**LAPORAN**

**TUGAS BESAR GRAFIKA KOMPUTER**

**MEMBUAT MENARA BIG BEN LONDON 3D**

*Diajukan untuk Memenuhi Tugas Besar*

*Mata Kuliah Grafika Komputer*

Dosen : Hendri Karisma, S. Kom

Oleh:

Muhammad Rizky Saragih - 10109363

Rico Oktavian Adhi Wibowo - 10109390

Rintho Rante Rerung - 10109400

Kelas IF-9/VIII



**PROGRAM STUDI S1**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

**2013**

**KATA PENGANTAR**



*Assalaamu’alaikum wr. wb,*

Puji dan syukur penulis panjatkan kepada Allah SWT atas rahmat dan karunia-Nya yang telah dilimpahkan, shalawat dan salam tidak lupa dicurahkan kepada Nabi Muhammad SAW, sehingga penulis dapat menyelesaikan penyusunan Tugas Besar ini tepat pada waktunya dengan judul “**MEMBUAT MENARA BIG BEN LONDON 3D**”.

Penyusunan Tugas Besar ini diajukan untuk memenuhi mata kuliah Grafika Komputer Program Strata Satu Jurusan Teknik Informatika Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Indonesia Bandung.

Sebagai manusia biasa penulis menyadari masih terdapat banyak kekurangan di dalam penulisan laporan Tugas Besar ini karena keterbatasan pengetahuan penulis. Oleh karena itu, saran dan kritik sangat penulis harapkan. Semoga laporan ini dapat bermanfaat bagi semua pihak yang memerlukan.

*Wassalaamu’alaikum wr. wb.*

Bandung, Juli 2013

Penulis

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Menara Big Ben nama yang diberi kepada untuk loceng besar dalam menara jam di hujung utara Istana Westminster di London. Nama samaran sering juga digunakan untuk memaksudkan jam atau menara berkenaan. Ini adalah jam berperanggu loceng bermuka empat yang terbesar di dunia, dan menara jam berdiri bebas ketiga tertinggi di dunia..

Menara Big Ben London 3D yang akan dibanagun nantinya terdapat jarum jam yang dapat berputar seperti jam-jam pada umumnya. Terdapat bangunan gedung-gedung yang dibuat sekitar menara Big Ben London. Selain terdapat beberapa objek yang menjadi pelengkap untuk gambaran kota London seperti aslinya seperti jalan, lampu taman, batu, pepohonan, wartel dan bus. Kesesuaian penglihatan kamera dikontrol menggunakan beberapa key yang ada di keyboard.

* 1. **Perumusan Masalah**

Bagaimana membangun suatu objek 3D Menara Big Ben London dengan jarum jam yang dapat berputar secara otaomatis beserta membuat beberapa objek yang ada di sekitar Menara Big Ben.

* 1. **Maksud dan Tujuan**

**1.3.1 Maksud**

Membangun suatu objek 3D Menara Big Ben dengan jarum jam yang berputar atomatis atau bergerak dengan menggunakan OpenGL.

**1.3.2 Tujuan**

Adapun tujuan dari pembuatan objek Menara Big Ben iniadalah sebagai berikut

1. Menampilkan bangunan menara Big Ben yang memiliki jam yang bisa berputar dan disekitarnya terdapat beberapa objek diantaranya gedung-gedung, jalan, pohon dan bus.
2. Mengetahui fungsi-fungsi pada openGL dengan menggunakan empat unsur yaitu coloring, lighting, blending, dan mapping.
   1. **Batasan Masalah**

Agar pembahasan masalah tidak menyimpang dari objek pokok bahasan, maka batasan masalah dalam pembuatan perangkat lunak ini adalah sebagai berikut :

1. Hanya menampilkan menara Big Ben London 3D, gedung-gedung, pepohonan, jalan, lampu-lampu jalan dan bus.
2. Menampilkan pergerakan jarum jam.
3. Objek ditampilkan dalam 3 Dimensi.
4. Menggunakan OpenGL.
5. Bahasa yang digunakan bahasa pemograman C++
   1. **Objek List**
6. Menara Big Ben : Bangunan akan dibuat menyerupai bangunan asli menara kincir angin yang ada di Belanda dan kincir angin bisa berputar
7. Gedung-gedung : Gedung-gedung dibuat sesuai dengan gedung-gedung yang berada tepat disamping menara Big Ben London.
8. Bus : Bus dibuat tepat di jalan samping menara Big Ben.
9. Jalan : Jalan dibuat selayaknya aslinya, jalan bersebelahan dengan Menara Big Ben London.
10. Pohon : pohon akan dibuat ditepian jalan dan disekitar bangunan Menara Big Ben.
11. Lampu-lampu jalan : lampu-lampu jalan dibuat tepat berada disamping jalan.
12. Wartel : Bangunan wartel dibuat berdekatan dengan Menara Big Ben London.
    1. **Objek Pembangun**

|  |  |
| --- | --- |
| **Bentuk Bangun** | **Penjabaran** |
| Bola | Bola digunakan untuk membuat atap wartel, pohon bagian atas atau bagian daun. Diibaratkan sebagai kumpulan daun yang bergumpal |
| Persegi | Persegi digunakan untuk membuat jendela |
| Kerucut | Kerucut digunakan untuk membuat bagian atas menara Big Ben dan gedung-gedung yang dapat dimodifikasi menjadi bentuk limas. |
| Balok | Balok digunakan untuk membuat menara, gedung, wartel, bus, dll |
| Kubus | Kubus digunakan untuk membuat menara Big Ben |

**BAB II**

**LANDASAN TEORI**

## Pengertian Grafika Komputer

Grafika komputer (*Computer graphics*) adalah bagian dari [ilmu komputer](http://id.wikipedia.org/wiki/Ilmu_komputer) yang berkaitan dengan pembuatan dan manipulasi gambar secara digital. Bentuk sederhana dari grafika komputer adalah [grafika komputer 2D](http://id.wikipedia.org/w/index.php?title=Grafika_komputer_2D&action=edit&redlink=1) yang kemudian berkembang menjadi [grafika komputer 3D](http://id.wikipedia.org/wiki/Grafika_komputer_3D), [pemrosesan citra](http://id.wikipedia.org/w/index.php?title=Pemrosesan_citra&action=edit&redlink=1) *(image processing),* dan [pengenalan pola](http://id.wikipedia.org/wiki/Pengenalan_pola) *(pattern recognition).* Grafika komputer sering dikenal juga dengan istilah [visualisasi](http://id.wikipedia.org/wiki/Visualisasi) data.

