PEMBENTUKAN TRANSFORMASI OBJEK 2 DIMENSI DENGAN OPENGL

Ina Agustina, Fauziah
Jurusan Sistem Informasi Universitas Nasional
Jl. Sawo Manila No.61 Pasar Minggu
E-Mail : ina_agustina2007@yahoo.com, fauziah_z2@yahoo.com

Abstrak
Pembuatan makalah ini membahas Pemindahan, Pemutaran, Pembesaran, Pencerminan. Program-program di atas dibuat dengan menggunakan software Microsoft Visual c++ ditambah dengan library openGL untuk menampilkan gambar grafiknya. Di samping itu juga akan dilengkapi dengan penjelasan dari program tersebut.

Keyword : *Transformasi 2D, Visual C++,OpenGL*

1. PENDAHULUAN

1.1 Latar Belakang Masalah

Semakin berkembangnya ilmu pengetahuan dan teknologi grafis pada masa sekarang ini, maka pengetahuan mengenai grafik computer semakin banyak dipakai. Namun sebelum kita membahas lebih jauh mengenai grafik komputer, kita akan mengetahui dahulu definisi sebenarnya tentang apa itu grafik komputer. Secara umum grafik komputer adalah gambar yang dihasilkan oleh komputer, yang hasilnya sudah sering kita lihat seperti di majalah dan televisi. Disini dibahas bahwa tiap karakter yang dihasilkan diambil dari *library* dari bentuk karakter yang ada pada memori komputer. Gambar-gambar yang berada pada majalah atau televisi tersebut ada beberapa yang terlihat sangat natural, sehingga kita para pembaca akan sulit membedakan mereka buatan atau hasil dari fotografi asli.

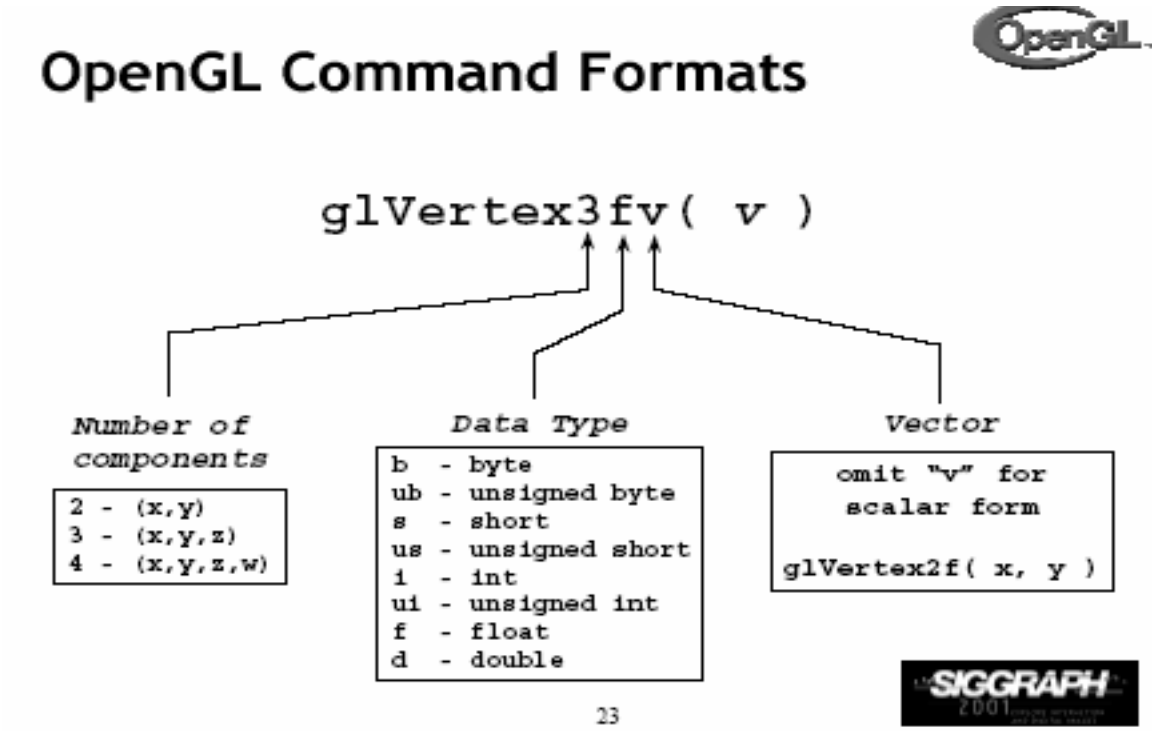
Selain itu grafik komputer juga digunakan untuk membuat *design* dari *background* ataupun objek-objek dalam *game* di komputer. Gambar-gambar yang ada pada *game* itu adalah gabungan antara kenyataan dan imajinasi dari *programmer*-nya. Dalam bidang lain, grafik komputer digunakan dalam dunia seni, entertainment, dan publikasi. Khususnya dipakai dalam produksi film, animasi, dan spesial efek. Animasi dibuat dengan membuat sebuah gambar yang berkelanjutan di film atau videotape. Gambar satu dan yang selanjutnya hanya mempunyai perbedaan yang tipis (slightly different). Bidang-bidang lain yang menggunakan grafika komputer adalah untuk browsing di World Wide Web, untuk pengawasan proses, dan banyak lagi.

