Praktikum
Interaksi antar Objek

Sebagai usaha untuk meningkatkan fungsi dan kegunaan dari project openGL mengharuskan kita untuk melakukan interaksi antar objek. Interaksi yang ada tidak lain untuk mencapai scenario yang kita inginkan. Seperti Untuk mendorong objek dengan objek lainnya atau menabrak objek lainnya. Sayangnya ada keterbatasan interaksi pada openGL dimana tidak terdapat pendeteksi untuk mengetahui apakah dua objek sedang saling berinteraksi/bersentuahn atau tidak. Hal ini tentu saja merugikan dalam mengembangkan skenario yang diinginkan.

Sebagai solusi untuk tetap mendapatkan skenario yang diinginkan interaksi dilakukan dengan melakukan perkondisian dari koordinat lokasi objek-objek yang ingin diinteraksi. Seperti jika ingin objek A mendorong objek B maka dibuat permisalan jika koordinat sisi objek A telah menyentuk koordinat sisi objek B maka objek B akan mulai bergerak searah dengan objek A. Untuk lebih memahami silahkan mencoba contoh berikut.

Tendangan

Main.cpp

```
#include <math.h>
#include <GL/glut.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define PI 3.14

float angle=0.0, deltaAngle = 0.0, ratio;
float x=-5.0f,y=12.0f,z=40.0f; // posisi awal kamera
float lx=0.0f,ly=0.0f,lz=-1.0f;
int deltaMove = 0,h,w;
static int rotAngleX =0, rotAngleY =0, rotAngleZ =0;
float posXKaki=10, posXBola=-10, posYKaki=6, posYBola=-5;
float rotKaki=0.0;
int kick=0, roll=0,touch=0;
float jarak=1;

GLUquadricObj *IDquadric;
// Variable untuk pencahayaan
const GLfloat light_ambient[] = { 0.5f, 0.5f, 0.5f, 0.0f };
const GLfloat light_diffuse[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_position[] = { 0.0f, 20.0f, 10.0f, 1.0f };
const GLfloat mat_ambient[] = { 0.7f, 0.7f, 0.7f, 1.0f };
const GLfloat mat_diffuse[] = { 0.8f, 0.8f, 0.8f, 1.0f };
const GLfloat mat_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat high_shininess[] = { 100.0f };

void init(void)
{
    glEnable (GL_DEPTH_TEST);
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
    IDquadric=gluNewQuadric();      // Create A Pointer To The Quadric
Object ( NEW )
```

```
    gluQuadricNormals(IDquadric, GLU_SMOOTH);  // Create Smooth Normals
( NEW )
    gluQuadricTexture(IDquadric, GL_TRUE);    // Create Texture Coords (
NEW )
}

void Reshape(int w1, int h1)
{
    // Fungsi reshape
    if(h1 == 0)
    h1 = 1;
    w = w1;
    h = h1;
    ratio = 1.0f * w / h;
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glViewport(0, 0, w, h);
    gluPerspective(45,ratio,0.1,1000);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
}

void orientMe(float ang)
{
    // Fungsi ini untuk memutar arah kamera (tengok kiri/kanan)
    lx = sin(ang/10);
    lz = -cos(ang/10);
    glLoadIdentity();
    gluLookAt(x, y, z,x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
}

void moveMeFlat(int i)
{
    // Fungsi ini untuk maju mundur kamera
    x = x + i*(lx)*0.1;
    z = z + i*(lz)*0.1;
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
}

void keyboard(unsigned char key, int x, int y)
{
     switch (key)
    {
          case 'w':
          rotAngleX += 2;  break;
          case 's':
          rotAngleX -= 2;  break;
          case 'a':
                rotAngleY += 2; break;
          case 'd':
          rotAngleY -= 2; break;
          case 'q':
          rotAngleZ += 2; break;
          case 'e':
          rotAngleZ -= 2; break;
        case 'o':
          posXKaki -= 1;
          //deteksi apakah menempel
          if (posXBola<-2.9){
