**TUGAS BESAR GRAFIKA KOMPUTER**

**PEMBUATAN ANIMASI SEPEDA 3D**

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



**Disusun Oleh :**

Irfan Budi Santoso - 10109903

Budi Arisandi - 10109911

Hendra - 10109928

Kelas : IF17K

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

**BANDUNG**

**2013**

**BAB I**

**PENDAHULUAN**

* 1. **LATAR BELAKANG**

Pada saat ini grafika komputer sudah digunakan pada bidang sains, Design, engineering, kedokteran, bisnis, industri, pemerintahan, seni, hiburan, iklan, pendidikan, dan lain-lain. Oleh karena itu penulis mengambil judul “ PEMBUATAN ANIMASI SEPEDA 3D”. Dalam rangka mengimplementasikan grafika komputer di dalam salah satu bidang yaitu bidang Design.

Untuk pembuatan animasi sepeda 3D, penulis merancang detail sepeda tersebut dengan beberapa objek yang ada di dalam sepeda itu sendiri yaitu diantaranya :

1. Roda Set yang di dalamnya terdapat objek : Ban, Jeruji,tromol,Velg
2. Gear set yang di dalamnya terdapat objek : Gear Depan, Gear belakang, rantai
3. Pedal set yang di dalamnya terdapat pedal kanan dan kiri dan Bahu pedal
4. Setang set yang di dalamnya terdapat stang dan handle stang
5. Rangka Utama yang berfungsi untuk tempat dimana menempelnya objek objek pembangun sepeda yang sudah disebutkan diatas.

Latar belakang penulis mengambil tema Animasi Sepeda 3d yaitu penulis ingin mencoba mengkombinasikan beberapa bentuk benda yang akan kami buat dalam bentuk 3D sehingga bentuk-bentuk tersebut dapat disajikan ke dalam satu tampilan layar dan bergerak .

* 1. **IDENTIFIKASI MASALAH**

Berdasarkan latar belakang dapat didefinisikan beberapa masalah yaitu :

1. Bagaimana cara memodelkan Sepeda 3D dalam bentuk 3D pada openGL dengan menggunakan IDE Microsoft Visual C++ 6.0 yang mendukung openGL.
2. Bagaimana membuat animasi Sepeda 3D sehingga Objek yang dibuat tersebut dapat bergerak.

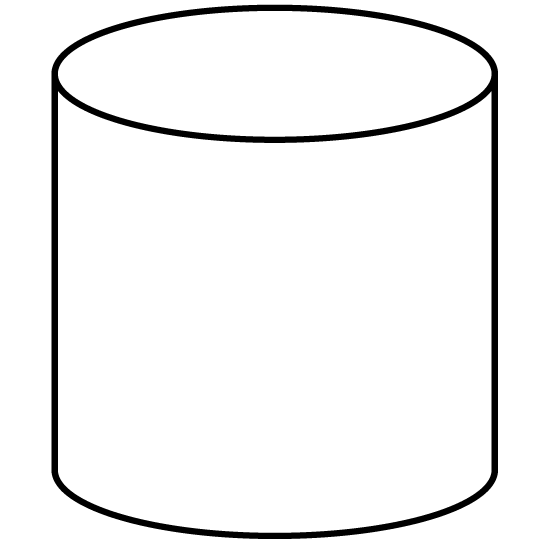
* 1. **MAKSUD DAN TUJUAN**

Berdasarkan permasalahan yang ada, maka maksud dari penulisan tugas besar ini adalah untuk membangun Animasi Sepeda 3D dengan berbasis desktop. Sedangkan tujuan yang akan dicapai adalah :

1. Menciptakan Sepeda dalam bentuk 3D.
2. Membuat Animasi sepeda 3D .
3. Mengaplikasikan materi openGL dari mata kuliah Grafika Komputer.
   1. **Objek Pembangun**

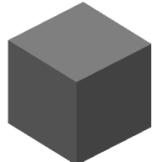
Objek pembangun yang akan digunakan dalam pemodelan Animasi sepeda 3D adalah

* 1. glucylinder



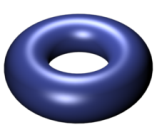
Object ini akan mendominasi pembuatan sepeda 3d karena sepeda pada umunnya lebih banyak menggunakan material yang berbentuk Tabung / Cylinder

* 1. glutsolidcube



Kubus digunakan untuk pembuatan Sadel, Bahu Pedal dan Pedal

* 1. glutsolidtorus



Object Torus digunakan untuk pembuatan ban

1. glVertex

Digunakan untuk membangun gear, yaitu dengan menentukan titik-object yang akan dibangun secara spesifik .

* 1. **BATASAN MASALAH**

Dalam pembahasan tugas Animasi Sepeda 3D dibatasi sebagai berikut :

1. Bahasa yang digunakan bahasa pemrograman C++
2. IDE yang digunakan Visual C++ 6.0 dengan openGL
3. Program ini hanya menampilkan Animasi sepeda 3D
4. Sepeda yang akan dibangun hanya dapat bergerak Maju dan Setang dapat digerakkan ke Kanan dan Kiri

**BAB II**

**KONTRIBUSI TUGAS**

1. **Source Code Irfan Budi Santoso**

**void rangka()**

{

glColor3f(1.0f,0.0f,0.0f);

glPushMatrix();

glPushMatrix();

glColor3f(0.0f,1.0f,0.0f);

glPushMatrix();

glTranslatef(0.0f,0.0f,0.06f);

glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);

gear(0.08f,0.3f,0.03f,30,0.03f);

glPopMatrix();

glColor3f(1.0f,0.0f,0.0f); //warna rumah

glTranslatef(0.0f,0.0f,-0.2f);

Silinder1(0.08f,0.32f);

glPopMatrix();

//sambungan fork belakang edit lagi

glPushMatrix();

glColor3f(1.0f,0.0f,0.0f);

glTranslatef(-0.87f,0.85f,-0.16f);

Silinder1(0.05f,0.32f);

glPopMatrix();

//TUTUP LAS SAMBUNGAN belakang kanan

glPushMatrix();

glTranslatef(-0.87f, 0.85f, 0.16f);

tutuplas();

glPopMatrix();

//TUTUP LAS SAMBUNGAN belakang kiri

glPushMatrix();

glTranslatef(-0.87f, 0.85f, -0.16f);

tutuplas();

glPopMatrix();

//rekaan sambungan fork belakang tengah

glPushMatrix();

glColor3f(1.0f,0.0f,0.0f);

glTranslatef(-0.475f,0.0f,-0.16f);

Silinder1(0.05f,0.32f);

glPopMatrix();

//TUTUP LAS fork belakang tengah kanan

glPushMatrix();

glTranslatef(-0.475f,0.0f,-0.16f);

tutuplas();

glPopMatrix();

//TUTUP LAS fork belakang tengah kiri

glPushMatrix();

glTranslatef(-0.475f,0.0f,0.16f);

tutuplas();

glPopMatrix();

