalam program miniatur villa ini terdapat beberapa fitur yang bisa digunakan untuk melihat tampilan villa dari dari berbagai arah. Berikut ini adalah tombol-tombol yang bisa digunakan untuk menggerakan tampilan miniatur villa, yaitu :

1. Tombol Up

Digunakan untuk memutarkan villa ke arah atas

1. Tombol Down

Digunakan untuk memutarkan villa ke arah bawah.

1. Tombol Left

Digunakan untuk memutarkan villa ke arah kiri

1. Tombol Right

Digunakan untuk memutarkan villa ke arah kanan

1. Tombol 1

Digunakan untuk menentukan koordinat kamera.

1. Tombol 2

Digunakan untuk menyalakan lampu.

1. Tombol 3

Digunakan untuk mematikan lampu.

1. Tombol W

Digunakan untuk menggeser villa ke depan.

1. Tombol S

Digunakan untuk menggeser villa ke belakang.

1. Tombol A

Digunakan untuk menggeser villa ke kiri.

1. Tombol D

Digunakan untuk menggeser villa ke kanan.

1. Tombol ESC

Digunakan untuk menutup program.

1. Mouse

Digunakan untuk mengatur pergerakan villa.

|  |
| --- |
| void KeyBoard(unsigned char key, int x, int y)  {  switch (key)  {  case '1':  //menentukan koordinat camera  printf("%f %f %f \n\tat %f %f %f \n\tup %f %f %f",  myView.position().x,myView.position().y,myView.position().z,  myView.view().x,myView.view().y,myView.view().z,  myView.upVector().x,myView.upVector().y,myView.upVector().z);  break;  case '2':  bbolalampu += 0.4;  glEnable(GL\_LIGHT2);  break;  case '3':  bbolalampu -= 0.4;  glDisable(GL\_LIGHT2);  break;  case 'w': //move forward  myView.move(speed);  break;  case 's': //move back  myView.move(-speed);  break;  case 'a': //strafe left  myView.strafe(-speed);  break;  case 'd': //strafe right  myView.strafe(speed);  break;  case 27: // “esc” on keyboard  exit(0);  break;  }  glutPostRedisplay();  }  void specialKeyHandler(int key, int x, int y)  {  switch(key)  {  case GLUT\_KEY\_UP:  myView.pitch(10);  break;  case GLUT\_KEY\_DOWN:  myView.pitch(-10);  break;  case GLUT\_KEY\_LEFT:  myView.turn(10);  break;  case GLUT\_KEY\_RIGHT:  myView.turn(-10);  break;  }  glutPostRedisplay();  }  void mouseButtonHandler(int button, int state, int x, int y)  {  if(button == GLUT\_LEFT\_BUTTON) {  if(state == GLUT\_DOWN) {  xPrev = x;  yPrev = y;  dragging = true;  }  else //state == GLUT\_UP  dragging = false;  }  glutPostRedisplay();  }  void mouseMotionHandler(int x, int y){  int xDiff, yDiff;  if(dragging) {  xDiff = x - xPrev;  yDiff = y - yPrev;  if(xDiff != 0)  myView.turn(-xDiff);  if(yDiff != 0)  myView.pitch(-yDiff);  xPrev = x;  yPrev = y;  glutPostRedisplay();  }  glutPostRedisplay();  } |

**BAB III**

**KESIMPULAN**

Setelah melakukan proses implementasi dari hasil perancangan pada program miniatur Villa maka dapat diambil beberapa kesimpulan yaitu :

1. Dengan adanya program miniatur villa ini dapat di jadikan media promosi kepada pengunjung yang ingin membelinya atau menyewanya.
2. Merepresentasikan keadaan villa yang sesungguhnya tanpa melihat langsung bentuk fisik villa ini.
3. Dapat melakukan interaksi terhadap program miniatur villa.
4. Villa ini juga dapat menjadi suatu objek yang menawan yang pantas diberi harga yang sesuai akan pesona alam yang memukau dan strategis.

