Perancangan dan Pembuatan Bangunan

“GPIB Taman Sari Salatiga”

Berbasis Vektor dengan OpenGL

672016143– Septian Triiriawan Parapak

672016064 – Rival Jekson Belora

672016195 – Raimundus Teniwut

Grafika Komputer

Fakultas Teknologi Informasi

Universitas Kristen Satya Wacana

2019

1. **PENDAHULUAN**

Jika kita berjalan – jalan di kota Salatiga , Provinsi Jawa Tengah, kita pasti akan melihat beberapa bangunan yang ada di kota tersebut. Nah, kami tertarik dengan salah satu bangunan di kota tersebut **GPIB Taman Sari**. Kabarnya gereja tersebut merupakan salah satu gereja tertua di daerah kota Salatiga, ini semakin membuat kami tertarik memilihnya sebagai bahan obyek Tugas Rancang kami.

Laporan yang kami buat ini berisi tentang panduan sederhana tentang penulisan laporan Tugas Rancang kami dengan obyek salah satu Bangunan yang ada di Salatiga yaitu **GPIB Taman Sari** Berbasis Vektor dengan OpenGL . Tugas Rancang yang kami buat adalah sebuah miniatur atau duplikat bangunan tesebut dalam bentuk aplikasi program untuk menampilkan vektor – vektor dengan OpenGL dari obyek yang kami pilih, jadi disini kami membuat suatu gambar 3 Dimensi dengan menggunakan bahasa pemrograman C++. Laporan ini kami susun sebagai pemenuhan Tugas Akhir ( Tugas Rancang ) mata kuliah Grafika Komputer ( Grafkom ) di semester 3 tahun ajaran 2018 / 2019 yang diampu oleh dosen kami Evangs Mailoa. Yang telah mengajarkan kami cara pembuatan bangunan 3D atau bidang 2D melalui bahasa pemrograman C++ berbasis Vektor dengan Open GL. Dalam perancangan dan pembuatan ini bisa menggunakan banyak aplikasi seperti Visual Studio, GIMP dan lain-lain. Disini kami menggunakan aplikasi Visual Studio Community 2019.

Metode penilitan yang kami gunakan adalah dengan survey lokasi, untuk mengambil beberapa dokumentasi dari bergai sudut agar untuk membantu penilitian kami supaya dalam pembuataan nanti kurang lebih mirip dengan object aslinya. Kan tidak mungkin juga kita membawa bangunan kemana-mana kecuali bangunannya kecil dan bisa digenggam tangan. Untuk itu kami mendokumentasikan object yang ingin kami buat vektornya.

Disini kami akan membahas secara singkat tentang, bagaimana cara kami membuat suatu bangunan dengan menggunakan bahasa pemrograman C++. Dalam pembahasan ini menganut konsep vektor dan bidang, dengan *software* bantu OpenGL.

**2. TINJAUAN PUSTAKA**

Pada artikel lain seperti pada laporan kelompok lain mungkin juga membahas pembuatan suatu bangunan tetapi pada dasarnya cara membuatnya sama-sama memakai bahasa pemrograman C++ dan *library* GLUT. Dan perlu diketahui GLUT merupakan *library* yang berisi kode pemorograman yang digunakan unutk mendesain dan menentukan titik-titik vektor. Seperti untuk menentukan bidang segi empat itu deperlukan empat titik, nah titik-titik tersebut mempunyai *syntax* program yang sama tetapi letak tiap titik tersebut berbeda, contoh kode program:

glBegin(GL\_POLYGON); memulai vektor dengan *polygon* yaitu ada isi(warna) bidang.

glColor3ub(255, 255, 255); menentukan warna dengan urutan *Red*, *Green*, *Blue*.

glVertex3f(1.0f, 1.0f, 0.0f); titik pertama dengan koordinat x,y,z (1,1,0);

glVertex3f(1.0f ,2.0f, 0.0f); titik kedua dengan koordinat x,y,z (1,2,0);

glVertex3f(2.0f, 2.0f, 0.0f); titik ketiga dengan koordinat x,y,z (2,2,0);

