**TUGAS BESAR**

**RUANG KAMAR 3D DENGAN OPENGL**

Diajukan Untuk Memenuhi

Salah Satu Tugas Mata Kuliah Komputer Grafika

Dosen : Hendri Karisma, S. Kom.



Disusun Oleh : 10108389 – Diki Taurens Sia

10108400 – Sopyan

10108405 – Andika Tanjung J.

10108407 – M. Arif Ansyari

Kelas : IF-8/S1/VIII

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

**2012**

**SOURCE CODE**

#include "stdafx.h"

#include <windows.h>

#include <stdio.h>

#include <GL\glut.h>

#include <GL\GL.h>

#include <GL\GLU.h>

#include <math.h>

GLuint texture[5];

GLint slices = 16;

GLint stacks = 16;

static float ypoz = 0, zpoz = 0, xpoz = 0,a = 5, b = -5,c = -7, rPintu=0,

jKiri=0, jKanan=0, lampu=45, lemari1=0, lemari2=0;

int z=0;

struct Image {

unsigned long sizeX;

unsigned long sizeY;

char \*data;

};

typedef struct Image Image; //struktur data untuk

//ukuran image #bisa di set sesuai kebutuhan

#define checkImageWidth 164

#define checkImageHeight 164

GLubyte checkImage[checkImageWidth][checkImageHeight][3];

//mengambil gambar BMP

int ImageLoad(char \*filename, Image \*image) {

FILE \*file;

unsigned long size; // ukuran image dalam bytes

unsigned long i; // standard counter.

unsigned short int plane; // number of planes in image

unsigned short int bpp; // jumlah bits per pixel

char temp; // temporary color storage for var warna sementara untuk memastikan filenya ada

if ((file = fopen(filename, "rb")) == NULL) {

printf("File Not Found : %s\n", filename);

return 0;

}

// mencari file header bmp

fseek(file, 18, SEEK\_CUR);

// read the width

if ((i = fread(&image->sizeX, 4, 1, file)) != 1) {

printf("Error reading width from %s.\n", filename);

return 0;

}

//printf("Width of %s: %lu\n", filename, image->sizeX);

// membaca nilai height

if ((i = fread(&image->sizeY, 4, 1, file)) != 1) {

printf("Error reading height from %s.\n", filename);

return 0;

}

//printf("Height of %s: %lu\n", filename, image->sizeY);

//menghitung ukuran image(asumsi 24 bits or 3 bytes per pixel).

size = image->sizeX \* image->sizeY \* 3;

// read the planes

if ((fread(&plane, 2, 1, file)) != 1) {

printf("Error reading planes from %s.\n", filename);

return 0;

}

if (plane != 1) {

printf("Planes from %s is not 1: %u\n", filename, plane);

return 0;

}

// read the bitsperpixel

if ((i = fread(&bpp, 2, 1, file)) != 1) {

printf("Error reading bpp from %s.\n", filename);

return 0;

}

if (bpp != 24) {

printf("Bpp from %s is not 24: %u\n", filename, bpp);

return 0;

}

// seek past the rest of the bitmap header.

fseek(file, 24, SEEK\_CUR);

// read the data.

image->data = (char \*) malloc(size);

if (image->data == NULL) {

printf("Error allocating memory for color-corrected image data");

return 0;

}

if ((i = fread(image->data, size, 1, file)) != 1) {

printf("Error reading image data from %s.\n", filename);

return 0;

}

for (i = 0; i < size; i += 3) { // membalikan semuan nilai warna (gbr - > rgb)

temp = image->data[i];

image->data[i] = image->data[i + 2];

image->data[i + 2] = temp;

}

// we're done.

return 1;

}

//mengambil tekstur

Image \* loadTexture() {

Image \*image1;

// alokasi memmory untuk tekstur

image1 = (Image \*) malloc(sizeof(Image));

if (image1 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("zebra.bmp", image1)) {

exit(1);

}

return image1;

}

Image \* loadTexture2() {

Image \*image2;

// alokasi memmory untuk tekstur

image2 = (Image \*) malloc(sizeof(Image));

if (image2 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("snsd1.bmp", image2)) {

exit(1);

}

return image2;

}

Image \* loadTexture3() {

Image \*image3;

// alokasi memmory untuk tekstur

image3 = (Image \*) malloc(sizeof(Image));

if (image3 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("wiwid.bmp", image3)) {

exit(1);

}

return image3;

}

Image \* loadTexture4() {

Image \*image4;

