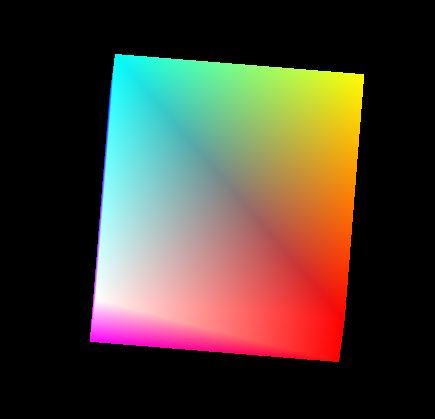
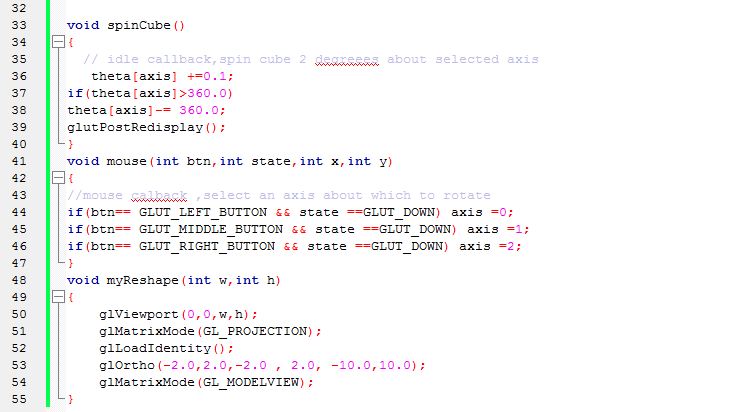
****

****

**Rectangle 3D Animation**

**With OpenGLES**

1. **Tujuan Penelitian** 
   1. **Definisi Masalah**

Dalam penelitian ini memuat bagaimana cara membuat sebuah animasi 3D dengan menggunakan OpenGLES. Dalam membuat game pastinya tidak terlepas dari animasi dan grafik 3D (Dimensi). Animasi adalah sebuah hal penting dalam pengembangan sebuah game, tanpa adanya animasi, sebuah game pasti tidak akan menarik bahkan saat sekarang ini tidak ada lagi Game tanpa animasi. Begitu pun Grafik 3D, dimana didalam grafik 3D memuat efek warna, shadow, tekstur, dan lain-lain. Tujuan dari 3D ini adalah agar supaya animasi ataupun game yang dibuat atau dikembangkan akan terlihat sepertinya kenyataan atau realita.

* 1. **Analisa Masalah**

Dalam menganalisis masalah tentunya kita harus menguraikan beberapa hal :

* Aplikasi apa yang akan digunakan untuk membuat animasi 3D ini
* Bahasa pemograman dalam pembuatan animasi ini
* Bentuk animasi yang akan dibuat
* Apa tujuan dari pembuatan animasi ini
* Pengembangan animasi ini untuk masa depan
  + 1. **Aplikasi/Software yang digunakan**

Dalam pembuatan animasi ini saya menggunakan aplikasi/software Codeblocks 17.12. Dimana saya memasukkan library OpenGL didalamnya. Alasan mengapa saya menggunakan aplikasi ini adalah :

1. Software ini sangat ringan dari segi memory
2. Software ini sangat mudah digunakan dan diinstall
3. Tidak harus menggunakan laptop dengan spesifikasi tinggi
4. Sangat mudah dalam menginstal library untuk OpenGL
   * 1. **Bahasa Pemrograman**

Dalam Penelitian pembuatan animasi 3D ini saya menggunakan Bahasa pemrograman C++. Karena, saya adalah seorang C++ Programmer, dan Bahasa pemrograman yang paling saya kuasai adalah C++.

* + 1. **Bentuk Animasi**

Sesuai dengan judul diatas, saya akan membuat sebuah animasi 3D dengan OpenGLES dengan bentuk kotak (rectangle). Tidak ada alasan khusus mengapa saya memilih bentuk kotak. Saya lebih focus pada animasi dan tekstur.

* + 1. **Tujuan animasi**

Pembuatan animasi ini bertujuan untuk meneliti gerakan animasi dan tekstur dari bentuk animasi seperti: warna, efek, dan shadow.

