**TECHNICAL REPORT**

**GRAFIKA KOMPUTER**

**MODUL 6**

**“Tiga Dimensi”**

****

**Disusun Oleh:**

TGL PRAKTIKUM : 30 April 2018

NAMA : Rahma Nur Layla Sari

NRP : 160411100104

KELAS/KELOMPOK : E2

DOSEN :FITRIYATUL QAMARIYAH, S.KOM,. M.KOM

ASISTEN : Sofian Eka Sandra

DOSEN PENGAMPU :

|  |
| --- |
| Disetujui : ....../……./………../Bangkalan |
| Sofian Eka Sandra  150411100108 |

**PRODI INFORMATIKA**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

**2017/ 2018**

**BAB I**

**PENDAHULUAN**

1. **Pengantar**

Alam fisik dalam persepsi manusia adalah sebuah ruang yang berformat 3 dimensi. Benda-benda yang ada di dalamnya umum direpresentasikan menggunakan format 3 dimensi: panjang, lebar, dan tinggi.

Dalam matematika, 3 dimensi ini biasa dinyatakan dalam sistem koordinat kartesian. Koordinat kartesian 3 dimensi memiliki 3 bidang yang saling tegak lurus satu dengan yang lainnya. Tiap bidang memiliki sumbu yang koordinat yang biasa disebut sumbu x, y, dan z.

* 1. **Dimensi di OpenGL**

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 pelajaran-pelajaran sebelumnya obyek dibuat dalam 2 dimensi, sebenarnya obyek-obyek tersebut adalah obyek 3 dimensi. Hanya saja dimensi ketiga diabaikan. Termasuk didalam konsep ini adalah transformasi. Transformasi selalu dilakukan dalam format 3 dimensi.

* 1. **Proyeksi**

Proyeksi diperlukan untuk menggambarkan obyek 3 dimensi kedalam format 2 dimensi, contohnya: arsitek perlu menuangkan idenya tentang sebuah obyek bangunan 3 dimensi diatas kertas (2 dimensi). Contoh lain adalah penggambaran dunia OpenGL yang 3 dimensi ke layar monitor yang 2 dimensi. Perubahan format dari 3 dimensi menjadi 2 dimensi ini memerlukan proses/aturan khusus. Proses/aturan ini disebut proyeksi grafis.

Ada 2 jenis proyeksi :

1. Perspektif

Cara mata manusia dan kamera menangkap gambar obyek sekelilingnya. Obyek yang jauh terlihat kecil, obyek yang dekat terlihat besar. 2 garis sejajar akan terlihat menyatu di kejauhan.

1. Parallel

Garis proyeksi selalu sejajar baik di obyek 3 dimensi maupun di penggambaran 2 dimensinya. Jenis proyeksi ini digunakan oleh orang -orang teknik (Arsitek, teknik mesin, teknik sipil) dalam menggambar pekerjaannya

* 1. **Proyeksi di OpenGL**

Dalam OpenGL, diperlukan beberapa perubahan setting agar sebuah aplikasi dapat menampilkan obyek 3 dimensi. Jika pada aplikasi-aplikasi sebelumnya perintah proyeksi yang digunakan adalah gluOrtho2D(kiri, kanan, bawah, atas) untuk menampilkan obyek 2 dimensi, perintah yang sama harus diganti agar aplikasi dapat menampilkan obyek 3 dimensi:

1. Untuk proyeksi parallel, gunakan glOrtho(kiri, kanan, atas, dekat, jauh);
2. Untuk proyeksi perspektif, gunakan gluPerspective(sudut buka, aspek rasio, dekat, jauh);

Program 6.1 menghasilkan tampilan kotak 3 dimensi

|  |
| --- |
| void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glColor3f(1.0, 1.0, 1.0);  glRotated(35,1,1,1);  glutWireCube(2);  glFlush();  }  void init(void){  glClearColor (0.0, 0.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-5.0, 5.0, -5.0, 5.0, -5.0, 5.0);  glMatrixMode(GL\_MODELVIEW);  glShadeModel(GL\_FLAT);  glEnable(GL\_DEPTH\_TEST);  }  int main(int argc, char\*\* argv){  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_RGB | GLUT\_DEPTH);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Kubus");  init();  glutDisplayFunc(display);  glutMainLoop();  return 0;  } |

**Program 6.1 Kubus 3 dimensi**

Perhatikan perbedaan program diatas dengan program-program sebelumnya:

1. Menggunakan glOrtho, bukan gluOrtho2D
2. Perintah glEnable(GL\_DEPTH\_TEST) diperlukan disini
3. Modifikasi: glutInitDisplayMode(GLUT\_RGB | GLUT\_DEPTH);
4. Modifikasi: glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);
5. Pada beberapa kondisi, diperlukan juga modifikasi:
6. glShadeModel(GL\_FLAT);

Program diatas menggunakan perintah glutWireCube(ukuran) untuk membuat sebuah kubus 3 dimensi. Perintah ini adalah perintah bawaan dari glut. Perintah bawaan lain untuk membuat obyek 3 dimensi adalah:

