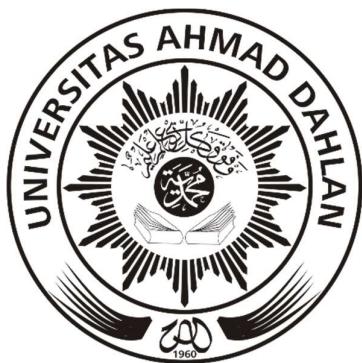


LAPORAN AKHIR PRAKTIKUM
GRAFIKA KOMPUTER



DISUSUN OLEH
ALDI TULUS P (2200018097)
RABU 12.15 - C

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN

JULI 2024

**LEMBAR PENGESAHAN ASISTEN
RESPONSI
LAPORAN AKHIR PRAKTIKUM GRAFIKA KOMPUTER**

Disusun oleh:

ALDI TULUS P

2200018097

**Laboratorium Informatika
Program Studi S1 Informatika
Fakultas Teknologi Industri
Universitas Ahmad Dahlan**

Telah disetujui oleh:

PJ Asisten

[ttd]

**[Nama Lengkap PJ Asisten]
[NIM Asisten]**

DAFTAR ISI

LEMBAR PENGESAHAN ASISTEN	i
DAFTAR ISI	ii
DAFTAR GAMBAR.....	iv
KATA PENGANTAR.....	vii
BAB I PENGANTAR OPENGL.....	1
1.1 PRETEST 1.....	1
1.2 LANGKAH PRAKTIKUM 1	3
1.3 POSTEST 1.....	5
BAB II ALGORITMA GARIS	7
2.1 PRETEST 2.....	7
2.2 LANGKAH PRAKTIKUM 2	8
2.1 POSTEST 2.....	11
BAB III INTERPOLASI DAN KURVA	13
3.1 PRETEST 3.....	13
3.2 LANGKAH PRAKTIKUM 3	14
3.3 POSTEST 3.....	16
BAB IV TRANSFORMASI 2D DAN 3D	18
4.1 PRETEST 4.....	18
4.2 LANGKAH PRAKTIKUM 4	19
4.3 POSTEST 4.....	21
BAB V PROYEKSI 3D.....	23

5.1	PRETEST 5.....	23
5.2	LANGKAH PRAKTIKUM 5.....	24
5.3	POSTEST 5.....	25
	BAB VI REPRESENTASI OBYEK 3D	27
6.1	PRETEST 6.....	27
6.2	LANGKAH PRAKTIKUM 6.....	28
6.3	POSTEST 6.....	31
	BAB VII KURVA SPLINE.....	35
7.1	PRETEST 7.....	35
7.2	LANGKAH PRAKTIKUM 7.....	36
7.3	POSTEST 7.....	38
	BAB VIII TEKNIK REPRESENTASI PERMUKAAN	40
8.1	PRETEST 8.....	40
8.2	LANGKAH PRAKTIKUM 8	40
8.3	POSTEST 8.....	43
	BAB IX TEKNIK PEMODELAN OBYEK 3D	45
9.1	PRETEST 9.....	45
9.2	LANGKAH PRAKTIKUM 9	45
9.3	PRETEST.....	48
	BAB X TEKNIK SUBDIVISI	50
10.1	PRETEST 10.....	50
10.2	LANGKAH PRAKTIKUM 10	50
10.3	POSTEST 10.....	54
	DAFTAR PUSTAKAN.....	56

DAFTAR GAMBAR

Gambar 1. 1 Open GL Pipeline	1
Gambar 1. 2 Fungsi Drawobject LP1	3
Gambar 1. 3 Fungsi display LP1	3
Gambar 1. 4 Fungsi init dan reshape LP1	4
Gambar 1. 5 Fungsi keyboard LP1	4
Gambar 1. 6 Output LP 1	5
Gambar 1. 7 Fungsi Drawobject POST1	5
Gambar 1. 8 Output 1 POST1	6
Gambar 2. 1 Fungsi Algortima DDA X LP1	8
Gambar 2. 2 Fungsi Algortima DDA Y LP1	8
Gambar 2. 3 Fungsi drawObject LP2	9
Gambar 2. 4 Fungsi Display LP2	9
Gambar 2. 5 Fungsi Keyboard LP2	10
Gambar 2. 6 Output LP2	10
Gambar 2. 7 Fungsi drawObject POST2	11
Gambar 2. 8 Output POST2	12
Gambar 3. 1 Fungsi drawInterpolation LP3	14
Gambar 3. 2 Fungsi drawObject LP3	14
Gambar 3. 3 Fungsi display LP3	15
Gambar 3. 4 Fungsi Keyboard LP3	15
Gambar 3. 5 Output LP3	16
Gambar 3. 6 drawObject POST3	16
Gambar 3. 7 Output POST3	17
Gambar 4. 1 Fungsi drawObject LP4	19
Gambar 4. 2 Fungsi display LP4	19
Gambar 4. 3 Fungsi Keyboard LP4	20
Gambar 4. 4 Fungsi keyboard1 LP4	20
Gambar 4. 5 Output LP4	21
Gambar 4. 6 Fungsi drawObject POST4	21
Gambar 4. 7 Output POST4	22

Gambar 6. 1 Fungsi drawObject LP6	28
Gambar 6. 2 Fungsi display LP6	28
Gambar 6. 3 Fungsi keyboard LP6.....	29
Gambar 6. 4 output1 LP6	29
Gambar 6. 5 output2 LP6	30
Gambar 6. 6 output3 LP6	30
Gambar 6. 7 Fungsi drawObject POST6.....	31
Gambar 6. 8 output1 POST6	33
Gambar 6. 9 output2 POST6	33
Gambar 6. 10 output3 POST6	34
Gambar 6. 11 output4 POST6	34
Gambar 7. 1 Fungsi drawObject LP7	36
Gambar 7. 2 Fungsi display LP7	36
Gambar 7. 3 Fungsi keyboard LP7	37
Gambar 7. 4 Output LP7	37
Gambar 7. 5 Fungsi drawObject POST7	38
Gambar 7. 6 output POST7	39
Gambar 8. 1 Fungsi drawObject LP8	40
Gambar 8. 2 Fungsi display LP8	41
Gambar 8. 3 Fungsi init LP8	41
Gambar 8. 4 Fungsi keyboard LP8	42
Gambar 8. 5 Fungsi output LP8	42
Gambar 8. 6 Fungsi init POST8	43
Gambar 8. 7 output POST8	44
Gambar 9. 1 Variable inisiasi Metaball dan cahaya	45
Gambar 9. 2 Fungsi drawObject LP9	46
Gambar 9. 3 Fungsi display LP9	46
Gambar 9. 4 Fungsi init LP9	47
Gambar 9. 5 Fungsi keyboard LP9	47
Gambar 9. 6 Output LP9	48
Gambar 9. 7 Variable inisiasi Metaball dan cahaya POST9	48

Gambar 9. 8 Output POST9	49
Gambar 10. 1 Variable inisiasi warna dan jumlah subdiv.....	50
Gambar 10. 2 Fungsi drawObject LP10	51
Gambar 10. 3 fungsi display LP10	51
Gambar 10. 4 Fungsi init LP10	52
Gambar 10. 5 Fungsi Keyboard LP10.....	52
Gambar 10. 6 Output1 LP10	53
Gambar 10. 7 Output2 LP10	53
Gambar 10. 8 Output3 LP10	54
Gambar 10. 9 Fungsi drawObject POST10	54
Gambar 10. 10 Output POST10	55

KATA PENGANTAR

Assalamu'alaikum Wr. Wb

Segala puji kami panjatkan kepada Allah SWT yang telah memberikan kami nikmat sehingga kami dapat menyelesaikan makalahini dengan tepat waktu. Tanpa rahmat-Nya, tentu kami tidak akan mampu menyelesaikan laporan ini dengan baik. Shalawat serta salam semoga tercurah kepada Nabi Muhammad SAW yang selalu kita nantikan syafaatnya di akhirat kelak.

Kami selaku penulis menyampaikan rasa syukur kepada Allah SWT atas nikmat kesehatan, baik jasmani maupun rohani, yang telah diberikan, sehingga kami mampu menyelesaikan Laporan mata praktikum Grafika Komputer.

Kami menyadari bahwa Laporan ini masih jauh dari sempurna dan masih terdapat banyak kekurangan serta kesalahan di dalamnya. Oleh karena itu, kami sangat mengharapkan kritik dan saran dari para pembaca untuk perbaikan laporan ini ke depannya. Kami juga mohon maaf sebesar-besarnya jika terdapat banyak kesalahan dalam laporan ini.

Semoga laporan ini dapat memberikan manfaat. Terima kasih. Wassalamu'alaikum Wr. Wb.

Yogyakarta,

BAB I

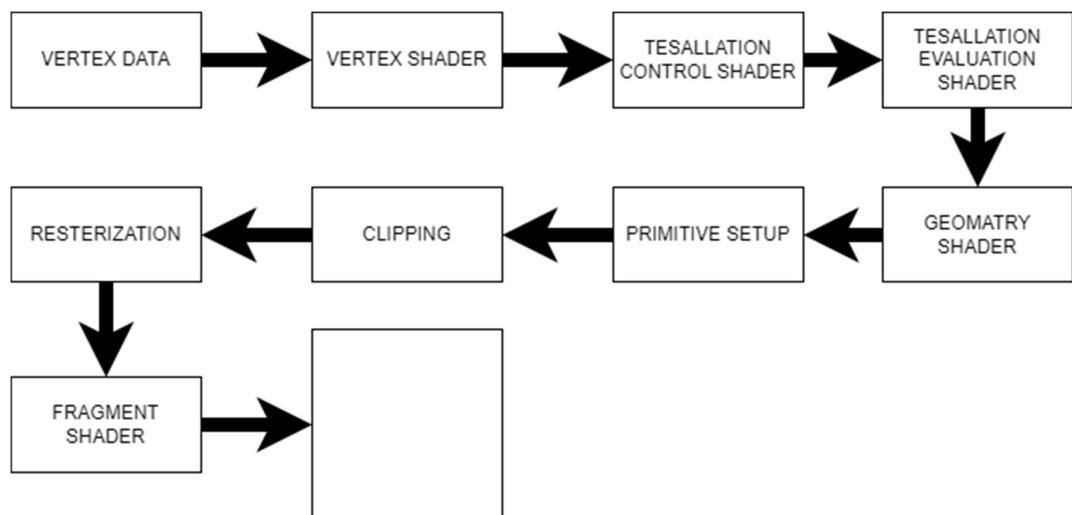
PENGANTAR OPENGL

1.1 PRETEST 1

1. Apa saja yang bisa dilakukan dengan library OpenGL?

(Karisma, 2013) OpenGL menyediakan set perintah untuk menggambar dan semua penggambaran yang lebih tinggi tingkatnya harus dilakukan dengan mengambil fungsi dasar dari perintah ini. Maka dari itu dapat dibuat library itu sendiri di atas program OpenGL yang mempermudah pemrograman lebih lanjut. Fungsi asli dari OpenGL sendiri selalu diawali dengan gl yang terdapat pada library opengl32.dll dan file header gl.h.