**BAB IV**

**SOURCE CODE**

/\*

Kelompok 9

Kelas IF-9

Nama:

1. 10107286 - Restiana Ayunita

2. 10108431 - Luthfiana Adityarini

3. 10108439 - R. Adzie Ramadani Kusumah

4. 10108445 - Alfiyah

\*/

#include<Windows.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

#include<gl/glew.h>

#include <GL/glut.h>

#include <GL/glu.h>

#include <GL/gl.h>

#include "imageloader.h"

#include "vec3f.h"

#include <ctype.h>

//include camera class:

#include "camera.h"

//texture

unsigned int texture\_id, langit;

//global variable untuk camera

float speed = 5;

CCamera myView;

int xPrev = 0, yPrev = 0;

bool dragging = false;

//variabel terrain

float lastx, lasty;

GLint stencilBits;

//lampu

static int lampu = 5;

GLUquadricObj \* qobj;

//keseluruhan villa

GLfloat cahaya[] ={1.0, 1.0, 1.0, 1.0};

const GLfloat light\_ambient[] = { 2.7f, 2.7f, 2.7f, 1.0f };

const GLfloat light\_diffuse[] = { 0.7f, 0.7f, 0.7f, 1.0f };

const GLfloat light\_specular[] = { 1.0f, 1.0f, 0.0f, 1.0f };

const GLfloat light\_position[] = { 1.0f, 1.0f, 1.0f, 1.0f };

const GLfloat light\_ambient2[] = { 0.3f, 0.3f, 0.3f, 0.0f };

const GLfloat light\_diffuse2[] = { 0.3f, 0.3f, 0.3f, 0.0f };

const GLfloat mat\_ambient[] = { 0.8f, 0.8f, 0.8f, 1.0f };

const GLfloat mat\_diffuse[] = { 0.8f, 0.8f, 0.8f, 1.0f };

const GLfloat mat\_specular[] = { 1.0f, 1.0f, 0.0f, 1.0f };

const GLfloat high\_shininess[] = { 100.0f };

GLfloat

cahaya\_kanan[] ={-1.0, 0.0, 1.0, 1.0},

cahaya\_kiri[] ={1.0, 0.0, 1.0, 0.0};

GLUquadricObj \*quadObj;

static float lmodel\_twoside[] =

{GL\_TRUE};

static float lmodel\_oneside[] =

{GL\_FALSE};

using namespace std;

float shineo = 0.02 ,shinev =0.02;

class Terrain {

private:

int w; //Width

int l; //Length

float\*\* hs; //Heights

Vec3f\*\* normals;

bool computedNormals; //Whether normals is up-to-date

public:

Terrain(int w2, int l2) {

w = w2;

l = l2;

hs = new float\*[l];

for (int i = 0; i < l; i++) {

hs[i] = new float[w];

}

normals = new Vec3f\*[l];

for (int i = 0; i < l; i++) {

normals[i] = new Vec3f[w];

}

computedNormals = false;

}

~Terrain() {

for (int i = 0; i < l; i++) {

delete[] hs[i];

}

delete[] hs;

for (int i = 0; i < l; i++) {

delete[] normals[i];

}

delete[] normals;

}

int width() {

return w;

}

int length() {

return l;

}

//Sets the height at (x, z) to y

void setHeight(int x, int z, float y) {

hs[z][x] = y;

computedNormals = false;

}

//Returns the height at (x, z)

float getHeight(int x, int z) {

return hs[z][x];

}

//Computes the normals, if they haven't been computed yet

void computeNormals() {

if (computedNormals) {

return;

}

//Compute the rough version of the normals

Vec3f\*\* normals2 = new Vec3f\*[l];

for (int i = 0; i < l; i++) {

normals2[i] = new Vec3f[w];

}

for (int z = 0; z < l; z++) {

for (int x = 0; x < w; x++) {

Vec3f sum(0.0f, 0.0f, 0.0f);

Vec3f out;

if (z > 0) {

out = Vec3f(0.0f, hs[z - 1][x] - hs[z][x], -1.0f);

}

Vec3f in;

if (z < l - 1) {

in = Vec3f(0.0f, hs[z + 1][x] - hs[z][x], 1.0f);

}

Vec3f left;

if (x > 0) {

left = Vec3f(-1.0f, hs[z][x - 1] - hs[z][x], 0.0f);

}

Vec3f right;

if (x < w - 1) {

right = Vec3f(1.0f, hs[z][x + 1] - hs[z][x], 0.0f);

}

if (x > 0 && z > 0) {

sum += out.cross(left).normalize();

}

if (x > 0 && z < l - 1) {

sum += left.cross(in).normalize();

}

if (x < w - 1 && z < l - 1) {

sum += in.cross(right).normalize();

}

if (x < w - 1 && z > 0) {

sum += right.cross(out).normalize();

}

normals2[z][x] = sum;