1. glutWireTeapot(GLdouble size);
2. glutWireSphere(GLdouble radius, GLint slices, GLint stacks);
3. glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks);
4. glutWireTetrahedron(void);
5. glutWireOctahedron(void);

Selain bentuk wireframe diatas, glut menyediakan juga obyek-obyek bawaan 3 dimensi dalam format solid dimana obyek ditampilkan dalam bentok penuh/solid. Format perintahnya sama seperti obyek glut wire diatas hanya tinggal mengganti kata Wire dengan kata Solid:

1. glutWireCube(GLdouble size) 🡪 glutSolidCube(GLdouble size);
2. glutWireTeapot(GLdouble size) 🡪 glutSolidTeapot(GLdouble size);
3. glutWireSphere(GLdouble radius, GLint slices, GLint stacks) 🡪 glutSolidSphere(GLdouble radius, GLint slices, GLint stacks)
4. glutWireCone(GLdouble base, GLdouble height, GLint slices, GLint stacks) 🡪 glutSolidCone(GLdouble base, GLdouble height, GLint slices, GLint stacks)
5. glutWireTetrahedron(void) 🡪 glutSolidTetrahedron(void)
6. glutWireOctahedron(void) 🡪 glutSolidOctahedron(void)

Jika dicoba, obyek-obyek solid ini tampak tidak jelas bentuknya di layar. Yang tampak hanyalah blok berwarna putih. **Ini wajar**. Di modul 8, pencahayaan yang benar akan menampilkan bentuk obyek-obyek ini dengan baik.

Ada kalanya obyek yang ingin dibuat/ditampilkan tidak tersedia dalam library bawaan glut. Untuk kasus seperti ini, programmer perlu membuat sendiri obyek tersebut. Di OpenGL, untuk membuat obyek 3 dimensi, salah satu caranya adalah dengan membuat tiap sisi obyek dari polygon 2 dimensi. Program 6.2 membuat kubus 3 dimensi dimana tiap sisinya dibuat dari persegi 2 dimensi.

|  |
| --- |
| void display(){  glClear(GL\_COLOR\_BUFFER\_BIT| GL\_DEPTH\_BUFFER\_BIT);  glRotated(-35, 1,1,1);  //depan  glColor3f(0.0,0.0,1.0);  glBegin(GL\_POLYGON);  glVertex3f(-1.0, -1.0, 1.0);  glVertex3f(1.0, -1.0, 1.0);  glVertex3f(1.0, 1.0, 1.0);  glVertex3f(-1.0, 1.0, 1.0);  glEnd();  //belakang  glColor3f(0.0,1.0,0.0);  glBegin(GL\_POLYGON);  glVertex3f(1.0, -1.0, -1.0);  glVertex3f(-1.0, -1.0, -1.0);  glVertex3f(-1.0, 1.0, -1.0);  glVertex3f(1.0, 1.0, -1.0);  glEnd();  //kiri  glColor3f(1.0,0.0,0.0);  glBegin(GL\_POLYGON);  glVertex3f(-1.0, -1.0, -1.0);  glVertex3f(-1.0, -1.0, 1.0);  glVertex3f(-1.0, 1.0, 1.0);  glVertex3f(-1.0, 1.0, -1.0);  glEnd();  //kanan  glColor3f(0.0,1.0,1.0);  glBegin(GL\_POLYGON);  glVertex3f(1.0, -1.0, -1.0);  glVertex3f(1.0, 1.0, -1.0);  glVertex3f(1.0, 1.0, 1.0);  glVertex3f(1.0, -1.0, 1.0);  glEnd();  //bawah  glColor3f(1.0,0.0,1.0);  glBegin(GL\_POLYGON);  glVertex3f(1.0, -1.0, 1.0);  glVertex3f(-1.0, -1.0, 1.0);  glVertex3f(-1.0, -1.0, -1.0);  glVertex3f(1.0, -1.0, -1.0);  glEnd();  //atas  glColor3f(1.0,1.0,0.0);  glBegin(GL\_POLYGON);  glVertex3f(-1.0, 1.0, 1.0);  glVertex3f(1.0, 1.0, 1.0);  glVertex3f(1.0, 1.0, -1.0);  glVertex3f(-1.0, 1.0, -1.0);  glEnd();  glFlush();  }  void myinit(){  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-3.0,3.0,-3.0,3.0,-3.0,3.0);  glMatrixMode(GL\_MODELVIEW);  glClearColor(0.0,0.0,0.0,1.0);  glColor3f(0.0,0.0,0.0);  glShadeModel(GL\_FLAT);  glEnable(GL\_DEPTH\_TEST);  }  int main(int argc, char\* argv[]){  glutInit(&argc,argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB | GLUT\_DEPTH);  glutInitWindowSize(400,400);  glutInitWindowPosition(100,100);  glutCreateWindow("Kotak");  glutDisplayFunc(display);  myinit();  glutMainLoop();  return 0;  } |

**Program 6.2 Kubus 3 dimensi yang dibangun dari 6 polygon**

Modul ini memiliki kemungkinan untuk membuat beberapa komputer tidak dapat menampilkan bentuk obyek sebagaimana mestinya. Jika memang program sudah benar sementara tampilan tidak juga sesuai, bergabunglah dengan teman atau cari pinjaman komputer.