**1.2.5** **Pengembangan animasi**

Dalam animasi ini ada beberapa hal yang dapat dikembangkan, yaitu:

* Gerakan animasi
* Warna dan tekstur
* Dapat digabung dengan bentuk-bentuk lain

1. **Solusi**
   1. **Desripsi Solusi**

Pada bab ini, berisi tentang penjelasan aplikasi dan animasi yang telah dirancang. Pada perancangan animasi ini, saya menggunakan aplikasi/software Codeblocks yang telah diinstal didalamnya library dari OpenGL. Bahasa pemrograman yang pakai dalam merancang aplikasi/animasi ini adalah Bahasa pemrograman c++

* + 1. **Aplikasi Codeblocks**

**Code::Blocks** adalah suatu program [lingkungan pengembangan terpadu](https://id.wikipedia.org/wiki/Lingkungan_pengembangan_terpadu) [bebas](https://id.wikipedia.org/wiki/Perangkat_lunak_bebas), [nirlaba](https://id.wikipedia.org/wiki/Perangkat_gratis), [bersumber terbuka](https://id.wikipedia.org/wiki/Sumber_terbuka) dan [lintas platform](https://id.wikipedia.org/wiki/Lintas_platform). Program yang ditulis dalam [C++](https://id.wikipedia.org/wiki/C++) beserta [wxWidgets](https://id.wikipedia.org/wiki/WxWidgets) untuk GUI-nya ini bisa digunakan bersama dengan berbagai macam [kompilator](https://id.wikipedia.org/wiki/Kompilator), contohnya [GCC](https://id.wikipedia.org/w/index.php?title=GCC&action=edit&redlink=1) dan [Visual C++](https://id.wikipedia.org/wiki/Visual_C++). Peralatannya yang tersedia tergantung dari "plugin" yang ada dipasang. Sekarang ini, Code::Blocks lebih tersedia sebagai perangkat pengembangan dalam bahasa [C](https://id.wikipedia.org/wiki/C_(bahasa_pemrograman)) dan [C++](https://id.wikipedia.org/wiki/C++), walaupun program ini juga bisa disesuaikan, dan mungkin akan membutuhkan pemasangan tambahan, untuk pengembangan perangkat lunak [ARM](https://id.wikipedia.org/wiki/Arsitektur_ARM), [AVR](https://id.wikipedia.org/wiki/Atmel_AVR), [DirectX](https://id.wikipedia.org/wiki/DirectX), FLTK, [Fortran](https://id.wikipedia.org/wiki/Fortran), GLFW, GLUT, [GTK+](https://id.wikipedia.org/wiki/GTK+), [Irrlicht](https://id.wikipedia.org/w/index.php?title=Irrlicht&action=edit&redlink=1), Lightfeather, MATLAB, OGRE, [OpenGL](https://id.wikipedia.org/wiki/OpenGL), [Qt](https://id.wikipedia.org/wiki/Qt_toolkit), SDL, SFML, STL, SmartWin dan wx. Code::Blocks tersedia di [sistem operasi](https://id.wikipedia.org/wiki/Sistem_operasi) [Windows](https://id.wikipedia.org/wiki/Microsoft_Windows), [Linux](https://id.wikipedia.org/wiki/Linux), [Mac OS X](https://id.wikipedia.org/wiki/OS_X) dan [FreeBSD](https://id.wikipedia.org/wiki/FreeBSD).