Bagian dari grafika komputer meliputi:

1. [Geometri](http://id.wikipedia.org/wiki/Geometri): mempelajari cara menggambarkan permukaan bidang
2. [Animasi](http://id.wikipedia.org/wiki/Animasi): mempelajari cara menggambarkan dan memanipulasi gerakan
3. [Rendering](http://id.wikipedia.org/w/index.php?title=Rendering&action=edit&redlink=1): mempelajari [algoritma](http://id.wikipedia.org/wiki/Algoritma) untuk menampilkan efek cahaya
4. [Citra](http://id.wikipedia.org/wiki/Citra) (Imaging) : mempelajari cara pengambilan dan penyuntingan gambar.

Teknik-teknik yang dipelajari dalam grafika komputer adalah teknik-teknik bagaimana membuat atau menciptakan gambar menggunakan komputer.Ada perbedaan yang sangat mendasar antara foto dan gambar, yaitu pada foto semua detail obyek terlihat sedangkan pada gambar (baik itu gambar manusia atau gambar komputer) tidak dapat memperlihatkan semua detail yang ada tetapi hanya detail-detail yang dianggap penting dalam menunjukkan pola suatu gambar.

## 2.2 Elemen Dasar Grafika

Ada beberapa elemen dasar dari grafika komputer antara lain:

1. Point

*Point* adalah sebuah titik yang digunakan untuk membangun obyek. Setiap titik dalam obyek 3 dimensi memiliki nilai dalam x, y dan z.

1. Polyline

*Polyline* adalah sebuah fungsi yang dibentuk dari beberapa garis yang saling berhubungan dan membentuk sebuh kurva yang terbuka.

1. Polygon

*Polygon* adalah suatu fungsi yang mirip dengan polyline hanya saja hasilnya adalah kurva tertutup, sedangkan *polyline* hasilnya kurva terbuka.

1. Filled Polygon ( Face )

*Filled Polygon* adalah sebuah *polygon* yang bagian dalamnya diwarnai atau dipenuhi dengan sebuah warna tertentu.*Filled polygon* biasanya digunakan sebagai *face* dari pembentukan obyek–obyek 3 Dimensi.

1. Gradate Polygon

*Gradate polygon* adalah sebuah *polygon* yang bagian dalamnya memiliki warna – warna yang bergradasi dari satu warna ke warna yang lainnya.

## 2.3 Grafik Komputer 2D

Grafik komputer 2D adalah pembuatan objek gambar dengan menggunakan 2 titik sebagai acuannya yaitu sumbu x dan y. Grafik 2D ini dapat digunakan dalam berbagai aplikasi yang pada awalnya dikembangkan pada teknologi cetak tradisional dan gambar, seperti tipografi, kartografi, gambar teknik, iklan, dan lain-lain.

Grafik komputer 2D ini merupakan langkah paling awal dalam membentuk model objek yang akan dibangun dalam grafik komputer 3D. Dalam aplikasi, gambar dua dimensi adalah bukan hanya representasi dari objek dunia nyata, tetapi sebuah artefak independen dengan nilai tambah semantik. Keseluruhan obyek 2D dapat dimasukkan dengan jumlah lebih dari satu, model yang akan dibentuk sesuai dengan kebutuhan. Tahap rekayasa hasil obyek 2D dapat dilakukan dengan aplikasi program grafis seperti Adobe Photoshop, Corel Draw, dan lain sebagainya.

## 2.4 Grafik Komputer 3D

Grafik komputer 3D merupakan representasi dari data geometrik 3 dimensi sebagai hasil dari pemrosesan dan pemberian efek cahaya terhadap grafik komputer 2D.hasilnya dapat ditampilkan secara real time untuk keperluan simulasi. Prinsip yang dipakai mirip dengan grafik komputer 2D dalam penggunaan algoritma, grafika vektor, model frame kawat (wire frame model), dan grafik rasternya.

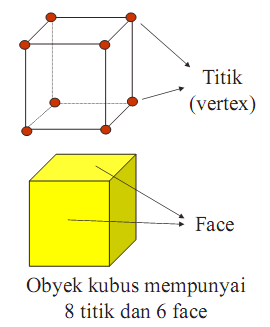
Grafik komputer 3D sering disebut sebagai model 3D. Namun, model 3D ini lebih menekankan pada representasi matematis untuk objek 3 dimensi. Obyek pada grafik 3D adalah sekumpulan titik-titik 3D (x,y,z) yang membentuk suatu face (bidang) yang digabungkan menjadi satu kesatuan. Face sendiri adalah gabungan titik-titik yang membentuk bidang tertentu. Data matematis ini belum bisa dikatakan sebagai gambar grafis hingga saat ditampilkan secara visual pada layar komputer atau printer. Proses penampilan suatu model matematis ke bentuk citra 2 D biasanya dikenal dengan proses 3D rendering.

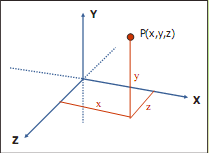
## 2.5 Perbedaan Grafik 2D dan 3D

Perbedaan yang paling mendasar dan terlihat dengan sangat jelas adalah tampilan gambarnya. Gambar 2D tampil flat adn frame tampilannya cenderung terbatas karena objek gambarnya disajikan hanya dengan sumbu x dan y. Sedangkan pada grafik 3D, gambar yang ditampilkan lebih hidup, membentuk ruang, tidak flat, serta framenya lebih luas yang dikarenakan gambar 3D disajikan dengan 3 sumbu, yaitu x, y, dan z.

* 1. **Definisi Objek 3 Dimensi**

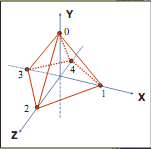
Obyek 3-D adalah sekumpulan titik-titik 3-D (x,y,z) yang membentuk luasan-luasan (face) yang digabungkan menjadi satu kesatuan. Face adalah gabungan titik-titik yang membentuk luasan tertentu atau sering dinamakan dengan sisi.





Sistem Koordinat 3 Dimensi

**Contoh pernyataan Objek : Limas Segiempat**

Titik-titik yang membentuk obyek:

Titik 0 (0,150,0)

Titik 1 (100,0,0)

Titik 2 (0,0,100)

Titik 3 (-100,0,0)

Titik 4 (0,0,-100)

## Pengertian OpenGl

OpenGL adalah API (Application Programing Interface) yang dikenalkan oleh SGI (Silicon Graphics Inc) kali pertama pada tahun 1980-an. Awalnya OpenGL diciptakan untuk mendukung proses rendering, pada saat itu hanya digunakan untuk mendukung proses redering yang dilakukan oleh komputer graphic produksi SGI. Namun akhirnya, OpenGL dijadikan standar oleh berbagai perusahaan software dan hardware.

open GL (program java for openGL). adalah bahasa yang dapat dijalankan dimanapun dan di sembarang platform apapun, diberagam lingkungan : internet, intranets, consumer electronic products, dan computer applications. Bahasa pemrograman berorientasi objek telah menjadi aliran utama (mainstream), java benar-benar berorientasi objek sejati, melebihi C++.Segala sesuatu dijava kecuali sedikit type dasar (int, float, double, char) adalah objek.