1. **Percobaan**
2. Untuk program 6.1, modifikasi parameter glRotated, lalu amati perubahan tampilannya. Kombinasikan dengan memodifikasi parameter pada glutSolidCube; amati perubahannya juga.
3. Untuk program 6.1, ganti glutWireCube dengan obyek-obyek 3 dimensi bawaan yang lain: baik sesama wire maupun solid. Modifikasi parameter glRotated, lalu amati perubahan tampilannya.
4. Kombinasikan dengan memodifikasi parameter pada masing-masing obyek. Amati perubahannya juga.
5. Untuk program 6.2, modifikasi parameter glRotated, lalu amati perubahan tampilannya. Kombinasikan dengan perintah transformasi yang lain; amati perubahannya juga.

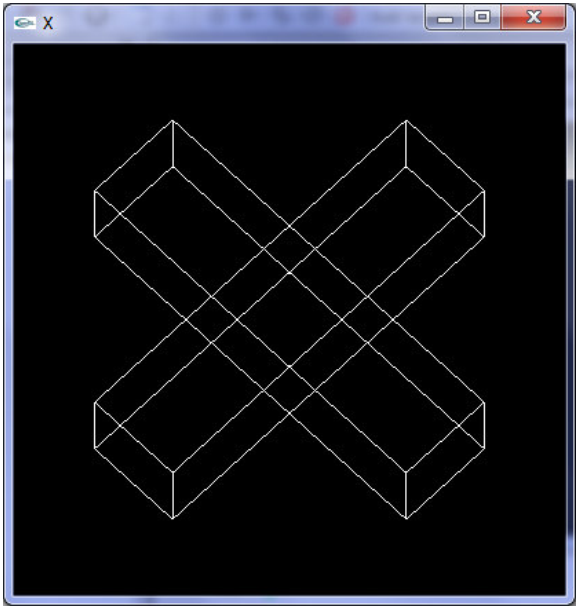
**BAB II**

**LAPORAN TUGAS**

* 1. **TUGAS**

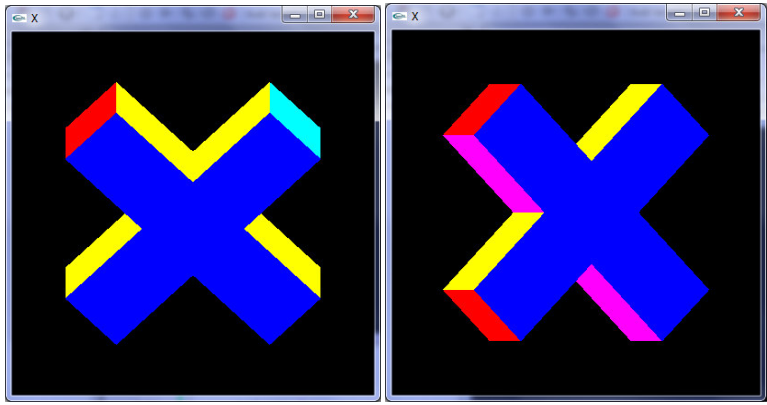
Selesaikan soal-soal berikut :

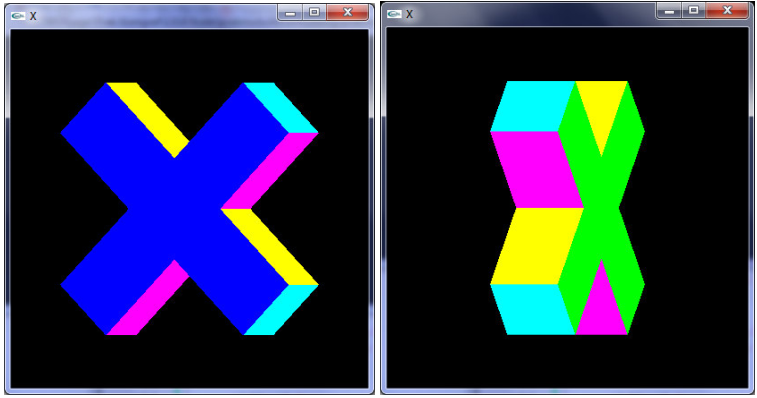
1. Buat dua balok bersilang sebagai berikut dengan menggunakan glutWireCube.