* + 1. **Bahasa pemrograman C++**

**C++** adalah [bahasa pemrograman](https://id.wikipedia.org/wiki/Bahasa_pemrograman) [komputer](https://id.wikipedia.org/wiki/Komputer) yang di buat oleh Bjarne Stroustrup, yang merupakan perkembangan dari bahasa C dikembangkan di [Bong Labs](https://id.wikipedia.org/wiki/Bell_Labs) ([Dennis Ritchie](https://id.wikipedia.org/wiki/Dennis_Ritchie)) pada awal tahun 1970-an, Bahasa itu diturunkan dari bahasa sebelumnya, yaitu B, Pada awalnya, bahasa tersebut dirancang sebagai bahasa pemrograman yang dijalankan pada sistem [Unix](https://id.wikipedia.org/wiki/Unix), Pada perkembangannya, versi ANSI (American National Standart Institute) [Bahasa pemrograman C](https://id.wikipedia.org/wiki/Bahasa_pemrograman_C) menjadi versi dominan, Meskipun versi tersebut sekarang jarang dipakai dalam pengembangan sistem dan jaringan maupun untuk sistem embedded, Bjarne Stroustrup pada Bel labs pertama kali mengembangkan C++ pada awal [1980](https://id.wikipedia.org/wiki/1980)-an. Untuk mendukung fitur-fitur pada C++, dibangun efisiensi dan sistem support untuk pemrograman tingkat rendah (low level coding).[[1]](https://id.wikipedia.org/wiki/C++) Pada C++ ditambahkan konsep-konsep baru seperti class dengan sifat-sifatnya seperti inheritance dan overloading.[[*butuh rujukan*](https://id.wikipedia.org/wiki/Wikipedia:Kutip_sumber_tulisan)] Salah satu perbedaan yang paling mendasar dengan bahasa C adalah dukungan terhadap konsep pemrograman berorientasi objek ([Object Oriented Programming](https://id.wikipedia.org/wiki/Object_Oriented_Programming)).[[2]](https://id.wikipedia.org/wiki/C++)

tercipta untuk mu .[[3]](https://id.wikipedia.org/wiki/C++)

* + 1. **Library OpenGL**

**OpenGL** (**Open G**raphics **L**ibrary)[[3]](https://id.wikipedia.org/wiki/OpenGL) adalah spesifikasi standar yang mendefinisikan sebuah lintas-bahasa, [lintas platform](https://id.wikipedia.org/wiki/Lintas_platform) [API](https://id.wikipedia.org/wiki/API) untuk mengembangkan aplikasi yang menghasilkan grafis komputer [2D](https://id.wikipedia.org/wiki/2D) maupun[3D](https://id.wikipedia.org/wiki/3D). Antarmuka terdiri dari lebih dari 250 panggilan fungsi yang berbeda yang dapat digunakan untuk menggambar tiga dimensi yang adegan-adegan kompleks dari bentuk-bentuk primitif sederhana. OpenGL dikembangkan oleh [Silicon Graphics Inc](https://id.wikipedia.org/w/index.php?title=Silicon_Graphics_Inc&action=edit&redlink=1) (SGI) pada tahun 1992 [2] dan secara luas digunakan dalam [CAD](https://id.wikipedia.org/wiki/CAD), [realitas maya](https://id.wikipedia.org/wiki/Realitas_maya), visualisasi ilmiah, visualisasi informasi, dan [simulasi penerbangan](https://id.wikipedia.org/wiki/Simulasi_penerbangan). Hal ini juga digunakan dalam video game, di mana bersaing dengan Direct3D on Microsoft Windows platform (lihat vs OpenGL Direct3D). OpenGL dikelola oleh sebuah teknologi [konsorsium](https://id.wikipedia.org/wiki/Konsorsium) nirlaba yaitu Khronos Group.

* 1. **Plan**

Dalam bab ini saya akan membahas bagaimana perencanaan dalam merancang animasi ini mulai dari pembuatan bentuk, warna, hingga jalannya animasi. Mari kita bahas satu persatu.

* + 1. **Perencanaan Bentuk**

Sesuai dengan judul dari animasi ini, saya akan membuat sebuah animasi yang berbentuk kubus (kotak). Sebenarnya tidak ada alasan khusus kenapa saya memilih membuat bentuk kotak. Dari beberapa referensi yang saya baca dan saya coba praktikkan, saya melihat bentuk kotak adalah bentuk simple atau sederhana, namum cocok dijadikan untuk sebuah contoh atau rujukan dalam merancang animasi atau game yang lebih kompleks. Mengapa?, karena dalam bentuk kotak ini kita dapat memperhitungkan setiap sisi dalam untuk dijadikan bentuk yang utuh, atau bahasa simplenya bentuk ini bisa dikembangkan menjadi bentuk apa saja.