* 1. **Fungsi OpenGL 3D**

1. **Membersihkan *Windows***

Pada komputer, *memory* untuk menampilkan gambar biasanya diisi dengan gambar yang berasal dari perintah gambar paling akhir, jadi perlu dibersihkan dengan warna latar belakang sebelum digambar lagi. Contoh berikut ini perintah yang digunakan untuk membersihkan layar latar belakang dengan warna hitam dan *buffer* apa yang akan dibersihkan. Dalam hal ini, *buffer* warna yang akan dibersihkan karena *buffer* warna merupakan tempat gambar disimpan.

glClearColor 0, 0, 0, 0

glClear GL\_COLOR\_BUFFER\_BIT Or GL\_DEPTH\_BUFFER\_BIT

1. **Spesifikasi Warna**

Pada OpenGL mendeskripsikan objek dengan warna objek adalah proses yang berjalan sendiri-sendiri. Sebelum warna diubah maka semua objek yang digambar sesudah perintah tersebut akan menggunakan warna terakhir yang terdapat pada *coloring scheme.* Untuk warna digunakan perintah *glColor3f*. Contoh berikut menunjukkan urutan langkah dalam proses spesifikasi warna sebelum objek digambar. Warna yang ditampilkan adalah warna merah.

glColor3f 1, 0, 0

1. **Memaksa Proses Menggambar Sampai Selesai**

Kebanyakan sistem grafik modern sudah menggunakan sistem *graphics pipeline*. Dimana CPU utama memberikan *issue* perintah menggambar dan hardware lain yang melakukan *transformasi,* *clipping*, *shading*, *texturing* dan lain-lain. Pada arsitektur yang demikian, proses tidak dilakukan pada satu computer karena setiap komputer mempunyai tugas sendiri. CPU utama tidak harus menunggu proses pada masing-masing komputer tadi selesai, tapi bisa dengan memberikan *issue* perintah gambar yang berikutnya. Untuk inilah OpenGL menyediakan perintah *glFlush* yang memaksa *client* untuk segera mengirim paket *network* walaupun belum penuh. Program sebaiknya ditulis menggunakan perintah ini karena *glFlush* tidak memaksa proses gambar untuk selesai tetapi memaksa proses gambar untuk segera dieksekusi, sehingga dijamin semua perintah gambar yang sebelumnya akan segera dieksekusi dalam suatu waktu tertentu.

GlFlush

1. **Fungsi untuk keperluan Transformasi**
2. **Fungsi Translasi (Translation)**

Translasi merupakan bentuk transformasi yang **memindahkan** posisi suatu objek, baik pada sumbu x, sumbu y, atau sumbu z. Fungsi yang digunakan untuk melakukan translasi adalah :

glTranslatef Tx, Ty, Tz

Parameter *Tx* digunakan untuk menentukan arah dan seberapa jauh suatu benda akan dipindahkan berdasarkan sumbu x. Parameter *Ty* digunakan untuk menentukan arah dan seberapa jauh suatu benda akan dipindahkan berdasarkan sumbu y. Sedangkan parameter *Tz* digunakan untuk menentukan arah dan seberapa jauh suatu benda akan dipindahkan berdasarkan sumbu z.

Contohnya :

glTranslatef 2, 2, 2

1. **Fungsi Rotasi (Rotation)**

Rotasi merupakan bentuk transformasi yang digunakan untuk **memutar** posisisuatu benda. Fungsi yang digunakan untuk melakukan rotasi ialah **glRotatef(θ, Rx, Ry, Rz)**

Parameter yang dibutuhkan pada fungsi tersebut ada 4 macam, yaitu parameter **θ** untuk besar sudut putaran, parameter ***Rx*** untuk putaran berdasarkan sumbu x, parameter ***Ry*** untuk putaran berdasarkan sumbu y, dan parameter ***Rz*** untuk putaran berdasarkan sumbu z. Jika parameter **θ** bernilai postif, maka objek akan diputar berlawanan arah jarum jam. Sedangkan jika parameter **θ** bernilai negatif, maka objek akan diputar searah jarum jam. Contohnya :

**glRotatef -30, 2, 2, 1**

1. **Fungsi Skala (Scalling)**

Skalasi merupakan bentuk transformasi yang dapat mengubah ukuran **(besar-kecil)** suatu objek. Fungsi yang digunakan untuk melakukan skalasi ialah :

**glScalef(Sx, Sy, Sz)**

Perubahan ukuran suatu objek diperoleh dengan mengalikan semua titik atau atau vertex pada objek dengan faktor skala pada masing-masing sumbu (parameter *Sx* untuk sumbu x, *Sy* untuk sumbu y, dan *Sz* untuk sumbu z).

Contohnya :

**glScalef(2, 2, 2)**

## Lighting (pencahayaan)

Lighting merupakan proses menghitung intensitas cahaya terutama pada 3-Dimensi point, biasanya diatas suatu permukaan.

Beberapa cara mengatasi masalah pencahayaan, antara lain :

* Mengerti persepsi dari cahaya (warna)
* Membuat sebuah solusi untuk merepresentasikan dan menghasilkan warna menggunakan komputer.
* Mengerti akan pengaruh cahaya dan objek

**` Bayangan**

* Bayangan akan muncul saat cahaya jatuh menyinari suatu objek.
* Pada dunia maya, layaknya cahaya, terdapat beberapa jenis bayangan yang dapat dihasilkan oleh komputer.

Bayangan bekerja sama dengan cahaya untuk memberi kesan natural atau realistic pada scene yang ada. Bayangan dapat membantu mendefinisikan posisi objek-objek, apakah berada di lantai atau melayang di udara.Bayangan yang dihasilkan bisa tajam dan solid namun bisa juga lembut dan buram (blurry).Keberadaan bayangan atau ketiadaannya dapat digunakan untuk memberi keseimbangan dan kontras pada objek-objek di dalam scene.