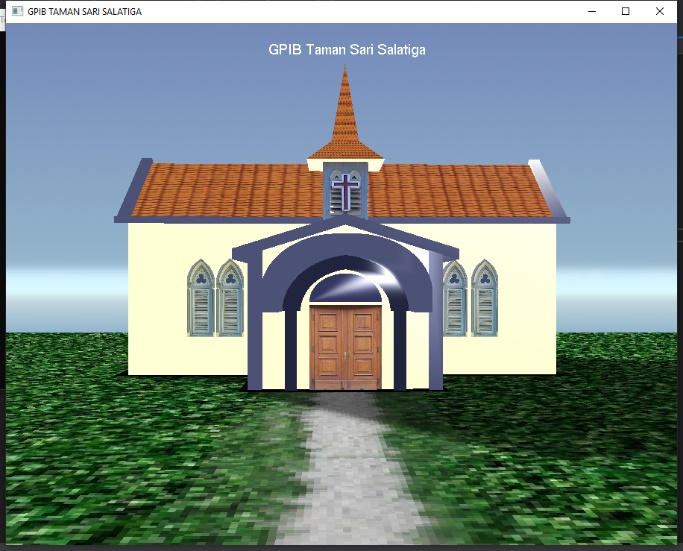
glVertex3f(2.0f, 1.0f, 0.0f); titik keempat dengan koordinat x,y,z (2,1,0);

glEnd(); akhir vektor.

Nah contoh program diatas adalah untuk menentukan bidang segi empat. glVertex3f merupkan kode program untuk menentukan vektor atau titik – titik yang kita inginkan dalam 3f (3 koordinat x,y,z dengan tipe data *float* (f) ). Dalam pembuatan vektor bidang atau bangunan mempunyai banyak titik – titik, dan untuk mengakalinya agar tidak ribet kita harus bisa logika bahasa pemrograman seperti dalam pengetahuan konsep *Dasar-Dasar Pemrograman* , *Pemrograman Berorientasi Object* ,dll. Nah setiap orang pastinya mempunyai logikanya masing-masing yang berbeda-beda dengan orang lain. Untuk itu kami tidak memberikan contoh artikel dari orang lain yang juga membahas tentang ini, karena kita tahu pasti bahasa yang mereka gunakan adalah Bahasa C++, tetapi perbedaannya ada di aplikasi software C++ yang mereka gunakan dan juga logika yang mereka gunakan. Tetapi konsep dasarnya sama yaitu menentukan titik – titik, pewarnaan, dan lain sebagainya, yang diperlukan untuk project kita.

**3. PEMBAHASAN**

Untuk pembahasan, diatas tadi sudah disebutkan bahwa kami membuat duplikat bangunan GPIB Taman Sari di Salatiga dalam bentuk vektor dengan menggunakan GLUT bahasa Pemrograman C++. Nah kami akan menjelasakan sedikit tentang fungsi–fungsi program yang kami gunakan dalam pembuatan tersebut.

Gambar 1.1. Gambar Asli Gambar 1.2. Model 3D

Penjelasan mengenai *library* yang dibutuhkan dalam perancangan model 3D.

Kami menggunakan :

**1. stdio.h**

*Library* pemograman c / c++ yang memiliki fungsi untuk memproses standar operasi *input* atau *output*.

**2. glut.h**

*Library* ini yang berperan penting dalam pengembangan menggunakan OpenGL. Berisi fungsi-fungsi untuk memanipulasi vektor.

**3. imageloader.h**

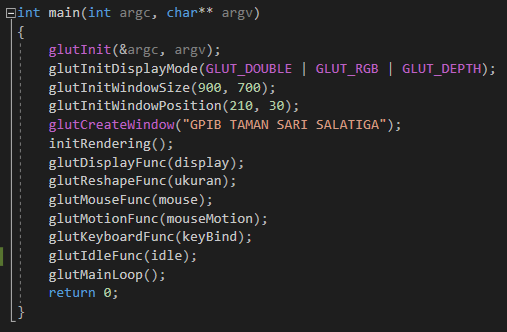
*Library* ini digunakan untuk memasukkan tekstur ke dalam obyek yang memiliki *fill*.

**4. object.h**

Kami yang merancang *library* ini untuk menampung fungsi-fungsi yang akan kami gunakan di *main* *class* agar pemograman menjadi lebih ringkas. Sebagai contoh untuk membuat balok yang mana kita tahu terdiri dari 6 sisi (GL\_QUADS). Sehingga apabila ada kubus yang ingin kita panggil hanya tinggal memanggil fungsinya dengan melemparkan *parameter* berupa letak x, y, z, panjang, lebar, dan tinggi tanpa harus mengetikkan GL\_QUADS satu persatu.