}

}

//Smooth out the normals

const float FALLOUT\_RATIO = 0.5f;

for (int z = 0; z < l; z++) {

for (int x = 0; x < w; x++) {

Vec3f sum = normals2[z][x];

if (x > 0) {

sum += normals2[z][x - 1] \* FALLOUT\_RATIO;

}

if (x < w - 1) {

sum += normals2[z][x + 1] \* FALLOUT\_RATIO;

}

if (z > 0) {

sum += normals2[z - 1][x] \* FALLOUT\_RATIO;

}

if (z < l - 1) {

sum += normals2[z + 1][x] \* FALLOUT\_RATIO;

}

if (sum.magnitude() == 0) {

sum = Vec3f(0.0f, 1.0f, 0.0f);

}

normals[z][x] = sum;

}

}

for (int i = 0; i < l; i++) {

delete[] normals2[i];

}

delete[] normals2;

computedNormals = true;

}

//Returns the normal at (x, z)

Vec3f getNormal(int x, int z) {

if (!computedNormals) {

computeNormals();

}

return normals[z][x];

}

};

Terrain\* loadTerrain(const char\* filename, float height) {

Image\* image = loadBMP(filename);

Terrain\* t = new Terrain(image->width, image->height);

for (int y = 0; y < image->height; y++) {

for (int x = 0; x < image->width; x++) {

unsigned char color = (unsigned char) image->pixels[3 \* (y

\* image->width + x)];

float h = height \* ((color / 255.0f) - 0.5f);

t->setHeight(x, y, h);

}

}

delete image;

t->computeNormals();

return t;

}

Terrain\* \_terrainRumput;

Terrain\* \_terrainTanah;

Terrain\* \_terrainAir;

void light\_select(GLenum which)

{

glEnable(which);

glLightfv(which, GL\_DIFFUSE, cahaya);

glutPostRedisplay();

}

void initRendering() {

glEnable(GL\_DEPTH\_TEST);

glEnable(GL\_COLOR\_MATERIAL);

glEnable(GL\_LIGHTING);

//glEnable(GL\_LIGHT0);

glEnable(GL\_NORMALIZE);

glShadeModel(GL\_SMOOTH);

}

void drawSceneTerra(Terrain \*terrain, GLfloat r, GLfloat g, GLfloat b) {

float scale = 500.0f / max(terrain->width() - 1, terrain->length() - 1);

glScalef(scale, scale, scale);

glTranslatef(-(float) (terrain->width() - 1) / 2, 0.0f,

-(float) (terrain->length() - 1) / 2);

glColor3f(r, g, b);

for (int z = 0; z < terrain->length() - 1; z++) {

//Makes OpenGL draw a triangle at every three consecutive vertices

glBegin(GL\_TRIANGLE\_STRIP);

for (int x = 0; x < terrain->width(); x++) {

Vec3f normal = terrain->getNormal(x, z);

glNormal3f(normal[0], normal[1], normal[2]);

glVertex3f(x, terrain->getHeight(x, z), z);

normal = terrain->getNormal(x, z + 1);

glNormal3f(normal[0], normal[1], normal[2]);

glVertex3f(x, terrain->getHeight(x, z + 1), z + 1);

}

glEnd();

}

}

//Load texture BMP

int load\_texture(char \*file\_name, unsigned int width, unsigned int height){

GLubyte \*imgbitmap;

FILE \*file;

unsigned short int depth=3;

if ((file = fopen(file\_name, "rb"))==NULL){

printf("File tidak ditemukan: %s!\n",file\_name);

exit(1);

}

imgbitmap = (GLubyte \*) malloc (width \* height \* depth \* (sizeof(GLubyte)));

if (imgbitmap == NULL){

printf("Alokasi textures ke Memory gagal!\n");

fclose(file);

exit(1);

}

fread(imgbitmap, width \* height \* depth, 1, file);

fclose(file);

glGenTextures(1, &texture\_id);

glBindTexture(GL\_TEXTURE\_2D, texture\_id);

glPixelStorei(GL\_UNPACK\_ALIGNMENT,1);

//menyesuaikan ukuran textur ketika image lebih besar dari texture

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MAG\_FILTER, GL\_NEAREST); //

//menyesuaikan ukuran textur ketika image lebih kecil dari texture

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_MIN\_FILTER, GL\_NEAREST); //