2. TINJAUAN PUSTAKA

Transformasi dua dimensi adalah suatu model atau bentuk atau teknik memindahkan atau mengubah nilai posisi objek dalam sistem koordinat dua dimensi. Pemindahan objek ini dapat diartikan sebagai pemindahan titik. Ada dua cara yang dapat digunakan untuk mentransformasi gambar yaitu :

- **Transformasi objek**
  Definisinya adalah mengubah koordinat-koordinat dari tiap titik di objek dengan beberapa aturan , meninggalkan *underlying* sistem koordinat yang tidak bisa diubah lagi.
- **Transformasi koordinat**
  Definisinya adalah system koordinat yang baru dibuat sebelumnya merupakan perwakilan dari semua titik objek dalam sistem yang baru.

OpenGL adalah *Interface* (*library*) untuk *software* dalam berinteraksi dengan *graphic hardware*.



Gambar 1. Perintah dalam OpenGL

Menggambar bentuk Geometrik openGL :

- dimulai dengan glBegin(nama bentuk);
- diakhiri dengan glEnd();

Contoh :

```
glBegin( GL_QUADS );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(-1.0f, 1.0f, -10.0f );
        glColor3f( 0.0, 1.0, 0.0 );
        glVertex3f( 1.0f,1.0f, -10.0f );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(1.0f, -1.0f, -10.0f );
        glColor3f( 1.0, 0.0, 0.0 );
        glVertex3f(-1.0f, -1.0f, -10.0f );
glEnd();
```

2.1 Perpindahan (Translasi)

Pengertian translasi adalah perpindahan objek dari titik P(x, y) ke titik P'(x', y') secara linier. Dalam translasi dua dimensi diperlukan dua parameter, yaitu pemindahan kearah sumbu x dan pemindahan kearah sumbu y. Translasi (dx, dy) didefinisikan dengan :

$x' = x + dx$

$y' = y + dy$

Bentuk umum :

- glTranslatef(x, y, z)
- glTranslated(x, y, z)

```
        glVertex3f(1.0f, -1.0f, -10.0f );
        glColor3f( 1.0, 0.0, 0.0 );
        glVertex3f(-1.0f, -1.0f, -10.0f );
glEnd();
```

2.2 Penskalaan (Scale)

Definisi dari penskalaan m adalah perpindahan objek dari titik P ke titik P', dimana jarak titik P' adalah m kali titik P atau suatu operasi yang membuat suatu objek berubah ukurannya baik menjadi mengecil ataupun membesar secara seragam atau tidak seragam tergantung pada faktor penskalaan yang diberikan.

Proses penskalaan dapat didefinisikan dengan :

$x' = m_x x$

$y' = m_y y$

Bentuk umum :

- glScalef(xf, yf, zf);

2.3 Perputaran (Rotate)

Definisi rotasi adalah perpindahan objek dari titik P(x, y) ke titik P'(x',y') yang berupa pemindahan berputar sebesar sudut θ atau suatu operasi yang menyebabkan objek bergerak berputar pada titik pusat atau pada sumbu putar yang dipilih berdasarkan sudut putaran tertentu.