                      posXBola+=1;
                }
                break;
        case 'p':
          posXKaki += 1;
                posXBola -= 1;
```

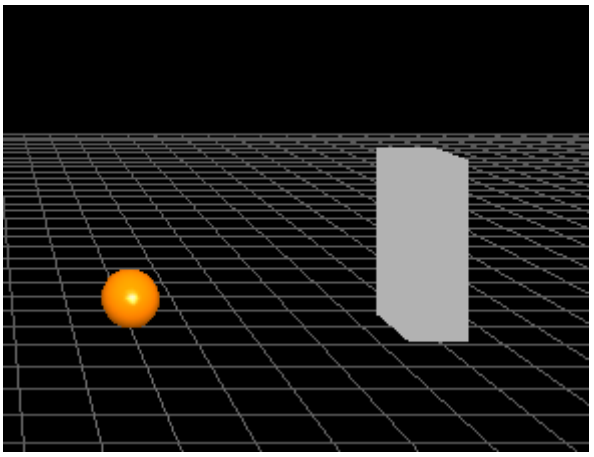
```
                break;
        case 'k':
          kick = 1; break;
        case 32:
          rotAngleX=rotAngleY=rotAngleZ=0;
                posXKaki=10, posXBola=-10, posYKaki=6, posYBola=-5;
                rotKaki=kick=roll=0;
                break;
          case 27:
          exit(0);
        default:
          break;
    } glutPostRedisplay();
}

void pressKey(int k, int x, int y)
{
    // Fungsi ini akan dijalankan saat tombol keyboard ditekan dan belum
dilepas
    // Selama tombol ditekan, variabel angle dan move diubah => kamera
bergerak
    switch (k)
    {
          case GLUT_KEY_UP : deltaMove = 1;break;
          case GLUT_KEY_DOWN : deltaMove = -1;break;
          case GLUT_KEY_LEFT : deltaAngle = -0.01f;break;
          case GLUT_KEY_RIGHT : deltaAngle = 0.01f;break;
    }
}

void releaseKey(int key, int x, int y)
{
    // Fungsi ini akan dijalankan saat tekanan tombol keyboard dilepas
    // Saat tombol dilepas, variabel angle dan move diset nol => kamera
berhenti
    switch (key)
    {
          case GLUT_KEY_UP : if (deltaMove > 0)
                deltaMove = 0;
                break;
          case GLUT_KEY_DOWN : if (deltaMove < 0)
                deltaMove = 0;
                break;
          case GLUT_KEY_LEFT : if (deltaAngle < 0.0f)
          deltaAngle = 0.0f;
                break;
          case GLUT_KEY_RIGHT : if (deltaAngle > 0.0f)
                deltaAngle = 0.0f;
                break;
    }
}


void lighting(){
    // Fungsi mengaktifkan pencahayaan
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LESS);
    glEnable(GL_LIGHT0);
    glEnable(GL_NORMALIZE);
    glEnable(GL_COLOR_MATERIAL);
    glEnable(GL_LIGHTING);
    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
    glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
    glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
```

```
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
}


//------------------------------------------------------------------------
-
void Grid()
{
    // Fungsi untuk membuat grid di "lantai"
    double i;
    const float Z_MIN = -50, Z_MAX = 50;
    const float X_MIN = -50, X_MAX = 50;
    const float gap = 2;
    glColor3f(0.5, 0.5, 0.5);
    glBegin(GL_LINES);
     for(i=Z_MIN;i<Z_MAX;i+=gap)
     {
        glVertex3f(i, 0, Z_MIN);
        glVertex3f(i, 0, Z_MAX);
     }

     for(i=X_MIN;i<X_MAX;i+=gap)
     {
        glVertex3f(X_MIN, 0, i);
        glVertex3f(X_MAX, 0, i);
     }
    glEnd();
}

void Grid2()
{
    glColor3f(1.0f,1.0f,1.0f);
    glBegin(GL_QUADS);
          glVertex3f(-50, 0, 50);
         glVertex3f(-50, 0, -50);
         glVertex3f(50, 0, -50);
          glVertex3f(50, 0, 50);

    glEnd();

}

void Balok(float panjang,float lebar,float tinggi)
{
    glPushMatrix();
          float p=panjang/2;
          float l=lebar/2;
          float t=tinggi/2;
          //depan
        glBegin(GL_QUADS);
              glVertex3f(-p,0,l);
              glVertex3f(p,0,l);
              glVertex3f(p,-t*2,l);
              glVertex3f(-p,-t*2,l);
          glEnd();

      // belakang
          glBegin(GL_QUADS);
              glVertex3f(-p,0,-l);
              glVertex3f(p,0,-l);
              glVertex3f(p,-t*2,-l);
              glVertex3f(-p,-t*2,-l);