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_S, GL\_REPEAT);

glTexParameteri(GL\_TEXTURE\_2D, GL\_TEXTURE\_WRAP\_T, GL\_REPEAT);

glTexEnvi(GL\_TEXTURE\_ENV, GL\_TEXTURE\_ENV\_MODE, GL\_REPLACE);

gluBuild2DMipmaps(GL\_TEXTURE\_2D, GL\_RGB8, width, height, GL\_BGR, GL\_UNSIGNED\_BYTE, imgbitmap);

glTexImage2D(GL\_TEXTURE\_2D, 0, GL\_RGB8, width, height, 0, GL\_BGR, GL\_UNSIGNED\_BYTE, imgbitmap);

free(imgbitmap);

return texture\_id;

}

void init(void) {

myView.positionCamera(0.0,30.0,-170.0, 0.0,0.0,0.0, 0.0,1.0,0.0);

glEnable(GL\_DEPTH\_TEST);

glDepthFunc(GL\_LESS);

glEnable(GL\_NORMALIZE);

glEnable(GL\_COLOR\_MATERIAL);

glDepthFunc(GL\_LEQUAL);

glShadeModel(GL\_SMOOTH);

glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT, GL\_NICEST);

glEnable(GL\_CULL\_FACE);

initRendering();

\_terrainRumput = loadTerrain("heightmap.bmp", 20);

\_terrainTanah = loadTerrain("heightmapTanah.bmp", 20);

\_terrainAir = loadTerrain("heightmapAir.bmp", 20);

glLightfv(GL\_LIGHT0, GL\_POSITION, cahaya\_kiri);

glLightfv(GL\_LIGHT1, GL\_POSITION, cahaya\_kanan);

glLightfv(GL\_LIGHT0, GL\_SPECULAR, cahaya);

glLightfv(GL\_LIGHT1, GL\_SPECULAR, cahaya);

light\_select(GL\_LIGHT0);

light\_select(GL\_LIGHT1);

langit =load\_texture("nigth1.bmp", 251, 201);

glEnable(GL\_TEXTURE\_2D);

}

float skala=15;//skala vila

void allmodel(){

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.0, 4.22, 12.5);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1,1.3,0.8); // villa utama

glColor3f(1.5,0.5,1.5);

glutSolidCube(6.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.0, 2.1, 10);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(-1.3,1.8,0.05);// Jendela kanan Border bawah

glColor3f(0,0,0);

glutSolidCube(1.7);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.0, 2.1, 10);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1,1.4,0.08);

glColor3f(0.3, 0.2, 0.1);// Jendela kanan bawah @zietuck

glutSolidCube(1.9);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.3, 2, 11.4);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1,1.3,0.1);

glColor3f(0.7, 0.7, 1);// Jendela kiri bawah

glutSolidCube(1.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.3, 2, 11.5);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(-1.3,1.8,0.1);

glColor3f(0,0,0);// Jendela kiri Border

glutSolidCube(1.3);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.3, 6, 11.4);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,1.0,0.1);

glColor3f(0.7, 0.7, 1);// Jendela kiri Atas

glutSolidCube(1.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.3, 6, 11.7);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1,1.0,0.3);

glColor3f(0,0,0);// Jendela Border kiri Atas

glutSolidCube(1.8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.1, 6, 10);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,1.0,0.03);

glColor3f(0.7, 0.7, 1);// Jendela kanan Atas

glutSolidCube(1.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.1, 6, 10.1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,1.0,0.05);

glColor3f(0,0,0);// Jendela Border kanan Atas

glutSolidCube(1.8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.35, 4.3, 13.24);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,2.2,0.95); // villa kiri

glColor3f(1.5,1.5,0.5);

glutSolidCube(3.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-2.2, 2.8, 8.0);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.5,1.8,0.2); // villa penyangga depan kiri

glColor3f(0,1,1);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.95, 6.5, 10.9);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.2,1.0,0.5); // villa penyangga samping kiri atas

glColor3f(0.5,0.5,1.5);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.95, 2.5, 10.9);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.2,1.37,0.5); // villa penyangga samping kiri bawah

glColor3f(0.5,0.5,1.5);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(2.2, 2.8, 8.0);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.5,1.8,0.2); // villa penyangga depan kanan

glColor3f(0,1,1);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.0, 5.3, 8.0);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(2.0,0.7,0.2); // villa penyangga depan atas

glColor3f(0,1,1);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(0.0, 4.8,9.2);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(2.0,0.3,0.6); // villa atap depan

glColor3f(0.5,0.5,0.5);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.6, 7.7,10.9);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,0.3,0.5); // villa atap kiri atas

glColor3f(0.5,0.5,1.5);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.6, 4.8,10.9);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(1.1,0.3,0.5); // villa atap kiri bawah

glColor3f(0.5,0.5,1.5);

glutSolidCube(3.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.7, 0.6, 4.0);