Operasi rotasi dapat diuraikan sebagai berikut :

1. Rotasi dari titik P yang berada di sumbu x, P(x, 0) menjadi P'(x', y') dapat digambarkan dengan :
   $x' = x .\cos \theta$
   $y' = x.\sin \theta$
2. Rotasi dari titik P yang berada di sumbu y, P(y, 0) menjadi P'(x', y') dapat digambarkan dengan :
   $x' = -y .\cos \theta$
   $y' = y.\sin \theta$
3. Rotasi dari titik P(x, y) menjadi P'(x', y') merupakan kombinasi dari kedua macam transformasi di atas R = Rx + Ry.
   $x' = x .\cos \theta - y.\sin \theta$
   $y' = x.\sin \theta + y.\cos \theta$

Bentuk umum :

- glRotatef(sudut, sx, sy, sz);

2.4 Pencerminan (Reflection)

Definisi pencerminan adalah transformasi objek yang mengubah lokasi objek pada kedudukan yang berlawanan dengan sembarang sumbu yang dipilih. Sumbu ini bisa sumbu x, sumbu y,sumbu z, atau sumbu lain yang ditentukan oleh pengguna. Transformasi ini menghasilkan citra cermin objek terhadap sumbu tertentu bagi objek yang dicerminkan.

Contoh soal, diberikan titik A dengan koordinat A(2, 3). Hitunglah koordinat baru dari titik A yang ditransformasi cerminan terhadap (a) sumbu y, (b) sumbu x, (c) titik pusat (0,0), dan (d) garis x = -1.
Jawabannya adalah sebagai berikut:

a. Transformasi cerminan terhadap sumbu y, maka tinggal membalik tanda koordinat x untuk titik A(2,3) sehingga menjadi A'(-2, 3).
b. Transformasi cerminan terhadap sumbu x, maka tinggal membalik tanda koordinat y untuk titik A(2,3) sehingga menjadi A"(2, -3).
c. Transformasi cerminan terhadap titik pusat (0, 0), maka tinggal membalik tanda koordinat x dan koordinat y untuk titik A(2,3) sehingga menjadi A'"(-2, -3).
d. Transformasi cerminan terhadap garis x = -1 akan menghasilkan titik A*(-4, 3).