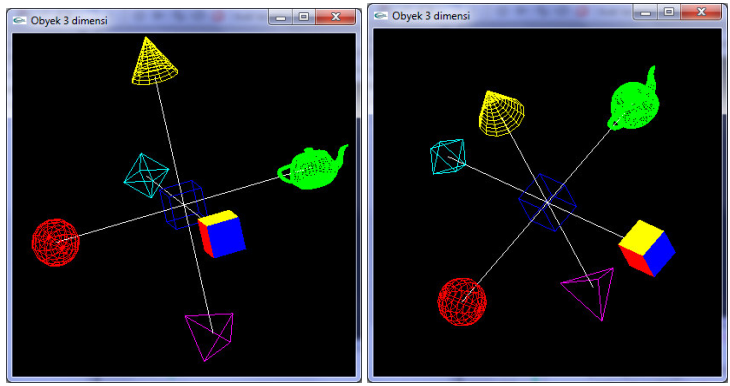
Gambar tidak harus persis, asal cukup mirip

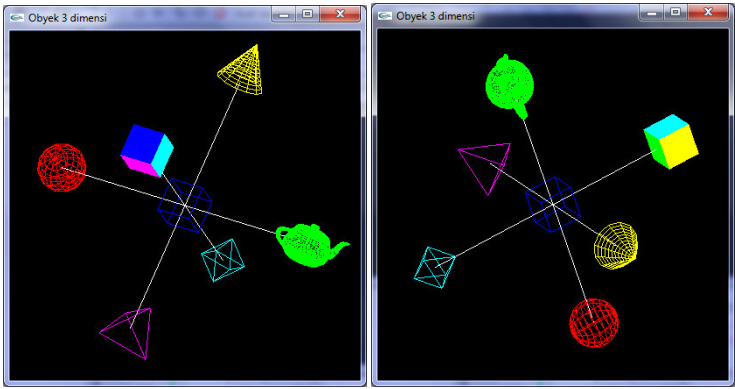
1. Buat dua balok bersilang sebagai berikut secara manual menggunakan kumpulan polygon.





1. Buat 7 obyek 3 dimensi yang dibahas di modul ini dalam satu tampilan. Ketujuh obyek tersebut diletakkan pada ujung-ujung dan titik pusat sumbu koordinat sebagai berikut:



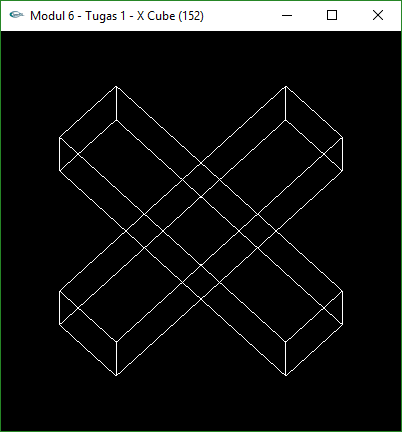


* 1. **PENYELESAIAN**

1. Membuat dua balok bersilang menggunakan glutWireCube

|  |
| --- |
| #include <windows.h>  #ifdef \_\_APPLE\_\_  #include <GLUT/glut.h>  #else  #include <GL/glut.h>  #endif  #include <stdlib.h>  void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glColor3f(1.0, 1.0, 1.0);  glScaled(-1,1,1);  glRotated(25,1,0,0);  glRotated(45,0,0,1);  glScaled(4,1,1);  glutWireCube(2);  glLoadIdentity();  glRotated(25, 1,0,0);  glRotated(45,0,0,1);  glScaled(4,1,1);  glutWireCube(2);  glPopMatrix();  glFlush();  }  void init(void){  glClearColor (0.0, 0.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-5.0, 5.0, -5.0, 5.0, -5.0, 5.0);  glMatrixMode(GL\_MODELVIEW);  glShadeModel(GL\_FLAT);  glEnable(GL\_DEPTH\_TEST);  }  int main(int argc, char\*\* argv){  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_RGB | GLUT\_DEPTH);  glutInitWindowSize(400, 400);  glutInitWindowPosition(100, 100);  glutCreateWindow("Modul 6 - Tugas 1 - X Cube (152)");  init();  glutDisplayFunc(display);  glutMainLoop();  return 0;  } |