          glEnd();

          // atas
          glBegin(GL_QUADS);
```

```
          glTexCoord2f(0.0f, 0.0f);
              glVertex3f(-p,0,-l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(p,0,-l);
              glTexCoord2f(1.0f, 0.0f);
              glVertex3f(p,0,l);
              glTexCoord2f(1.0f, 1.0f);
              glVertex3f(-p,0,l);
          glEnd();

          // bawah
          glBegin(GL_QUADS);
                glTexCoord2f(0.0f, 0.0f);
              glVertex3f(-p,-t*2,-l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(p,-t*2,-l);
              glTexCoord2f(1.0f, 0.0f);
              glVertex3f(p,-t*2,l);
              glTexCoord2f(1.0f, 1.0f);
              glVertex3f(-p,-t*2,l);
          glEnd();

          // kanan
        glBegin(GL_QUADS);
          glTexCoord2f(0.0f, 0.0f);
              glVertex3f(-p,-t*2,-l);
              glTexCoord2f(1.0f, 0.0f);
              glVertex3f(-p,-t*2,l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(-p,0,l);
              glTexCoord2f(1.0f, 1.0f);
              glVertex3f(-p,0,-l);
          glEnd();

          // kiri
          glBegin(GL_QUADS);
          glTexCoord2f(0.0f, 0.0f);
              glVertex3f(p,-t*2,-l);
              glTexCoord2f(1.0f, 0.0f);
              glVertex3f(p,-t*2,l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(p,0,l);
              glTexCoord2f(1.0f, 1.0f);
              glVertex3f(p,0,-l);
          glEnd();
    glPopMatrix();

}

//================================================
//kondisi yang mengubah nilai putaran
void pergerakanKaki(){
    //kondisi menarik kaki
    if (kick==1){
          if (rotKaki<=45){
                rotKaki += 0.03;
          }
          if (rotKaki>44.9){
                      kick = 2;
          }
    }
    //deteksi apakah bola menempel
    if (posXBola>-2.9){
          touch=1;
    }else if(posXBola<-12){
          touch = 0;
    }
```

```
    //kondisi menendang
    if (kick==2){
          if (rotKaki>=-90){
                rotKaki -= 0.2;
                if(rotKaki <1 && touch ==1){
                      roll=1;
                }
          }
          if(rotKaki < -90){
                kick=3;
          }
    }
    //kondisi mengembalikan kaki
    if (kick ==3){
          if (rotKaki<=0){
                rotKaki +=0.05;
          }
          if (rotKaki > -1){
                kick = 0;
          }
    }
}

void pergerakanBola(){
    //kondisi jika kaki telah menyentuh bola
    if (roll == 1){
          if (jarak>0){
                posXBola-=0.03; //mengatur kecepatan & banyaknya jarak
yang ditempuh dlm 1 perulangan
                jarak-=0.01; // mengatur banyaknya perulangan
          }
          if (jarak <0){
                roll = 0;
                jarak = 1;
          }
    }
}

void Object()
{
    glPushMatrix();
          glTranslatef(posXKaki,posYKaki,0);
          glPushMatrix();
                pergerakanKaki();
                glRotatef(rotKaki,0,0,1); //eksekusi rotasi kaki
                glColor3f(1,1,1);
                Balok(2,3,6);
          glPopMatrix();

          glPushMatrix();
                pergerakanBola();
                glColor3f(0.8,0.4,0.0);
                glTranslatef(posXBola,posYBola,0);
                glutSolidSphere(1, 20,20);
          glPopMatrix();
    glPopMatrix();

    glFlush();
}

void display() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    // Kalau move dan angle tidak nol, gerakkan kamera...