3. HASIL DAN PEMBAHASAN
ALGORITMA PROGRAM OPENGL
3.1 LISTING PROGRAM
3.1.1 Perpindahan, penskalaan, pemutaran

```
#include <windows.h>
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glfw.h>

void mulaiOpenGL(void);

int main(void)
{
        GLuint mRunning = GL_TRUE;
if( glfwInit() == GL_FALSE )
        {
                MessageBox( NULL, "ERROR :: gagal menginisialisasi GLFW", "Error!", MB_OK);
                return(0);
        }
if( glfwOpenWindow( 640, 480, 0, 0, 0, 0, 24, 0, GLFW_WINDOW ) == GL_FALSE )
```

```
	{
		MessageBox( NULL, "ERROR :: gagal membuat window", "Error!", MB_OK );
		glfwTerminate();
		return(0);
	}
glfwSetWindowTitle( " Grafik Komputer " );
	glfwSwapInterval( 1 );
mulaiOpenGL();
while( mRunning )
	{
glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
		glLoadIdentity();
glTranslatef( -3.0f, 0.0f, 0.0f );
	glBegin( GL_QUADS );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(-1.0f, 1.0f, -10.0f );
		glColor3f( 0.0, 1.0, 0.0 );
		glVertex3f( 1.0f,1.0f, -10.0f );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(1.0f, -1.0f, -10.0f );
		glColor3f( 1.0, 0.0, 0.0 );
		glVertex3f(-1.0f, -1.0f, -10.0f );
	glEnd();
	glScalef( 7.0f, 7.0f, 10.0f );
	glBegin( GL_QUADS );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(-1.0f, 1.0f, -10.0f );
		glColor3f( 0.0, 1.0, 0.0 );
		glVertex3f( 1.0f,1.0f, -10.0f );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(1.0f, -1.0f, -10.0f );
		glColor3f( 1.0, 0.0, 0.0 );
		glVertex3f(-1.0f, -1.0f, -10.0f );
	glEnd();
	glRotatef( 45.0f, -2.0f, -6.0f, 10.0f );
	 glBegin( GL_QUADS );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(-1.0f, 1.0f, -10.0f );
		glColor3f( 0.0, 1.0, 0.0 );
		glVertex3f( 1.0f,1.0f, -10.0f );
		glColor3f( 1.0, 1.0, 0.0 );
		glVertex3f(1.0f, -1.0f, -10.0f );
		glColor3f( 1.0, 0.0, 0.0 );
		glVertex3f(-1.0f, -1.0f, -10.0f );
	glEnd();

	glfwSwapBuffers();
	mRunning = !glfwGetKey( GLFW_KEY_ESC ) &&
 glfwGetWindowParam( GLFW_OPENED );
}
	glfwTerminate();
	return(0);
}

void mulaiOpenGL(void)
{
	glViewport( 0, 0, 640, 480 );
	glMatrixMode( GL_PROJECTION );
	glLoadIdentity();
	gluPerspective( 60.0f, 640.0f/480.0f, 0.1f, 1000.0f );
```

```
        glMatrixMode( GL_MODELVIEW );
        glLoadIdentity();
        glShadeModel( GL_SMOOTH );
        glClearColor( 0.0f, 0.0f, 0.0f, 0.0f );
        glClearDepth( 1.0f );
        glEnable( GL_DEPTH_TEST );
        glDepthFunc( GL_LEQUAL );
        glHint( GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST );
}
```

3.1.2 Pencerminan

```
#include <windows.h>
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glfw.h>

void mulaiOpenGL(void);
int main(void)
{
    GLuint mRunning = GL_TRUE;
    if( glfwInit() == GL_FALSE )
    {
        MessageBox( NULL, "ERROR :: gagal menginisialisasi GLFW", "Error!", MB_OK);
        return(0);
    }
    if( glfwOpenWindow( 640, 480, 0, 0, 0, 0, 24, 0, GLFW_WINDOW ) == GL_FALSE )
    {
        MessageBox( NULL, "ERROR :: gagal membuat window", "Error!", MB_OK );
        glfwTerminate();
        return(0);
    }
    glfwSetWindowTitle( " RefLeCTiOn" );
    glfwSwapInterval( 1 );
    mulaiOpenGL();
    while( mRunning )
    {
        glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
        glLoadIdentity();
         glRotatef( 0.0f, 0.0f, 0.0f, -10.0f );
         glBegin( GL_QUADS );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(-1.0f, 1.0f, -10.0f );
                glColor3f( 0.0, 1.0, 0.0 );
                glVertex3f( 1.0f,1.0f, -10.0f );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(1.0f, -1.0f, -10.0f );
                glColor3f( 1.0, 0.0, 0.0 );
                glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();
        glTranslatef( -3.0f, 0.0f, 0.0f );
        glBegin( GL_QUADS );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(-1.0f, 1.0f, -10.0f );
                glColor3f( 0.0, 1.0, 0.0 );
                glVertex3f( 1.0f,1.0f, -10.0f );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(1.0f, -1.0f, -10.0f );
                glColor3f( 1.0, 0.0, 0.0 );
                glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();
```

```
        glTranslatef( 1.5f, 0.0f, 0.0f );
        glBegin( GL_LINES );
                glColor3f( 0.0, 0.0, 0.0 );
                glVertex3f(0.0f, 2.0f, -10.0f );
                glColor3f( 0.0, 0.0, 0.0 );
                glVertex3f( 0.0f,-2.0f, -10.0f );
        glEnd();
        glfwSwapBuffers();
        mRunning = !glfwGetKey( GLFW_KEY_ESC ) && glfwGetWindowParam( GLFW_OPENED );
    }
    glfwTerminate();
    return(0);
}
void mulaiOpenGL(void)
{
    glViewport( 0, 0, 640, 480 );
    glMatrixMode( GL_PROJECTION );
    glLoadIdentity();
    gluPerspective( 60.0f, 640.0f/480.0f, 0.1f, 1000.0f );
    glMatrixMode( GL_MODELVIEW );
    glLoadIdentity();
    glShadeModel( GL_SMOOTH );
    glClearColor( 1.0f, 1.0f, 1.0f, 0.0f );
    glClearDepth( 1.0f );
    glEnable( GL_DEPTH_TEST );
    glDepthFunc( GL_LEQUAL );
    glHint( GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST );
}
```