//glCallList(8);

glRotatef(-0.5, 0.0, 1.0, 0.0);

glScalef(0.8,0.05,1.5);

glColor3f(0.2,0.2,0.2);// tempat berjemur

glutSolidCube(3.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.2, 1, 4.5);

//glCallList(13);

glRotatef(-0.5, 0.0, 1.0, 0.0);

glScalef(0.8,0.05,1.5);

glColor3f(0,0.2,0.2);// kursi 1 kanan

glutSolidCube(0.6);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.2, 1, 5.0);

//glCallList(13);

glRotatef(-40, 1, 0, 0);

glScalef(0.8,0.05,1.5);

glColor3f(0.5,0,0.2);// kursi sandaran kanan

glutSolidCube(0.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.2, 1, 5.0);

//glCallList(13);

glRotatef(-40, 1, 0, 0);

glScalef(0.8,0.05,1.5);

glColor3f(0.5,0,0.2);// kursi sandaran kiri

glutSolidCube(0.5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.2, 1, 4.5);

//glCallList(13);

glRotatef(-0.5, 0.0, 1.0, 0.0);

glScalef(0.8,0.05,1.5);

glColor3f(0,0.2,0.2);// kursi 2 kiri

glutSolidCube(0.6);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.7, 1.8, 5.0);

//glCallList(17); // payung

glScalef(0.9,0.5,0.5);

glRotated(-90,1,0,0);

glColor3f(0.0,0,1);

glutSolidCone(0.8, 1.0, 10, 5);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.7, 1.8, 5.0);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.3,0.2,0.2); // Tiang Payung

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0,0,0);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.6, 1, 2.8);

//glCallList(18);

glRotatef(-0.5, 0.0, 1.0, 0.0);

glScalef(0.8,0.05,1.3);

glColor3f(0.3, 0.2, 0.1);// meja

glutSolidCube(0.6);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.8, 1.0, 3.1);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki meja kiri bagian belakang sebelah kiri

qobj = gluNewQuadric();

glColor3f(0.3, 0.2, 0.1);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.8, 1.0, 2.5);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki meja kiri bagian depan sebelah kiri

qobj = gluNewQuadric();

glColor3f(0.3, 0.2, 0.1);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.5, 1.0, 3.1);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki meja kanan bagian belakang sebelah kanan

qobj = gluNewQuadric();

glColor3f(0.3, 0.2, 0.1);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.5, 1.0, 2.5);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki meja kanan bagian depan sebelah kanan

qobj = gluNewQuadric();

glColor3f(0.3, 0.2, 0.1);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.0, 1.0, 4.2);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian depan sebelah kanan

qobj = gluNewQuadric();

glColor3f(0, 0, 0);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.4, 1.0, 4.2);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian depan sebelah kiri

qobj = gluNewQuadric();

glColor3f(0, 0, 0);

gluQuadricDrawStyle(qobj,GLU\_LINE);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.0, 1.0, 5.0);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian belakang sebelah kanan

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0, 0, 0);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-5.4, 1.0, 5.0);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kiri bagian belakang sebelah kiri

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0, 0, 0);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.0, 1.0, 4.2);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian depan sebelah kanan

qobj = gluNewQuadric();

glColor3f(0, 0, 0);

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0.3, 0.2, 0.1);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.4, 1.0, 4.2);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian depan sebelah kiri

qobj = gluNewQuadric();

glColor3f(0, 0, 0);

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0.3, 0.2, 0.1);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.0, 1.0, 5.0);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2);// kaki kursi kanan bagian belakng sebelah kanan

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0, 0, 0);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-4.4, 1.0, 5.0);

//glCallList(15);

glRotatef(90, 1.0, 0.0, 0.0);

glScalef(0.2,0.2,0.2); // kaki kursi kanan bagian belakng sebelah kiri

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj,GLU\_LINE);

glColor3f(0, 0, 0);

gluCylinder(qobj,0.2,0.2,5.0,50,8);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(-1.8, 0.6, 4.0);