//TUTUP LAS dekat pedal kiri

glPushMatrix();

glTranslatef(0.0f,0.0f,-0.2f);

glScalef(0.04f,0.04f,0.04f);

glRotatef(90.0f,0.0f,90.0f,0.0f);

glutSolidSphere(2.0, 2.0, 100.0);

glPopMatrix();

//TUTUP LAS dekat pedal kanan

glPushMatrix();

glTranslatef(0.0f,0.0f,0.1f);

glScalef(0.04f,0.04f,0.04f);

glRotatef(90.0f,0.0f,90.0f,0.0f);

glutSolidSphere(2.0, 2.0, 100.0);

glPopMatrix();

glRotatef(SUDUT\_KANAN,0.0f,0.0f,1.0f);

Silinder2(DMT\_ROD,1.67f); //rangka ubahan

glRotatef(SUDUT\_TENGAH-SUDUT\_KANAN,0.0f,0.0f,1.0f);

Silinder2(DMT\_ROD,TENGAH\_ROD);

glColor3f(1.0f,1.0f,0.0f);

glTranslatef(TENGAH\_ROD,0.0f,0.0f);

glRotatef(-SUDUT\_TENGAH,0.0f,0.0f,1.0f);

glScalef(0.3f,DMT\_ROD,0.25f);

jok();

glColor3f(1.0f,0.0f,0.0f);

glPopMatrix();

glPushMatrix();

glRotatef(-180.0f,0.0f,1.0f,0.0f);

Silinder2(DMT\_ROD,PHB\_B);

glPushMatrix();

glTranslatef(0.5f,0.0f,OFSET\_RODA);

Silinder2(DMT\_ROD,DMT\_RODA+LB\_BAN);

glPopMatrix();

glPushMatrix();

glTranslatef(0.5f,0.0f,-OFSET\_RODA);

Silinder2(DMT\_ROD,DMT\_RODA+LB\_BAN);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glTranslatef(-(PHB\_B+DMT\_RODA+LB\_BAN),0.0f,0.0f);

glPushMatrix();

glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);

ban();

glColor3f(0.0f,1.0f,0.0f);

gear(0.03f,0.15f,0.03f,20,0.03f);

glColor3f(1.0f,0.0f,0.0f);

glPopMatrix();

glRotatef(SUDUT\_KIRI,0.0f,0.0f,1.0f);

glPushMatrix();

glTranslatef(0.0f,0.0f,-OFSET\_RODA);

Silinder2(DMT\_ROD,LEN\_RODA);

glPopMatrix();

glPushMatrix();

glTranslatef(0.0f,0.0f,OFSET\_RODA);

Silinder2(DMT\_ROD,LEN\_RODA);

glPopMatrix();

glTranslatef(LEN\_RODA,0.0f,0.0f);

Silinder2(DMT\_ROD,ENGSEL\_ROD);

glTranslatef(ENGSEL\_ROD,0.0f,0.0f);

glRotatef(-SUDUT\_KIRI,0.0f,0.0f,1.0f);

Silinder2(DMT\_ROD,LEN\_ATAS);

glTranslatef(LEN\_ATAS,0.0f,0.0f);

glRotatef(-FRONT\_INCLINE,0.0f,0.0f,1.0f);

glPushMatrix();

glTranslatef(-0.1f,0.0f,0.0f);

Silinder2(DMT\_ROD,0.45f);

glPopMatrix();

glPushMatrix();

glRotatef(-stir,1.0f,0.0f,0.0f);

glTranslatef(-0.3f,0.0f,0.0f);

glPushMatrix();

glRotatef(FRONT\_INCLINE,0.0f,0.0f,1.0f);

glPushMatrix();

glTranslatef(0.0f,0.0f,-HANDEL\_ROD/2);

Silinder1(DMT\_ROD,HANDEL\_ROD);

glPopMatrix();

glPushMatrix();

glColor3f(1.0f,1.0f,0.0f);

glTranslatef(0.0f,0.0f,-HANDEL\_ROD/2);

Silinder1(0.07f,HANDEL\_ROD/4);

glTranslatef(0.0f,0.0f,HANDEL\_ROD\*3/4);

Silinder1(0.07f,HANDEL\_ROD/4);

glColor3f(1.0f,0.0f,0.0f);

glPopMatrix();

glPopMatrix();

glPushMatrix();

Silinder2(DMT\_ROD,ENGSEL\_ROD);

glTranslatef(ENGSEL\_ROD,0.0f,0.0f);

glRotatef(SUDUT\_ENGSEL,0.0f,0.0f,1.0f);

glPushMatrix();

glTranslatef(0.0f,0.0f,OFSET\_RODA);

Silinder2(DMT\_ROD,ENGSEL\_RODS);

glPopMatrix();

glPushMatrix();

glTranslatef(0.0f,0.0f,-OFSET\_RODA);

Silinder2(DMT\_ROD,ENGSEL\_RODS);

glPopMatrix();

//rekaan tutup fork depan

glPushMatrix();

glColor3f(1.0f,0.0f,0.0f);

glTranslatef(0.0f,0.0f,-0.16f);

Silinder1(0.05f,0.32f);

glPopMatrix();

//TUTUP LAS SAMBUNGAN FORK KIRI

glPushMatrix();

glTranslatef(0.0f, 0.0f, -0.16f);

tutuplas();

glPopMatrix();

//TUTUP LAS SAMBUNGAN FORK KANAN

glPushMatrix();

glTranslatef(0.0f, 0.0f, 0.16f);

tutuplas();

glPopMatrix();

//tTUTUP LAS handle kiri

glPushMatrix();

glTranslatef(-0.7f,0.09f,0.6f);

glScalef(0.03f,0.04f,0.04f);

glRotatef(90.0f,0.0f,90.0f,0.0f);

glutSolidSphere(2.0, 2.0, 100.0);

glPopMatrix();

//TUTUP LAS handle kiri

glPushMatrix();

glTranslatef(-0.7f,0.09f,-0.6f);

glScalef(0.03f,0.04f,0.04f);

glRotatef(90.0f,0.0f,90.0f,0.0f);

glutSolidSphere(2.0, 2.0, 100.0);

glPopMatrix();

glTranslatef(ENGSEL\_RODS,0.0f,0.0f);

glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);

ban();

glPopMatrix();

glPopMatrix();

glPopMatrix();

}

**void idle**(void)

{

updateScene();

glutPostRedisplay();

}

**void reset**()

{

anglex=angley=anglez=0.0f;

sudutpedal=stir=1.0f;

Mouse=GLUT\_UP;

sudutpedal=spid=stir=0.0f;

camx=camy=1.0f;

camz=5.0f;

xpos=zpos=0.0f;

direction=-10.0f;

}

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

{

GLfloat r=0.0f;

switch(key)