## Texture Mapping

Texture mapping merupakan teknik pemetaan sebuah tekstur pada pola gambar wireframe, dimana wireframe yang telah dibuat akan ditampilkan memiliki kulit luar seperti tekstur yang diinginkan. Dalam pemberian tekstur, perlu diperhatikan dasarnya seperti:

1. Menentukan tekstur
2. Membaca atau membangkitkan tekstur.
3. Menandai tekstur.
4. Mengenablekan tekstur.
5. Menandai koordinat tekstur pada vertek.
6. Menentukan parameter tekstur seperti : Wrapping , filtering, dsb.
   1. **Definisi Dev C++**

Dev C++ adalah sebuah produk IDE untuk bahasa pemrograman C dan C++ yang dikembangkan Microsoft. Bahasa C atau C++ adalah suatu bahasa pemrograman. Bahasa C termasuk sebagai bahasa pemrograman tingkat menengah, maksudnya bahasa C bisa dipelajari dengan lebih mudah karena mudah dimengerti tetapi mempunyai kemampuan yang tinggi.

Bahasa C bisa digunakan untuk merekayasa program untuk segala kebutuhan, baik untuk aplikasi bisnis, matematis atau bahkan game. Semua bahasa mempunyai kelemahan atau kelebihan sendiri-sendiri. Begitu juga dengan bahasa C. Adapun sebagian kelebihan dari bahasa C adalah sebagai berikut :

1. Banyak memiliki operator untuk mengolah / memanipulasi data.
2. Bahasa C termasuk sebagai bahasa yang terstruktur sehingga program dapat
3. lebih mudah dipahami atau dikembangkan.
4. Bahasa C lebih mudah dimengerti karena lebih mirip kepada bahasa manusia.
5. Kecepatan eksekusi tinggi.
6. Mengenal data pointer.

Sedangkan kelemahan dari bahasa C adalah :

1. Banyaknya operator atau cara penulisan program kadang menimbulkan
2. kebingungan para pemakainya.
3. Perlunya ketelitian dalam penulisan program karena perintah (*statement*)
4. dalam bahasa C bersifat *case sensitiv* (huruf kapital dan huruf kecil dibedakan

**BAB III**

**IMPLEMENTASI dan HASIL**

* 1. **Implementasi**

Pembuatan objek yang kami lakukan dengan pemanggilan fungsi-fungsi umum yang sudah tersedia di library OpenGL itu sendiri. Berikut objek-objek yang kami panggil :

* + 1. **Rincian Aplikasi**
       1. Terain yang dibuat terdiri dari terrain tanah dan terrain jalan raya
       2. Menara Big Ben London dibuat dengan mengabungakan kubus (*cube*) dengan kerucut yang *solid* di transformasikan, dimodifikasi menjadi bentuk limas dan di rotasi *.* Sama halnya dengan geddung-gedung yang berada sekitar menara Big Ben London sendiri menggunakan kubus balok tetapi dengan di *rotasi.* Jarum-jarum jam bisa berputar ada *fuction* nya yang menjadi sudut rotasi berubah ubah
       3. Pohon-pohonan menggabungkan kerucut (*cone*) sebagai batang pohon dan lingkaran (*sphere*) sebagai dedaunananya yang di transformasikan dan di rotasi sedemikian rupa sehingga terlihat seperti pohon
       4. Batu dibuat dengan lingkaran (*sphere*)
       5. Wartel menggabungkan glSolidCube dengan glSolidSphere(bola) yang di transformasikan sehingga menyerupai bentuk wartel.
    2. **Perangkat Lunak Pendukung**

Perangkat lunak yang dipasang pada sistem komputer yang digunakan untuk membangun aplikasi ini adalah sebagai berikut :

Sistem Operasi *Windows* 7.