// alokasi memmory untuk tekstur

image4 = (Image \*) malloc(sizeof(Image));

if (image4 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("olimpik.bmp", image4)) {

exit(1);

}

return image4;

}

Image \* loadTexture5() {

Image \*image5;

// alokasi memmory untuk tekstur

image5 = (Image \*) malloc(sizeof(Image));

if (image5 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("pemandangan.bmp", image5)) {

exit(1);

}

return image5;

}

Image \* loadTexture6() {

Image \*image6;

// alokasi memmory untuk tekstur

image6 = (Image \*) malloc(sizeof(Image));

if (image6 == NULL) {

printf("Error allocating space for image");

exit(0);

}

//pic.bmp is a 64x64 picture

if (!ImageLoad("black.bmp", image6)) {

exit(1);

}

return image6;

}

void myinit(void) {

glClearColor(0.5, 0.5, 0.5, 0.0);

glEnable(GL\_DEPTH\_TEST);

glDepthFunc(GL\_LESS);

Image \*image1 = loadTexture();

Image \*image2 = loadTexture2();

Image \*image3 = loadTexture3();

Image \*image4 = loadTexture4();

// Image \*image5 = loadTexture5();

Image \*image6 = loadTexture6();

if (image1 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

}

if (image2 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

}

if (image3 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

}

if (image4 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

}

/\* if (image5 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

} \*/

if (image6 == NULL) {

printf("Image was not returned from loadTexture\n");

exit(0);

}

// Generate texture/ membuat texture

glGenTextures(5,texture);

//binding texture untuk membuat texture 2D

glBindTexture(GL\_TEXTURE\_2D, texture[0]);

//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);

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, image1->sizeX, image1->sizeY, 0, GL\_RGB,

GL\_UNSIGNED\_BYTE, image1->data);

//tekstur air

//binding texture untuk membuat texture 2D

glBindTexture(GL\_TEXTURE\_2D, texture[1]);

//menyesuaikan ukuran textur ketika image lebih besar dari texture

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

//menyesuaikan ukuran textur ketika image lebih kecil dari texture

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

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, image2->sizeX, image2->sizeY, 0, GL\_RGB,

GL\_UNSIGNED\_BYTE, image2->data);

//tekstur 3

//binding texture untuk membuat texture 2D

glBindTexture(GL\_TEXTURE\_2D, texture[2]);

//menyesuaikan ukuran textur ketika image lebih besar dari texture

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

//menyesuaikan ukuran textur ketika image lebih kecil dari texture

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

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, image3->sizeX, image3->sizeY, 0, GL\_RGB,

GL\_UNSIGNED\_BYTE, image3->data);

//tekstur olimpik

//binding texture untuk membuat texture 2D

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

//menyesuaikan ukuran textur ketika image lebih besar dari texture

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

//menyesuaikan ukuran textur ketika image lebih kecil dari texture

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

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, image4->sizeX, image4->sizeY, 0, GL\_RGB,

GL\_UNSIGNED\_BYTE, image4->data);

//tekstur hitam

//binding texture untuk membuat texture 2D

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

//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); //

glTexImage2D(GL\_TEXTURE\_2D, 0, 3, image6->sizeX, image6->sizeY, 0, GL\_RGB,

GL\_UNSIGNED\_BYTE, image6->data);

glEnable(GL\_TEXTURE\_2D);

glShadeModel(GL\_FLAT);

}

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

{

if(state == GLUT\_DOWN) //ketika mouse ditekan

{

if(button == GLUT\_LEFT\_BUTTON) //klik kiri

{

if (rPintu <0)

{

rPintu=0; //pintu tertutup

}else

rPintu=-100; //pintu terbuka

glutPostRedisplay();

}

else if(button == GLUT\_MIDDLE\_BUTTON) //klik tengah mouse

{}

else if(button == GLUT\_RIGHT\_BUTTON) //klik kanan

{}

}

else //ketika mouse tidak ditekan atau dilepas

{

if(button == GLUT\_LEFT\_BUTTON)

{}

else if(button == GLUT\_MIDDLE\_BUTTON)

{}

else if(button == GLUT\_RIGHT\_BUTTON)

{}

}

}

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

{

switch (key)

{

//putar arah jarum jam (menambah derajat putaran)

case 'z':

ypoz=ypoz+5;

if (ypoz>360) ypoz=0;

glutPostRedisplay();

break;