{

case 'r':

reset();

break;

case 'a':

if(stir < BTS\_HENDEL)

stir += PLUS\_STIR;

break;

case 'd':

if(stir > -BTS\_HENDEL)

stir -= PLUS\_STIR;

break;

case '+':

spid += PLUS\_SPID;

break;

case '-':

spid -= PLUS\_SPID;

break;

case 27:

exit(1);

}

sudutpedal += spid;

if(spid < 0.0f)

spid = 0.0f;

if(sudutpedal < 0.0f)

sudutpedal = 0.0f;

if(sudutpedal >= 360.0f)

sudutpedal -= 360.0f;

glutPostRedisplay();

}

**void mouse**(int button,int state,int x,int y)

{

switch(button)

{

case GLUT\_LEFT\_BUTTON:

if(state==GLUT\_DOWN)

{

Mouse=GLUT\_DOWN;

prevx=x;

prevy=y;

}

if(state==GLUT\_UP)

{

Mouse=GLUT\_UP;

}

break;

case GLUT\_RIGHT\_BUTTON:

break;

}

glutPostRedisplay();

}

**void updateScene**()

{

GLfloat xDelta, zDelta;

GLfloat rotation;

GLfloat sin\_stir, cos\_stir;

if (-PLUS\_SPID < spid && spid < PLUS\_SPID) return;

if(spid < 0.0f)

sudutpedal = spid = 0.0f;

xDelta = spid\*cos(radians(direction + stir));

zDelta = spid\*sin(radians(direction + stir));

xpos += xDelta;

zpos -= zDelta;

sudutpedal = degrees(angleSum(radians(sudutpedal), spid/DMT\_RODA));

sin\_stir = sin(radians(stir));

cos\_stir = cos(radians(stir));

rotation = atan2(spid \* sin\_stir, PJG\_SPD + spid \* cos\_stir);

direction = degrees(angleSum(radians(direction),rotation));

}

1. **Source Code Budi Arisandi**

**void tutuplas**()

{

glPushMatrix();

glColor3f(1.0f,0.0f,0.0f);

glScalef(0.025f,0.025f,0.025f);

glRotatef(90.0f,0.0f,90.0f,0.0f);

glutSolidSphere(2.0, 2.0, 100.0);

glPopMatrix();

}

**void jok**()

{

glBegin(GL\_POLYGON);

glVertex3f(-0.1f, 1.0f, -0.5f);

glVertex3f( 1.0f, 1.0f, -0.3f);

glVertex3f( 1.0f, 1.0f, 0.3f);

glVertex3f(-0.1f, 1.0f, 0.5f);

glVertex3f(-0.5f, 1.0f, 1.0f);

glVertex3f(-1.0f, 1.0f, 1.0f);

glVertex3f(-1.0f, 1.0f, -1.0f);

glVertex3f(-0.5f, 1.0f, -1.0f);

glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-0.1f, -1.0f, -0.5f);

glVertex3f( 1.0f, -1.0f, -0.3f);

glVertex3f( 1.0f, -1.0f, 0.3f);

glVertex3f(-0.1f, -1.0f, 0.5f);

glVertex3f(-0.5f, -1.0f, 1.0f);

glVertex3f(-1.0f, -1.0f, 1.0f);

glVertex3f(-1.0f, -1.0f, -1.0f);

glVertex3f(-0.5f, -1.0f, -1.0f);

glEnd();

glBegin(GL\_QUADS);

glVertex3f(1.0f,1.0f,-0.3f);

glVertex3f(1.0f,1.0f,0.3f);

glVertex3f(1.0f,-1.0f,0.3f);

glVertex3f(1.0f,-1.0f,-0.3f);

glVertex3f(1.0f,1.0f,0.3f);

glVertex3f(-0.1f,1.0f,0.5f);

glVertex3f(-0.1f,-1.0f,0.5f);

glVertex3f(1.0f,-1.0f,0.3f);

glVertex3f(1.0f,1.0f,-0.3f);

glVertex3f(-0.1f,1.0f,-0.5f);

glVertex3f(-0.1f,-1.0f,-0.5f);

glVertex3f(1.0f,-1.0f,-0.3f);

glVertex3f(-0.1f,1.0f,0.5f);

glVertex3f(-0.5f,1.0f,1.0f);

glVertex3f(-0.5f,-1.0f,1.0f);

glVertex3f(-0.1f,-1.0f,0.5f);

glVertex3f(-0.1f,1.0f,-0.5f);

glVertex3f(-0.5f,1.0f,-1.0f);

glVertex3f(-0.5f,-1.0f,-1.0f);

glVertex3f(-0.1f,-1.0f,-0.5f);

glVertex3f(-0.5f,1.0f,1.0f);

glVertex3f(-1.0f,1.0f,1.0f);

glVertex3f(-1.0f,-1.0f,1.0f);

glVertex3f(-0.5f,-1.0f,1.0f);

glVertex3f(-0.5f,1.0f,-1.0f);

glVertex3f(-1.0f,1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,-1.0f);

glVertex3f(-0.5f,-1.0f,-1.0f);

glVertex3f(-1.0f,1.0f,1.0f);

glVertex3f(-1.0f,1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,1.0f);

glEnd();

}

**void pedal**()

{

glColor3f(0.0f,0.0f,1.0f);

glPushMatrix();

glTranslatef(0.0f,0.0f,0.105f);

glRotatef(-sudutpedal,0.0f,0.0f,1.0f);

glTranslatef(0.25f,0.0f,0.0f);

glPushMatrix();

glScalef(0.5f,0.1f,0.1f);

glutSolidCube(1.0f);

glPopMatrix();

glPushMatrix();

glTranslatef(0.25f,0.0f,0.15f);

glRotatef(sudutpedal,0.0f,0.0f,1.0f);

glScalef(0.2f,0.02f,0.3f);

glutSolidCube(1.0f);

glPopMatrix();

glPopMatrix();

glPushMatrix();

glTranslatef(0.0f,0.0f,-0.23f);

glRotatef(180.0f-sudutpedal,0.0f,0.0f,1.0f);

glTranslatef(0.25f,0.0f,0.0f);

glPushMatrix();

glScalef(0.5f,0.1f,0.1f);

glutSolidCube(1.0f);

glPopMatrix();

glPushMatrix();

glTranslatef(0.25f,0.0f,-0.15f);

glRotatef(sudutpedal-180.0f,0.0f,0.0f,1.0f);

glScalef(0.2f,0.02f,0.3f);

glutSolidCube(1.0f);

glPopMatrix();

glPopMatrix();

glColor3f(1.0f,0.0f,0.0f);

}

**void ban**(void)

{

int i;

// Pelek

glColor3f(0.5f,1.0f,1.0f);

glutSolidTorus(0.06f,0.92f,4,30);

// Silinder tengah

glColor3f(1.0f,1.0f,0.5f);

glPushMatrix();

glTranslatef(0.0f,0.0f,-0.06f);

Silinder1(0.02f,0.12f);

glPopMatrix();

glutSolidTorus(0.02f,0.02f,3,20);