* + 1. **Perencanaan warna**

Dalam perencanaan warna, sesuai dengan topic diatas saya akan menggunakan multi warna, agar telihat seperti bentuk 3D (Dimensi). Walaupun memang ini animasi 3D sederhana, tetapi saya akan berusaha membuat warna yang akan terlihat menjadi bentuk 3D.

* 1. **Implementation**

Impementation ini adalah bagian terpenting dari penelitian ini, dimana dibagian ini kita dapat melihat bagaimana cara membuat animasi ini serta jalannya aplikasi atau animasi ini. Dalam bagian implementasi ini kita akan membahaa cara membuat program hingga berjalannya program.

* + 1. **Merancang Aplikasi/Animasi**
* Create Project
* Buka aplikasi Codeblocks
* Pilih Create a new project
* Pilih GLUT Project, klik Next
* Tulis nama projek yang ingin di rancang
* Pilih tempat penyimpanan, pilih Next
* Pilih Finish
* Menulis Sourcecode Program

#include<stdlib.h>

#include<GL/glut.h>

GLfloat vertices[][3] = {{-1.0,-1.0,-1.0},{1.0,-1.0,-1.0},{1.0,1.0,-1.0},{-1.0,1.0,-1.0},{-1.0,-1.0,1.0},{1.0,-1.0,1.0},{1.0,1.0,1.0},{-1.0,1.0,1.0}};

GLfloat colors[][3] = {{0.0,0.0,0.0},{1.0,0.0,0.0},{1.0,0.0,0.0},{1.0,1.0,0.0},

{0.0,0.0,1.0},{1.0,0.0,1.0},{1.0,1.0,1.0},{0.0,1.0,1.0}};

static GLfloat theta[]={0.0,0.0,0.0};

GLint axis =1;

void polygon(int a, int b,int c,int d)

{

//draw a polygon via list of vertices

glBegin(GL\_POLYGON);

glColor3fv(colors[a]);

glVertex3fv(vertices[a]);

glColor3fv(colors[b]);

glVertex3fv(vertices[b]);

glColor3fv(colors[c]);

glVertex3fv(vertices[c]);

glColor3fv(colors[d]);

glVertex3fv(vertices[d]);

glEnd();

}

void colorcube(void)

{ //map vertices to faces

polygon(0,3,2,1);

polygon(2,3,7,6);

polygon(0,4,7,3);

polygon(1,2,6,5);

polygon(4,5,6,7);

polygon(0,1,5,4);

}

void spinCube()

{

// idle callback,spin cube 2 degreees about selected axis

theta[axis] +=0.1;

if(theta[axis]>360.0)

theta[axis]-= 360.0;

glutPostRedisplay();

}

void mouse(int btn,int state,int x,int y)

{

//mouse calback ,select an axis about which to rotate

if(btn== GLUT\_LEFT\_BUTTON && state ==GLUT\_DOWN) axis =0;

if(btn== GLUT\_MIDDLE\_BUTTON && state ==GLUT\_DOWN) axis =1;

if(btn== GLUT\_RIGHT\_BUTTON && state ==GLUT\_DOWN) axis =2;

}

void myReshape(int w,int h)

{

glViewport(0,0,w,h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-2.0,2.0,-2.0 , 2.0, -10.0,10.0);

glMatrixMode(GL\_MODELVIEW);

}

void display(void)

{

// display callback , clear frame buffer an Z buffer ,rotate cube and draw , swap buffer.

glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

glRotatef(theta[0],1.0,0.0,0.0);

glRotatef(theta[1],0.0,1.0,0.0);

glRotatef(theta[2],0.0,0.0,1.0);

colorcube();

glFlush();

glutSwapBuffers();

}

int main(int argc,char\*\* argv)

{

glutInit(&argc,argv);

//need both double buffering and Zbuffer

glutInitDisplayMode(GLUT\_DOUBLE|GLUT\_RGB|GLUT\_DEPTH);

glutInitWindowSize(500,500);

glutCreateWindow("Rotating a color cube ");

glutDisplayFunc(display);

glutReshapeFunc(myReshape);

glutIdleFunc(spinCube);

glutMouseFunc(mouse);

glEnable(GL\_DEPTH\_TEST); //Enable hidden surface removal

glutMainLoop();