//putar berlawanan arah jarum jam (mengurangi derajat putaran)

case 'x':

ypoz=ypoz-5;

if (ypoz>360) ypoz=0;

glutPostRedisplay();

break;

//vertikal atas (mengurangi koordinat y)

case 'w':

b = b + 1;

if (b>-2) b=-2;

glutPostRedisplay();

break;

//vertikal bawah (menambah koordinat y)

case 's':

b = b - 1;

if (b<-12) b=-12;

glutPostRedisplay();

break;

//horisontal kiri (mengurangi koordinat x)

case 'a':

a = a + 1;

if (a>15) a=15;

glutPostRedisplay();

break;

//horisontal kanan (menambah koordinat x)

case 'd':

a = a - 1;

if (a<-20) a=-20;

glutPostRedisplay();

break;

//memperbesar objek (menambah koordinat z)

case 'q':

c = c + 1;

if (c>15) c=15;

glutPostRedisplay();

break;

//memperkecil abjek(mengurangi koordinat z)

case 'e':

c = c - 1;

if (c<-10) c=-10;

glutPostRedisplay();

break;

//buka pintu

case '7':

rPintu=rPintu-5;

if (rPintu<-100) rPintu=-100;

glutPostRedisplay();

break;

//tutup pintu

case '9':

rPintu=rPintu+5;

if (rPintu>0) rPintu=0;

glutPostRedisplay();

break;

//buka jendela kiri

case '4':

jKiri=jKiri+5;

if (jKiri>75) jKiri=75;

glutPostRedisplay();

break;

//tutup jendela kiri

case '6':

jKiri=jKiri-5;

if (jKiri<0) jKiri=0;

glutPostRedisplay();

break;

//buka jendela kanan

case '1':

jKanan=jKanan-5;

if (jKanan<-75) jKanan=-75;

glutPostRedisplay();

break;

//tutup jendela kanan

case '3':

jKanan=jKanan+5;

if (jKanan>0) jKanan=0;

glutPostRedisplay();

break;

//buka pintu lemari kiri

case 'n':

lemari1=lemari1-5;

if (lemari1<-75) lemari1=-75;

glutPostRedisplay();

break;

//tutup pintu lemari kiri

case 'm':

lemari1=lemari1+5;

if (lemari1>0) lemari1=0;

glutPostRedisplay();

break;

//buka pintu lemari kanan

case 'k':

lemari2=lemari2+5;

if (lemari2>75) lemari2=75;

glutPostRedisplay();

break;

//tutup pintu lemari kanan

case 'l':

lemari2=lemari2-5;

if (lemari2<0) lemari2=0;

glutPostRedisplay();

break;

//menyalakan dan mematikan lampu

case '0':

if (lampu==45)

{

lampu=-45;

}

else if(lampu==-45)

{

lampu=45;

}

glutPostRedisplay();

break;

}

}

void dinding(float x1,float y1,float z1,float x2,float y2,float z2)

{

//depan

glTexCoord2f(0.0, 0.0);

glVertex3f(x1,y1,z1);

glTexCoord2f(0.0, 1.0);

glVertex3f(x2,y1,z1);

glTexCoord2f(1.0, 1.0);

glVertex3f(x2,y2,z1);

glTexCoord2f(1.0, 0.0);

glVertex3f(x1,y2,z1);

//atas

glTexCoord2f(0.0, 0.0);

glVertex3f(x1,y2,z1);

glTexCoord2f(0.0, 1.0);

glVertex3f(x2,y2,z1);

glTexCoord2f(1.0, 1.0);

glVertex3f(x2,y2,z2);

glTexCoord2f(1.0, 0.0);

glVertex3f(x1,y2,z2);

//belakang

glTexCoord2f(0.0, 0.0);

glVertex3f(x1,y2,z2);

glTexCoord2f(0.0, 1.0);

glVertex3f(x2,y2,z2);

glTexCoord2f(1.0, 1.0);

glVertex3f(x2,y1,z2);

glTexCoord2f(1.0, 0.0);

glVertex3f(x1,y1,z2);

//bawah

glTexCoord2f(0.0, 0.0);

glVertex3f(x1,y1,z2);

glTexCoord2f(1.0, 0.0);

glVertex3f(x2,y1,z2);

glTexCoord2f(1.0, 1.0);

glVertex3f(x2,y1,z1);

glTexCoord2f(0.0, 1.0);

glVertex3f(x1,y1,z1);