// Jari Jari

glColor3f(1.0f,1.0f,1.0f);

for(i=0;i<JML\_JARI2;++i)

{

glPushMatrix();

glRotatef(i\*SUDUT\_JARI2,0.0f,0.0f,1.0f);

glBegin(GL\_LINES);

glVertex3f(0.0f,0.02f,0.0f);

glVertex3f(0.0f,0.86f,0.0f);

glEnd();

glPopMatrix();

}

// BAN

glColor3f(0.0f,0.0f,0.5f);

glutSolidTorus(LB\_BAN-0.03,DMT\_RODA,10,30);

glColor3f(1.0f,0.0f,0.0f);

}

**void gerak**(int x,int y)

{

if(Mouse==GLUT\_DOWN)

{

int deltax,deltay;

deltax=prevx-x;

deltay=prevy-y;

anglex += 0.5\*deltax;

angley += 0.5\*deltay;

if(deltax!=0 && deltay!=0)

anglez += 0.5\*sqrt(deltax\*deltax + deltay\*deltay);

if(anglex < 0)

anglex+=360.0;

if(angley < 0)

angley+=360.0;

if(anglez < 0)

anglez += 360.0;

if(anglex > 360.0)

anglex-=360.0;

if(angley > 360.0)

angley-=360.0;

if(anglez > 360.0)

anglez-=360.0;

}

else

{

Mouse=GLUT\_UP;

}

prevx=x;

prevy=y;

glutPostRedisplay();

}

1. **Source Code Hendra**

**void Silinder1**(GLfloat radius,GLfloat length)

{

GLUquadricObj \*cylinder;

cylinder=gluNewQuadric();

glPushMatrix();

glTranslatef(0.0f,0.0f,0.0f);

gluCylinder(cylinder,radius,radius,length,15,5);

glPopMatrix();

}

**void Silinder2**(GLfloat radius,GLfloat length)

{

glPushMatrix();

glRotatef(90.0f,0.0f,1.0f,0.0f);

Silinder1(radius,length);

glPopMatrix();

}

**void rantai**()

{

GLfloat depth;

static int mode=0;

glColor3f(0.0f,1.0f,0.0f);

glEnable(GL\_LINE\_STIPPLE);

mode=(mode+1)%2;

if(mode==0 && spid>0)

glLineStipple(1,0x1c47);

else if(mode==1 && spid>0)

glLineStipple(1,0x00FF);

glBegin(GL\_LINES);

for(depth=0.06f;depth<=0.12f;depth+=0.01f)

{

glVertex3f(-1.6f,0.15f,DMT\_ROD);

glVertex3f(0.0f,0.3f,depth);

glVertex3f(-1.6f,-0.15f,DMT\_ROD);

glVertex3f(0.0f,-0.3f,depth);

}

glEnd();

glDisable(GL\_LINE\_STIPPLE);

}

**void special**(int key,int x,int y)

{

switch(key)

{

case GLUT\_KEY\_UP:

camz -= 0.1f;

break;

case GLUT\_KEY\_DOWN:

camz += 0.1f;

break;

case GLUT\_KEY\_LEFT:

camx -= 0.1f;

break;

case GLUT\_KEY\_RIGHT:

camx += 0.1f;

break;

}

glutPostRedisplay();

}

**void alas**(void)

{

GLfloat i;

glColor3f(0.0f,1.0f,0.0f);

glBegin(GL\_LINES);

for(i=-100.0f ; i<100.0f ; i += 1.0f)

{

glVertex3f(-100.0f,-DMT\_RODA,i);

glVertex3f( 100.0f,-DMT\_RODA,i);

glVertex3f(i,-DMT\_RODA,-100.0f);

glVertex3f(i,-DMT\_RODA,100.0f);

}

glEnd();

}

**void gear**( GLfloat inner\_radius, GLfloat outer\_radius, GLfloat width,

GLint teeth, GLfloat tooth\_depth )

{

GLint i;

GLfloat r0, r1, r2;

GLfloat angle, da;

GLfloat u, v, len;

const double pi = 3.14159264;

r0 = inner\_radius;

r1 = outer\_radius - tooth\_depth/2.0;

r2 = outer\_radius + tooth\_depth/2.0;

da = 2.0\*pi / teeth / 4.0;

glShadeModel( GL\_FLAT );

glNormal3f( 0.0, 0.0, 1.0 );

glBegin( GL\_QUAD\_STRIP );

for (i=0;i<=teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );

glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );

glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );

}

glEnd();

glBegin( GL\_QUADS );

da = 2.0\*pi / teeth / 4.0;

for (i=0;i<teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );

glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), width\*0.5 );

glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), width\*0.5 );

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );

}

glEnd();

glNormal3f( 0.0, 0.0, -1.0 );

glBegin( GL\_QUAD\_STRIP );

for (i=0;i<=teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );

glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );

glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );

}

glEnd();

glBegin( GL\_QUADS );

da = 2.0\*pi / teeth / 4.0;

for (i=0;i<teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );

glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), -width\*0.5 );

glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), -width\*0.5 );

glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );

}

glEnd();

glBegin( GL\_QUAD\_STRIP );

for (i=0;i<teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );

glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );

u = r2\*cos(angle+da) - r1\*cos(angle);

v = r2\*sin(angle+da) - r1\*sin(angle);

len = sqrt( u\*u + v\*v );

u /= len;

v /= len;

glNormal3f( v, -u, 0.0 );

glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), width\*0.5 );

glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), -width\*0.5 );

glNormal3f( cos(angle), sin(angle), 0.0 );

glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), width\*0.5 );

glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), -width\*0.5 );

u = r1\*cos(angle+3\*da) - r2\*cos(angle+2\*da);

v = r1\*sin(angle+3\*da) - r2\*sin(angle+2\*da);

glNormal3f( v, -u, 0.0 );

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );

glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );

glNormal3f( cos(angle), sin(angle), 0.0 );

}

glVertex3f( r1\*cos(0.0), r1\*sin(0.0), width\*0.5 );

glVertex3f( r1\*cos(0.0), r1\*sin(0.0), -width\*0.5 );

glEnd();

glShadeModel( GL\_SMOOTH );

glBegin( GL\_QUAD\_STRIP );

for (i=0;i<=teeth;i++) {

angle = i \* 2.0\*pi / teeth;

glNormal3f( -cos(angle), -sin(angle), 0.0 );

glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );

glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );

}

glEnd();