*Method* yang kami gunakan (Diurutkan berdasarkan yang pertama kali dieksekusi):

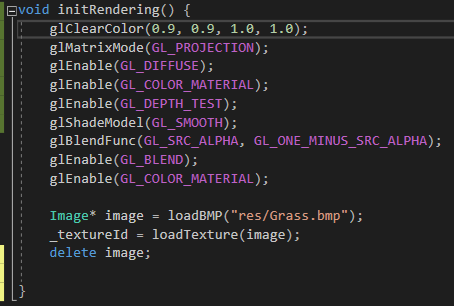
**i. main()**



Gambar 2.1. Perintah di dalam fungsi main()

*Method* ini berisi fungsi-fungsi yang akan dipanggil pertama kali ketika program dieksekusi. Seperti ukuran jendela, posisi jendela di *desktop*, *title* jendela, lalu fungsi lainnya. Lalu pada akhir *method* ditambahkan *glutMainLoop();* agar program kita di*refresh* terus menerus.

**ii. initRendering()**



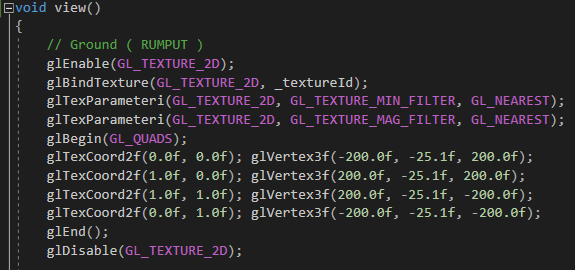
Gambar 2.2. Perintah di dalam fungsi initRendering()

*Method* ini berisi fungsi-fungsi untuk menginisialisasi *workspace* yang akan digunakan. Seperti warna dasar, fitur *lighting* yang akan diaktifkan, memasukkan lokasi tekstur ke dalam variabel sehingga nantinya hanya variabelnya yang dipanggil.

**iii. display()**

*Method* ini berfungsi sebagai direktornya. Di dalam *method* ini terdapat perintah kemana arah pandangan kamera atau direktornya, kemana arah *lighting*. Kami menempatkan gambar latar di method ini. Tepatnya sebelum perintah *gluLookAt();* agar gambar nantinya tidak ikut bergerak saat pandangan kamera dipindahkan.

**iv. view()**



Gambar 2.3. Perintah di dalam fungsi view()

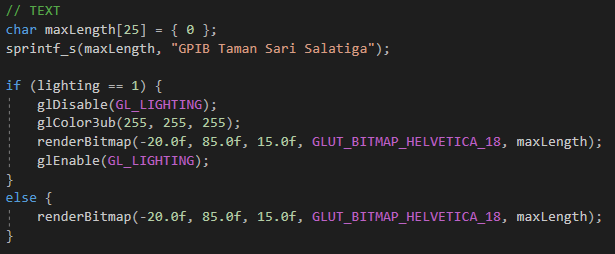
*Method* ini berisi perintah untuk memunculkan obyek ke dalam *workspace*.

Salah satu contoh untuk memasukkan gambar ke dalam bidang dapat dilihat pada Gambar 2.3. glTexCoord2f berfungsi untuk menempatkan koordinat gambar pada bidang.



Gambar 2.3.1. Contoh koordinat

Pada Gambar 2.3.1, bisa kita lihat untuk koordinat x = 0, y = 0 terletak pada sudut kiri bawah.

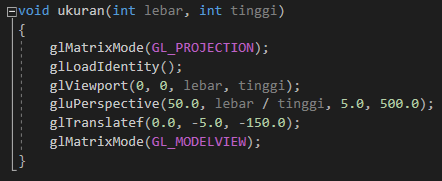


Gambar 2.3.2. Contoh tampilkan teks

Pada Gambar 2.3.2 terdapat contoh untuk menampilkan teks pada *workspace*. Pembuatan variabel yang menampung teks “GPIB Taman Sari Salatiga”. Akan tetapi apabila GL\_LIGHTING diaktifkan, teks tidak muncul sehingga untuk menyiasatinya kami menonaktifkan GL\_LIGHTING terlebih dahulu dan akan kembali diaktifkan diakhir pemanggilan fungsi teks.

 perintah ini digunakan untuk menampilkan atap. Pada saat program pertama diinisialisasikan, nilai xray adalah 0 sehingga atap akan ditampilkan. Saat *user* menekan tombol mouse tengah, nilai xray berubah menjadi 1 lalu saat program redisplay() dijalankan, atap tidak akan muncul karena apabila nilai xray != 0 maka perintah atap tidak akan dijalankan. Untuk merubah xray menjadi 0, tekan lagi tombol *mouse* tengah