2. Gambarkan kemudian jelaskan tahapan OpenGL pipeline!



Gambar 1. 1 Open GL Pipeline

Penjelasan singkat tentang setiap tahap dalam pipeline :

- a. Vertex Data: Tahap ini melibatkan pengiriman data vertex, seperti posisi, warna, dan tekstur, ke unit pemrosesan grafik.
- b. Vertex Shader: Tahap ini menjalankan program yang ditentukan oleh pengguna, yang mengubah posisi dan atribut vertex. Ini dapat digunakan untuk melakukan transformasi objek, menerapkan pencahayaan, atau menerapkan efek khusus lainnya.

- c. Tessellation Control Shader: Tahap ini opsional dan digunakan dalam tahap tessellation. Ini mengontrol seberapa banyak detail geometri yang akan dihasilkan selama proses tessellation.
- d. Tessellation Evaluation Shader: Tahap ini opsional dan digunakan dalam tahap tessellation. Ini menghasilkan detail geometri tambahan berdasarkan konfigurasi dan posisi vertex yang telah ditentukan sebelumnya.
- e. Geometry Shader: Tahap ini opsional dan digunakan untuk menghasilkan primitif geometri baru dari primitif input. Misalnya, satu titik dapat digunakan untuk menghasilkan beberapa titik, atau satu segitiga dapat digunakan untuk menghasilkan beberapa segitiga.
- f. Primitive Setup: Tahap ini mengubah data vertex yang telah ditransformasi dan diubah menjadi primitif yang sesuai, seperti titik, garis, atau segitiga.
- g. Clipping: Tahap ini memotong primitif yang ada di luar jangkauan penglihatan, berdasarkan pengaturan viewport dan frustum pandangan.
- h. Rasterization: Tahap ini mengubah primitif yang telah dipotong menjadi piksel-piksel di layar, menghasilkan fragmen-fragment yang akan dikirim ke fragment shader.
- i. Fragment Shader: Tahap ini menjalankan program yang ditentukan oleh pengguna, yang menghasilkan warna dan atribut lain untuk setiap fragmen. Ini digunakan untuk menerapkan pencahayaan, tekstur, dan efek khusus lainnya pada fragmen.

1.2 LANGKAH PRAKTIKUM 1

```

Project1 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug main.cpp
Project1 main.cpp
17 // fungsi untuk menggambar objek kubus
18 void drawObject()
19 {
20     // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
21     // Fungsi agar objek tidak terpengaruh atau mempengaruhi objek lain
22     // misalnya transformasi, alattransformasi dan sebagainya
23     // gunakan glPushMatrix();
24
25     // operasi transformasi rotasi objek ke arah kanan-kiri
26     glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
27
28     glPushMatrix();
29
30     // operasi transformasi rotasi objek ke arah atas-bawah
31     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
32
33     // set warna objek ke warna hijau (0.0f, 1.0f, 0.0f)
34     glColor3f(0.0f, 1.0f, 0.0f);
35
36     // bilah menggambar objek harus diawali glBegin(tipe objek) dan diakhiri dengan glEnd()
37     // kecuali menggunakan fungsi yang sudah ada di GLUT-OpenGL seperti dibawah ini
38     glutSolidSphere(1.0, 20, 20);
39
40     // glutSolidSphere(1.0, 20, 20);
41     //     glVertex3f(-0.5f, -0.5f, 0.0f); // v1
42     //     glVertex3f(0.5f, -0.5f, 0.0f); // v1
43     //     glVertex3f(0.5f, 0.5f, 0.0f); // v1
44     //     glEnd();
45
46     glPopMatrix();
47
48 }
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69

```

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation - Compilation Time: 0,59s
Line: 39 Col: 34 Sel: 0 Lines: 183 Length: 5998 Insert Done parsing in 0,016 seconds

Gambar 1. 2 Fungsi Drawobject LP1

Fungsi ini digunakan untuk menggambar objek yang akan ditampilkan nanti. Line 39 di gunakan untuk membuat objek berbentuk Bulat/bola

```

Project1 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug main.cpp
Project1 main.cpp
42
43     glPopMatrix();
44 }
45
46 // taruh semua objek yang akan digambar di fungsi display()
47 void display()
48 {
49     // bersihkan dan reset layar dan buffer
50     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
51     glLoadIdentity();
52
53     // posisikan kamera pandang
54     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
55     gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
56
57     // panggil fungsi untuk menggambar objek
58     drawObject();
59
60     // tampilkan objek ke layar
61     // gunakan glFlush() bila memakai single buffer
62     // gunakan glutSwapBuffers() bila memakai double buffer
63     glutSwapBuffers();
64 }
65
66 //inisialisasikan variabel, pencitraaan, tekstur dan pengaturan kamera pandang di fungsi init()
67 void init(void)
68 {
69     //inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation - Compilation Time: 0,36s
Line: 19 Col: 16 Sel: 10 Lines: 178 Length: 5828 Insert Done parsing in 2,125 seconds

Gambar 1. 3 Fungsi display LP1

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 58 untuk memanggil fungsi drawObject.

```

Project1 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glut | (globals)]
Project Classes Debug main.cpp
Project1 main.cpp
63     glutSwapBuffers();
64 }
65
66 //inisialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
67 void init(void)
68 {
69     //inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
70     glClearColor(1.0, 1.0, 1.0, 0.0); //mengaktifkan depth buffer
71     glEnable(GL_DEPTH_TEST);
72     glMatrixMode(GL_PROJECTION);
73     glLoadIdentity();
74     gluPerspective(45.0, 1.0, 1.0, 100.0); //set proyeksi ke perspektif
75     glMatrixMode(GL_MODELVIEW);
76     glLoadIdentity();
77     //inisialisasi kamera pandang
78     gluLookAt(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
79 }
80
81 //fungsi ini digunakan bila layar akan diresize (default)
82 void reshape(int w, int h)
83 {
84     glViewport(0, 0, (GLsizei)w, (GLsizei)h);
85     glMatrixMode(GL_PROJECTION);
86     glLoadIdentity();
87     gluPerspective(45, (GLfloat)w / (GLfloat)h, 1.0, 100.0);
88     glMatrixMode(GL_MODELVIEW);
89 }
90

```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,36s

Line: 19 Col: 16 Sel: 10 Lines: 178 Length: 5828 Insert Done parsing in 2,125 seconds

Gambar 1. 4 Fungsi init dan reshape LP1

Fungsi init di gunakan untuk menginisialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init(). Dan Fungsi reshape digunakan bila layar akan diresize (default)

```

Project1 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glut | (globals)]
Project Classes Debug main.cpp
Project1 main.cpp
83 void keyboard(int key, int x, int y)
84 {
85     float fraction = 0.1f;
86
87     switch (key)
88     {
89         //masukan perintah distini bila tombol kiri ditekan
90         case GLUT_KEY_LEFT:
91             //masukan perintah rotasi objek ke kiri sebanyak 1 derajat
92             objectanglex -= 1.0f;
93             glutPostRedisplay(); //update objek
94             break;
95         //masukan perintah distini bila tombol kanan ditekan
96         case GLUT_KEY_RIGHT:
97             //masukan perintah rotasi objek ke kanan sebanyak 1 derajat
98             objectanglex += 1.0f;
99             glutPostRedisplay(); //update objek
100            break;
101        //masukan perintah distini bila tombol bawah ditekan
102        case GLUT_KEY_DOWN:
103             //masukan perintah rotasi objek ke bawah sebanyak 1 derajat
104             objectanglez -= 1.0f;
105             glutPostRedisplay(); //update objek
106             break;
107         //masukan perintah distini bila tombol atas ditekan
108         case GLUT_KEY_UP:
109             //masukan perintah rotasi objek ke atas sebanyak 1 derajat
110             objectanglez += 1.0f;
111             glutPostRedisplay(); //update objek
112             break;
113         //zoom out
114         case GLUT_KEY_PAGE_UP:
115             //masukan perintah distini bila tombol PgUp ditekan
116             post -= rotX * fraction;
117             post += rotZ * fraction;
118             glutPostRedisplay(); //update objek
119             break;
120         //zoom in
121         case GLUT_KEY_PAGE_DOWN:
122             //masukan perintah distini bila tombol PgDn ditekan
123             post -= rotX * fraction;
124             post += rotZ * fraction;
125             glutPostRedisplay(); //update objek
126             break;
127     }
128 }
129

```

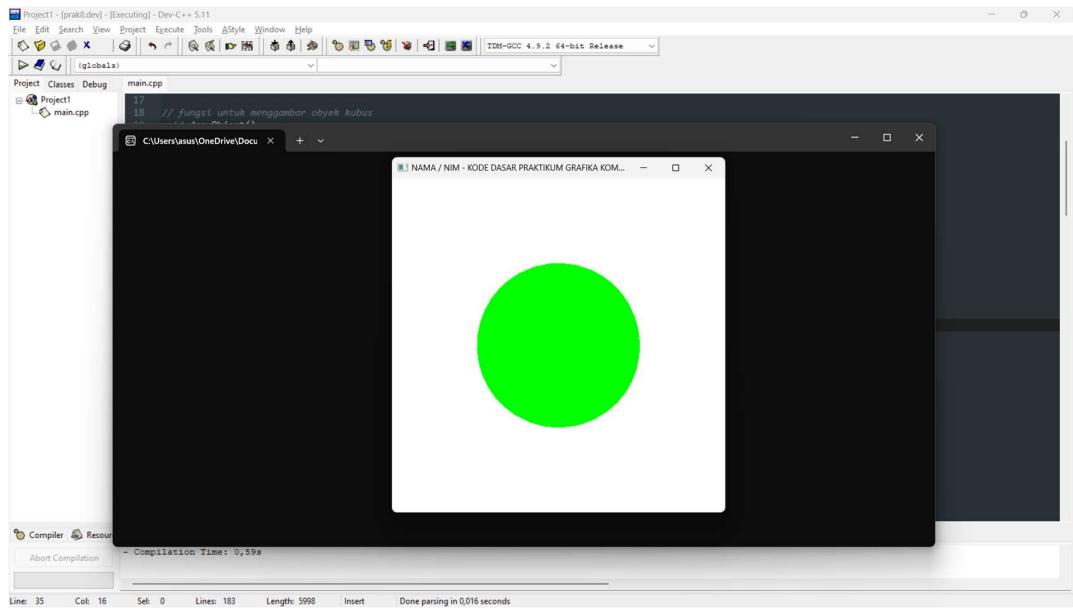
Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,36s

Line: 81 Col: 59 Sel: 44 Lines: 178 Length: 5828 Insert Done parsing in 2,125 seconds

Gambar 1. 5 Fungsi keyboard LP1

Fungsi ini di gunakan untuk mengatur pergerekian objek dengan menggunakan keyboard.



Gambar 1. 6 Output LP 1

1.3 POSTEST 1

1. Membuat Rumah

Untuk Kodingan Dasarnya sama dengan langkah praktikum