**Hasil Running**



1. Membuat dua balok bersilang secara manual menggunakan kumpulan polygon
2. Bagian 1

|  |
| --- |
| #include <windows.h>  #ifdef \_\_APPLE\_\_  #include <GLUT/glut.h>  #else  #include <GL/glut.h>  #endif  #include <stdlib.h>  void objek() {  glPushMatrix();  glRotated(45,0,0,1);  glColor3d(0,1,0);  glBegin(GL\_POLYGON); //Belakang +  glVertex3f(1.0, 1.0, -1.0);  glVertex3f(4.0, 1.0, -1.0);  glVertex3f(4.0, -1.0, -1.0 );  glVertex3f(1.0, -1.0, -1.0);  glVertex3f(1.0, -4.0, -1.0);  glVertex3f(-1.0, -4.0, -1.0);  glVertex3f(-1.0, -1.0, -1.0);  glVertex3f(-4.0, -1.0, -1.0);  glVertex3f(-4.0, 1.0, -1.0);  glVertex3f(-1.0,1.0, -1.0);  glVertex3f(-1.0, 4.0, -1.0);  glVertex3f(1.0, 4.0, -1.0);  glEnd();  glColor3d(0,0,1);  glBegin(GL\_POLYGON); //Depan +  glVertex3f(1.0, 1.0, 1.0);  glVertex3f(4.0, 1.0, 1.0);  glVertex3f(4.0, -1.0, 1.0 );  glVertex3f(1.0, -1.0, 1.0);  glVertex3f(1.0, -4.0, 1.0);  glVertex3f(-1.0, -4.0, 1.0);  glVertex3f(-1.0, -1.0, 1.0);  glVertex3f(-4.0, -1.0, 1.0);  glVertex3f(-4.0, 1.0, 1.0);  glVertex3f(-1.0,1.0, 1.0);  glVertex3f(-1.0, 4.0, 1.0);  glVertex3f(1.0, 4.0, 1.0);  glEnd();  glColor3d(1,0,0); //atas  glBegin(GL\_POLYGON);  glVertex3d(-1, 4, 1);  glVertex3d(1, 4, 1);  glVertex3d(1, 4, -1);  glVertex3d(-1, 4, -1);  glEnd();  glBegin(GL\_POLYGON); //kiri  glVertex3d(-4, -1, 1);  glVertex3d(-4, 1, 1);  glVertex3d(-4, 1, -1);  glVertex3d(-4, -1, -1);  glEnd();  glColor3f(0.5, 1, 1); //kanan  glBegin(GL\_POLYGON);  glVertex3d(4, -1, 1);  glVertex3d(4, 1, 1);  glVertex3d(4, 1, -1);  glVertex3d(4, -1, -1);  glEnd();  glBegin(GL\_POLYGON); //bawah  glVertex3d(-1, -4, 1);  glVertex3d(1, -4, 1);  glVertex3d(1, -4, -1);  glVertex3d(-1, -4, -1);  glEnd();  glColor3f(1,1,0); //Kuning 2 di kanan atas  glBegin(GL\_POLYGON);  glVertex3d(1, 4, 1);  glVertex3d(1, 4, -1);  glVertex3d(1, 1, -1);  glVertex3d(1, 1, 1);  glEnd();  glBegin(GL\_POLYGON);  glVertex3d(1, 1, 1);  glVertex3d(1, 1, -1);  glVertex3d(4, 1, -1);  glVertex3d(4, 1, 1);  glEnd();  glBegin(GL\_POLYGON); //Kuning bawah kanan vertikal  glVertex3d(1, -1, 1);  glVertex3d(1, -1, -1);  glVertex3d(1, -4, -1);  glVertex3d(1, -4, 1);  glEnd();  glBegin(GL\_POLYGON); //Kuning kiri atas horizontal  glVertex3d(-4, 1, 1);  glVertex3d(-4, 1, -1);  glVertex3d(-1, 1, -1);  glVertex3d(-1, 1, 1);  glEnd();  glColor3f(1,0,1); //Pink 2 di kiri bawah  glBegin(GL\_POLYGON);  glVertex3d(-4, -1, 1);  glVertex3d(-4, -1, -1);  glVertex3d(-1, -1, -1);  glVertex3d(-1, -1, 1);  glEnd();  glBegin(GL\_POLYGON);  glVertex3d(-1, -1, 1);  glVertex3d(-1, -1, -1);  glVertex3d(-1, -4, -1);  glVertex3d(-1, -4, 1);  glEnd();  glBegin(GL\_POLYGON); //Pink atas kiri vertikal  glVertex3d(-1, 4, 1);  glVertex3d(-1, 4, -1);  glVertex3d(-1, 1, -1);  glVertex3d(-1, 1, 1);  glEnd();  glBegin(GL\_POLYGON);  glVertex3d(4, -1, 1);  glVertex3d(4, -1, -1);  glVertex3d(1, -1, -1);  glVertex3d(1, -1, 1);  glEnd();  glPopMatrix();  }  void display(){  glClear(GL\_COLOR\_BUFFER\_BIT| GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(25, 1,0,0);  objek();  glPopMatrix();  glFlush();  }  void myinit(){  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-5.0,5.0,-5.0,5.0,-5.0,5.0);  glMatrixMode(GL\_MODELVIEW);  glClearColor(0.0,0.0,0.0,1.0);  glColor3f(0.0,0.0,0.0);  glShadeModel(GL\_FLAT);  glEnable(GL\_DEPTH\_TEST);  }  int main(int argc, char\* argv[]){  glutInit(&argc,argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB | GLUT\_DEPTH);  glutInitWindowSize(400,400);  glutInitWindowPosition(100,100);  glutCreateWindow("Modul 6 - Tugas 2 | Bag 1 - (152)");  glutDisplayFunc(display);  myinit();  glutMainLoop();  return 0;  } |