//glCallList(11);

glRotatef(-0.5, 0.0, 1.0, 0.0);

glScalef(2.2,0.02,1.5);

glColor3f(0.5, 0.5, 0.5);// alas

glutSolidCube(4.3);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(7.8, 2.0, 8.0);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.6,1.8,0.2);

glColor3f(0.1, 0.3, 0.3);// villa penyangga depan kanan garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(7.8, 2.0, 15.2);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.6,1.8,0.2);

glColor3f(0.1, 0.3, 0.3);// villa penyangga belakang kanan garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(4.1, 2.0, 8.0);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.5,1.8,0.2);

glColor3f(0.1, 0.3, 0.3);// villa penyangga samping kiri garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(4.1, 2.0, 15.1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.5,1.8,0.2);

glColor3f(0.1, 0.3, 0.3);// villa penyangga belakang kiri garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(6.0, 2.0, 15.1);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(0.5,1.8,0.2);

glColor3f(0.1, 0.3, 0.3);// villa penutup belakang garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glScalef(skala,skala,skala);

glPushMatrix();

glTranslatef(6.0, 4.0, 11.6);

//glCallList(1);

glRotatef(0, 0.0, 1.0, 0.0);

glScalef(2.4,0.5,3.8); // villa penyangga depan atas garasi

glutSolidCube(2.0);

glPopMatrix();

glPopMatrix();

glPopMatrix();

}

float rbolalampu =1.0 ,gbolalampu = 1.0 ,bbolalampu =0.0;

void lampuvila(){

glPushMatrix();

glPushMatrix();

glColor3f(0.3, 0.2, 0.1); //glColor3f(0.4, 0.8, 0.1);-->warna hijau

glTranslatef(0.1, 0.8, 0.0);

glScalef(0.1, 2.40, 0.1);

glutSolidCube(0.5);

glPopMatrix();

//++++++++++++++++dasar lampu++++++++++++++

glPushMatrix();

glColor3f(180, 180, 179);

glTranslatef(0.1, 0.1, 0.0);

glScalef(0.3, 0.04, 0.3);

glutSolidSphere(0.5, 30, 30);

glPopMatrix();

//++++++++++++++++penyangga lampu++++++++++++++

glPushMatrix();

glColor3f(0.1, 0.1, 0.1);

glTranslatef(0.1, 0.35, 0.0);

glScalef(0.2, 0.99, 0.2);

glutSolidCube(0.5);

glPopMatrix();

//++++++++++++++++penahan lampu1++++++++++++++

glPushMatrix();

glColor3f(0.3, 0.2, 0.1);

glTranslatef(0.35, 1.28, 0.0);

glScalef(1.5, 0.1, 0.1);

glutSolidCube(0.5);

glPopMatrix();

//++++++++++++++++penahan lampu2++++++++++++++

glPushMatrix();

glColor3f(0.3, 0.2, 0.1);

glTranslatef(0.20, 1.15, 0.0);

glScalef(0.7, 0.1, 0.1);

glutSolidCube(0.5);

glPopMatrix();

//++++++++++++++++penahan lampu3++++++++++++++

glPushMatrix();

glColor3f(0.3, 0.2, 0.1);

glTranslatef(0.20, 1.20, 0.0);

glScalef(0.1, 0.5, 0.1);

glutSolidCube(0.5);

glPopMatrix();

//++++++++++++++++atap lampu++++++++++++++

glPushMatrix();

glColor3f(0.5, 0.3, 0.2);

glTranslatef(0.75, 1.23, 0.0);

glScalef(0.3, 0.04, 0.3);

glutSolidSphere(0.8, 210, 10);

glPopMatrix();

//++++++++++++++paku penahan lampu+++++++++++

glPushMatrix();

glColor3f(180, 180, 179);

glTranslatef(0.75, 1.28, 0.0);

glScalef(0.3, 0.3, 0.3);

glutSolidSphere(0.10, 10, 10);

glPopMatrix();

//++++++++++++++bola lampu+++++++++++

glPushMatrix();

glColor3f(rbolalampu, gbolalampu, bbolalampu);

glTranslatef(0.75, 1.15, 0.0);

glScalef(0.8, 0.8, 0.8);

glutSolidSphere(0.10, 20, 20);

glPopMatrix();

glPopMatrix();