```

Project1 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[ ] (globals)
Project1 main.cpp Makefile.win
13
14 float objectAngleX = 0.0f; // sudut transformasi objek di sumbu X
15 float objectAngleY = 0.0f; // sudut transformasi objek di sumbu Y
16 float objectAngleZ = 0.0f; // sudut transformasi objek di sumbu Z
17
18 // fungsi untuk menggambar objek kubus
19 void drawObject()
20 {
21     // objek bisa dimulukkan diantara glPushMatrix() dan glPopMatrix()
22     // funginya agar objek tidak terpengaruh atau mempengaruhi objek lain
23     // yang dibuat sebelum atau setelah ulitransformasi dan sebagainya
24     glPushMatrix();
25
26     // operasi transformasi rotasi objek ke arah kanan-kiri
27     glRotatef(objectAngleX, 0.0f, 1.0f, 0.0f);
28
29     glPushMatrix();
30
31     // operasi transformasi rotasi objek ke arah atas-bawah
32     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
33
34     glBegin(GL_QUADS);
35     glColor3f(1.0f, 1.0f, 0.0f); // warna kuning untuk atap
36     glVertex3f(0.0f, 1.0f, 0.0f); // v1
37     glVertex3f(1.0f, 1.0f, 0.0f); // v2
38     glVertex3f(1.0f, 0.0f, 0.0f); // v3
39     glVertex3f(0.0f, 0.0f, 0.0f); // v4
40     glEnd();
41
42     glBegin(GL_QUADS);
43     glColor3f(0.5f, 0.0f, 0.0f); // warna merah untuk badan rumah
44     glVertex3f(0.5f, 0.0f, 0.0f); // v1
45     glVertex3f(0.5f, 0.5f, 0.0f); // v2
46     glVertex3f(0.5f, 1.0f, 0.0f); // v3
47     glVertex3f(0.5f, 1.0f, 0.5f); // v4
48     glEnd();
49
50     glPopMatrix();
51 }
52
53

```

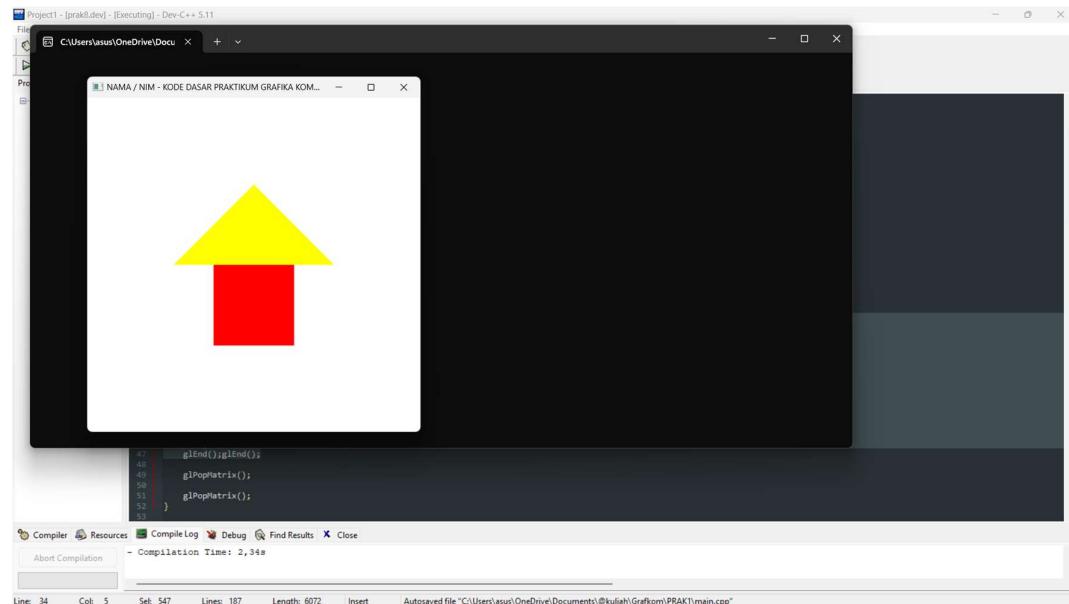
The screenshot shows the Dev-C++ IDE interface. The main window displays a C++ code editor with the file 'main.cpp' open. The code contains a function 'drawObject' which uses OpenGL commands to draw a house. The house has a yellow roof and a red body. Below the code editor is a terminal window showing the compilation output. The terminal window shows 'Compilation Time: 2,38s'.

Gambar 1. 7 Fungsi Drawobject POST1

Hanya merubah pada bagian drawobject, buat program seperti ini

```
glBegin(GL_TRIANGLES);
```

```
    glColor3f(1.0f, 1.0f, 0.0f); // Warna kuning untuk atap  
    glVertex3f(0.0f, 1.0f, 0.0f); // v1  
    glVertex3f(-1.0f, 0.0f, 0.0f); // v2  
    glVertex3f(1.0f, 0.0f, 0.0f); // v3  
  
    glEnd();  
  
    glBegin(GL_QUADS);  
  
        glColor3f(1.0f, 0.0f, 0.0f); // Warna merah untuk badan rumah  
        glVertex3f(-0.5f, 0.0f, 0.0f); // v1  
        glVertex3f(0.5f, 0.0f, 0.0f); // v2  
        glVertex3f(0.5f, -1.0f, 0.0f); // v3  
        glVertex3f(-0.5f, -1.0f, 0.0f); // v4  
  
    glEnd();glEnd();
```



Gambar 1. 8 Output 1 POST1

BAB II

ALGORITMA GARIS

2.1 PRETEST 2

1. Jelaskan tahapan pembangkitan garis dengan algoritma DDA!

Tahapan pembangkitan garis dengan algoritma DDA melibatkan langkah-langkah berikut:

- a. Inisialisasi: Tentukan titik awal (x_0, y_0) dan titik akhir (x_1, y_1) dari garis.
 - b. Hitung Selisih: Tentukan perubahan dalam koordinat x (Δx) dan y (Δy).
 - c. Hitung Langkah: Tentukan jumlah langkah yang dibutuhkan, yaitu nilai maksimum antara $|\Delta x|$ dan $|\Delta y|$.
 - d. Inkrementasi: Hitung nilai inkrementasi untuk x dan y pada setiap langkah, yaitu $\Delta x/steps$ dan $\Delta y/steps$.
 - e. Iterasi: Mulai dari titik awal, tambahkan nilai inkrementasi ke koordinat x dan y pada setiap langkah, dan plot titik-titik tersebut hingga mencapai titik akhir.
2. Jelaskan tahapan pembangkitan garis dengan algoritma Bresenham!
- Tahapan pembangkitan garis dengan algoritma Bresenham melibatkan langkah-langkah berikut:
- a. Inisialisasi: Tentukan titik awal (x_0, y_0) dan titik akhir (x_1, y_1) dari garis.
 - b. Hitung Selisih: Tentukan perubahan dalam koordinat x (Δx) dan y (Δy).
 - c. Tentukan Parameter Keputusan: Inisialisasi parameter keputusan $p_0 = 2\Delta y - \Delta x$.
 - d. Iterasi: Mulai dari titik awal, untuk setiap langkah x, lakukan:
 - Jika $p < 0$, titik berikutnya adalah $(x+1, y)$, dan perbarui parameter keputusan $p = p + 2\Delta y$.
 - Jika $p \geq 0$, titik berikutnya adalah $(x+1, y+1)$, dan perbarui parameter keputusan $p = p + 2\Delta y - 2\Delta x$.
 - e. Plot: Plot titik-titik yang dihasilkan hingga mencapai titik akhir.

2.2 LANGKAH PRAKTIKUM 2

The screenshot shows the Dev-C++ IDE interface with the main.cpp file open. The code implements the DDA algorithm for drawing a line based on its gradient relative to the X-axis. It uses OpenGL functions like glBegin(GL_POINTS) and glVertex3f to draw points along the line. The code is annotated with comments explaining each step: calculating gradients, starting points, and then iterating through the line segments to calculate intermediate points (px = px + 1; py = py + m;).

```

Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
D Project1 main.cpp
31  };
32
33 // fungsi untuk menggambar garis dengan algoritma DDA bila terhadap X
34 void lineDDAX(Vec3 point1, Vec3 point2)
35 {
36     // hitung gradient garis m
37     int dY = point2.Y - point1.Y;
38     int dX = point2.X - point1.X;
39     float m = (float)dY / dX;
40     float im = 1.0f/m;
41
42     // mulai menggambar titik-titik
43     glBegin(GL_POINTS);
44     // koordinat titik awal
45     glVertex3f(point1.X, point1.Y, point1.Z);
46     float pX = point1.X, pY = point1.Y, pZ = point1.Z;
47
48     // kenaikan terhadap X
49     for (int i = point1.X; i < point2.X; i++)
50     {
51         pX = pX + 1; // Xn+1 = Xn + 1
52         pY = pY + m; // Yn+1 = Yn + m
53         glVertex3f(pX, pY, pZ);
54     }
55     // koordinat titik akhir
56     glVertex3f(point2.X, point2.Y, point2.Z);
57     glEnd();
58 }
59
60 // fungsi untuk menggambar garis dengan algoritma DDA bila terhadap Y.

```

Gambar 2. 1 Fungsi Algortima DDA X LP1

fungsi untuk menggambar garis dengan algoritma DDA bila terhadap X

The screenshot shows the Dev-C++ IDE interface with the main.cpp file open. The code implements the DDA algorithm for drawing a line based on its gradient relative to the Y-axis. It uses OpenGL functions like glBegin(GL_POINTS) and glVertex3f to draw points along the line. The code is annotated with comments explaining each step: calculating gradients, starting points, and then iterating through the line segments to calculate intermediate points (px = px + im; py = py + 1;).