//samping kiri

glTexCoord2f(0.0, 0.0);

glVertex3f(x1,y1,z1);

glTexCoord2f(1.0, 0.0);

glVertex3f(x1,y2,z1);

glTexCoord2f(1.0, 1.0);

glVertex3f(x1,y2,z2);

glTexCoord2f(0.0, 1.0);

glVertex3f(x1,y1,z2);

//samping kanan

glTexCoord2f(0.0, 0.0);

glVertex3f(x2,y1,z1);

glTexCoord2f(1.0, 0.0);

glVertex3f(x2,y2,z1);

glTexCoord2f(1.0, 1.0);

glVertex3f(x2,y2,z2);

glTexCoord2f(0.0, 1.0);

glVertex3f(x2,y1,z2);

}

void timer (int value){

if (z <= 360){

z +=15;

}

if (z == 360){

z =0;

}

glutPostRedisplay();

glutTimerFunc(5,timer,0);

}

void display(void) {

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity ();

glEnable(GL\_DEPTH\_TEST);

//perpindahan

glTranslatef(a,b,c);

//putaran

glRotatef(xpoz,1,0,0);

glRotatef(ypoz,0,1,0);

glRotatef(zpoz,0,0,1);

//dinding kiri

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

dinding(-10,0,0,-9,15,-20);

glEnd();

glPopMatrix();

//dinding kanan

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

dinding(17,0,0,18,15,-20);

glEnd();

glPopMatrix();

//dinding belakang

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

//bawah

glBegin(GL\_QUADS);

dinding(-10,0,-20,18,7,-21);

glEnd();

//atas

glBegin(GL\_QUADS);

dinding(-10,14,-20,18,15,-21);

glEnd();

//samping kiri

glBegin(GL\_QUADS);

dinding(-10,7,-20,-1,14,-21);

glEnd();

//samping kanan

glBegin(GL\_QUADS);

dinding(7,7,-20,18,14,-21);

glEnd();

glPopMatrix();

//dinding depan

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

dinding(-3,0,0,18,15,-1);

glEnd();

glPopMatrix();

//lorong pintu

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

dinding(-9,13,0,-3,15,-1);

glTexCoord2f(0.0, 0.0);

glVertex3f(-9,0,0);

glTexCoord2f(1.0, 0.0);

glVertex3f(-3,0,0);

glTexCoord2f(1.0, 1.0);

glVertex3f(-3,0,-1);

glTexCoord2f(0.0, 1.0);

glVertex3f(-9,0,-1);

glEnd();

glPopMatrix();

//pintu

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(-9,0,0);

glRotated(rPintu,0,1,0);

glTranslated(9,0,0);

glBegin(GL\_QUADS);

dinding(-9,0,0,-3,13,-1);

glEnd();

glPopMatrix();

//jendela kiri

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glTranslated(-1,7,-21);

glRotated(jKiri,0,1,0);

glTranslated(1,-7,21);

glBegin(GL\_QUADS);

dinding(-1,7,-20,3,14,-21);

glEnd();

glPopMatrix();

//jendela kanan

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glTranslated(7,7,-21);

glRotated(jKanan,0,1,0);

glTranslated(-7,-7,21);

glBegin(GL\_QUADS);

dinding(7,7,-20,3,14,-21);

glEnd();

glPopMatrix();

//lantai

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

glTexCoord2f(0.0, 0.0);

glVertex3d(-9,0,-1);

glTexCoord2f(1.0, 0.0);

glVertex3d(17,0,-1);

glTexCoord2f(1.0, 1.0);

glVertex3d(17,0,-20);

glTexCoord2f(0.0, 1.0);

glVertex3d(-9,0,-20);

glEnd();

glPopMatrix();

//atap

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

glTexCoord2f(0.0, 0.0);

glVertex3d(-9,15,-1);

glTexCoord2f(1.0, 0.0);

glVertex3d(17,15,-1);

glTexCoord2f(1.0, 1.0);

glVertex3d(17,15,-20);

glTexCoord2f(0.0, 1.0);

glVertex3d(-9,15,-20);

glEnd();

glPopMatrix();

//stop kontak

glPushMatrix();

glTranslated(-8.85,8,-2);

glScalef(0.15,1,1);

glutSolidCube(1);

glPopMatrix();

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[3]);

glTranslated(-8.85,8,-2);

glRotated(lampu,0,0,1);

glScalef(0.1,0.5,0.2);

glutSolidCube(1);

glPopMatrix();

//kasur

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[0]);

glBegin(GL\_QUADS);

dinding(9.2,2,-4.2,16.8,3,-19);

glEnd();

glPopMatrix();