}

**BAB III**

**IMPLEMENTASI PROGRAM**

Sourcecode Program OpenGL :

|  |
| --- |
| **Sepeda.h**  #include<GL/glut.h>  #include <GL/glu.h>  #include<stdio.h>  #include<stdlib.h>  #include<math.h>  #define ENGSEL\_ROD 0.7f  #define ENGSEL\_RODS 1.12f  #define SUDUT\_ENGSEL 8.0f  #define PI 3.14159  #define LEBAR 600  #define TINGGI 600  #define PJG\_SPD 3.3f  #define DMT\_ROD 0.05f  #define JML\_JARI2 20  #define SUDUT\_JARI2 18  #define DMT\_RODA 1.0f  #define LB\_BAN 0.08f  #define KANAN\_ROD 1.6f //  #define SUDUT\_KANAN 48.0f  #define TENGAH\_ROD 1.7f  #define SUDUT\_TENGAH 106.0f  #define PHB\_B 0.5f  #define SUDUT\_KIRI 50.0f  #define OFSET\_RODA 0.11f  #define LEN\_RODA 1.1f  #define LEN\_ATAS 1.5f  #define HANDEL\_ROD 1.2f  #define FRONT\_INCLINE 70.0f //  #define BTS\_HENDEL 70.0f  #define PLUS\_STIR 2.0f  #define PLUS\_SPID 0.05f  GLfloat sudutpedal, spid, stir;  GLfloat camx,camy,camz;  GLfloat anglex,angley,anglez;  int prevx,prevy;  GLenum Mouse;  GLfloat Abs(GLfloat);  GLfloat degrees(GLfloat);  GLfloat radians(GLfloat);  GLfloat angleSum(GLfloat, GLfloat);  GLfloat xpos,zpos,direction;  void init(void);  void display(void);  void reshape(int w,int h);  void glSetupFuncs(void);  void reset(void);  void gear( GLfloat inner\_radius, GLfloat outer\_radius, GLfloat width,GLint teeth, GLfloat tooth\_depth );  void silinder1(GLfloat radius,GLfloat length);  void silinder2(GLfloat radius,GLfloat length);  void help(void);  void pedal(void);  void jok(void);  void rantai(void);  void alas(void);  **Sepeda.cpp**  #include "sepeda.h"  void alas(void)  {  GLfloat i;  glColor3f(0.0f,1.0f,0.0f);  glBegin(GL\_LINES);  for(i=-100.0f ; i<100.0f ; i += 1.0f)  {  glVertex3f(-100.0f,-DMT\_RODA,i);  glVertex3f( 100.0f,-DMT\_RODA,i);  glVertex3f(i,-DMT\_RODA,-100.0f);  glVertex3f(i,-DMT\_RODA,100.0f);  }  glEnd();  }  void updateScene()  {  GLfloat xDelta, zDelta;  GLfloat rotation;  GLfloat sin\_stir, cos\_stir;  if (-PLUS\_SPID < spid && spid < PLUS\_SPID) return;  if(spid < 0.0f)  sudutpedal = spid = 0.0f;  xDelta = spid\*cos(radians(direction + stir));  zDelta = spid\*sin(radians(direction + stir));  xpos += xDelta;  zpos -= zDelta;  sudutpedal = degrees(angleSum(radians(sudutpedal), spid/DMT\_RODA));  sin\_stir = sin(radians(stir));  cos\_stir = cos(radians(stir));  rotation = atan2(spid \* sin\_stir, PJG\_SPD + spid \* cos\_stir);  direction = degrees(angleSum(radians(direction),rotation));  }  GLfloat Abs(GLfloat a)  {  if(a < 0.0f)  return -a;  else  return a;  }  GLfloat degrees(GLfloat a)  {  return a\*180.0f/PI;  }  GLfloat radians(GLfloat a)  {  return a\*PI/180.0f;  }  void tutuplas()  {  glPushMatrix();  glColor3f(1.0f,0.0f,0.0f);  glScalef(0.025f,0.025f,0.025f);  glRotatef(90.0f,0.0f,90.0f,0.0f);  glutSolidSphere(2.0, 2.0, 100.0);  glPopMatrix();  }  void rantai()  {  GLfloat depth;  static int mode=0;  glColor3f(0.0f,1.0f,0.0f);  glEnable(GL\_LINE\_STIPPLE);  mode=(mode+1)%2;    if(mode==0 && spid>0)  glLineStipple(1,0x1c47);  else if(mode==1 && spid>0)  glLineStipple(1,0x00FF);    glBegin(GL\_LINES);  for(depth=0.06f;depth<=0.12f;depth+=0.01f)  {  glVertex3f(-1.6f,0.15f,DMT\_ROD);  glVertex3f(0.0f,0.3f,depth);    glVertex3f(-1.6f,-0.15f,DMT\_ROD);  glVertex3f(0.0f,-0.3f,depth);  }  glEnd();  glDisable(GL\_LINE\_STIPPLE);  }  void jok()  {  glBegin(GL\_POLYGON);  glVertex3f(-0.1f, 1.0f, -0.5f);  glVertex3f( 1.0f, 1.0f, -0.3f);  glVertex3f( 1.0f, 1.0f, 0.3f);  glVertex3f(-0.1f, 1.0f, 0.5f);  glVertex3f(-0.5f, 1.0f, 1.0f);  glVertex3f(-1.0f, 1.0f, 1.0f);  glVertex3f(-1.0f, 1.0f, -1.0f);  glVertex3f(-0.5f, 1.0f, -1.0f);  glEnd();  glBegin(GL\_POLYGON);  glVertex3f(-0.1f, -1.0f, -0.5f);  glVertex3f( 1.0f, -1.0f, -0.3f);  glVertex3f( 1.0f, -1.0f, 0.3f);  glVertex3f(-0.1f, -1.0f, 0.5f);  glVertex3f(-0.5f, -1.0f, 1.0f);  glVertex3f(-1.0f, -1.0f, 1.0f);  glVertex3f(-1.0f, -1.0f, -1.0f);  glVertex3f(-0.5f, -1.0f, -1.0f);  glEnd();  glBegin(GL\_QUADS);  glVertex3f(1.0f,1.0f,-0.3f);  glVertex3f(1.0f,1.0f,0.3f);  glVertex3f(1.0f,-1.0f,0.3f);  glVertex3f(1.0f,-1.0f,-0.3f);  glVertex3f(1.0f,1.0f,0.3f);  glVertex3f(-0.1f,1.0f,0.5f);  glVertex3f(-0.1f,-1.0f,0.5f);  glVertex3f(1.0f,-1.0f,0.3f);  glVertex3f(1.0f,1.0f,-0.3f);  glVertex3f(-0.1f,1.0f,-0.5f);  glVertex3f(-0.1f,-1.0f,-0.5f);  glVertex3f(1.0f,-1.0f,-0.3f);  glVertex3f(-0.1f,1.0f,0.5f);  glVertex3f(-0.5f,1.0f,1.0f);  glVertex3f(-0.5f,-1.0f,1.0f);  glVertex3f(-0.1f,-1.0f,0.5f);  glVertex3f(-0.1f,1.0f,-0.5f);  glVertex3f(-0.5f,1.0f,-1.0f);  glVertex3f(-0.5f,-1.0f,-1.0f);  glVertex3f(-0.1f,-1.0f,-0.5f);  glVertex3f(-0.5f,1.0f,1.0f);  glVertex3f(-1.0f,1.0f,1.0f);  glVertex3f(-1.0f,-1.0f,1.0f);  