```

Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
D Project1 main.cpp
58 }
59
60 // fungsi untuk menggambar garis dengan algoritma DDA bila terhadap Y
61 void lineDDAY(Vec3 point1, Vec3 point2)
62 {
63     // hitung gradient garis m
64     int dY = point2.Y - point1.Y;
65     int dX = point2.X - point1.X;
66     float m = (float)dY / dX;
67     float im = 1.0f/m;
68
69     // mulai menggambar titik-titik
70     glBegin(GL_POINTS);
71     // koordinat titik awal
72     glVertex3f(point1.X, point1.Y, point1.Z);
73     float pX = point1.X, pY = point1.Y, pZ = point1.Z;
74
75     // kenaikan terhadap Y
76     for (int i = point1.Y; i < point2.Y; i++)
77     {
78         pX = pX + im; // Xn+1 = Xn + 1/m
79         pY = pY + 1; // Yn+1 = Yn + 1
80         glVertex3f(pX, pY, pZ);
81     }
82     // koordinat titik akhir
83     glVertex3f(point2.X, point2.Y, point2.Z);
84     glEnd();
85 }
86
87 // fungsi untuk menggambar garis dengan algoritma DDA

```

Gambar 2. 2 Fungsi Algortima DDA Y LP1

fungsi untuk menggambar garis dengan algoritma DDA bila terhadap Y.

```

Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1] (globals)
Project Classes Debug main.cpp
Project1 main.cpp
109 // fungsi untuk menggambar objek
110 void drawObject()
111 {
112     glPushMatrix();
113     // operasi transformasi rotasi objek ke arah kanan-kiri
114     glRotatef(objectAngle, 0.0f, 1.0f, 0.0f);
115     glTranslatef(0.0f, 0.0f, 0.0f);
116     // operasi transformasi rotasi objek ke arah atas-bawah
117     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
118     // set warna objek ke warna hijau (0.0f, 1.0f, 0.0f)
119     glColor3f(0.0f, 1.0f, 0.0f);
120     // gambar sumbu
121     glBegin(GL_LINES);
122     // V
123     glVertex3f(0.0f, 0.0f, 0.0f);
124     glVertex3f(0.0f, 1.0f, 0.0f);
125     // kuadran 1
126     Vec3 point1 = Vec3( 50.0f, 0.0f, 0.0f);
127     Vec3 point2 = Vec3(-50.0f, 0.0f, 0.0f);
128     lineObj(point1, point2);
129     // kuadran 2
130     Vec3 point1 = Vec3(-50.0f, 0.0f, 0.0f);
131     Vec3 point2 = Vec3( 50.0f, 0.0f, 0.0f);
132     lineObj(point1, point2);
133     // kuadran 3
134     Vec3 point1 = Vec3(-50.0f, -50.0f, 0.0f);
135     Vec3 point2 = Vec3( 50.0f, -50.0f, 0.0f);
136     lineObj(point1, point2);
137     // kuadran 4
138     Vec3 point1 = Vec3( 50.0f, 0.0f, 0.0f);
139     Vec3 point2 = Vec3(-50.0f, 0.0f, 0.0f);
140     lineObj(point1, point2);
141     // garis diagonal
142     Vec3 point1 = Vec3(-50.0f, 10.0f, 0.0f);
143     Vec3 point2 = Vec3(-50.0f, 50.0f, 0.0f);
144     lineObj(point1, point2);
145     gPopMatrix();
146     gPopMatrix();
147 }

```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 3,16s

Line: 60 Col: 4 Sel: 66 Lines: 284 Length: 8469 Insert Done parsing in 0,109 seconds

Gambar 2. 3 Fungsi *drawObject* LP2

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 126 - 144 di gunakan untuk membuat objek berbentuk U.

```

Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1] (globals)
Project Classes Debug main.cpp
Project1 main.cpp
145     glPopMatrix();
146     glPopMatrix();
147 }
148 }
149 // taruh semua objek yang akan digambar di fungsi display()
150 void display()
151 {
152     // bersihkan dan reset layar dan buffer
153     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
154     glLoadIdentity();
155
156     // posisikan kamera pandang
157     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
158     gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
159
160     // panggil fungsi untuk menggambar objek
161     drawObject();
162
163     // tampilkan objek ke layar
164     // gunakan glFlush() bila memakai single buffer
165     // gunakan glutSwapBuffers() bila memakai double buffer
166     glutSwapBuffers();
167 }
168 }
169
170 // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
171 void init(void)
172 {
173     // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
174     glClearColor(1.0, 1.0, 1.0, 0.0);
175 }

```

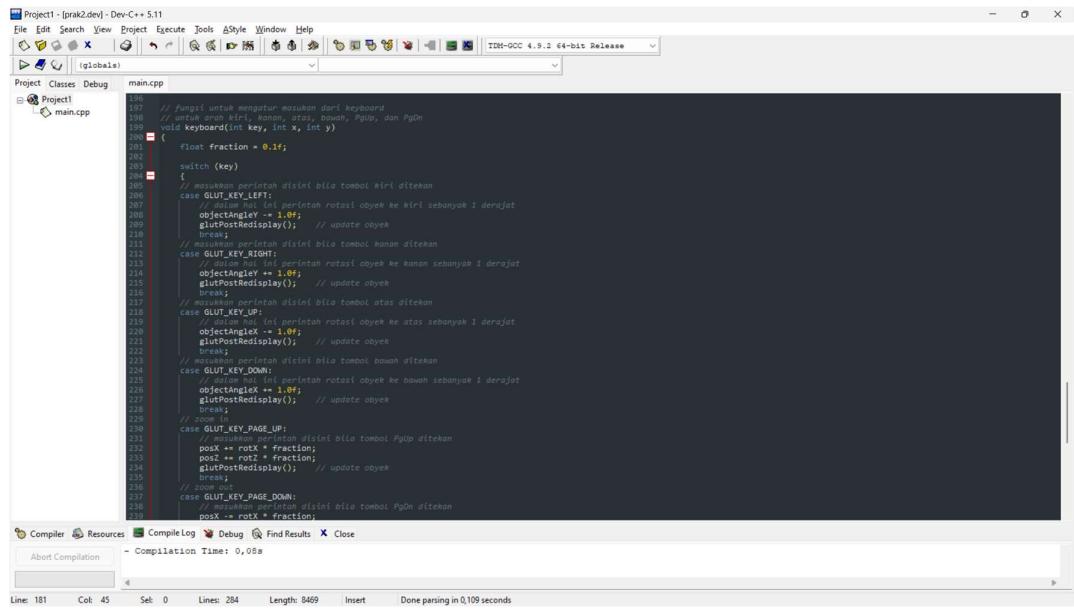
Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,00s

Line: 181 Col: 45 Sel: 0 Lines: 284 Length: 8469 Insert Done parsing in 0,109 seconds

Gambar 2. 4 Fungsi *Display* LP2

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 162 untuk memanggil fungsi *drawObject*.



```

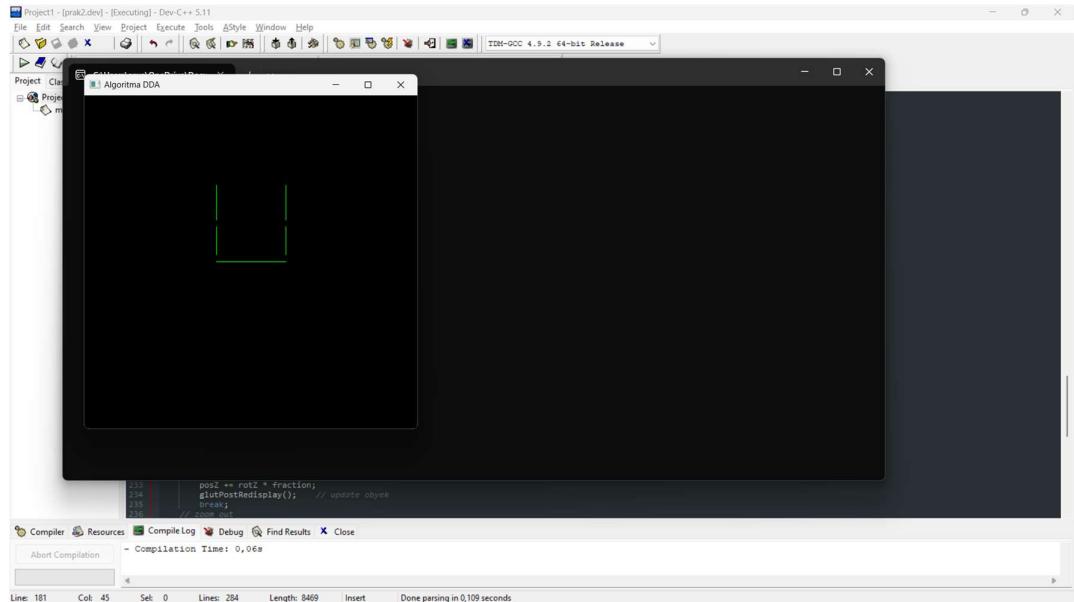
Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
| X | (globals)
Project Classes Debug main.cpp
Project1 main.cpp
196 // fungsi untuk mengatur masukan dari keyboard
197 // pada antara tombol PgUp, PgDn, Atas, Bawah, PgUp, dan PgDn
198 void keyboard(int key, int x, int y)
199 {
200     float fraction = 0.1f;
201
202     switch (key)
203     {
204         // mosukkan perintah distri bila tombol kiri ditekan
205         case GLUT_KEY_LEFT:
206             // dalam hal ini perintah rotasi objek ke kiri sebanyak 1 derajat
207             objectAngleX -= 1.0f;
208             glutPostRedisplay(); // update objek
209             break;
210         // mosukkan perintah distri bila tombol kanan ditekan
211         case GLUT_KEY_RIGHT:
212             // dalam hal ini perintah rotasi objek ke kanan sebanyak 1 derajat
213             objectAngleX += 1.0f;
214             glutPostRedisplay(); // update objek
215             break;
216         // mosukkan perintah distri bila tombol atas ditekan
217         case GLUT_KEY_UP:
218             // dalam hal ini perintah rotasi objek ke atas sebanyak 1 derajat
219             objectAngleY -= 1.0f;
220             glutPostRedisplay(); // update objek
221             break;
222         // mosukkan perintah distri bila tombol bawah ditekan
223         case GLUT_KEY_DOWN:
224             // dalam hal ini perintah rotasi objek ke bawah sebanyak 1 derajat
225             objectAngleY += 1.0f;
226             glutPostRedisplay(); // update objek
227             break;
228         // zoom in
229         case GLUT_KEY_PAGE_UP:
230             posZ += rotX * fraction;
231             posZ += rotY * fraction;
232             posZ += rotZ * fraction;
233             glutPostRedisplay(); // update objek
234             break;
235         // zoom out
236         case GLUT_KEY_PAGE_DOWN:
237             posZ -= rotX * fraction;
238             posZ -= rotY * fraction;
239             posZ -= rotZ * fraction;
240             glutPostRedisplay(); // update objek
241             break;
242     }
243 }

```

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation - Compilation Time: 0,08s
Line: 181 Col: 45 Sel: 0 Lines: 284 Length: 8469 Insert Done parsing in 0,109 seconds

Gambar 2. 5 Fungsi Keyboard LP2

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.

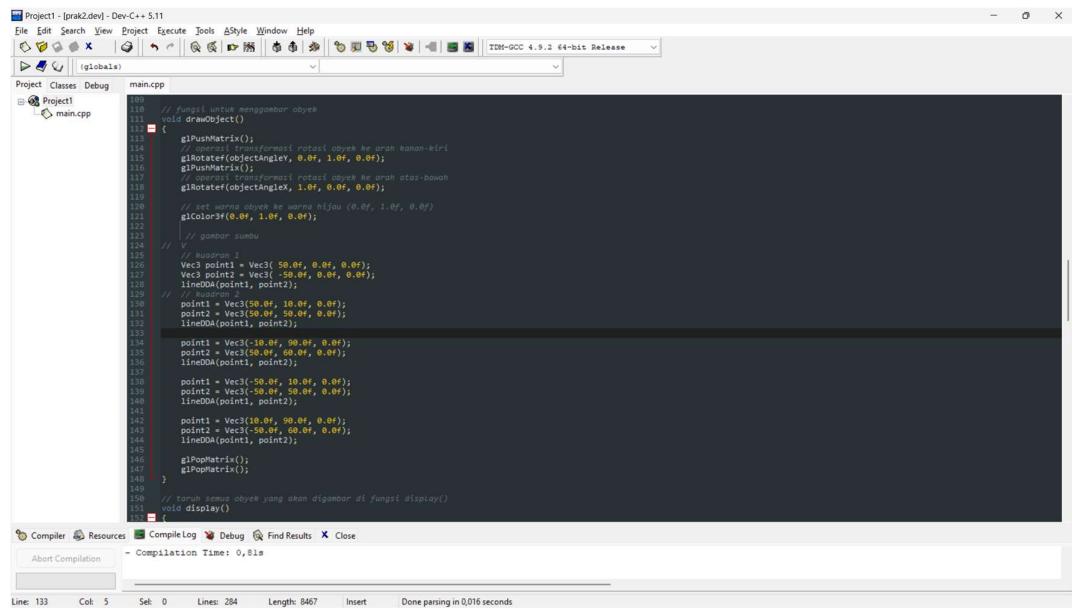


Gambar 2. 6 Output LP2

2.1 POSTEST 2

1. Membuat Rumah

Untuk Kodingan Dasarnya sama dengan langkah praktikum



The screenshot shows the Dev-C++ IDE interface with the main.cpp file open. The code defines a `drawObject` function that performs matrix operations like pushing and popping matrices, rotating around angles, and drawing lines using the `lineDDA` function. It also sets colors and draws a house-like structure with specific points and lines.