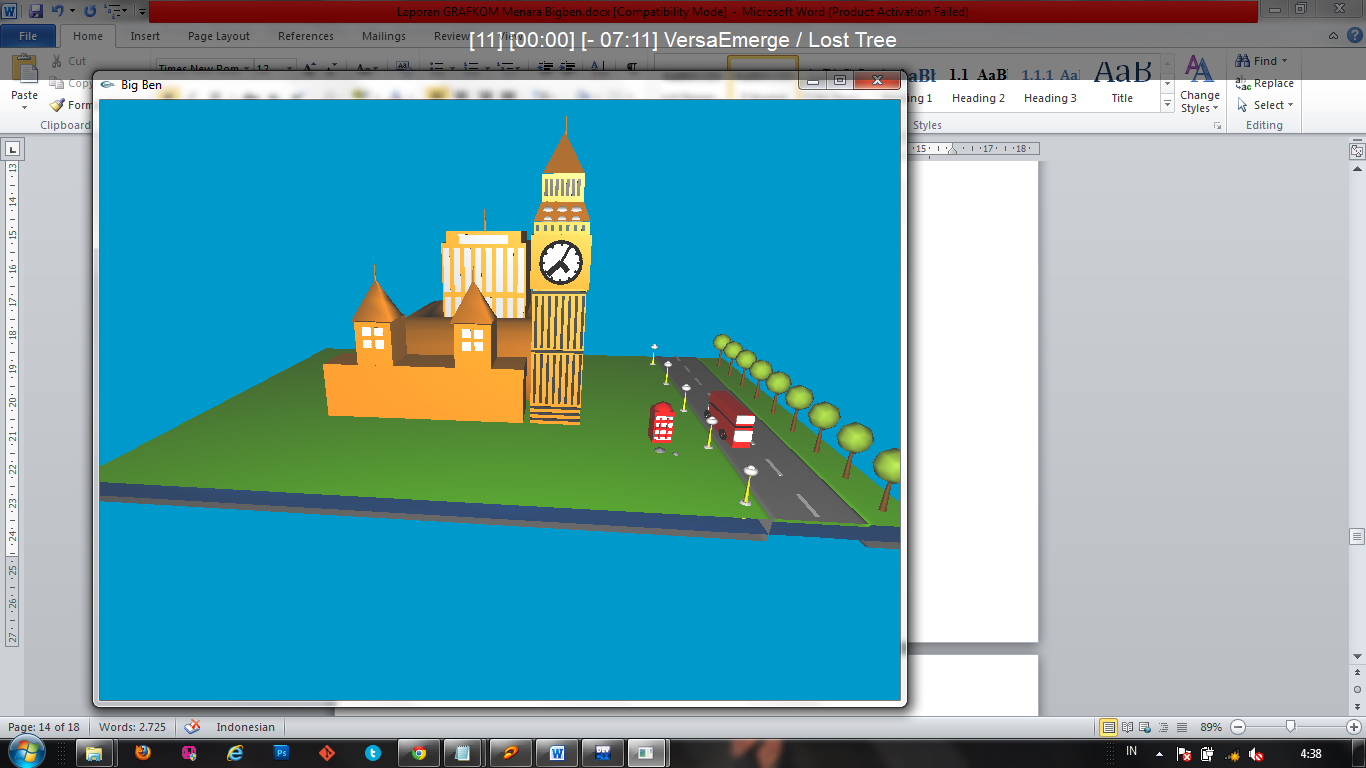
Dev C++ v.4.9.9.2

Glut32 dll

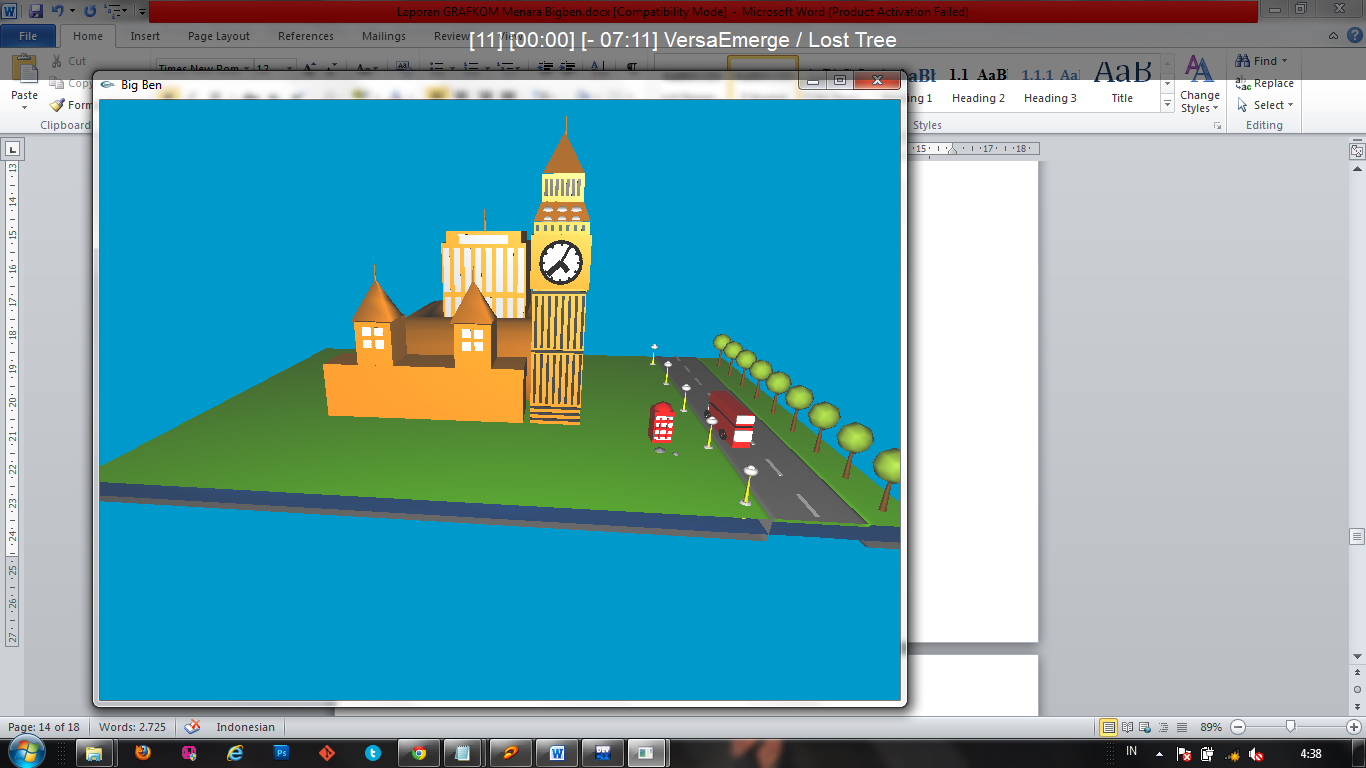
* + 1. **Perangkat Keras Pendukung**

Kebutuhan minimal perangkat keras (*hardware*) yang diperluka untuk mengimplementasikan program aplikasi ini adalah sebagai berikut :

* + - 1. Prosesor 1.60 Ghz
      2. Ram 2 Gb
      3. VGA 128MB
      4. Keyboard dan Mouse
  1. **Hasil Tampilan Aplikasi** 
     1. **Gambar Tampilan Awal Aplikasi menara bigben london**



**3.2.2 Gambar Tampilan Menara Big Ben**



**Source Program Fungsi Menara**

void menara()

{

//kotak yang tinggi

glPushMatrix();

glTranslatef(50.0, 75.0, 35.5);

glColor3f(1,0.49,0);

glScalef(40.0, 155.0, 40.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//kotak anu gede na men

glPushMatrix();

glTranslatef(50.0, 125.0, 36.5);

glColor3f(1, 0.49, 0);

glScalef(43.0, 38.0, 48.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//limas runving

glPushMatrix();

glTranslatef(50.9,200.5, 36.5);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(1,0.49,0);

glutSolidCone(1.5, 31.86,4,2); //kerucut

glPopMatrix();

//limas1

glPushMatrix();

glTranslatef(50.0, 153.5, 35.5);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(1,0.49,0);

glutSolidCone(28.5, 50.8, 4, 2); //kerucut

glPopMatrix();

for(int n=0;n<2;n++)

{

for(int t=0;t<8;t++)

{ glPushMatrix();

glTranslatef(33+(t\*5), 80.5-(n\*45),55);

glScalef(1.0, 20.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

}

}

//atas

for(int t=0;t<7;t++)

{ glPushMatrix();

glTranslatef(34+(t\*5), 148.5,85);

glScalef(1.0, 2.0, 1.0);

glColor3f(0.0f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

}

//atas pisan

for(int t=0;t<7;t++)

{ glPushMatrix();

glTranslatef(39+(t\*3.5), 173.5,85);

glScalef(1.0, 5.0, 1.0);

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

glutSolidCube(2); //kerucut

glPopMatrix();

}

for(int n=0;n<2;n++)

{

for(int t=0;t<8;t++)

{ glPushMatrix();

glTranslatef(33+(t\*5), 80.5-(n\*45),16);

glScalef(1.0, 20.0, 1.0);

glColor3f(0.4f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

}

}

//sekat gedung tengah

glPushMatrix();

glTranslatef(50,58,55);

glScalef(20.0, 1.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

//sekat gedung bawah

glPushMatrix();

glTranslatef(50,12,55);

glScalef(20.0, 1.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

//sekat gedung bawah

glPushMatrix();

glTranslatef(50,7,55);

glScalef(20.0, 1.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

//sekat gedung bawah

glPushMatrix();

glTranslatef(50,2,55);

glScalef(20.0, 1.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

//sekat gedung atas

glPushMatrix();

glTranslatef(50,103,55);

glScalef(20.0, 1.0, 1.0);

glColor3f(0.1f, 0.1f, 0.1f);

glutSolidCube(2); //kerucut

glPopMatrix();