}

void pohon(){

glPushMatrix();

glPushMatrix();

glRotatef(-90,1.0,0.0,0.0);

glColor3f(0.5, 0.3, 0.2);

glutSolidCone(4,20,100,100);

glPopMatrix();

glPushMatrix();

glTranslatef(0.0,10.0,0.0);

glPushMatrix();

glRotatef(-90,1.0,0.0,0.0);

glColor3f(0.6, 1, 0.6);

glutSolidCone(9,20,100,10);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glTranslatef(0.0,15.0,0.0);

glPushMatrix();

glRotatef(-90,1.0,0.0,0.0);

glColor3f(0.5, 1, 0.5);

glutSolidCone(8,20,100,10);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glTranslatef(0.0,20.0,0.0);

glPushMatrix();

glRotatef(-90,1.0,0.0,0.0);

glColor3f(0.4, 1, 0.4);

glutSolidCone(7,20,100,10);

glPopMatrix();

glPopMatrix();

glPopMatrix();

}

void pepohonan(){

glPushMatrix();

glTranslatef(100,-5,-100);

glScalef(2,2,2);

pohon();

glPopMatrix();

glPushMatrix();

glTranslatef(65,-5,-20);

glScalef(1.5,1.5,1.5);

pohon();

glPopMatrix();

glPushMatrix();

glTranslatef(135,-5,10);

glScalef(1.5,1.5,1.5);

pohon();

glPopMatrix();

glPushMatrix();

glTranslatef(-110,-5,-80);

glScalef(1.7,1.7,1.7);

pohon();

glPopMatrix();

glPushMatrix();

glTranslatef(-120,-5,-20);

glScalef(1.4,1.4,1.4);

pohon();

glPopMatrix();

glPushMatrix();

glTranslatef(-130,-5,20);

glScalef(1.4,1.4,1.4);

pohon();

glPopMatrix();

}

void display(void) {

glClearColor(1.0, 1.0, 1.0, 1);

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT | GL\_STENCIL\_BUFFER\_BIT); //clear the buffers

glLoadIdentity();

myView.look();//camera

///\*

GLfloat lightColor1[] = {1.0f, 1.0f, 1.0f, 1.0f };//0.0f, 0.0f, -8.0f

GLfloat lightPos1[] = {-80, 30, 0.0,1.0};

glLightfv(GL\_LIGHT2, GL\_DIFFUSE, lightColor1);

glMaterialf(GL\_LIGHT2,GL\_SHININESS,4.0);

glLightfv(GL\_LIGHT2, GL\_POSITION, lightPos1);

glPushMatrix();

drawSceneTerra(\_terrainRumput, 0.3f, 0.9f, 0.0f);

glPopMatrix();

glPushMatrix();

drawSceneTerra(\_terrainTanah, 0.7f, 0.2f, 0.1f);

glPopMatrix();

glPushMatrix();

drawSceneTerra(\_terrainAir, 0.0f, 0.2f, 0.5f);

glPopMatrix();//\*/

glPushMatrix();

glTranslatef(0.0,-10.0,-90.0);

allmodel();

glPopMatrix();

glPushMatrix();

glTranslatef(-100.0,-10.0,0);

glScalef(30,40,30);

lampuvila();

glPopMatrix();

glPushMatrix();

pepohonan();

glPopMatrix();

glPushMatrix();

glTranslatef(-250,-20,-185);

//glScalef(500,500,500);

glPushMatrix();

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,langit);

//glColor3f(0,0,0);

glBegin(GL\_QUADS);

glTexCoord2f(0,0);glVertex3f(0,0,0);

glTexCoord2f(1,0);glVertex3f(500,0,0);

glTexCoord2f(1,1);glVertex3f(500,250,0);

glTexCoord2f(0,1);glVertex3f(0,250,0);

glEnd();

glDisable(GL\_TEXTURE\_2D);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glRotatef(180,0.0,1.0,0.0);

glTranslatef(-250,-20,-185);

//glScalef(500,500,500);

glPushMatrix();

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,langit);

glColor3f(0,0,0);

glBegin(GL\_QUADS);

glTexCoord2f(0,0);glVertex3f(0,0,0);

glTexCoord2f(1,0);glVertex3f(500,0,0);

glTexCoord2f(1,1);glVertex3f(500,250,0);

glTexCoord2f(0,1);glVertex3f(0,250,0);

glEnd();

glDisable(GL\_TEXTURE\_2D);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glRotatef(90,0.0,1.0,0.0);

glTranslatef(-250,-20,-185);

//glScalef(500,500,500);

glPushMatrix();

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,langit);

glColor3f(0,0,0);

glBegin(GL\_QUADS);

glTexCoord2f(0,0);glVertex3f(0,0,0);

glTexCoord2f(1,0);glVertex3f(500,0,0);

glTexCoord2f(1,1);glVertex3f(500,250,0);

glTexCoord2f(0,1);glVertex3f(0,250,0);

glEnd();

glDisable(GL\_TEXTURE\_2D);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glRotatef(270,0.0,1.0,0.0);

glTranslatef(-250,-20,-185);