**Hasil Running**

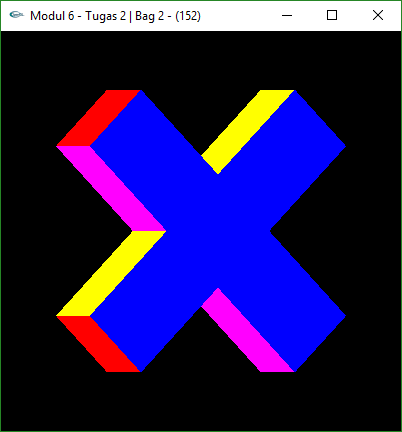


1. Bagian 2

Hanya melakukan perubahan sudut pandang rotasi menjadi glRotated(25, 0,1,0); pada void display()

|  |
| --- |
| void display(){  glClear(GL\_COLOR\_BUFFER\_BIT| GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(25, 0,1,0);  objek();  glPopMatrix();  glFlush();  } |

**Hasil Running**

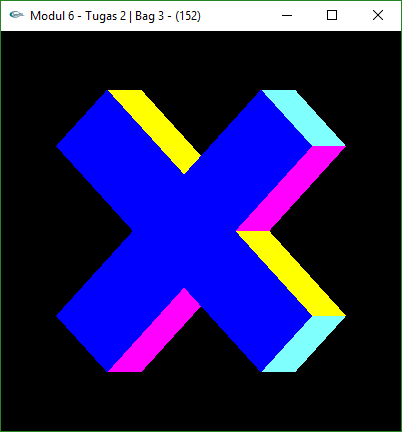


1. Bagian 3

Hanya melakukan perubahan sudut pandang rotasi menjadi glRotated(-25, 0,1,0); pada void display()

|  |
| --- |
| void display(){  glClear(GL\_COLOR\_BUFFER\_BIT| GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(-25, 0,1,0);  objek();  glPopMatrix();  glFlush();  } |

**Hasil Running**

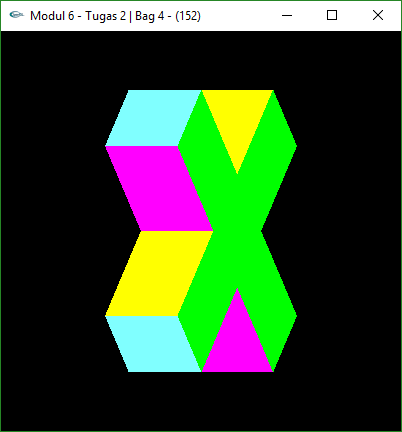


1. Bagian 4

Hanya melakukan perubahan sudut pandang rotasi menjadi glRotated(-115, 0,1,0); pada void display()

|  |
| --- |
| void display(){  glClear(GL\_COLOR\_BUFFER\_BIT| GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(-115, 0,1,0);  objek();  glPopMatrix();  glFlush();  } |

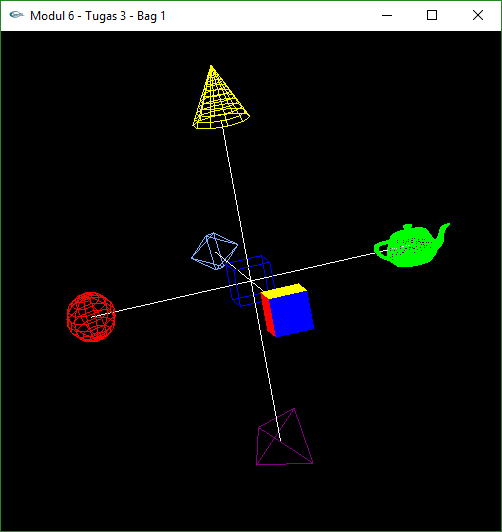
**Hasil Running**



1. Membuat 7 obyek 3 dimensi yang dibahas pada modul 6 dalam satu tampilan, ketujuh obyek diletakkan pada ujung-ujung dan titik pusat sumbu koordinat
2. Bagian 1