//jendela

for(int z=0; z<3;z++)

{

int y=0;

for(int o=0; o<2;o++)

{

glPushMatrix();

glTranslatef(40+(z\*10), 160.5-(o\*6),47+y);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(0.9f, 0.9f, 0.9f);

glutSolidTorus(3,4,5,100); //kerucut

glPopMatrix();

y=y+2;

}

}

//kotak anu luhur pisan

glPushMatrix();

glTranslatef(50.0, 170.0, 35.5);

glColor3f(1,0.49,0);

glScalef(30.0, 30.0, 30.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//limas2

glPushMatrix();

glTranslatef(50.0, 185.0, 40.5);

glRotatef(45.0,0.0,1.0,0.0);

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

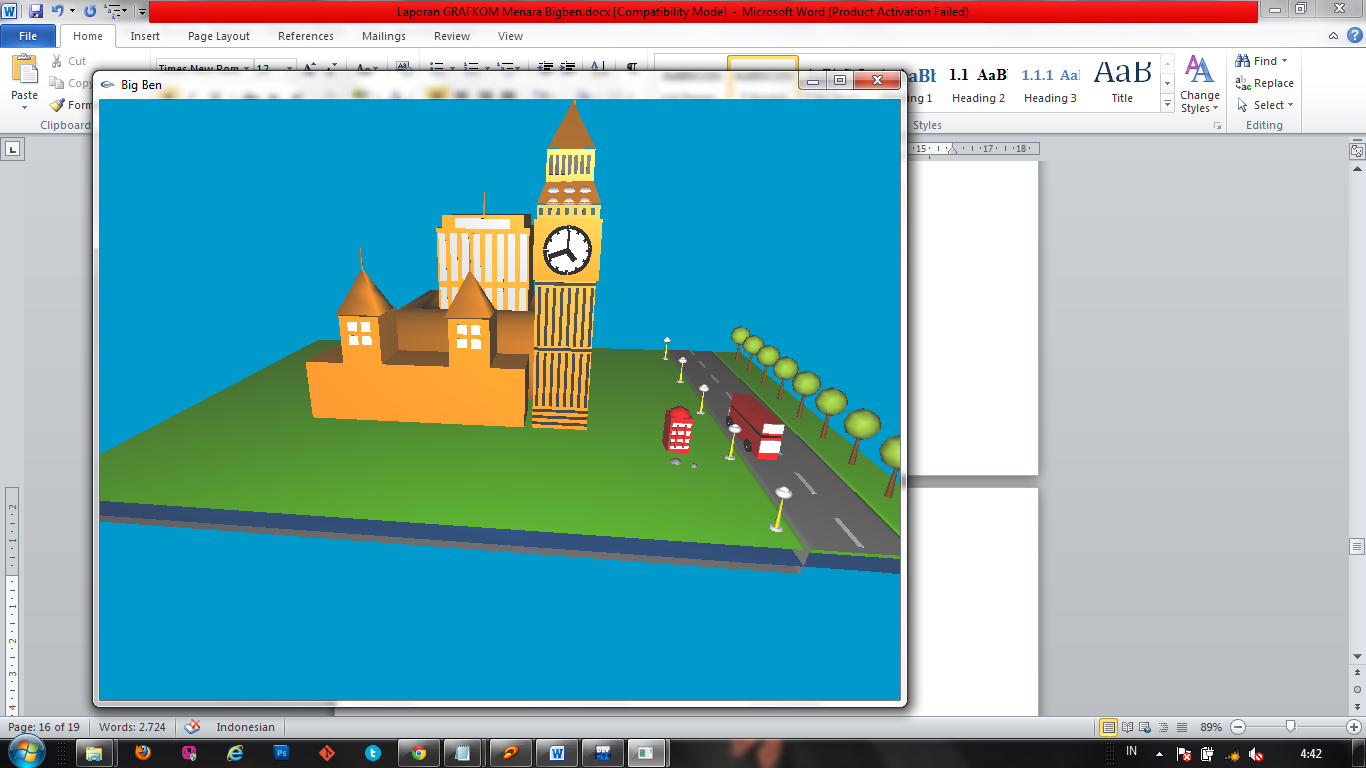
glColor3f(1,0.49,0);

glutSolidCone(20.5, 27.86,4,2); //kerucut

glPopMatrix();

}

* + 1. **Gambar Tampilan Gedung Pelengkap**



**Source Program Fungsi Gedung Pelengkap**

void gedungsebelah(int y)

{

//gedungpersegipanjangmen

glPushMatrix();

glColor3f(1,0.49,0);

glTranslatef(-57.9, 20.0, 35.7);

//glTranslatef(pindahx,pindahy,pindahz);

glScalef(328.8,91.8,81.8);

glutSolidCube(0.5); //kubus solid

glPopMatrix();

//jendela depan

for(int n=0;n<2;n++)

{

for(int t=0;t<8;t++)

{

glPushMatrix();

glTranslatef(-56+(t\*10), 110.5-(n\*45),-46);

glScalef(3.0, 20.0, 1.0);

glColor3f(0.7f, 0.7f, 0.7f);

glutSolidCube(2); //kerucut

glPopMatrix();

}

}

//jendela belakang

for(int n=0;n<2;n++)

{

for(int t=0;t<8;t++)

{

glPushMatrix();

glTranslatef(-56+(t\*10), 110.5-(n\*45),-136);

glScalef(3.0, 20.0, 1.0);

glColor3f(0.7f, 0.7f, 0.7f);

glutSolidCube(2); //kerucut

glPopMatrix();

}

}

glPushMatrix();

glTranslatef(-21, 138.5,-46);

glScalef(23.0, 4.0, 1.0);

glColor3f(0.7f, 0.7f, 0.7f);

glutSolidCube(2); //kerucut

glPopMatrix();

//gedungbelakang

glPushMatrix();

glColor3f(1,0.49,0);

glTranslatef(-30, 18.0, -90.7);

glRotatef(45.0,0.0,1.0,0.0);

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

glutSolidTorus(50,80,20,4); //donat

glPopMatrix();

//limas runving

glPushMatrix();

glTranslatef(-20.9,140.5, -56.5);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(1,0.49,0);

glutSolidCone(1.5, 31.86,4,2); //kerucut

glPopMatrix();

//menara tengah

glPushMatrix();

glColor3f(1,0.49,0);

glTranslatef(-20.9, 80.0, -90.7);

glScalef(158.8,220.8,181.8);

glutSolidCube(0.5); //kubus solid

glPopMatrix();