    if (deltaMove)
          moveMeFlat(deltaMove);
    if (deltaAngle) {
```

```
          angle += deltaAngle;
          orientMe(angle);        }

    glPushMatrix();
          glRotated(rotAngleX, 1, 0, 0);
        glRotated(rotAngleY, 0, 1, 0);
        glRotated(rotAngleZ, 0, 0, 1);
          // Gambar grid
          Grid();
          //Grid2();
          // Gambar objek di sini...
          Object();
    glPopMatrix();
    glFlush();
    glutSwapBuffers();
}


int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    glutInitWindowPosition(100,100);
    glutInitWindowSize(640,480);
    glutCreateWindow("Tendangan");
    glutSpecialFunc(pressKey);
    glutSpecialUpFunc(releaseKey);
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutIdleFunc(display);
    glutReshapeFunc(Reshape);
    lighting();
    init();
    glutMainLoop();
    return(0);
}
```



Grab

```
#include <math.h>
#include <GL/glut.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define PI 3.14

float angle=0.0, deltaAngle = 0.0, ratio;
float x=5.0f,y=10.0f,z=40.0f; // posisi awal kamera
float lx=0.0f,ly=0.0f,lz=-1.0f;
int deltaMove = 0,h,w;
static int rotAngleX =0, rotAngleY =0, rotAngleZ =0;
float posXBadan=10, posXKotak=0, posYBadan=7, posYKotak=6;
float rotTangan1=0.0, rotTangan2=0.0,rotTangan3=0.0, rotTangan4=0.0;
int  kick=0,  roll=0,hit=0,  gerakTangan=0,  drop=0,  bring=0,  grab=0,
tabrak=0;
float jarak=1;

GLUquadricObj *IDquadric;
// Variable untuk pencahayaan
const GLfloat light_ambient[] = { 0.5f, 0.5f, 0.5f, 0.0f };
const GLfloat light_diffuse[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat light_position[] = { 0.0f, 20.0f, 10.0f, 1.0f };
const GLfloat mat_ambient[] = { 0.7f, 0.7f, 0.7f, 1.0f };
const GLfloat mat_diffuse[] = { 0.8f, 0.8f, 0.8f, 1.0f };
const GLfloat mat_specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
const GLfloat high_shininess[] = { 100.0f };

void init(void)
{
     glEnable (GL_DEPTH_TEST);
     glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
     IDquadric=gluNewQuadric();      // Create A Pointer To The Quadric
Object ( NEW )
     gluQuadricNormals(IDquadric, GLU_SMOOTH);  // Create Smooth Normals
( NEW )
     gluQuadricTexture(IDquadric, GL_TRUE);     // Create Texture Coords
( NEW )
}

void Reshape(int w1, int h1)
{
     // Fungsi reshape
     if(h1 == 0)
     h1 = 1;
     w = w1;
     h = h1;
     ratio = 1.0f * w / h;
     glMatrixMode(GL_PROJECTION);
     glLoadIdentity();
     glViewport(0, 0, w, h);
     gluPerspective(45,ratio,0.1,1000);
     glMatrixMode(GL_MODELVIEW);
     glLoadIdentity();
     gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
}

void orientMe(float ang)
{
     // Fungsi ini untuk memutar arah kamera (tengok kiri/kanan)
     lx = sin(ang/10);
     lz = -cos(ang/10);
     glLoadIdentity();
     gluLookAt(x, y, z,x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
```

```
}

void moveMeFlat(int i)
{
     // Fungsi ini untuk maju mundur kamera
     x = x + i*(lx)*0.1;
     z = z + i*(lz)*0.1;
     glLoadIdentity();
     gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,1.0f,0.0f);
}

void keyboard(unsigned char key, int x, int y)
{
    switch (key)
     {
     case 'w':
           rotAngleX += 2;  break;
     case 's':
           rotAngleX -= 2;  break;
           case 'a':
                 rotAngleY += 2; break;
     case 'd':
           rotAngleY -= 2; break;