**v. ukuran()**

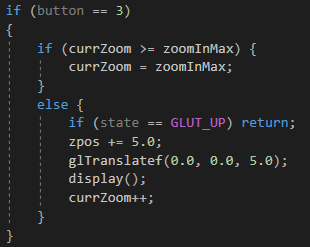


Gambar 2.4. Perintah di dalam fungsi ukuran()

*Method* ini berisi fungsi untuk mengatur ukuran *workspace* yang akan digunakan. Perbedaan mengenai *workspace* dan jendela adalah ketika ukuran jendela lebih besar dari *workspace*, akan ada *space* antara gambar dengan pinggiran jendela. Jadi *workspace* adalah letak vektor-vektor yang nantinya akan digambar.

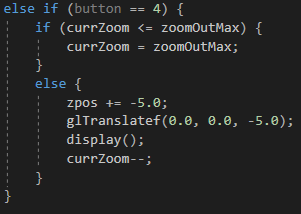
**vi. glutMouseFunc(mouse);**

*Method* ini sebagai *listener* apabila ada perintah yang dilakukan pada *mouse*. Seperti klik kiri, klik tengah, klik kanan, scroll up, dan scroll down*.*



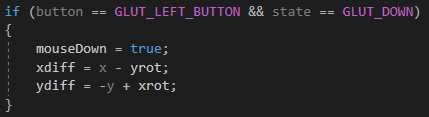
Gambar 2.5.1 Scroll Up

Pada Gambar 2.5.1 terdapat *listener* untuk *mouse* *scroll* *up*. *Mouse* *scroll* *up* pada GLUT didefinisikan sebagai *button* 3. Di dalam bagan if diatas, apabila scroll up sudah mencapai batas maksimum yang bisa dilakukan, tidak dapat melakukan *scroll* *up* lagi. Ini bertujuan untuk membatasi *zoom* *in* agar gambar masih tetap bisa dilihat.



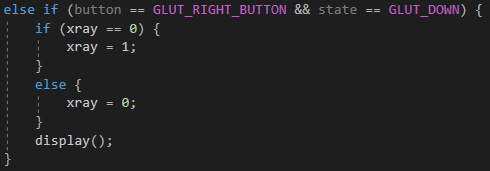
Gambar 2.5.2 Scroll Down

Pada Gambar 2.5.2 terdapat *listener* untuk mouse *scroll* *down*. *Mouse* *scroll* *down* pada GLUT didefinisikan sebagai *button* 4. Di dalam bagan if diatas, apabila *scroll* *down* sudah mencapai batas maksimum yang bisa dilakukan, tidak dapat melakukan *scroll* *down* lagi. Ini bertujuan untuk membatasi *zoom* *out* agar gambar masih tetap bisa dilihat.



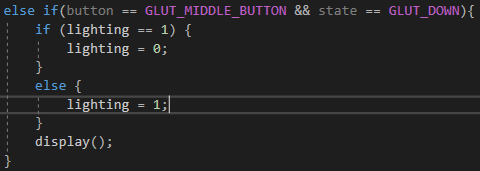
Gambar 2.5.3 Klik Kiri

Pada Gambar 2.5.3 terdapat *listener* untuk klik kiri. Apabila mouse digerakkan tanpa klik kiri terlebih dahulu, tidak akan ada perubahan pada *workspace*. Sedangkan apabila klik kiri dilakukan, variabel mouseDown akan berubah menjadi true.



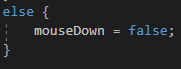
Gambar 2.5.4 Klik Kanan

Pada Gambar 2.5.4 terdapat *listener* untuk klik kanan. Apabila klik kanan dilakukan dan nilai xray bernilai 0, maka variable xray akan bernilai 1 (lihat hal. 11) *vice versa*. Lalu perintah display(); akan dieksekusi yang mana perintah *display* terdapat perintah untuk *refresh*.



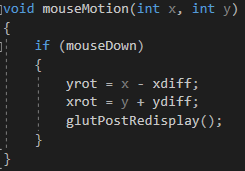
Gambar 2.5.5 Klik Tengah

Pada Gambar 2.5.5 terdapat *listener* untuk klik tengah. Apabila klik tengah dilakukan dan *lighting* bernilai 1, maka nilai *lighting* akan bernilai 0 *vice versa.* Lalu perintah display(); akan dieksekusi yang mana perintah *display* terdapat perintah untuk *refresh*.