3.2 LOGIKA PROGRAM

3.2.1 Perpindahan, penskalaan, pemutaran

Untuk membuat program transformasi menggunakan openGL dengan Pelles C, terlebih dahulu kita deklarasikan file-file include yang dibutuhkan, seperti :

- #include <windows.h> sebagai header file untuk windows
- #include <GL/gl.h> sebagai header file untuk OpenGL32 Library
- #include <GL/glu.h> sebagai header file untuk GLu32 Library
- #include <GL/glfw.h> sebagai header file untuk The Glfw Library

Pada program transformasi ini terdapat sebuah fungsi bernama mulaiOpenGL yang harus dideklarasikan menggunakan tipe data void. Fungsi ini digunakan untuk menjalankan program openGL Di dalam fungsi ini terdapat fungsi main bertipe data int yang juga harus dideklarasikan.

```
void mulaiOpenGL(void);
 int main(void)

{       GLuint mRunning = GL_TRUE;
```

Perintah diatas digunakan untuk mendeklarasikan variable mRunning yang nilainya diinisialisasikan sama dengan GL_TRUE. Jika mRunning bernilai TRUE maka aplikasi akan berjalan dan jika mRunning bernilai FALSE maka akan terjadi sebaliknya. Untuk mendeklarasikan sebuah variable digunakan perintah GLuint.

```
        if( glfwInit() == GL_FALSE )
        {
                MessageBox( NULL, "ERROR :: gagal menginisialisasi GLFW", "Error!", MB_OK);
                return(0);
        }
```

Perintah di atas digunakan untuk menginisialisasikan GLFW. GLFW merupakan Library tambahan yang menjembatani antara platform (OS) dengan OpenGL. GLFW Bukan bagian resmi dari OpenGL dan Multiplatform. Pada perintah di atas, Pertama dilakukan seleksi apakah metode glfwinit() bernilai GL_FALSE atau salah. Jika kondisi terpenuhi maka akan tampil MessageBox dengan tulisan "ERROR :: gagal menginisialisasi GLFW", "Error!".

```
if( glfwOpenWindow( 640, 480, 0, 0, 0, 0, 24, 0, GLFW_WINDOW ) == GL_FALSE )
{
        MessageBox( NULL, "ERROR :: gagal membuat window", "Error!", MB_OK );
        glfwTerminate();
        return(0);
}
```

Perintah di atas digunakan untuk membuat sebuah window berukuran 640 x 280 yang akan digunakan untuk menggambar. Jika gagal mambuat window maka akan tampil message box dengan tulisan ""ERROR :: gagal membuat window", "Error!", dengan sebuah command button OK yang dibuat dengan perintah MB_OK.

```
glfwSetWindowTitle( "Grafik Komputer " );
glfwSwapInterval( 1 );
```

Perintah di atas digunakan untuk menset judul yang ada di window dan Swap interval. glfwSetWindowTitle digunakan untuk menset judul yang ada di window sedangkan glfwSwapInterval digunakan untuk menset judul yang ada swap.

```
mulaiOpenGL();
```

Perintah di atas digunakan untuk memanggil fungsi mulaiOpenGL() secara rekursif. Fungsi mulaiOpenGL() merupakan fungsi untuk melakukan setting awal OpenGL.