|  |
| --- |
| #include <windows.h>  #ifdef \_\_APPLE\_\_  #include <GLUT/glut.h>  #else  #include <GL/glut.h>  #endif  #include <stdlib.h>  void kubusWarna(){  glPushMatrix();  glScaled(0.8,0.8,0.8);  //depan  glColor3f(0,0,1);  glBegin(GL\_POLYGON);  glVertex3f(-3,-3,3);  glVertex3f(3,-3,3);  glVertex3f(3,3,3);  glVertex3f(-3,3,3);  glEnd();  //belakang  glColor3f(0,1,0);  glBegin(GL\_POLYGON);  glVertex3f(3,-3,-3);  glVertex3f(-3,-3,-3);  glVertex3f(-3,3,-3);  glVertex3f(3,3,-3);  glEnd();  //kiri  glColor3f(1,0,1);  glBegin(GL\_POLYGON);  glVertex3f(-3,-3,-3);  glVertex3f(-3,-3,3);  glVertex3f(-3,3,3);  glVertex3f(-3,3,-3);  glEnd();  //kanan  glColor3f(1,1,0);  glBegin(GL\_POLYGON);  glVertex3f(3,-3,-3);  glVertex3f(3,3,-3);  glVertex3f(3,3,3);  glVertex3f(3,-3,3);;  glEnd();  //bawah  glColor3f(0,1,1);  glBegin(GL\_POLYGON);  glVertex3f(3,-3,3);  glVertex3f(-3,-3,3);  glVertex3f(-3,-3,-3);  glVertex3f(3,-3,-3);  glEnd();  //atas  glColor3f(1,0,0);  glBegin(GL\_POLYGON);  glVertex3f(-3,3,3);  glVertex3f(3,3,3);  glVertex3f(3,3,-3);  glVertex3f(-3,3,-3);  glEnd();  glPopMatrix();  }  void objek() {  glColor3f(0,0,1);  glutWireCube(4.5); //Kubus Tengah  glPushMatrix();  glColor3f(1,1,0);  glTranslated(0,20,0);  glRotated(-90,1,0,0);  glutWireCone(3.5, 7, 10, 10); //Cone atas  glPopMatrix();  glPushMatrix();  glColor3f(0.5, 0, 0.5);  glTranslated(0,-20,0);  glScaled(5,5,5);  glRotated(90,0,0,1);  glutWireTetrahedron(); //Tetrahedron Bawah  glPopMatrix();  glPushMatrix();  glColor3f(0,1,0);  glTranslated(20,0,0);  glutWireTeapot(3); //Teapot Kanan  glPopMatrix();  glPushMatrix();  glColor3f(1,0,0);  glRotated(45,1,0,0);  glTranslated(-20,0,0);  glutWireSphere(3, 10, 10); //Bola Kiri  glPopMatrix();  glPushMatrix();  glTranslated(0,0,20);  glRotated(90,0,0,1);  kubusWarna(); //Kubus depan  glPopMatrix();  glPushMatrix();  glColor3f(0.5,0.7,1);  glTranslated(0,0,-20);  glRotated(30,1,0,0);  glRotated(5,0,1,0);  glutWireSphere(3, 4, 2); //Sphere belakang  glPopMatrix();  }  garis() {  glColor3f(1,1,1);  glBegin(GL\_LINES);  glVertex3f(0,20,0);  glVertex3f(0,-20,0);  glVertex3f(20,0,0);  glVertex3f(-20,0,0);  glVertex3f(0,0,20);  glVertex3f(0,0,-20);  glEnd();  }  void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(20,1,1,1);  objek();  garis();  glPopMatrix();  glFlush();  }  void init(void){  glClearColor (0.0, 0.0, 0.0, 0.0);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glOrtho(-30.0, 30.0, -30.0, 30.0, -30.0, 30.0);  glMatrixMode(GL\_MODELVIEW);  glShadeModel (GL\_FLAT);  glEnable(GL\_DEPTH\_TEST);  }  int main(int argc, char\*\* argv){  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_RGB | GLUT\_DEPTH);  glutInitWindowSize(500, 500);  glutInitWindowPosition(100, 100);  glutCreateWindow("Modul 6 - Tugas 3 - Bag 1");  init();  //glutKeyboardFunc(kunci);  glutDisplayFunc(display);  glutMainLoop();  return 0;  } |

**Hasil Running**

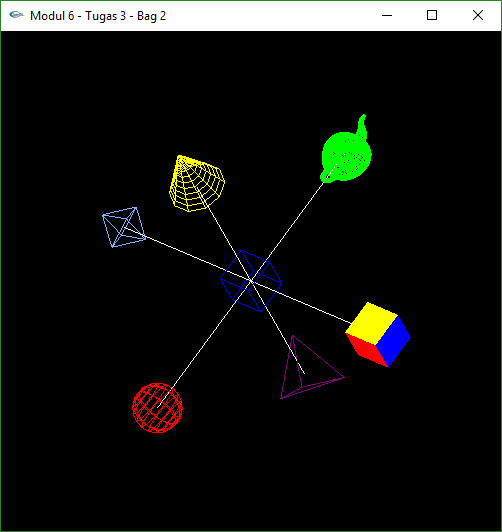


1. Bagian 2

Program sama dengan Bagian 1 diatas, hanya mengubah nilai rotasi menjadi glRotated(70,1,1,1); pada void display();

|  |
| --- |
| void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(70,1,1,1);  objek();  garis();  glPopMatrix();  glFlush();  } |

**Hasil Running**



1. Bagian 3

Program sama dengan Bagian 1 diatas, hanya mengubah nilai rotasi menjadi glRotated(-30,1,1,1); pada void display();

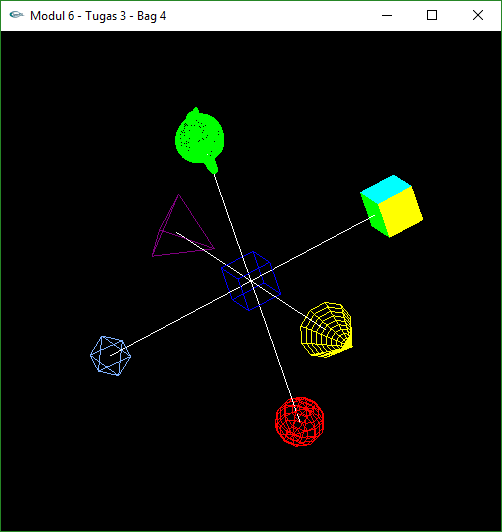
|  |
| --- |
| void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(-30,1,1,1);  objek();  garis();  glPopMatrix();  glFlush();  } |