 Apabila tidak ada dari salah satu fungsi *mouse* yang dilakukan, mouseDown akan bernilai *false*.

**vii. glutMotionFunc(mouseMotion);**

*Method* ini berfungsi untuk menjalankan perintah apabila klik and *drag* dilakukan.

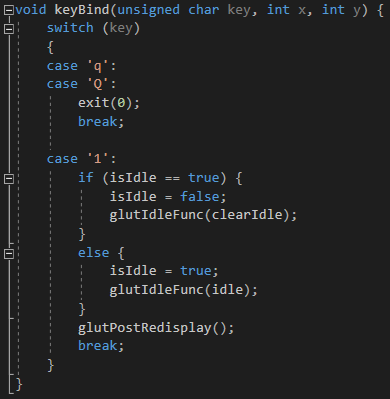


Gambar 2.6 Mouse Drag

Pada Gambar 2.6 apabila mouseDown bernilai *true*, maka rotasi *workspace* akan dilakukan berdasarkan gerakan *mouse*.

**viii. glutKeyboardFunc(keyBind);**

*Method* ini berfungsi sebagai *listener* apabila ada tombol yang ditekan pada *keyboard* dan akan menjalankan perintah berdasarkan tombol apa yang ditekan sesuai dan akan menjalankan perintah apa yang terjadi ketika tombol itu ditekan.



Gambar 2.7 Perintah di dalam fungsi keyBind()

Pada Gambar 2.7 apabila tombol Q atau q pada *keyboard* ditekan, maka program akan ditutup, apabila tombol 1 pada *keyboard* ditekan maka akan memeriksa apakah sedang *idle* atau tidak. Apabila sedang *idle*, maka glutIdleFunc() akan dikosongkan. Apabila tidak sedang *idle*, maka glutIdleFunc() akan di set dengan fungsi *idle* yang mana fungsi *idle* adalah fungsi untuk merotasi gambar secara otomatis.

**ix. glutIdleFunc(idle);**

*Method* ini berisi perintah yang akan dijalankan ketika program sedang *idle* atau tidak ada perintah apa-apa yang diberikan *user*. Perintah di dalam fungsi ini berfungsi untuk memutar gambar berdasarkan sumbu y secara otomatis

*Klarifikasi : Sebenarnya kesembilan fungsi diatas bisa saja disatukan hanya menjadi fungsi yang lebih sedikit akan tetapi untuk mengelompokkannya berdasarkan kapan dieksekusi itulah yang membuat kami untuk mengelompokkannya ke dalam beberapa fungsi*

**4. COMPATIBILITY TEST**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Platform** | **isSuccess** |
| Testing run project menggunakan IDE Code::Blocks dengan library imageloader | x64 Linux (Debian Based) | **TIDAK** |
| Testing run project menggunakan IDE Code::Blocks tanpa library imageloader | x64 Linux (Debian Based) | **YA** |
| Testing run executable menggunakan wineHQ | x64 Linux (Debian Based) | **YA** |

**5. SOFTWARE**

IDE : Microsoft Visual Studio Community 2019

Debugger : Local Windows Debugger

GLUT : FreeGlut & glew 1.11.0

**6. KESIMPULAN**

Dari penjelasan diatas, kami dapat menyimpulkan bahwa dalam bahasa C++ dapat digunakan untuk mendesain suatu bangunan 3D maupun 2D seperti yang kita kerjaan. Dan selain itu dalam bahasa C++ terutama software OpenGL juga dapat membuat suatu Game, animasi gerak, dan masih banyak lagi. Dan juga dalam program tersebut banyak sekali macam-macam fungsi yang dapat kita gunakan seperti yang telah kami kami jelaskan diatas, seperti glVertex, glColor, dan lain- lain. Untuk itu pintar-pintarlah kita dalam memahami bahasa pemrograman ini, jika kita banayak belajar mungkin kita bisa menajdi programer yang handal dan bisa membuat aplikasi yang berguna bagi masyarakat. Sekian laporan dari kami kiranya pambaca sekalian dapat memahami apa yang telah kami sampaikan dalam laporan ini. Dan apabila kami ada kesalahan dalam penulisan maupun penjelasan yang sengaja maupun tidak sengaja kami mohon maaf. Terima kasih.