           case 'q':
           rotAngleZ += 2; break;
     case 'e':
           rotAngleZ -= 2; break;
        case 'o':
           if(drop==0){      //kondisi jika kotak tidak jatuh ke tanah
           if (posXBadan > 4){ //kondisi jika tidak menabrak meja
                 posXBadan -= 1;
                 if (bring == 1 ) {
                              posXKotak -= 1;
                       }
           }
        }else{
           if  (posXBadan  >posXKotak+3){  //kondisi  jika  tidak  menabrak
kotak
                 posXBadan -= 1;
           }
        }
                 break;
        case 'p':
           posXBadan += 1;
           if (bring == 1){
                 posXKotak += 1;
           }
                 break;
        case 'g':
           gerakTangan = 1; break;
        case 't':
           if (posXBadan>=8){ //kondisi kotak telah diluar meja
                 drop = 1;
                 gerakTangan =3;
                 }
                 break;

        case 32:
           rotAngleX=rotAngleY=rotAngleZ=0;
                 posXBadan=10, posXKotak=0, posYBadan=7, posYKotak=6;

     rotTangan1=rotTangan2=rotTangan3=rotTangan4=kick=roll=gerakTangan=d
rop=hit=bring=grab=0;
                 break;
     case 27:
           exit(0);
        default:
```

```
            break;
    }glutPostRedisplay();
}

void pressKey(int k, int x, int y)
{
     // Fungsi ini akan dijalankan saat tombol keyboard ditekan dan belum
dilepas
     // Selama tombol ditekan, variabel angle dan move diubah => kamera
bergerak
     switch (k)
     {
           case GLUT_KEY_UP : deltaMove = 1;break;
     case GLUT_KEY_DOWN : deltaMove = -1;break;
     case GLUT_KEY_LEFT : deltaAngle = -0.01f;break;
     case GLUT_KEY_RIGHT : deltaAngle = 0.01f;break;
     }
}

void releaseKey(int key, int x, int y)
{
     // Fungsi ini akan dijalankan saat tekanan tombol keyboard dilepas
     // Saat tombol dilepas, variabel angle dan move diset nol => kamera
berhenti
     switch (key)
     {
     case GLUT_KEY_UP : if (deltaMove > 0)
                deltaMove = 0;
                break;
     case GLUT_KEY_DOWN : if (deltaMove < 0)
                deltaMove = 0;
                break;
     case GLUT_KEY_LEFT : if (deltaAngle < 0.0f)
           deltaAngle = 0.0f;
                break;
     case GLUT_KEY_RIGHT : if (deltaAngle > 0.0f)
                deltaAngle = 0.0f;
                break;
     }
}

void lighting(){
     // Fungsi mengaktifkan pencahayaan
     glEnable(GL_DEPTH_TEST);
     glDepthFunc(GL_LESS);
     glEnable(GL_LIGHT0);
     glEnable(GL_NORMALIZE);
     glEnable(GL_COLOR_MATERIAL);
     glEnable(GL_LIGHTING);
     glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
     glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
     glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
     glLightfv(GL_LIGHT0, GL_POSITION, light_position);
     glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
     glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
     glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
     glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
}

//-----------------------------------------------------------------------
void Grid()
{
     // Fungsi untuk membuat grid di "lantai"
     double i;
     const float Z_MIN = -50, Z_MAX = 50;
     const float X_MIN = -50, X_MAX = 50;
     const float gap = 2;