Selanjutnya kita mulai looping utama program. Pada bagian program di bawah digunakan perulangan while untuk mengulang program selama variable mrunning masih bernilai true.

```
while( mRunning )
        {

                glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
                glLoadIdentity();
```

Perintah di atas digunakan untuk membersihkan layar dan depth buffer. Sedangkan glLoadIdentity() digunakan untuk menampilkan program yang ingin ditampilkan.

Untuk membuat atau menampilkan program translasi digunakan perintah glTranslatef(0.0f, 0.0f, 0.0f), besarnya perpindahan (translasi diisikan didalam kurung). Sebagai contoh glTranslatef(-3.0f, 0.0f, 0.0f), angka -3 menunjukan dilakukannya translasi sebesar tiga satuan kearah kiri sumbu x , kita juga bisa melakukan translasi kea rah sumbu y dan sumbu z sesuai yang kita inginkan. Selanjutnya bidang yang akan kita translasikan, kita deklarasikan dibawahnya. Tetapi, sebelumnya kita gunakan perintah glBegin(GL_QUADS) sebagai keyword untuk memulai melakukan penulisan program. Bidang yang akan kita tampilkan merupakan gabungan dari beberapa titik yang dihubungkan sesuai dengan yang kita inginkan dengan menggunakan perintah-perintah openGL. Untuk membuat titik digunakan perintah glVertex3f, kemudian kita masukan nilainya terhadap sumbu x,y dan z dan untuk memberi warna kita gunakan perintah glColor3f. Angka tiga menunjukan bahwa kita akan menggambarkan bangun tiga dimensi. Bila titik-titik tersebut akan kita hubungkan menjadi sebuah persegi maka kita tuliskan perintah GL_QUADS. Hal yang sama juga kita lakukan untuk program perubahan ukuran (scale) dan program perputaran (rotate). Untuk mengakhiri masing-masing program, digunakan perintah glEnd(). Jumalah glBegin harus sama dengan jumlah glEnd().

```
glTranslatef( -3.0f, 0.0f, 0.0f );
glBegin( GL_QUADS );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(-1.0f, 1.0f, -10.0f );
        glColor3f( 0.0, 1.0, 0.0 );
        glVertex3f( 1.0f,1.0f, -10.0f );
        glColor3f( 1.0, 1.0, 0.0 );
```

```
                glVertex3f(1.0f, -1.0f, -10.0f );
                glColor3f( 1.0, 0.0, 0.0 );
                glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();

        glScalef( 7.0f, 7.0f, 10.0f );
        glBegin( GL_QUADS );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(-1.0f, 1.0f, -10.0f );
                glColor3f( 0.0, 1.0, 0.0 );
                glVertex3f( 1.0f,1.0f, -10.0f );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(1.0f, -1.0f, -10.0f );
                glColor3f( 1.0, 0.0, 0.0 );
                glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();

        glRotatef( 45.0f, -2.0f, -6.0f, 10.0f );
         glBegin( GL_QUADS );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(-1.0f, 1.0f, -10.0f );
                glColor3f( 0.0, 1.0, 0.0 );
                glVertex3f( 1.0f,1.0f, -10.0f );
                glColor3f( 1.0, 1.0, 0.0 );
                glVertex3f(1.0f, -1.0f, -10.0f );
                glColor3f( 1.0, 0.0, 0.0 );
                glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();


        glfwSwapBuffers();
```

Perintah diatas digunakan untuk menampilkan gambar ke layar (swap double buffer)

```
                mRunning = !glfwGetKey( GLFW_KEY_ESC ) &&


glfwGetWindowParam( GLFW_OPENED );}
glfwTerminate();
return(0);
}
```

Perintah diatas digunakan untuk mengecek inputan, apakah tombol esc yang ditekan atau tombol "close yang diclick

Untuk mengeset ukuran layar dengan resolusi 640x480 digunakan perintah glViewport. Viewport bisa diibaratkan sebagai layar monitor anda.

```
        void mulaiOpenGL(void)
{

        glViewport( 0, 0, 640, 480 );
```