//tiang kasur

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glBegin(GL\_QUADS);

dinding(16.2,0,-19,16.8,1.5,-20);

dinding(16.2,0,-4,16.8,1.5,-5);

dinding(9,0,-19,9.8,1.5,-20);

dinding(9,0,-4,9.8,1.5,-5);

//alas

dinding(9,1.5,-4,16.8,2,-20);

//tempat sandar belakang

dinding(9.2,2,-19,16.8,5,-20);

glEnd();

glPopMatrix();

//bantal

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[0]);

glBegin(GL\_QUADS);

dinding(9.3,3,-16.5,13,3.5,-18.7);

dinding(13.3,3,-16.5,16.5,3.5,-18.7);

glEnd();

glPopMatrix();

//selimut

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glBegin(GL\_QUADS);

dinding(8.8,1,-4.2,9,3.2,-13);

dinding(9,3,-4.2,16.8,3.2,-13);

dinding(16.8,1,-4.2,17,3.2,-13);

glEnd();

glPopMatrix();

//lemari

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glBegin(GL\_QUADS);

//belakang

dinding(-9,0,-19.7,-2,11.5,-20);

//samping kiri

dinding(-9,0,-19.7,-8.7,11.5,-17);

//samping kanan

dinding(-2.3,0,-19.7,-2,11.5,-17);

//atas

dinding(-9,11.5,-17,-2,11.2,-20);

//dasar

dinding(-9,0,-17,-2,0.3,-20);

//sekat atas

dinding(-9,9,-17,-2,9.3,-20);

glEnd();

glPopMatrix();

//pintu lemari kiri

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glTranslated(-9,0,-17);

glRotated(lemari1,0,1,0);

glTranslated(9,0,17);

glBegin(GL\_QUADS);

//depan kiri

dinding(-9,0,-17,-5.5,11.5,-16.7);

glEnd();

glPopMatrix();

//pintu lemari kanan

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glTranslated(-2.5,0,-17);

glRotated(lemari2,0,1,0);

glTranslated(2.5,0,17);

glBegin(GL\_QUADS);

//depan kanan

dinding(-2,0,-17,-5.5,11.5,-16.7);

glEnd();

glPopMatrix();

//gantungan lemari

glPushMatrix();

glTranslated(-9, 8 ,-18);

glRotatef(90,0,1,0);

GLUquadricObj \* qobj;

qobj = gluNewQuadric();

gluQuadricDrawStyle(qobj, GLU\_FLAT);

gluCylinder(qobj, 0.1, 0.1, 7, 8,8);//silinder dengan diameter 0.1

glPopMatrix();

//meja

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[4]);

glTranslated(9, -3 ,-16.7);

glBegin(GL\_QUADS);

dinding(-10,3,0,-9.5,5,0.3);

dinding(-2.5,3,0,-2,5,0.3);

dinding(-10,3,-3,-9.5,5,-3.3);

dinding(-2.5,3,-3,-2,5,-3.3);

dinding(-10,5,0.3,-2,5.3,-3.3);

glEnd();

glPopMatrix();

//tv

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(11, 3 ,-1.5);

glBegin(GL\_QUADS);

glTexCoord2f(0.0, 0.0);

glVertex3f(-9.7,2.3,-0.1);

glTexCoord2f(0.0, 1.0);

glVertex3f(-5.3,2.3,-0.1);

glTexCoord2f(1.0, 1.0);

glVertex3f(-5.3,5.1,-0.1);

glTexCoord2f(1.0, 0.0);

glVertex3f(-9.7,5.1,-0.1);

dinding(-10,2,0,-5,5.5,0.3);

dinding(-10,3,0,-6.3,3.3,1);

glEnd();

glPopMatrix();

//laptop

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(11, -0.65 ,-18.7);

glBegin(GL\_QUADS);

dinding(-10,3,0,-6.3,5.5,0.3);

dinding(-10,3,0,-6.3,3.3,2.5);

glEnd();

glPopMatrix();

//gagang kiri

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(-9,0,-17);

glRotated(lemari1,0,1,0);

glTranslated(9,0,17);

glBegin(GL\_QUADS);

dinding(-6,5.3,-16.5,-5.8,5.5,-17);

dinding(-6,6.5,-16.5,-5.8,6.7,-17);

dinding(-6,6.8,-16.3,-5.8,5.2,-16.5);

glEnd();

glPopMatrix();