```

```
     glColor3f(0.5, 0.5, 0.5);
     glBegin(GL_LINES);
    for(i=Z_MIN;i<Z_MAX;i+=gap)
    {
       glVertex3f(i, 0, Z_MIN);
       glVertex3f(i, 0, Z_MAX);
    }

    for(i=X_MIN;i<X_MAX;i+=gap)
    {
       glVertex3f(X_MIN, 0, i);
       glVertex3f(X_MAX, 0, i);
    }
     glEnd();
}

void Grid2()
{
     glColor3f(1.0f,1.0f,1.0f);
     glBegin(GL_QUADS);
           glVertex3f(-50, 0, 50);
         glVertex3f(-50, 0, -50);
         glVertex3f(50, 0, -50);
           glVertex3f(50, 0, 50);
     glEnd();
}

void Balok(float panjang,float lebar,float tinggi)
{
     glPushMatrix();
           float p=panjang/2;
           float l=lebar/2;
           float t=tinggi/2;
           //depan
         glBegin(GL_QUADS);
              glVertex3f(-p,0,l);
              glVertex3f(p,0,l);
              glVertex3f(p,-t*2,l);
              glVertex3f(-p,-t*2,l);
           glEnd();

       // belakang
           glBegin(GL_QUADS);
              glVertex3f(-p,0,-l);
              glVertex3f(p,0,-l);
              glVertex3f(p,-t*2,-l);
              glVertex3f(-p,-t*2,-l);
           glEnd();

           // atas
           glBegin(GL_QUADS);
           glTexCoord2f(0.0f, 0.0f);
              glVertex3f(-p,0,-l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(p,0,-l);
              glTexCoord2f(1.0f, 0.0f);
              glVertex3f(p,0,l);
              glTexCoord2f(1.0f, 1.0f);
              glVertex3f(-p,0,l);
           glEnd();

           // bawah
           glBegin(GL_QUADS);
                 glTexCoord2f(0.0f, 0.0f);
              glVertex3f(-p,-t*2,-l);
              glTexCoord2f(0.0f, 1.0f);
              glVertex3f(p,-t*2,-l);
```

```
                glTexCoord2f(1.0f, 0.0f);
                glVertex3f(p,-t*2,l);
                glTexCoord2f(1.0f, 1.0f);
                glVertex3f(-p,-t*2,l);
            glEnd();

            // kanan
          glBegin(GL_QUADS);
            glTexCoord2f(0.0f, 0.0f);
                glVertex3f(-p,-t*2,-l);
                glTexCoord2f(1.0f, 0.0f);
                glVertex3f(-p,-t*2,l);
                glTexCoord2f(0.0f, 1.0f);
                glVertex3f(-p,0,l);
                glTexCoord2f(1.0f, 1.0f);
                glVertex3f(-p,0,-l);
            glEnd();

            // kiri
            glBegin(GL_QUADS);
            glTexCoord2f(0.0f, 0.0f);
                glVertex3f(p,-t*2,-l);
                glTexCoord2f(1.0f, 0.0f);
                glVertex3f(p,-t*2,l);
                glTexCoord2f(0.0f, 1.0f);
                glVertex3f(p,0,l);
                glTexCoord2f(1.0f, 1.0f);
                glVertex3f(p,0,-l);
            glEnd();
      glPopMatrix();

}

//================================================
//prosedur kondisi merubah posisi kotak
void perubahKotak(){
      if(drop==1  &&  grab==1){  //kondisi  saat  ada  perintah  jatuh  saat
dipegang
            if(posYKotak>=3){       //batas jatuh
                  posYKotak-=0.01;
            }
            if (posYKotak<3){ //saat sudah berhenti
                  bring=0;
                  hit=0;
                  grab=0;
            }
      }
}

//prosedur kondisi merubah tangan dan badan
void pengubahTangan(){
      if (posXBadan!=4){      //kondisi jika tubuh menyentuh kotak bawah
                  hit = 0;
            }else {
                  hit = 1;
            }

      if  (hit==1  &&  grab==1){   //kondisi  jika  menyentuh  kotak  dan  ada