```
Project1 - [prak2.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools Style Window Help
(globals)
Project Classes Debug main.cpp
Project1 main.cpp
110 // fungsi untuk menggambar objek
111 void drawObject()
112 {
113     glPushMatrix();
114     glRotatef(transFormasi rotasi objek ke arah kanan-kiri
115     glRotatef(objectAngleX, 0.0f, 1.0f, 0.0f);
116     glPushMatrix();
117     // operasi transformasi rotasi objek ke arah atas-bawah
118     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
119
120     // set warna objek ke warna hijau (0.0f, 1.0f, 0.0f)
121     glColor3f(0.0f, 1.0f, 0.0f);
122
123     // gambar sumbu
124     Vec3 point1 = Vec3( 50.0f, 0.0f, 0.0f);
125     Vec3 point2 = Vec3(-50.0f, 0.0f, 0.0f);
126     lineDDA(point1, point2);
127
128     // gambar bingkai rumah
129     Vec3 point1 = Vec3(50.0f, 10.0f, 0.0f);
130     point1 += Vec3(50.0f, 10.0f, 0.0f);
131     lineDDA(point1, point2);
132
133
134     point1 = Vec3(-10.0f, 90.0f, 0.0f);
135     point2 = Vec3(50.0f, 60.0f, 0.0f);
136     lineDDA(point1, point2);
137
138     point1 = Vec3(-50.0f, 10.0f, 0.0f);
139     point2 = Vec3(50.0f, 10.0f, 0.0f);
140     lineDDA(point1, point2);
141
142     point1 = Vec3(10.0f, 90.0f, 0.0f);
143     point2 = Vec3(-50.0f, 60.0f, 0.0f);
144     lineDDA(point1, point2);
145
146     glPopMatrix();
147     glPopMatrix();
148 }
149
150 // tampil semua objek yang akan digambar di fungsi display()
151 void display()
152 {
153 }
```

Compiler Resources Compiler Log Debug Find Results Close
Line: 133 Col: 5 Sel: 0 Lines: 284 Length: 8467 Insert Done parsing in 0.016 seconds

Gambar 2. 7 Fungsi `drawObject` POST2

Hanya merubah pada bagian `drawobject`, buat program seperti ini

```
Vec3 point1 = Vec3( 50.0f, 0.0f, 0.0f);
```

```
Vec3 point2 = Vec3( -50.0f, 0.0f, 0.0f);
```

```
lineDDA(point1, point2);
```

```
point1 = Vec3(50.0f, 10.0f, 0.0f);
```

```
point2 = Vec3(50.0f, 50.0f, 0.0f);
```

```
lineDDA(point1, point2);
```

```
point1 = Vec3(-10.0f, 90.0f, 0.0f);
```

```
point2 = Vec3(50.0f, 60.0f, 0.0f);
```

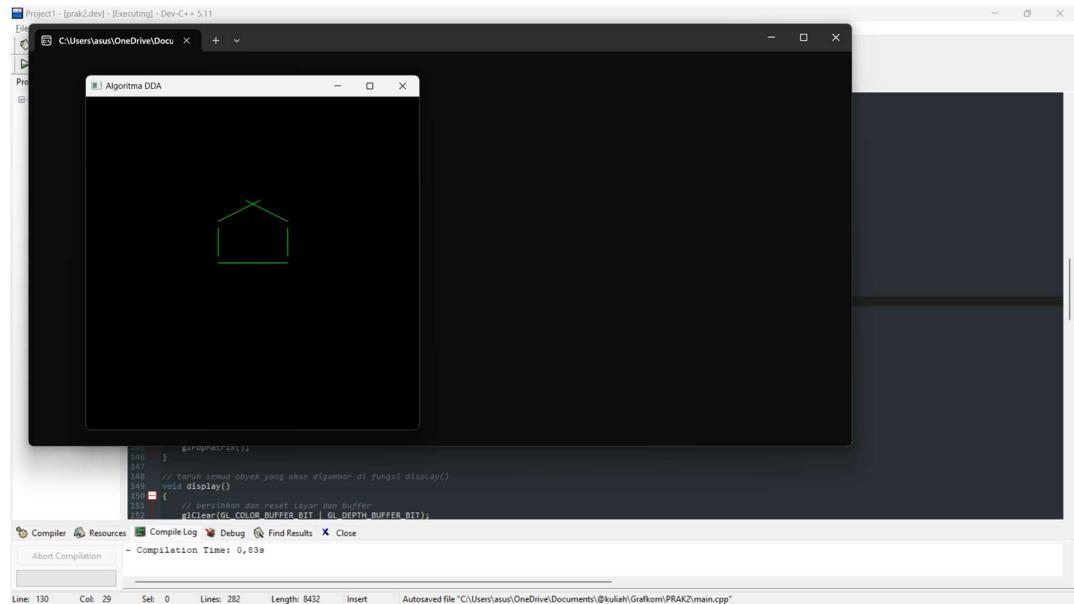
```
lineDDA(point1, point2);
```

```
point1 = Vec3(-50.0f, 10.0f, 0.0f);
```

```
point2 = Vec3(-50.0f, 50.0f, 0.0f);
```

```
lineDDA(point1, point2);
```

```
point1 = Vec3(10.0f, 90.0f, 0.0f);
point2 = Vec3(-50.0f, 60.0f, 0.0f);
lineDDA(point1, point2);
```



Gambar 2. 8 Output POST2

BAB III

INTERPOLASI DAN KURVA

3.1 PRETEST 3

1. Jelaskan tahapan interpolasi linear!

Interpolasi linear melibatkan estimasi nilai di antara dua titik data dengan garis lurus di antara keduanya. Tahapannya meliputi:

- a. Pemahaman Data: Memahami data yang ada, yaitu pasangan titik (x_i, y_i) (x_{-i}, y_{-i}) yang telah diketahui.
- b. Menghitung Gradien: Menghitung gradien atau slope dari garis yang menghubungkan dua titik data.
- c. Interpolasi: Menggunakan persamaan garis lurus untuk memperkirakan nilai y di antara dua titik dengan persamaan $y = y_i + m(x - x_i)$ $y = y_{-i} + m(x - x_{-i})$.

2. Jelaskan tahapan interpolasi kubik!

Interpolasi kubik menggunakan fungsi polinomial kubik untuk menghubungkan titik-titik data yang diketahui. Tahapannya meliputi:

- a. Pemahaman Data: Seperti pada interpolasi linear, memahami data yang ada.
- b. Menghitung Koefisien: Menghitung koefisien dari polinomial kubik yang melewati setiap tiga titik data berturut-turut.
- c. Interpolasi: Membangun polinomial kubik $f(x) = a_1x^3 + b_1x^2 + c_1x + d_1$ jika $f(x) = a_1x^3 + b_1x^2 + c_1x + d_1$ untuk setiap interval $[x_i, x_{i+1}]$ (x_{-i}, x_{-i+1}) $[x_i, x_{i+1}]$.

3.2 LANGKAH PRAKTIKUM 3