//gagang kanan

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(-2.5,0,-17);

glRotated(lemari2,0,1,0);

glTranslated(2.5,0,17);

glBegin(GL\_QUADS);

dinding(-5.2,5.3,-16.5,-5,5.5,-17);

dinding(-5.2,6.5,-16.5,-5,6.7,-17);

dinding(-5.2,6.8,-16.3,-5,5.2,-16.5);

glEnd();

glPopMatrix();

//box kipas

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[6]);

glTranslated(17,9.75,0);

glScalef(0.8,0.4,1);

glBegin(GL\_QUADS);

//belakang

dinding(-9,0,-19.7,-2,11.5,-20);

//samping kiri

dinding(-9,0,-20.7,-8.7,11.5,-19);

//samping kanan

dinding(-2.3,0,-19.7,-2,11.5,-19);

//atas

dinding(-9,11.5,-19,-2,11.2,-20);

//dasar

dinding(-9,0,-19,-2,0.3,-20);

glEnd();

glPopMatrix();

// baling2 kipas

glPushMatrix();

if(lampu==-45)

{

glTranslatef(12.7,12,-19.5);

glRotatef(z,0,0,1);

glTranslatef(-12.7,-12,19.5);

}

glBegin (GL\_POLYGON);

//segitiga atas

glVertex3f(12.7,12,-19.5);

glVertex3f(11.7,14,-19.5);

glVertex3f(13.7,14,-19.5);

//segitiga bawah

glVertex3f(12.7,12,-19.5);

glVertex3f(11.7,10,-19.5);

glVertex3f(13.7,10,-19.5);

glEnd();

glPopMatrix();

//poster

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[1]);

glTranslated(-9.8,7,-12);

glBegin (GL\_QUADS);

dinding(1,1,1,1,5,7);

glEnd ();

glPopMatrix();

//jam

float alpha, radius, cx, cy;

cx=5;

cy=5;

alpha = 0;

radius = 1;

glPushMatrix();

glTranslated(-8.8,6,-8);

glRotated(90,0,1,0);

//glBindTexture(GL\_TEXTURE\_2D, texture[0]);

glBegin(GL\_POLYGON);

for(float i=0; i<=360; i++)

{

alpha+=1;

glTexCoord2f(1.0, 1.0);

glVertex2f(radius \* cos (alpha/180 \* 3.1415) + cx, radius \* sin(alpha/180 \* 3.1415) + cy);

glTexCoord2f(1.0, 1.0);

}

glEnd();

glPopMatrix();

//layar laptop

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[2]);

glTranslated(11.4, -0.1 ,-18.4);

glBegin(GL\_QUADS);

dinding(-10,3,0,-7,4.7,0.1);

glEnd();

glPopMatrix();

//pemandangan jendela

glPushMatrix();

glBindTexture(GL\_TEXTURE\_2D, texture[2]);

glTranslated(9, 3.2 ,-21.5);

glBegin(GL\_QUADS);

dinding(-10,3,0,-2,11,0.1);

glEnd();

glPopMatrix();

glutSwapBuffers();

}

void myReshape(int w, int h) {

glViewport(0, 0, w, h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(60.0, 1.0 \* (GLfloat) w / (GLfloat) h, 1.0, 40.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(0.0, 0.0, -3.6);

}

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

printf("Tugas Besar Grafika Komputer");

printf("\nKelompok 1: ");

printf("\n\nDiki Taurens Sia 10108396");

printf("\nSopyan 10108400");

printf("\nAndika Tanjung 10108405");

printf("\nMuhamad Arif A. 10108407");

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowSize(1024, 640);

glutInitWindowPosition(0, 75);

glutCreateWindow("Tugas Besar Grafkom");

myinit();

glutReshapeFunc(myReshape);

glutDisplayFunc(display);

glutKeyboardFunc(keyboard);

glutMouseFunc(processMouse);

glutTimerFunc(1,timer,0);

glutMainLoop();

return 0;

}

**PRINTSCREEN**

****

**Gambar 1 Keadaan Awal**

****

**Gambar 2 Keadaan Saat Dilakukan Masuk Ruangan**

****

**Gambar 3 Keadaan Saat Dilakukan Pembukaan Lemari**

****

**Gambar 4 Keadaan Saat Dilakukan Pembukaan Jendela**

****

**Gambar 5 Keadaan Saat Menghadap Poster**

****

**Gambar 5 Keadaan Saat Menghadap Televisi**

****

**Gambar 5 Keadaan Saat Menghadap Kasur**