//menara tengah yang atas

glPushMatrix();

glColor3f(1,0.49,0);

glTranslatef(-20.9, 140.0, -90.7);

glScalef(138.8,20.8,181.8);

glutSolidCube(0.5); //kubus solid

glPopMatrix();

//gedungpersegimen

glPushMatrix();

glColor3f(1,0.49,0);

glTranslatef(-95.9+(y\*80), 60.0, 35.7);

glScalef(58.8,71.8,81.8);

glutSolidCube(0.5); //kubus solid

glPopMatrix();

//limasnaieumah

glPushMatrix();

glTranslatef(-97.9+(y\*80), 77.5, 35.5);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(1,0.49,0);

glutSolidCone(23.5, 31.86,4,2); //kerucut

glPopMatrix();

//limas runving

glPushMatrix();

glTranslatef(-97.9+(y\*80), 97.5, 35.5);

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(1,0.49,0);

glutSolidCone(1.5, 31.86,4,2); //kerucut

glPopMatrix();

//jendela atas kiri

glPushMatrix();

glTranslatef(-100.9+(y\*80), 70.5, 55.5);

glRotatef(3,0,1,0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(7.0, 7.0, 1.7);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela atas kiri

glPushMatrix();

glTranslatef(-90.9+(y\*80), 70.5, 55.5);

glRotatef(3,0,1,0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(7.0, 7.0, 1.7);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela bawah kiri

glPushMatrix();

glTranslatef(-100.9+(y\*80), 60.5, 55.5);

glRotatef(3,0,1,0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(7.0, 7.0, 1.7);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela bawah kiri

glPushMatrix();

glTranslatef(-90.9+(y\*80), 60.5, 55.5);

glRotatef(3,0,1,0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(7.0, 7.0, 1.7);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 40, 18.5);

glRotatef(3,0,1,0);

glColor3f(0.7f, 0.3f, 0.1f);

glScalef(9.0, 9.0, 1.5);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 44.0, 18.5);

glRotatef(3,0,1,0);

glRotatef(45,0,0,1);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 1.7);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 44.0, 18.5);

glRotatef(3,0,1,0);

glRotatef(45,0,0,1);

glColor3f(0.7f, 0.3f, 0.1f);

glScalef(7.0, 7.0, 1.5);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 40.0, 18.5);

glRotatef(3,0,1,0);

glColor3f(0.7f, 0.3f, 0.1f);

glScalef(1.0, 9.0, 2.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 40, 18.5);

glRotatef(3,0,1,0);

glColor3f(0.7f, 0.3f, 0.1f);

glScalef(9.0, 1.0, 2.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(-93.5, 40, 18.5);

glRotatef(3,0,1,0);

glColor3f(0.7f, 0.3f, 0.1f);

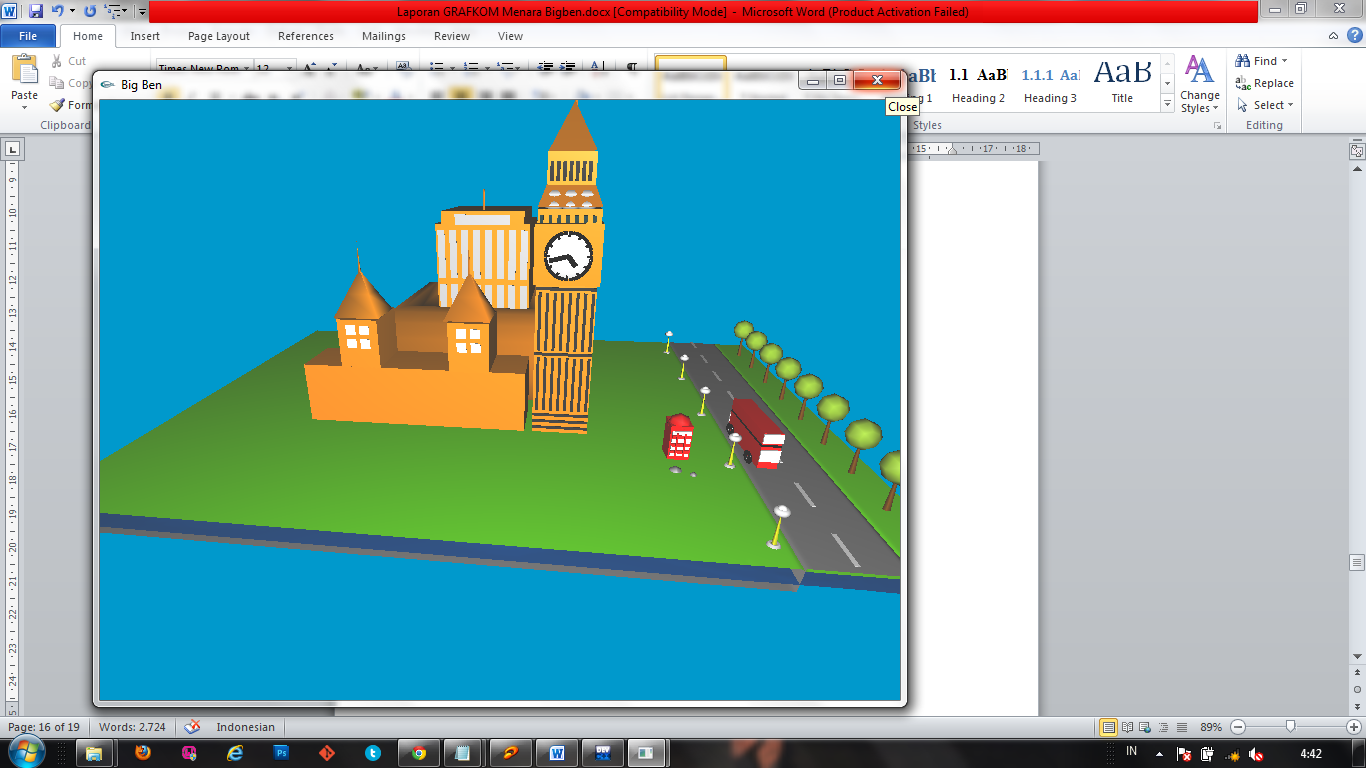
glScalef(9.0, 1.0, 2.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

}

**3.2.4 Gambar Tampilan Jalan,Bus Tingkat dan Telepon Umum**



**Source Program Fungsi Beberapa Objek**

void telepon(){

//wartel

glPushMatrix();

glTranslatef(128.5, 10.0, 78.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(14.0, 25.0, 15.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

glPushMatrix();

glColor3f(1.0, 0.0, 0.0);

glTranslatef(128.8, 20.0, 78.5);

glScaled(16, 16, 16);

glutSolidSphere(0.5, 8, 5); //bola

glPopMatrix();