```

Project1 - [Prak3.deg] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1 | main.cpp | (globals)]
Project Classes Debug main.cpp
Project1 main.cpp
67 // fungsi garis hasil Interpolasi
68 // n adalah jumlah titik yang dibuat
69 // type n tipe Interpolasi yang digunakan
70 void drawInterpolation(
71     Vec3 point0,
72     Vec3 point1, // point1 adalah titik awal
73     Vec3 point2, // point2 adalah titik akhir
74     Vec3 point3,
75     int n,
76     INTERP_TYPE type1,
77     INTERP_TYPE type2)
78 {
79     float u = 0;
80     float stepU = 1.0f / n; // kenaikan u
81     float stepX = fabs(point2.X - point1.X) / n; // kenaikan x
82     float px = point1.X, py = point1.Y, pz = point1.Z;
83     // mulai menggambar titik
84     glPointSize(5);
85     // mulai menggambar garis
86     if (type1 == INTERP_POINTS)
87         glBegin(GL_POINTS);
88     // kodingan garis part 1
89     else if (type1 == INTERP_LINES)
90         glBegin(GL_LINES);
91     for (int i = 0; i < n; i++)
92     {
93         glVertex3f(px, py, pz);
94         px += stepX;
95         u += stepU;
96         // Bila Interpolasi linear
97         if (type2 == INTERP_LINEAR)
98             py = linearInterpolate(point1.Y, point2.Y, u);
99         // Bila Interpolasi cubic
100        else if (type2 == INTERP_CUBIC)
101            py = cubicInterpolate(point0.Y, point1.Y, point2.Y, point3.Y, u);
102        // Bila Interpolasi cosine
103        else if (type2 == INTERP_COSINE)
104            py = cosineInterpolate(point0.Y, point1.Y, point2.Y, point3.Y, u);
105        // Bila Interpolasi const
106        else
107            py = point2.Y;
108         glVertex3f(px, py, pz);
109     }
110     glEnd();
111 }

```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 2,568

Line: 88 Col: 32 Sel: 0 Lines: 275 Length: 8413 Insert Done parsing in 0.094 seconds

Gambar 3. 1 Fungsi *drawInterpolation LP3*

Gambar garis hasil interpolasi n adalah jumlah titik yang dibuat type adalah tipe interpolasi yang digunakan.

```

Project1 - [Prak3.deg] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1 | main.cpp | (globals)]
Project Classes Debug main.cpp
Project1 main.cpp
69 // fungsi untuk menggaris obyek
70 void drawObject()
71 {
72     glPushMatrix();
73     // operasi transformasi rotasi obyek ke arah kanan-kiri
74     glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
75     glPushMatrix();
76     // operasi transformasi rotasi obyek ke arah atas-bawah
77     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
78     // set warna obyek ke warna hijau (0.0f, 1.0f, 0.0f)
79     glColor3f(0.0f, 1.0f, 0.0f);
80     // kodingan garis part 2
81     Vec3 point0 = Vec3(-300.0f, 600.0f, 0.0f);
82     Vec3 point1 = Vec3(-100.0f, 400.0f, 0.0f);
83     Vec3 point2 = Vec3(100.0f, 200.0f, 0.0f);
84     Vec3 point3 = Vec3(-300.0f, -180.0f, 0.0f);
85
86     drawInterpolation(point0, point1, point2, point3, 5, INTERP_POINTS, INTERP_CUBIC);
87     drawInterpolation(point0, point1, point3, 5, INTERP_LINES, INTERP_CUBIC);
88
89     drawInterpolation(point3, point2, point1, point0, 5, INTERP_POINTS, INTERP_CUBIC);
90     drawInterpolation(point3, point2, point1, point0, 5, INTERP_LINES, INTERP_CUBIC);
91
92     glPopMatrix();
93     glPopMatrix();
94 }

```

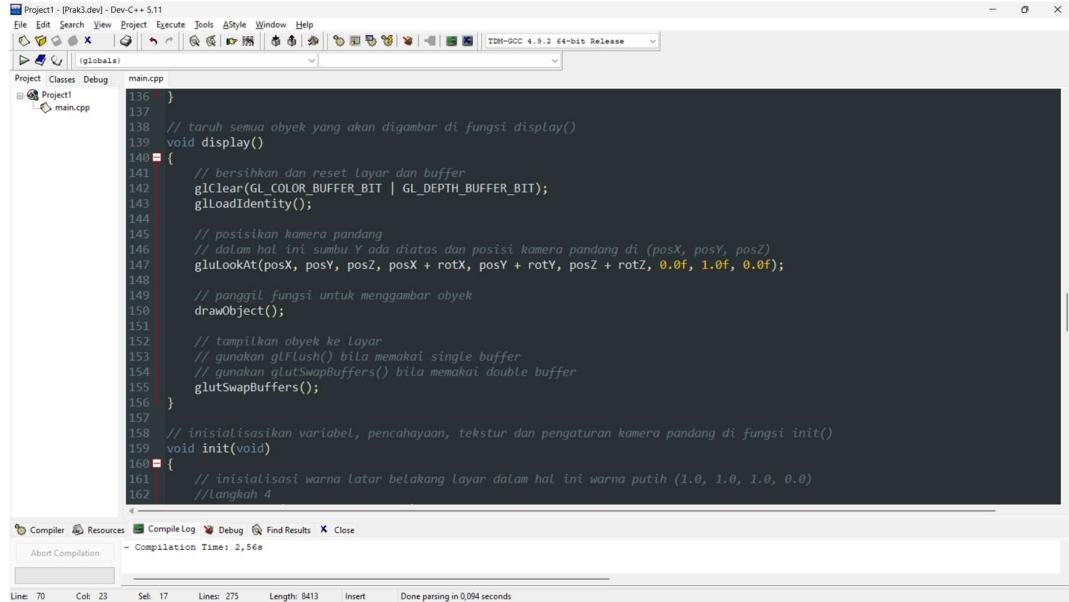
Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 2,568

Line: 70 Col: 23 Sel: 17 Lines: 275 Length: 8413 Insert Done parsing in 0.094 seconds

Gambar 3. 2 Fungsi *drawObject LP3*

Fungsi ini digunakan untuk menggambar objek yang akan di tampilkan nanti. Line 122 – 131 di gunakan untuk membuat objek berbentuk W.



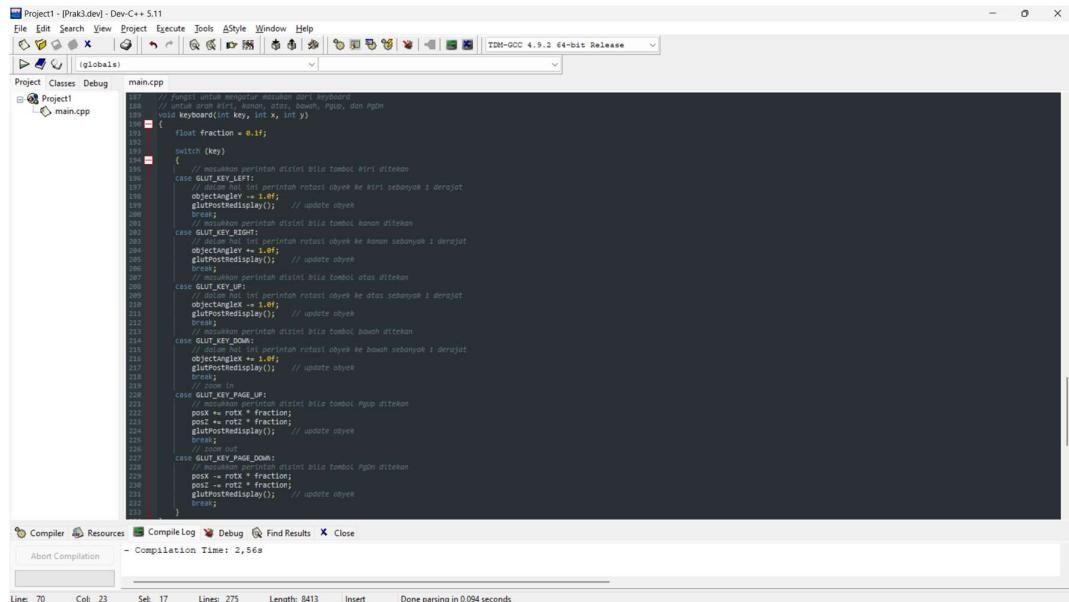
```

Project1 - [Prak3.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug main.cpp
Project1 main.cpp
136    }
137
138 // taruh semua obyek yang akan digambar di fungsi display()
139 void display()
140 {
141     // bersihkan dan reset Layar dan buffer
142     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
143     glLoadIdentity();
144
145     // posisikan kamera pandang
146     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
147     gluLookAt(posx, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
148
149     // panggil fungsi untuk menggambar objek
150     drawObject();
151
152     // tampilkan objek ke layar
153     // gunakan glFlush() bila memakai single buffer
154     // gunakan glutSwapBuffers() bila memakai double buffer
155     glutSwapBuffers();
156 }
157
158 // inisialisasikan variabel, pencitraan, tekstur dan pengaturan kamera pandang di fungsi init()
159 void init(void)
160 {
161     // inisialisasi warna Latar belakang Layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
162     // Langkah 4

```

Gambar 3. 3 Fungsi display LP3

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 150 untuk memanggil fungsi drawObject.



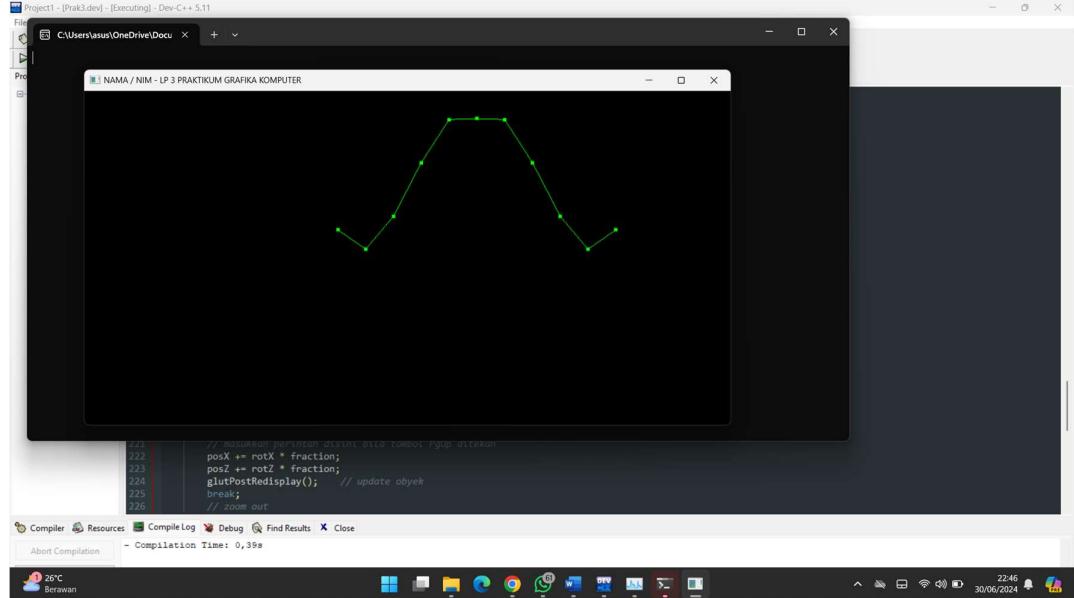
```