glVertex3f(-0.5f,-1.0f,1.0f);  glVertex3f(-0.5f,1.0f,-1.0f);  glVertex3f(-1.0f,1.0f,-1.0f);  glVertex3f(-1.0f,-1.0f,-1.0f);  glVertex3f(-0.5f,-1.0f,-1.0f);  glVertex3f(-1.0f,1.0f,1.0f);  glVertex3f(-1.0f,1.0f,-1.0f);  glVertex3f(-1.0f,-1.0f,-1.0f);  glVertex3f(-1.0f,-1.0f,1.0f);  glEnd();  }  void pedal()  {  glColor3f(0.0f,0.0f,1.0f);  glPushMatrix();  glTranslatef(0.0f,0.0f,0.105f);  glRotatef(-sudutpedal,0.0f,0.0f,1.0f);  glTranslatef(0.25f,0.0f,0.0f);  glPushMatrix();  glScalef(0.5f,0.1f,0.1f);  glutSolidCube(1.0f);  glPopMatrix();  glPushMatrix();  glTranslatef(0.25f,0.0f,0.15f);  glRotatef(sudutpedal,0.0f,0.0f,1.0f);  glScalef(0.2f,0.02f,0.3f);  glutSolidCube(1.0f);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,0.0f,-0.23f);  glRotatef(180.0f-sudutpedal,0.0f,0.0f,1.0f);  glTranslatef(0.25f,0.0f,0.0f);  glPushMatrix();  glScalef(0.5f,0.1f,0.1f);  glutSolidCube(1.0f);  glPopMatrix();  glPushMatrix();  glTranslatef(0.25f,0.0f,-0.15f);  glRotatef(sudutpedal-180.0f,0.0f,0.0f,1.0f);  glScalef(0.2f,0.02f,0.3f);  glutSolidCube(1.0f);  glPopMatrix();  glPopMatrix();  glColor3f(1.0f,0.0f,0.0f);  }  GLfloat angleSum(GLfloat a, GLfloat b)  {  a += b;  if (a < 0) return a+2\*PI;  else if (a > 2\*PI) return a-2\*PI;  else return a;  }  void Silinder1(GLfloat radius,GLfloat length)  {  GLUquadricObj \*cylinder;  cylinder=gluNewQuadric();  glPushMatrix();  glTranslatef(0.0f,0.0f,0.0f);  gluCylinder(cylinder,radius,radius,length,15,5);  glPopMatrix();  }  void Silinder2(GLfloat radius,GLfloat length)  {  glPushMatrix();  glRotatef(90.0f,0.0f,1.0f,0.0f);  Silinder1(radius,length);  glPopMatrix();  }  void ban(void)  {  int i;  // Pelek  glColor3f(0.5f,1.0f,1.0f);  glutSolidTorus(0.06f,0.92f,4,30);  // Silinder tengah  glColor3f(1.0f,1.0f,0.5f);  glPushMatrix();  glTranslatef(0.0f,0.0f,-0.06f);  Silinder1(0.02f,0.12f);  glPopMatrix();  glutSolidTorus(0.02f,0.02f,3,20);  // Jari Jari  glColor3f(1.0f,1.0f,1.0f);  for(i=0;i<JML\_JARI2;++i)  {  glPushMatrix();  glRotatef(i\*SUDUT\_JARI2,0.0f,0.0f,1.0f);  glBegin(GL\_LINES);  glVertex3f(0.0f,0.02f,0.0f);  glVertex3f(0.0f,0.86f,0.0f);  glEnd();  glPopMatrix();  }  // BAN  glColor3f(0.0f,0.0f,0.5f);  glutSolidTorus(LB\_BAN-0.03,DMT\_RODA,10,30);  glColor3f(1.0f,0.0f,0.0f);  }  void rangka()  {  glColor3f(1.0f,0.0f,0.0f);  glPushMatrix();  glPushMatrix();  glColor3f(0.0f,1.0f,0.0f);  glPushMatrix();  glTranslatef(0.0f,0.0f,0.06f);  glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);  gear(0.08f,0.3f,0.03f,30,0.03f);  glPopMatrix();  glColor3f(1.0f,0.0f,0.0f); //warna rumah  glTranslatef(0.0f,0.0f,-0.2f);  Silinder1(0.08f,0.32f);  glPopMatrix();  //sambungan fork belakang edit lagi  glPushMatrix();  glColor3f(1.0f,0.0f,0.0f);  glTranslatef(-0.87f,0.85f,-0.16f);  Silinder1(0.05f,0.32f);  glPopMatrix();  //TUTUP LAS SAMBUNGAN belakang kanan  glPushMatrix();  glTranslatef(-0.87f, 0.85f, 0.16f);  tutuplas();  glPopMatrix();  //TUTUP LAS SAMBUNGAN belakang kiri  glPushMatrix();  glTranslatef(-0.87f, 0.85f, -0.16f);  tutuplas();  glPopMatrix();  //rekaan sambungan fork belakang tengah  glPushMatrix();  glColor3f(1.0f,0.0f,0.0f);  glTranslatef(-0.475f,0.0f,-0.16f);  Silinder1(0.05f,0.32f);  glPopMatrix();  //TUTUP LAS fork belakang tengah kanan  glPushMatrix();  glTranslatef(-0.475f,0.0f,-0.16f);  tutuplas();  glPopMatrix();  //TUTUP LAS fork belakang tengah kiri  glPushMatrix();  glTranslatef(-0.475f,0.0f,0.16f);  tutuplas();  glPopMatrix();  //TUTUP LAS dekat pedal kiri  glPushMatrix();  glTranslatef(0.0f,0.0f,-0.2f);  glScalef(0.04f,0.04f,0.04f);  glRotatef(90.0f,0.0f,90.0f,0.0f);  glutSolidSphere(2.0, 2.0, 100.0);  glPopMatrix();  //TUTUP LAS dekat pedal kanan  glPushMatrix();  glTranslatef(0.0f,0.0f,0.1f);  glScalef(0.04f,0.04f,0.04f);  glRotatef(90.0f,0.0f,90.0f,0.0f);  glutSolidSphere(2.0, 2.0, 100.0);  glPopMatrix();  glRotatef(SUDUT\_KANAN,0.0f,0.0f,1.0f);  Silinder2(DMT\_ROD,1.67f); //rangka ubahan  glRotatef(SUDUT\_TENGAH-SUDUT\_KANAN,0.0f,0.0f,1.0f);  Silinder2(DMT\_ROD,TENGAH\_ROD);  glColor3f(1.0f,1.0f,0.0f);  glTranslatef(TENGAH\_ROD,0.0f,0.0f);  glRotatef(-SUDUT\_TENGAH,0.0f,0.0f,1.0f);  glScalef(0.3f,DMT\_ROD,0.25f);  jok();  glColor3f(1.0f,0.0f,0.0f);  glPopMatrix();  glPushMatrix();  glRotatef(-180.0f,0.0f,1.0f,0.0f);  Silinder2(DMT\_ROD,PHB\_B);  glPushMatrix();  glTranslatef(0.5f,0.0f,OFSET\_RODA);  Silinder2(DMT\_ROD,DMT\_RODA+LB\_BAN);  glPopMatrix();  glPushMatrix();  glTranslatef(0.5f,0.0f,-OFSET\_RODA);  Silinder2(DMT\_ROD,DMT\_RODA+LB\_BAN);  glPopMatrix();  glPopMatrix();  glPushMatrix();  glTranslatef(-(PHB\_B+DMT\_RODA+LB\_BAN),0.0f,0.0f);  glPushMatrix();  glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);  