perintah mengambil
            bring = 1;
      }
      //memegang
      if(gerakTangan == 1){   //kondisi pelebarkan tangan samping
            if (rotTangan1>=-90){
                  rotTangan1 -=0.1;
            }
            if (rotTangan1 <-90){
```

```
                  gerakTangan = 2;
            }
      }
      if (gerakTangan==2){    //kondisi merapatkan tangan depan
            if (rotTangan2>=-90){
                  rotTangan2 -=0.1;
            }
            if (rotTangan2 <-90 && hit==1){
                  grab = 1;
            }
      }
      //melepas
      if (gerakTangan==3){  //konsisi melebarkan tangan depan
            if (rotTangan2<=0){
                  rotTangan2 +=0.1;
            }
            if (rotTangan2 >0){
                  gerakTangan = 4;
            }
      }
      if (gerakTangan==4){   //kondisi merapatkan tangan samping
            if (rotTangan1<=0){
                  rotTangan1 +=0.1;
            }
            if (rotTangan1 >0){
                  gerakTangan = 0;
            }
      }
}
void Object()
{
      //Meja
      glPushMatrix();
            glColor3f(0.1,0.1,0.2);
            glTranslatef(0,3,0);
            Balok(5,5,3);
      glPopMatrix();
      // Kotak pink
      glPushMatrix();
            perubahKotak();
            glColor3f(0.8,0.3,0.3);
            glTranslatef(posXKotak,posYKotak,0);
            Balok(3,3,3);
      glPopMatrix();

      glPushMatrix();
            pengubahTangan();
            // objek yang dirubah
            glColor3f(0.3,0.3,0.8);
            glTranslatef(posXBadan,posYBadan,0);
            Balok(3,3,7); //badan
            // tangan kiri
            glPushMatrix();
                  glColor3f(0.2,0.5,0.2);
                  glTranslatef(0,-2,2.5);
                  glRotatef(rotTangan1,1,0,0);
                  glRotatef(rotTangan2,0,0,1);
                  Balok(2,2,4);
            glPopMatrix();
            //tangan kanan
            glPushMatrix();
                  glColor3f(0.2,0.5,0.2);
                  glTranslatef(0,-2,-2.5);
                  glRotatef(-rotTangan1,1,0,0);
                  glRotatef(rotTangan2,0,0,1);
                  Balok(2,2,4);
            glPopMatrix();
```

```
      glPopMatrix();
}

void display() {
      glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
      // Kalau move dan angle tidak nol, gerakkan kamera...
      if (deltaMove)
      moveMeFlat(deltaMove);
      if (deltaAngle) {
            angle += deltaAngle;
      orientMe(angle);        }

      glPushMatrix();
            glRotated(rotAngleX+10, 1, 0, 0);
          glRotated(rotAngleY, 0, 1, 0);
          glRotated(rotAngleZ, 0, 0, 1);
            // Gambar grid
            Grid();
            Grid2();
            // Gambar objek di sini...
            Object();
      glPopMatrix();
      glFlush();
      glutSwapBuffers();
}

int main(int argc, char **argv)
{
      glutInit(&argc, argv);
      glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
      glutInitWindowPosition(100,100);
      glutInitWindowSize(640,480);
      glutCreateWindow("Grab");
      glutSpecialFunc(pressKey);
      glutSpecialUpFunc(releaseKey);
      glutDisplayFunc(display);
      glutKeyboardFunc(keyboard);
      glutIdleFunc(display);
      glutReshapeFunc(Reshape);
      lighting();
      init();
      glutMainLoop();
      return(0);
}
```