Project1 - [Prak3.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug main.cpp
Project1 main.cpp
147 // fungsi untuk mendeklarasikan event keyboard
148 // dengan fungsi keyboard() kita bisa mendeklarasikan PgUp, PgDn, dan PgR
149 void keyboard(int key, int x, int y)
150 {
151     float fraction = 0.1f;
152
153     switch (key)
154     {
155         // masukan perintah distini bila tombol kiri ditekan
156         case GLUT_KEY_LEFT:
157             // int perintah rotasi objek ke kiri sebanyak 1 derajat
158             objectAngleX += 1.0f;
159             glutPostRedisplay(); // update objek
160             break;
161         // masukan perintah distini bila tombol kanan ditekan
162         case GLUT_KEY_RIGHT:
163             // int perintah rotasi objek ke kanan sebanyak 1 derajat
164             objectAngleX -= 1.0f;
165             glutPostRedisplay(); // update objek
166             break;
167         // masukan perintah distini bila tombol atas ditekan
168         case GLUT_KEY_UP:
169             // int hal ini perintah rotasi objek ke atas sebanyak 1 derajat
170             objectAngleY += 1.0f;
171             glutPostRedisplay(); // update objek
172             break;
173         // masukan perintah distini bila tombol bawah ditekan
174         case GLUT_KEY_DOWN:
175             // int perintah rotasi objek ke bawah sebanyak 1 derajat
176             objectAngleY -= 1.0f;
177             glutPostRedisplay(); // update objek
178             break;
179         // zoom out
180         case GLUT_KEY_PAGE_UP:
181             // masukan perintah distini bila tombol PgUp ditekan
182             posx += rotX * fraction;
183             posy += rotY * fraction;
184             glutPostRedisplay(); // update objek
185             break;
186         // zoom in
187         case GLUT_KEY_PAGE_DOWN:
188             // masukan perintah distini bila tombol PgDn ditekan
189             posx -= rotX * fraction;
190             posy -= rotY * fraction;
191             glutPostRedisplay(); // update objek
192             break;
193     }
194 }

```

Gambar 3. 4 Fungsi Keyboard LP3

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.



Gambar 3. 5 Output LP3

3.3 POSTEST 3

1. Buat L terbalik

Untuk Kodingan Dasarnya sama dengan langkah praktikum

```

110 // fungsi untuk menggarap objek
111 void drawObject()
112 {
113     glPushMatrix();
114     // operasi transformasi rotasi objek ke arah kanan-kiri
115     glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
116     glPopMatrix();
117     // operasi transformasi rotasi objek ke arah atas-bawah
118     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
119     // set posisi objek ke posisi hijau (0.0f, 1.0f, 0.0f)
120     glColor3f(0.0f, 1.0f, 0.0f);
121
122     Vec3 point0 = Vec3(100.0f, 0.0f, 0.0f); // Posisi awal huruf U, bagian bawah kiri
123     Vec3 point1 = Vec3(100.0f, 50.0f, 0.0f); // Titik atas huruf U
124     Vec3 point2 = Vec3(100.0f, -50.0f, 0.0f); // Titik kanan huruf U, setengah bagian bawah
125     Vec3 point3 = Vec3(100.0f, 0.0f, 0.0f); // Posisi akhir huruf U, bagian bawah kanan
126
127     // Gambar bagian vertikal dari huruf U
128     drawInterpolation(point0, point1, point2, point3, 4, INTERP_POINTS, INTERP_LINEAR);
129     void drawInterpolation (Vec3 point0, Vec3 point1, Vec3 point2, Vec3 point3, int n, INTERP_TYPE type1, INTERP_TYPE type2)
130     {
131         // Bagian Interpolasi horizontal atas huruf U
132         point0 = Vec3(100.0f, 0.0f, 0.0f); // Posisi awal huruf U, bagian bawah kiri
133         point1 = Vec3(100.0f, 50.0f, 0.0f); // Titik atas huruf U
134         point2 = Vec3(150.0f, 50.0f, 0.0f); // Titik kanan huruf U, setengah bagian atas
135         point3 = Vec3(150.0f, 0.0f, 0.0f); // Posisi akhir huruf U, bagian bawah kanan
136
137         // Gambar Interpolasi horizontal atas huruf U
138         drawInterpolation(point0, point1, point2, point3, 1, INTERP_POINTS, INTERP_LINEAR);
139         drawInterpolation(point0, point1, point2, point3, 1, INTERP_LINES, INTERP_LINEAR);
140
141         glPopMatrix();
142         glPopMatrix();
143     }

```

Gambar 3. 6 drawObject POST3

Hanya merubah pada bagian drawobject, buat program seperti ini

```

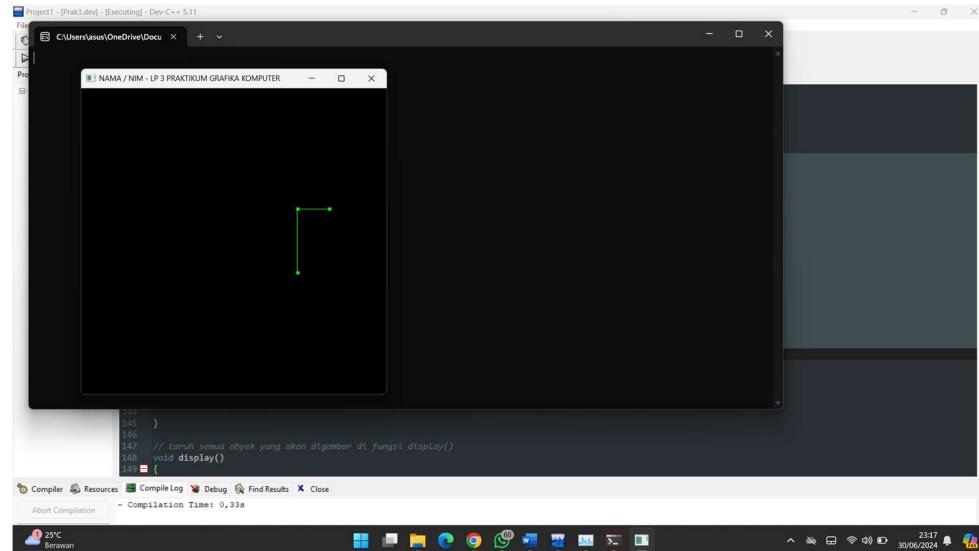
Vec3 point0 = Vec3(100.0f, 0.0f, 0.0f); // Posisi awal huruf U, bagian bawah kiri
Vec3 point1 = Vec3(100.0f, 50.0f, 0.0f); // Titik atas huruf U
Vec3 point2 = Vec3(100.0f, -50.0f, 0.0f); // Titik kanan huruf U, setengah bagian bawah
Vec3 point3 = Vec3(100.0f, 0.0f, 0.0f); // Posisi akhir huruf U, bagian bawah kanan

// Gambar bagian vertikal dari huruf U
drawInterpolation(point0, point1, point2, point3, 1, INTERP_POINTS, INTERP_LINEAR);
drawInterpolation(point0, point1, point2, point3, 1, INTERP_LINES, INTERP_LINEAR);

// Bagian lengkungan horizontal atas huruf U
point0 = Vec3(100.0f, 0.0f, 0.0f); // Posisi awal huruf U, bagian bawah kiri
point1 = Vec3(100.0f, 50.0f, 0.0f); // Titik atas huruf U
point2 = Vec3(150.0f, 50.0f, 0.0f); // Titik kanan huruf U, setengah bagian atas
point3 = Vec3(150.0f, 0.0f, 0.0f); // Posisi akhir huruf U, bagian bawah kanan

// Gambar lengkungan horizontal atas huruf U
drawInterpolation(point0, point1, point2, point3, 1, INTERP_POINTS, INTERP_LINEAR);
drawInterpolation(point0, point1, point2, point3, 1, INTERP_LINES, INTERP_LINEAR);

```



Gambar 3. 7 Output POST3

BAB IV

TRANSFORMASI 2D DAN 3D

4.1 PRETEST 4

Diketahui suatu garis P dengan titik-titik ada di koordinat A (-1, -1) dan B (1, 1). Apabila garis P dikenakan transformasi berikut, Berapa koordinat titik-titik yang baru?

1. Garis P ditranslasi sejauh T (2, 2)!

Translasi menggeser titik-titik garis PPP sejauh T(2,2)T(2, 2)T(2,2).

Koordinat titik-titik yang baru A'A'A' dan B'B'B' dapat dihitung dengan cara menambahkan vektor translasi ke koordinat asli.

Koordinat titik A baru: $A' = (-1+2, -1+2) = (1,1)$ $A' = (-1 + 2, -1 + 2) = (1, 1)$ $A' = (-1+2, -1+2) = (1,1)$

Koordinat titik B baru: $B' = (1+2, 1+2) = (3,3)$ $B' = (1 + 2, 1 + 2) = (3, 3)$ $B' = (1+2, 1+2) = (3,3)$

Jadi, setelah translasi sejauh T(2,2)T(2, 2)T(2,2), titik A menjadi (1,1)(1, 1)(1,1) dan titik B menjadi (3,3)(3, 3)(3,3).

2. Garis P discaling sebesar S (2, 1)!

Scaling mengubah jarak antara titik-titik garis PPP berdasarkan faktor scaling S(2,1)S(2, 1)S(2,1). Koordinat titik-titik yang baru A'A'A' dan B'B'B' dapat dihitung dengan mengalikan koordinat asli dengan faktor scaling.

Koordinat titik A baru: $A' = (-1 \times 2, -1 \times 1) = (-2, -1)$ $A' = (-1 \times 2, -1 \times 1) = (-2, -1)$ $A' = (-1 \times 2, -1 \times 1) = (-2, -1)$

Koordinat titik B baru: $B' = (1 \times 2, 1 \times 1) = (2,1)$ $B' = (1 \times 2, 1 \times 1) = (2, 1)$ $B' = (1 \times 2, 1 \times 1) = (2,1)$

Jadi, setelah scaling sebesar S(2,1)S(2, 1)S(2,1), titik A menjadi (-2,-1)(-2, -1)(-2,-1) dan titik B menjadi (2,1)(2, 1)(2,1).

3. Garis P dirotasi sejauh 30 derajat!

Rotasi mengubah orientasi garis PPP sejauh 30 derajat. Untuk menghitung koordinat titik-titik yang baru A'A'A' dan B'B'B', kita akan menggunakan rumus rotasi matriks.

4.2 LANGKAH PRAKTIKUM 4

```

Project2 - [Proj4Dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glPushMatrix();]
[glPopMatrix();]
[glTranslatef(objectPositionx, objectPositiony, objectPositionz);]
[glScalef(objectscalex, objectscaley, objectscalez);]
[glRotatef(objectanglex, 1.0f, 0.0f, 0.0f);]
[glColor3f(1.0f, 1.0f, 0.0f);]
[glLoadIdentity();]
[glutSolidTorus(0.5, 1.5, 50, 30);]
[glPopMatrix();]
[glPopMatrix();]

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation - Compilation Time: 2,23s
Line: 64 Col: 20 Sel: 0 Lines: 304 Length: 11550 Insert Done parsing in 0.062 seconds

```

Gambar 4. 1 Fungsi *drawObject* LP4

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 64 di gunakan untuk membuat objek berbentuk donat.