//glScalef(500,500,500);

glPushMatrix();

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,langit);

glColor3f(0,0,0);

glBegin(GL\_QUADS);

glTexCoord2f(0,0);glVertex3f(0,0,0);

glTexCoord2f(1,0);glVertex3f(500,0,0);

glTexCoord2f(1,1);glVertex3f(500,250,0);

glTexCoord2f(0,1);glVertex3f(0,250,0);

glEnd();

glDisable(GL\_TEXTURE\_2D);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glRotatef(90,1.0,0.0,0.0);

glTranslatef(-180,-190,-190);

//glScalef(500,500,500);

glPushMatrix();

glEnable(GL\_TEXTURE\_2D);

glBindTexture(GL\_TEXTURE\_2D,langit);

glColor3f(0,0,0);

glBegin(GL\_QUADS);

glTexCoord2f(0,0);glVertex3f(0,0,0);

glTexCoord2f(1,0);glVertex3f(500,0,0);

glTexCoord2f(1,1);glVertex3f(500,500,0);

glTexCoord2f(0,1);glVertex3f(0,500,0);

glEnd();

glDisable(GL\_TEXTURE\_2D);

glPopMatrix();

glPopMatrix();

glutSwapBuffers();

glFlush();

}

void reshape(int w, int h) {

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

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(40, (GLfloat) w / (GLfloat) h, 0.1, 1000.0);

glMatrixMode(GL\_MODELVIEW);

}

void KeyBoard(unsigned char key, int x, int y)

{

switch (key)

{

case '1':

//menentukan koordinat camera

printf("%f %f %f \n\tat %f %f %f \n\tup %f %f %f",

myView.position().x,myView.position().y,myView.position().z,

myView.view().x,myView.view().y,myView.view().z,

myView.upVector().x,myView.upVector().y,myView.upVector().z);

break;

case '2':

bbolalampu += 0.4;

glEnable(GL\_LIGHT2);

break;

case '3':

bbolalampu -= 0.4;

glDisable(GL\_LIGHT2);

break;

case 'w': //move forward

myView.move(speed);

break;

case 's': //move back

myView.move(-speed);

break;

case 'a': //strafe left

myView.strafe(-speed);

break;

case 'd': //strafe right

myView.strafe(speed);

break;

case 27: // “esc” on keyboard

exit(0);

break;

}

glutPostRedisplay();

}

void specialKeyHandler(int key, int x, int y)

{

switch(key)

{

case GLUT\_KEY\_UP:

myView.pitch(10);

break;

case GLUT\_KEY\_DOWN:

myView.pitch(-10);

break;

case GLUT\_KEY\_LEFT:

myView.turn(10);

break;

case GLUT\_KEY\_RIGHT:

myView.turn(-10);

break;

}

glutPostRedisplay();

}

void mouseButtonHandler(int button, int state, int x, int y)

{

if(button == GLUT\_LEFT\_BUTTON) {

if(state == GLUT\_DOWN) {

xPrev = x;

yPrev = y;

dragging = true;

}

else //state == GLUT\_UP

dragging = false;

}

glutPostRedisplay();

}

void mouseMotionHandler(int x, int y){

int xDiff, yDiff;

if(dragging) {

xDiff = x - xPrev;

yDiff = y - yPrev;

if(xDiff != 0)

myView.turn(-xDiff);

if(yDiff != 0)

myView.pitch(-yDiff);

xPrev = x;

yPrev = y;

glutPostRedisplay();

}

glutPostRedisplay();

}

int main(int argc, char \*\*argv) {

quadObj = gluNewQuadric();

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGBA | GLUT\_STENCIL | GLUT\_DEPTH); //add a stencil buffer to the window

glutInitWindowSize(800, 600);

glutInitWindowPosition(100, 100);

glutCreateWindow("Tugas Besar Grafika Komputer Kelompok 9 Kelas IF-9 ");

glutFullScreen();

init();

glutDisplayFunc(display);

glutIdleFunc(display);

glutReshapeFunc(reshape);

glutSpecialFunc(specialKeyHandler);

glutKeyboardFunc(KeyBoard);

glutMouseFunc(mouseButtonHandler);

glutMotionFunc(mouseMotionHandler);

glutMainLoop();

return 0;

}