Untuk mengeset mode OpenGL ke mode proyeksi (Projection) digunakan perintah GL_PROJECTION dan untuk mengeset proyeksi menggunakan proyeksi perspective, dengan sudut pandang (Field OfView) 60 derajat digunakan perintah gluPerspective.

```
        glMatrixMode( GL_PROJECTION );
        glLoadIdentity();
        gluPerspective( 60.0f, 640.0f/480.0f, 0.1f, 1000.0f );
```

```
glMatrixMode( GL_MODELVIEW );
glLoadIdentity();


glShadeModel( GL_SMOOTH );
glClearColor( 0.0f, 0.0f, 0.0f, 0.0f );
```

Perintah glShadeMode digunakan untuk mengeset mode gradasi dengan warna yang halus. Sedangkan perintah glClearColor digunakan untuk memberikan warna pada layar dengan tujuan untuk membersihkan layer.

```
glClearDepth( 1.0f );
glEnable( GL_DEPTH_TEST );
glDepthFunc( GL_LEQUAL );
```

Perintah glClearDepth diguanakan untuk memberikan nilai yang digunakan untuk membersihkan depth buffer (1.0f). Perintah GL_Depth_TEST digunakan untuk menghindari polygon yang tumpang tindih.

```
glHint( GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST );
}
```

Perintah glHint digunakan untuk memberitahu OpenGL untuk menggunakan perhitungan perspective yang terbaik (perhitungan ini tidak bisa selalu 100% akurat).

3.2.2 Pencerminan

Seperti definisi dari pencerminan yaitu transformasi objek yang mengubah lokasi objek yang berlawanan dengan sembarang sumbu yang dipilih. Sedangkan dalam program OpenGL sendiri menurut kami tidak ada perintah yang spesifik untuk membuat tampilan pencerminan ini. Maka dari itu, kali membuat program yang seolah-olah menggambarkan sebuah objek yang diletakkan di depan cermin dan pada lokasi yang berlawanan akan menghasilkan bayangan dari objek tersebut. Program pencerminan ini hampir sama dengan program di atas, yang membedakan hanya pada blok program di while. Maka kami akan menjelaskan pada blok program tersebut.

```
glRotatef( 0.0f, 0.0f, 0.0f, -10.0f );
    glBegin( GL_QUADS );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(-1.0f, 1.0f, -10.0f );
        glColor3f( 0.0, 1.0, 0.0 );
        glVertex3f( 1.0f,1.0f, -10.0f );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(1.0f, -1.0f, -10.0f );
        glColor3f( 1.0, 0.0, 0.0 );
        glVertex3f(-1.0f, -1.0f, -10.0f );
    glEnd();
```

Pada program ini kami misalkan objek bendanya adalah sebuah persegi. Perintah untuk membuat persegi dengan menggunakan GL_QUADS. Untuk membuat persegi kita membutuhkan empat titik, yang telah dideklarasikan seperti program di atas dengan glVertex3f. Sedangkan, glColor3f adalah perintah untuk mrwarnai persegi tersebut. Agar mendapatkan posisi yang sesuai dengan keinginan maka kami menggunakan glRotated untuk memutar objek. Dalam hal ini, objek pertama tidak diputar atau dengan mendeklarasikan sudutnya sebesar nol derajat, dan mengubah posisi koordinat z dengan memberi nilai minus sepuluh agar gambar seolah-olah ada di depan kita.

```
glTranslatef( -3.0f, 0.0f, 0.0f );
glBegin( GL_QUADS );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(-1.0f, 1.0f, -10.0f );
        glColor3f( 0.0, 1.0, 0.0 );
        glVertex3f( 1.0f,1.0f, -10.0f );
        glColor3f( 1.0, 1.0, 0.0 );
        glVertex3f(1.0f, -1.0f, -10.0f );
        glColor3f( 1.0, 0.0, 0.0 );
```