ban();  glColor3f(0.0f,1.0f,0.0f);  gear(0.03f,0.15f,0.03f,20,0.03f);  glColor3f(1.0f,0.0f,0.0f);  glPopMatrix();  glRotatef(SUDUT\_KIRI,0.0f,0.0f,1.0f);  glPushMatrix();  glTranslatef(0.0f,0.0f,-OFSET\_RODA);  Silinder2(DMT\_ROD,LEN\_RODA);  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,0.0f,OFSET\_RODA);  Silinder2(DMT\_ROD,LEN\_RODA);  glPopMatrix();  glTranslatef(LEN\_RODA,0.0f,0.0f);  Silinder2(DMT\_ROD,ENGSEL\_ROD);  glTranslatef(ENGSEL\_ROD,0.0f,0.0f);  glRotatef(-SUDUT\_KIRI,0.0f,0.0f,1.0f);  Silinder2(DMT\_ROD,LEN\_ATAS);  glTranslatef(LEN\_ATAS,0.0f,0.0f);  glRotatef(-FRONT\_INCLINE,0.0f,0.0f,1.0f);  glPushMatrix();  glTranslatef(-0.1f,0.0f,0.0f);  Silinder2(DMT\_ROD,0.45f);  glPopMatrix();  glPushMatrix();  glRotatef(-stir,1.0f,0.0f,0.0f);  glTranslatef(-0.3f,0.0f,0.0f);  glPushMatrix();  glRotatef(FRONT\_INCLINE,0.0f,0.0f,1.0f);  glPushMatrix();  glTranslatef(0.0f,0.0f,-HANDEL\_ROD/2);  Silinder1(DMT\_ROD,HANDEL\_ROD);  glPopMatrix();  glPushMatrix();  glColor3f(1.0f,1.0f,0.0f);  glTranslatef(0.0f,0.0f,-HANDEL\_ROD/2);  Silinder1(0.07f,HANDEL\_ROD/4);  glTranslatef(0.0f,0.0f,HANDEL\_ROD\*3/4);  Silinder1(0.07f,HANDEL\_ROD/4);  glColor3f(1.0f,0.0f,0.0f);  glPopMatrix();  glPopMatrix();  glPushMatrix();  Silinder2(DMT\_ROD,ENGSEL\_ROD);  glTranslatef(ENGSEL\_ROD,0.0f,0.0f);  glRotatef(SUDUT\_ENGSEL,0.0f,0.0f,1.0f);  glPushMatrix();  glTranslatef(0.0f,0.0f,OFSET\_RODA);  Silinder2(DMT\_ROD,ENGSEL\_RODS);  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,0.0f,-OFSET\_RODA);  Silinder2(DMT\_ROD,ENGSEL\_RODS);  glPopMatrix();  //rekaan tutup fork depan  glPushMatrix();  glColor3f(1.0f,0.0f,0.0f);  glTranslatef(0.0f,0.0f,-0.16f);  Silinder1(0.05f,0.32f);  glPopMatrix();  //TUTUP LAS SAMBUNGAN FORK KIRI  glPushMatrix();  glTranslatef(0.0f, 0.0f, -0.16f);  tutuplas();  glPopMatrix();  //TUTUP LAS SAMBUNGAN FORK KANAN  glPushMatrix();  glTranslatef(0.0f, 0.0f, 0.16f);  tutuplas();  glPopMatrix();  //tTUTUP LAS handle kiri  glPushMatrix();  glTranslatef(-0.7f,0.09f,0.6f);  glScalef(0.03f,0.04f,0.04f);  glRotatef(90.0f,0.0f,90.0f,0.0f);  glutSolidSphere(2.0, 2.0, 100.0);  glPopMatrix();  //TUTUP LAS handle kiri  glPushMatrix();  glTranslatef(-0.7f,0.09f,-0.6f);  glScalef(0.03f,0.04f,0.04f);  glRotatef(90.0f,0.0f,90.0f,0.0f);  glutSolidSphere(2.0, 2.0, 100.0);  glPopMatrix();  glTranslatef(ENGSEL\_RODS,0.0f,0.0f);  glRotatef(-2\*sudutpedal,0.0f,0.0f,1.0f);  ban();  glPopMatrix();  glPopMatrix();  glPopMatrix();  }  void gear( GLfloat inner\_radius, GLfloat outer\_radius, GLfloat width,  GLint teeth, GLfloat tooth\_depth )  {  GLint i;  GLfloat r0, r1, r2;  GLfloat angle, da;  GLfloat u, v, len;  const double pi = 3.14159264;  r0 = inner\_radius;  r1 = outer\_radius - tooth\_depth/2.0;  r2 = outer\_radius + tooth\_depth/2.0;  da = 2.0\*pi / teeth / 4.0;  glShadeModel( GL\_FLAT );  glNormal3f( 0.0, 0.0, 1.0 );  glBegin( GL\_QUAD\_STRIP );  for (i=0;i<=teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );  glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );  glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );  }  glEnd();  glBegin( GL\_QUADS );  da = 2.0\*pi / teeth / 4.0;  for (i=0;i<teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );  glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), width\*0.5 );  glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), width\*0.5 );  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );  }  glEnd();  glNormal3f( 0.0, 0.0, -1.0 );  glBegin( GL\_QUAD\_STRIP );  for (i=0;i<=teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );  glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );  glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );  }  glEnd();  glBegin( GL\_QUADS );  da = 2.0\*pi / teeth / 4.0;  for (i=0;i<teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );  glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), -width\*0.5 );  glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), -width\*0.5 );  glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );  }  glEnd();  glBegin( GL\_QUAD\_STRIP );  for (i=0;i<teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glVertex3f( r1\*cos(angle), r1\*sin(angle), width\*0.5 );  glVertex3f( r1\*cos(angle), r1\*sin(angle), -width\*0.5 );  u = r2\*cos(angle+da) - r1\*cos(angle);  v = r2\*sin(angle+da) - r1\*sin(angle);  len = sqrt( u\*u + v\*v );  u /= len;  v /= len;  glNormal3f( v, -u, 0.0 );  glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), width\*0.5 );  glVertex3f( r2\*cos(angle+da), r2\*sin(angle+da), -width\*0.5 );  glNormal3f( cos(angle), sin(angle), 0.0 );  glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), width\*0.5 );  glVertex3f( r2\*cos(angle+2\*da), r2\*sin(angle+2\*da), -width\*0.5 );  u = r1\*cos(angle+3\*da) - r2\*cos(angle+2\*da);  v = r1\*sin(angle+3\*da) - r2\*sin(angle+2\*da);  glNormal3f( v, -u, 0.0 );  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), width\*0.5 );  glVertex3f( r1\*cos(angle+3\*da), r1\*sin(angle+3\*da), -width\*0.5 );  glNormal3f( cos(angle), sin(angle), 0.0 );  }  glVertex3f( r1\*cos(0.0), r1\*sin(0.0), width\*0.5 );  glVertex3f( r1\*cos(0.0), r1\*sin(0.0), -width\*0.5 );  glEnd();  glShadeModel( GL\_SMOOTH );  glBegin( GL\_QUAD\_STRIP );  for (i=0;i<=teeth;i++) {  angle = i \* 2.0\*pi / teeth;  glNormal3f( -cos(angle), -sin(angle), 0.0 );  glVertex3f( r0\*cos(angle), r0\*sin(angle), -width\*0.5 );  glVertex3f( r0\*cos(angle), r0\*sin(angle), width\*0.5 );  }  glEnd();  }  void init()  {  GLfloat mat\_specular[]={1.0,1.0,1.0,1.0};  GLfloat mat\_shininess[]={100.0};  GLfloat light\_directional[]={1.0,1.0,1.0,1.0};  GLfloat light\_positional[]={1.0,1.0,1.0,0.0};  GLfloat light\_diffuse[]={1.0,1.0,1.0};  reset();  glShadeModel(GL\_SMOOTH);  glEnable(GL\_NORMALIZE);  glEnable(GL\_LIGHTING);  glEnable(GL\_LIGHT0);  glEnable(GL\_COLOR\_MATERIAL);  glEnable(GL\_DEPTH\_TEST);  }  void display(void)  {  glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);  glEnable(GL\_NORMALIZE);  glPushMatrix();  glRotatef(angley,1.0f,0.0f,0.0f);  glRotatef(anglex,0.0f,1.0f,0.0f);  glRotatef(anglez,0.0f,0.0f,1.0f);  alas();  glPushMatrix();  glTranslatef(xpos,0.0f,zpos);  glRotatef(direction,0.0f,1.0f,0.0f);  rangka();  rantai();  pedal();    glPopMatrix();  glPopMatrix();  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  gluLookAt(camx,camy,camz, camx,0.0,0.0, 0.0,1.0,0.0);  glutSwapBuffers();  }  void idle(void)  {  updateScene();  glutPostRedisplay();  }  void special(int key,int x,int y)  {  switch(key)  {  case GLUT\_KEY\_UP:  camz -= 0.1f;  break;  case GLUT\_KEY\_DOWN:  camz += 0.1f;  break;  case GLUT\_KEY\_LEFT:  camx -= 0.1f;  break;  case GLUT\_KEY\_RIGHT:  camx += 0.1f;  break;  }  glutPostRedisplay();  }  void reset()  {  anglex=angley=anglez=0.0f;  Mouse=GLUT\_UP;  camx=camy=1.0f;  camz=5.0f;  xpos=zpos=0.0f;  direction=-10.0f;  }  //main CODE  void reshape(int w,int h)  {  glViewport(0,0,(GLsizei)w,(GLsizei)h);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluPerspective(60.0,(GLfloat)w/(GLfloat)h,0.1,100.0);  glMatrixMode(GL\_MODELVIEW);  glLoadIdentity();  gluLookAt(camx,camy,camz, 0.0,0.0,0.0, 0.0,1.0,0.0);  }  void keyboard(unsigned char key,int x,int y)  {  GLfloat r=0.0f;  switch(key)  {  case 'r':  reset();  break;  case 'a':  if(stir < BTS\_HENDEL)  stir += PLUS\_STIR;  break;  case 'd':  if(stir > -BTS\_HENDEL)  stir -= PLUS\_STIR;  break;  case '+':  spid += PLUS\_SPID;  break;  case '-':  spid -= PLUS\_SPID;  break;  case 27:  exit(1);  }    sudutpedal += spid;  if(spid < 0.0f)  spid = 0.0f;  if(sudutpedal < 0.0f)  sudutpedal = 0.0f;  if(sudutpedal >= 360.0f)  sudutpedal -= 360.0f;  glutPostRedisplay();  }  void mouse(int button,int state,int x,int y)  {  switch(button)  {  case GLUT\_LEFT\_BUTTON:  if(state==GLUT\_DOWN)  {  Mouse=GLUT\_DOWN;  prevx=x;  prevy=y;  }  if(state==GLUT\_UP)  {  Mouse=GLUT\_UP;  }  break;  case GLUT\_RIGHT\_BUTTON:  break;  }  glutPostRedisplay();  }  void passive(int x,int y)  {  }  void gerak(int x,int y)  {  if(Mouse==GLUT\_DOWN)  {  int deltax,deltay;  deltax=prevx-x;  deltay=prevy-y;  anglex += 0.5\*deltax;  angley += 0.5\*deltay;  if(deltax!=0 && deltay!=0)  anglez += 0.5\*sqrt(deltax\*deltax + deltay\*deltay);  if(anglex < 0)  anglex+=360.0;  if(angley < 0)  angley+=360.0;  if(anglez < 0)  anglez += 360.0;  if(anglex > 360.0)  anglex-=360.0;  if(angley > 360.0)  angley-=360.0;  if(anglez > 360.0)  anglez-=360.0;  }  else  {  Mouse=GLUT\_UP;  }  prevx=x;  prevy=y;  glutPostRedisplay();  }  void glSetupFuncs(void)  {  glutDisplayFunc(display);  glutReshapeFunc(reshape);  glutIdleFunc(idle);  glutSpecialFunc(special);  glutKeyboardFunc(keyboard);  glutMouseFunc(mouse);  glutMotionFunc(gerak);  glutPassiveMotionFunc(passive);  glutSetCursor(GLUT\_CURSOR\_CROSSHAIR);  }  void help(void)  {  printf("ANIMASI SEPEDA 3D\n");  printf("IF17K - KOMPUTER GRAFIKA\n");    }  void main(int argc,char \*argv[])  {  help();  glutInit(&argc,argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowPosition(100,100);  glutInitWindowSize(LEBAR,TINGGI);  glutCreateWindow("ANIMASI SEPEDA 3D");  init();  glSetupFuncs();  glutMainLoop();  } |

Setelah dicompile dan run, maka hasilnya seperti screenshoot dibawah ini :

|  |
| --- |
|  |
|  |
|  |
|  |