1. Bagian 4

Program sama dengan Bagian 1 diatas, hanya mengubah nilai rotasi menjadi glRotated(160,1,1,1); pada void display();

|  |
| --- |
| void display(void){  glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);  glPushMatrix();  glRotated(160,1,1,1);  objek();  garis();  glPopMatrix();  glFlush();  } |

**Hasil Running**



1. **PENJELASAN PROGRAM**
2. Membuat dua balok bersilang dengan menggunakan glutWireCube(). Pembuatan program pertama ini menggunakan glutWireCube() dengan ukuran 2. Objek kami gambarkan di tengah-tengah titik sumbu. Kemudian supaya kubus menjadi memanjang membentuk balok, kami menggunakan glScale dengan menscale pada sumbu *x* dimana nantinya akan terbentuk gambar balok panjang horizontal. Setelah itu kami melakukan rotasi supaya obyek terlihat miring ke kanan. Dan yang terakhir, untuk menciptakan obyek disebaliknya (kiri) dilakukan mirror obyek dengan memanfaatkan glScaled(-1,1,1);
3. Membuat dua balok bersilang secara manual menggunakan kumpulan polygon. Pembuatan program kedua ini menggunakan GL\_POLYGON yang digambar dengan vertex3. Untuk memudahkan dalam menggambar obyek, pertama kali kami membuatnya membentuk seperti tanda plus (+), kemudian kami warnai sesuai dengan soal yang berikan. Setelah itu supaya gambar terlihat membentuk X, kami melakukan rotasi sebesar 45 derajat pada sumbu *z*. Maka gambar akan terlihat seperti huruf X dua dimensi.

Karena gambar terlihat seperti 2 dimensi, maka untuk membuatnya terlihat bahwa obyek tersebut 3 dimensi dilakukan perubahan sudut pandang dengan menggunakan fungsi rotasi sebesar 25 derajat pada sumbu *x* (untuk gambar 1). Rotasi sebesar 25 derajat pada sumbu *y* (gambar 2), rotasi sebesar -25 derajat pada sumbu *y* (gambar 3), dan yang terakhir dilakukan rotasi sebesar -115 pada sumbu *y* (gambar 4). Kami juga membuat program dengan fungsi input dari user untuk merotasi gambar

1. Membuat 7 obyek 3 dimensi yang dibahas pada modul 6 ini dalam satu tampilan. Ketujuh obyek tersebut diletakkan pada ujung dan titik pusat sumbu koordinat.

Pembuatan program 3 dimulai dengan membuat garis di sumbu *x,y* dan *z* berwarna putih dengan panjang 40 per sisi. Untuk obyek nya yang pertama adalah kerangka kubus yang terletak di tengah-tengah sumbu *x,y,z*, kedua obyek *cone* berwarna kuning yang ditranslasi sebesar 20 pada sumbu *y* jadi obyek berada di atas, ketiga obyek *tetrahedron* yang juga di translasi sebesar -20 pada sumbu *y* (berada di bawah), keempat obyek *teapot* yang ditranslasi sebesar 20 pada sumbu *x* (kanan), kelima obyek kerangka bola merah ditranslasi -20 pada sumbu *x* (kiri), keenam obyek kubus berwarna sesuai dengan contoh program 2 pada modul 6 yang ditranslasi 20 pada sumbu *z* (depan), dan yang terakhir *sphere* di bagian belakang.

Perubahan sudut pandang perlu dilakukan untuk membuat tampilan program menjadi 3 dimensi, detail rotasi yang dilakukan telah kami tulis pada source code tugas 3 diatas. Kami juga menyertakan program dengan tambahan fitur inputan user menggunakan keyboardfunc pada dokumen ini.

**BAB III**

**PENUTUP**

* 1. **KESIMPULAN**

Jika sebelumnya kita hanya dapat menggambar obyek 2 dimensi, dalam modul kali ini kita dapat belajar bagaimana untuk membuat obyek 3 dimensi. dengan menambahkan sumbu z, obyek dapat terlihat 3 dimensi. Selain itu untuk memudahkan penggambaran obyek kita juga dapat melakukan translasi obyek, merotasi, memperbesar/memperkecil ukuran obyek.

* 1. **SARAN**

Pemahaman tentang kombinasi terhadap obyek 3 dimensi akan sangat membantu kami dalam menggambar obyek. Karena pengaplikasiannya agak sedikit membingungkan terutama dibagian rotasi, mungkin beberapa mahasiswa memiliki masalah yang berbeda. Selain itu, contoh-contoh referensi yang lebih banyak bagi mahasiswa juga akan membantu untuk mengimplemenatasikan fungsi-fungsi pada OpenGL.