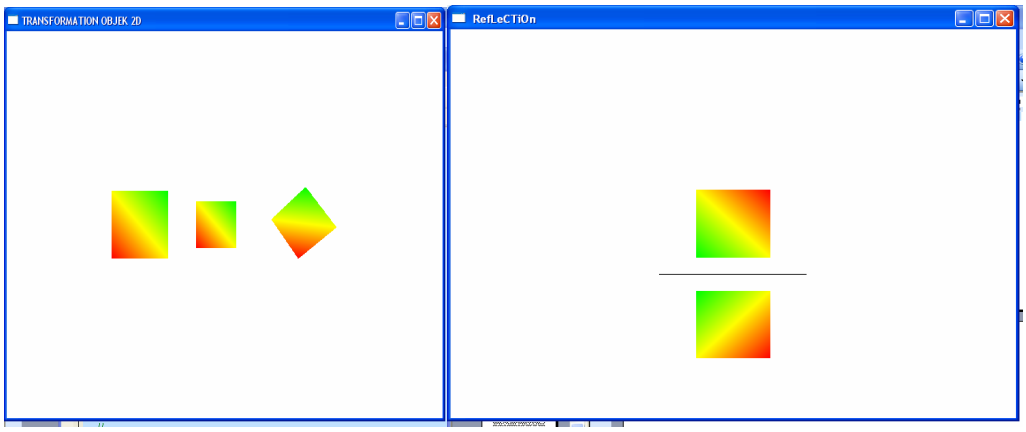
```
            glVertex3f(-1.0f, -1.0f, -10.0f );
        glEnd();
```

Sama seperti penjelasan di atas, bahwa kita buat satu persegi lagi. Supaya tempat antara persegi kesatu dan kedua tidak tumpang tindih, maka kita menggunakan glTranslatef untuk memindahkan objek kedua yang berperan sebagai bayangannya, dengan memindahkan berdasarkan sumbu x sebesar tiga satuan kearah kiri karena membubuhkan nilai negative.

```
        glTranslatef( 1.5f, 0.0f, 0.0f );
        glBegin( GL_LINES );
            glColor3f( 0.0, 0.0, 0.0 );
            glVertex3f(0.0f, 2.0f, -10.0f );
            glColor3f( 0.0, 0.0, 0.0 );
            glVertex3f( 0.0f,-2.0f, -10.0f );
        glEnd();
```

Objek yang ketiga adalah objek garis yang dibuat dengan perintah GL_LINES yang pada program ini berperan sebagai cermin antara dua objek yang dicerminkan. Mungkin sekian saja penjelasan dari program pencerminan.

3.3 OUTPUT PROGRAM



Gambar Output

4. KESIMPULAN

Grafika komputer akhir-akhir ini mulai dirasa sangat penting dan mencakup hampir semua bidang kehidupan seiring dengan semakin pentingnya sistem komputer dalam berbagai kegiatan. Grafika komputer merupakan gambar atau grafik yang dihasilkan oleh computer. Teknik-teknik yang dipelajari dalam grafika komputer adalah teknik-teknik bagaimana membuat atau manciptakan gambar dengan menggunakan komputer, salah satunya adalah Teknik Transformasi dua dimensi yaitu suatu model atau bentuk atau teknik memindahkan atau mengubah nilai posisi objek dalam sistem koordinat dua dimensi. Pemindahan objek ini dapat diartikan sebagai pemindahan titik. Transformasi dua dimensi ini mencakup proses translasi, penskalaan, rotasi, dll.

Dari makalah ini diketahui cara-cara membuat kurva tertutup dengan menggunakan GL_QUADS. Kemudian dapat memindahkan polygon dengan GL Translatef, memutar objek dengan GL Rotatef, membuat polygon tersebut lebih besar atau lebih kecil dengan menggunakan GL Scalef. Di samping perintah-perintah yang kami sebutkan di atas, masih banyak lagi gambar-gambar yang dapat dihasilkan dengan menggunkan transformasi dua dimensi.

Atas terselesainya makalah ini semoga dapat memberi manfaat bagi yang pembaca. Namun, kami juga menyadari bahwa isi makalah ini masih jauh dari sempurna.

5. DAFTAR PUSTAKA

FS. Hill, Jr., 2000, *Computer Graphics Using OpenGL*, Second Edition, Prentice Hall.

Basuki , Achmad & Nana Ramadijanti,2006, *Grafika Komputer Teori dan Implementasi*, Penerbit Andi. Yogyakarta.