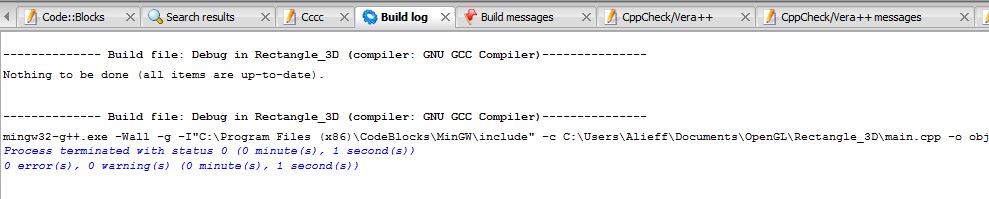
return 0;

} Source diatas adalah source untuk membuat bentuk kotak tanpa warna.

* + 1. **Menjalankan Animasi/Aplikasi**

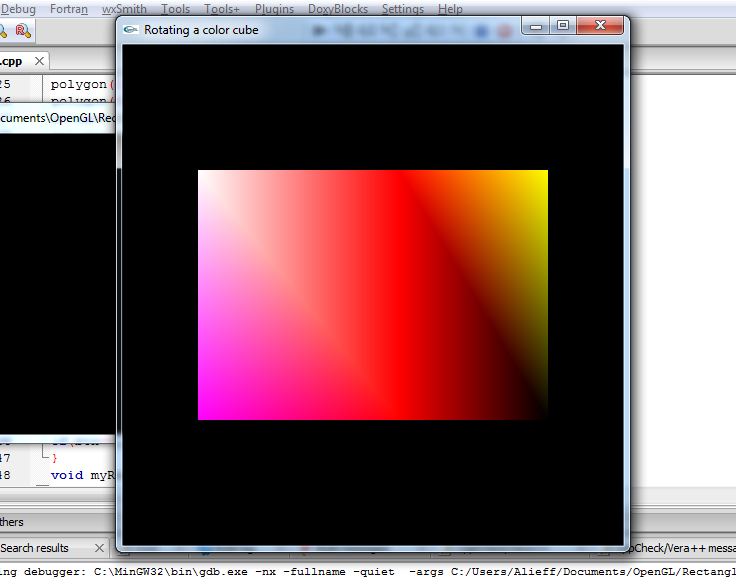
Ini adalah tahapan terakhir dari proses perancangan animasi/aplikasi Rectangle 3D ini. Pada tahapan ini kita dapat melihat bagaimana berjalannya aplikasi ini.

Tahapan :

* Klik kanan pada file main.cpp, lalu pilih build
* Maka akan muncul seperti pada gambar dibawah ini pada IDE Codeblock
* Lalu klik icon debug pada bagian atas aplikasi codeblock



* Maka aplikasi akan berjalan seperti gambar dibawah ini



* 1. **Kesimpulan**

Kesimpulan akhir pada penelitian ini adalah. Dengan OpenGLES kita dapat membuat sebuah animasi dan aplikasi 3D (dimensi) yang sangat besar kemugkinan untuk dapat dikembangkan menjadi aplikasi apa saja, baik itu dalam Game, maupun aplikasi desain lainnya. Pada penelitian ini memang masih banyak kekurangan, baik itu dari segi bentuk, warna, tekstur, maupun animasi. Tetapi itu bukan dikarenakan karean keterbatasan dari OpenGLES itu sendiri, melainkan keterbatasan kemampuan peneliti (researcher). Saya pikir dengan OpenGLES ini dapat mengembangkan game yang lebih kompleks lagi

* 1. **Ide Pengembangan**

Saya memiliki ide sekaligus impian untuk membuat sebuah game bebasis Augmented Reality. Mungkin ini bukan suatu ide yang baru, tapi saya ingin membuat sebuah game yang real, maksudnya real adalah pemain dari game tersebut memang menggunakan seluruh bagian tubuhnya untuk memainkan game tersebut. Pemain dapat langsung berinteraksi dengan game secara langsung