//jendela hareupna

glPushMatrix();

glTranslatef(128.5, 11.2, 88.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 1.0, 1.0);

glScalef(12.0, 18.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela hareupna tapi anu garis beureum anu ka hiji

glPushMatrix();

glTranslatef(128.5, 5.8, 89.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(12.0, 2.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela hareupna tapi anu garis beureum anu ka dua

glPushMatrix();

glTranslatef(128.5, 10.8, 89.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(12.0, 2.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela hareupna tapi anu garis beureum anu ka tilu

glPushMatrix();

glTranslatef(128.5, 16.5, 89.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(12.0, 2.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela hareupna tapi anu garis bodas anu ka hiji

glPushMatrix();

glTranslatef(125.5, 11.5, 90.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(1.0, 16.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//jendela hareupna tapi anu garis bodas anu ka dua

glPushMatrix();

glTranslatef(130.5, 11.5, 90.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(1.0, 16.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

}

void jalan()

{

glPushMatrix();

glTranslatef(195.0, -2.5, 170.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, 120.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, 70.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, 20.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, -30.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, -80.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, -130.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

glPushMatrix();

glTranslatef(195.0, -2.5, -180.0);

glColor3f(0.9f, 0.9f, 0.9f);

glScalef(5.0, 5.0, 50.5);

glutSolidCube(0.5f); //kubus solid

glPopMatrix();

}

void bus(){

//bus bawah

glPushMatrix();

glTranslatef(178.5, 10.0, 68.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(12.0, 15.0, 55.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//bus atas

glPushMatrix();

glTranslatef(178.5, 23.0, 68.5);

glRotatef(3,0,1,0);

glColor3f(1.0, 0.0, 0.0);

glScalef(12.0, 8.0, 55.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//sekat

glPushMatrix();

glTranslatef(178.5, 18, 68.5);

glRotatef(3,0,1,0);

glColor3f(0.0, 0.0, 0.0);

glScalef(12.0, 1.0, 55.1);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//roda depan kiri

glPushMatrix();

glColor3f(0.9f, 0.9f, 0.9f);

glTranslatef(161.0, 18.5, 112.0);

glRotatef(0.0,0.0,1.0,1.0);

glRotatef(90.0,0.0,1.0,1.0);

glColor3f(0.0f, 0.0f, 0.f);

glutSolidTorus(1,3,100,200); //kerucut

glPopMatrix();

//roda belakang kiri

glPushMatrix();

glColor3f(0.9f, 0.9f, 0.9f);

glTranslatef(161.0, 18.5, 75);

glRotatef(0.0,0.0,1.0,1.0);

glRotatef(90.0,0.0,1.0,1.0);

glColor3f(0.0f, 0.0f, 0.f);

glutSolidTorus(1,3,100,200); //kerucut

glPopMatrix();

//kaca depan bawah

glPushMatrix();

glTranslatef(178.9, 12.8, 99);

glRotatef(3,0,1,0);

glColor3f(1.0, 1.0, 1.0);

glScalef(12.0, 8.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//kaca depan atas

glPushMatrix();

glTranslatef(178.9, 24.8, 99);

glRotatef(3,0,1,0);

glColor3f(1.0, 1.0, 1.0);

glScalef(12.0, 6.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

//kaca belakang

glPushMatrix();

glTranslatef(177.9, 11.8, 40.0);

glRotatef(3,0,1,0);

glColor3f(1.0, 1.0, 1.0);

glScalef(14.0, 8.0, 1.0);

glutSolidCube(1.0f); //kubus solid

glPopMatrix();

}

void pohon()

{

//pohon deket jalan

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, 170.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, 170.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, 130.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, 130.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, 90.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, 90.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, 50.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, 50.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, 10.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, 10.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, -30.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, -30.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, -70.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, -70.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, -110.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, -110.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.4f, 0.2f, 0.0f);

glTranslatef(235.0, -2.5, -150.0);

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3); //kerucut

glPopMatrix();

glPushMatrix();

glColor3f(0.5f, 0.7f, 0.1f);

glTranslatef(235.0, 25.5, -150.0);

glScaled(20,20,20);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

}

void lampujalan(int z)

{

glPushMatrix();

glColor3f(1.1f, 1.2f, 0.0f);

glTranslatef(161.0, -12.5, -150.0+(z\*80));

glRotatef(270, 1, 0, 0);

glScaled(4,4,35);

glutSolidCone(0.5,1,5,3);

glPopMatrix();

glPushMatrix();

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

glTranslatef(161.0, 19.5, -150.0+(z\*80));

glScaled(20,20,20);

glutSolidSphere(0.12,80,50); //bola

glPopMatrix();

glPushMatrix();

glColor3f(0.9f, 0.9f, 0.9f);

glTranslatef(161.0, 18.5, -150.0+(z\*80));

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(0.9f, 0.9f, 0.9f);

glutSolidTorus(2,2,15,100); //kerucut

glPopMatrix();

glPushMatrix();

glTranslatef(161.0, -2.5, -150.0+(z\*80));

glRotatef(45.0,0.0,1.0,0.0);

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

glColor3f(0.9f, 0.9f, 0.9f);

glutSolidTorus(2,2,5,100); //kerucut

glPopMatrix();

}

void batu()

{

glPushMatrix();

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

glTranslatef(125.0, -4.5, 100.0);

glScaled(10,10,10);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

glPushMatrix();

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

glTranslatef(135.0, -2.5, 105.0);

glScaled(5,5,5);

glutSolidSphere(0.5,8,5); //bola

glPopMatrix();

}

**BAB IV**

**PENUTUP**

## Kesimpulan

**Berdasarkan sistem yang telah dikembangkan dan hasil pengujian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut :**

1. Dapat mengetahui beberapa fungsi-fungsi OpenGL yang dapat membangun beberpa objek.
2. Dapat melihat perubahan dalam pembuatan objek seperti menjalankan perintah glScalef atau glTranslatef.

## Saran

**Ada beberapa hal yang dapat dijadikan saran dalam proses pembuatan menara Big Ben London 3D, antara lain :**

1. Penyesuaian bentuk yang dibuat harus disesuaikan dengan skala mapping.
2. Pembuatan program dengan menggunakan code seefisien mungkin.

**DAFTAR PUSTAKA**

<http://id.wikipedia.org/wiki/Grafika_komputer_3D>

<http://risaagustiana.blogspot.com/2012/10/seputar-grafik-komputer.html>

[http://badcoding.wordpress.com](http://badcoding.wordpress.com/)

<http://aflah7.wordpress.com/2010/10/14/konsep-pemodelan-grafik-2d-dan-3d/>