```

Project2 - [Proj4Dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glPopMatrix();]
[glPopMatrix();]
[glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);]
[glLoadIdentity();]
[gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);]
[glutSwapBuffers();]
[glutSwapBuffers();]

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation - Compilation Time: 2,23s
Line: 64 Col: 20 Sel: 0 Lines: 304 Length: 11550 Insert Done parsing in 0.062 seconds

```

Gambar 4. 2 Fungsi *display* LP4

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 150 untuk memanggil fungsi *drawObject*.

```

Project2 - [Prak4.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
D X (globals)
Project Classes Debug main.cpp
Project2 main.cpp
131 // fungsi untuk mengatur posisi objek ke arah keyboard
132 // untuk arah kiri, kanan, atas, bawah, PgUp, dan PgDn
133 void keyboard(int key, int x, int y)
134 {
135     float fraction = 0.1f;
136
137     switch (key)
138     {
139         // masukan perintah distin bila tombol kiri ditekan
140         case GLUT_KEY_LEFT:
141             rotX -= 1.0f; // int perintah rotasi objek ke kiri sebanyak 1 derajat
142             objectAngleX -= 1.0f; // update objek
143             glutPostRedisplay(); // update display
144             break;
145         // masukan perintah distin bila tombol kanan ditekan
146         case GLUT_KEY_RIGHT:
147             rotX += 1.0f; // int perintah rotasi objek ke kanan sebanyak 1 derajat
148             objectAngleX += 1.0f; // update objek
149             glutPostRedisplay(); // update display
150             break;
151         // zoom in
152         case GLUT_KEY_UP:
153             rotY -= 1.0f; // int perintah rotasi objek ke atas sebanyak 1 derajat
154             objectAngleY -= 1.0f; // update objek
155             glutPostRedisplay(); // update display
156             break;
157         // zoom out
158         case GLUT_KEY_DOWN:
159             rotY += 1.0f; // int perintah rotasi objek ke bawah sebanyak 1 derajat
160             objectAngleY += 1.0f; // update objek
161             glutPostRedisplay(); // update display
162             break;
163         // zoom in
164         case GLUT_KEY_PAGE_UP:
165             posZ -= rotZ * fraction; // int perintah distin bila tombol PgUp ditekan
166             posX -= rotX * fraction; // update objek
167             posY -= rotY * fraction; // update objek
168             glutPostRedisplay(); // update display
169             break;
170         // zoom out
171         case GLUT_KEY_PAGE_DOWN:
172             posZ += rotZ * fraction; // int perintah distin bila tombol PgDn ditekan
173             posX += rotX * fraction; // update objek
174             posY += rotY * fraction; // update objek
175             glutPostRedisplay(); // update display
176             break;
177     }
178 }

```

Compiler Resources Compile Log Debug Find Results Close
About Compilation - Compilation Time: 2,23s
Line: 91 Col: 1 Sel: 0 Lines: 304 Length: 11550 Insert Done parsing in 0.062 seconds

Gambar 4. 3 Fungsi Keyboard LP4

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.

```

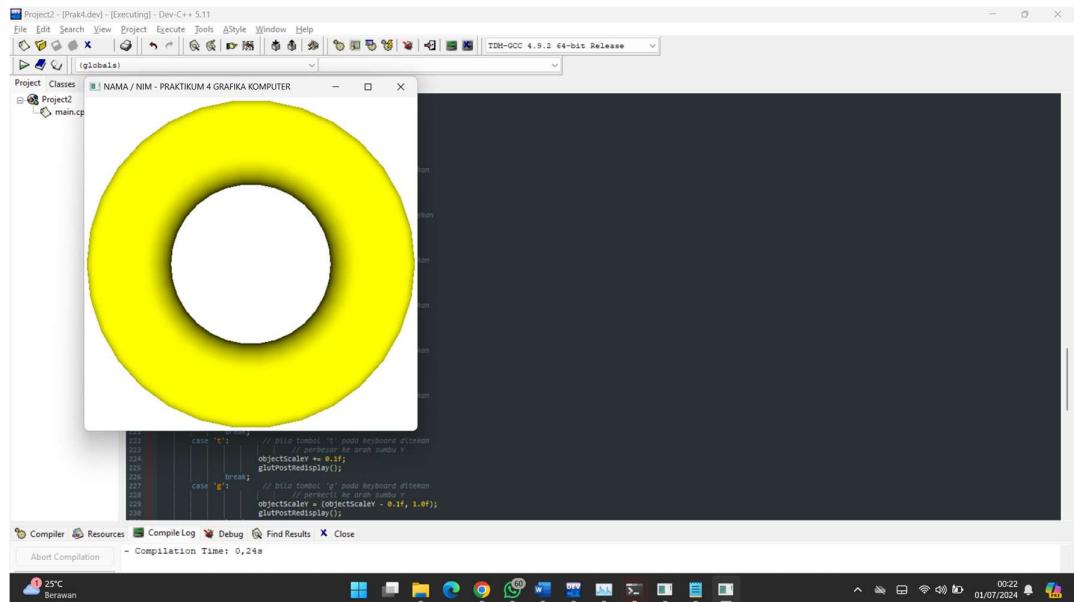
Project2 - [Prak4.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
D X (globals)
Project Classes Debug main.cpp
Project2 main.cpp
185 // fungsi untuk mengatur posisi objek
186 void keyboard(unsigned char key, int x, int y)
187 {
188     float fraction = 0.5f;
189
190     switch (key)
191     {
192         // bila tombol 'w' pada keyboard ditekan
193         case 'w':
194             // transisi ke atas
195             objectPositionY += fraction;
196             glutPostRedisplay();
197             break;
198         // bila tombol 's' pada keyboard ditekan
199         case 's':
200             // transisi ke bawah
201             objectPositionY -= fraction;
202             glutPostRedisplay();
203             break;
204         case 'd':
205             // bila tombol 'd' pada keyboard ditekan
206             // transisi ke kanan
207             objectPositionX += fraction;
208             glutPostRedisplay();
209             break;
210         case 'a':
211             // bila tombol 'a' pada keyboard ditekan
212             // transisi ke kiri
213             objectPositionX -= fraction;
214             glutPostRedisplay();
215             break;
216         case 'q':
217             // bila tombol 'q' pada keyboard ditekan
218             // transisi ke arah sumbu z
219             objectPositionZ += fraction;
220             glutPostRedisplay();
221             break;
222         case 'e':
223             // bila tombol 'e' pada keyboard ditekan
224             // transisi ke arah sumbu y
225             objectScaleY += 0.1f;
226             glutPostRedisplay();
227             break;
228         case 'r':
229             // bila tombol 'r' pada keyboard ditekan
230             // transisi ke arah sumbu x
231             objectScaleX += (objectScaleX - 0.1f, 1.0f);
232             glutPostRedisplay();
233             break;
234     }
235 }

```

Compiler Resources Compile Log Debug Find Results Close
About Compilation - Compilation Time: 2,23s
Line: 91 Col: 1 Sel: 0 Lines: 304 Length: 11550 Insert Done parsing in 0.062 seconds

Gambar 4. 4 Fungsi keyboard1 LP4

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan tombol w a s d.



Gambar 4. 5 Output LP4

4.3 POSTEST 4

1. Membuat Cone

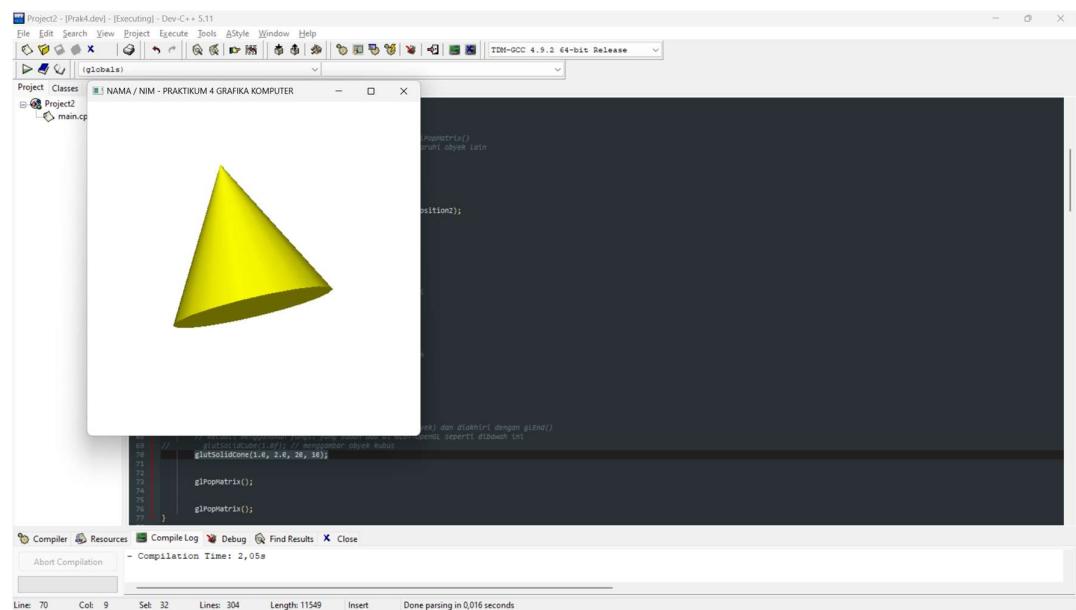
Untuk Kodingan Dasarnya sama dengan langkah praktikum.

```
Project2 - [Prak4-dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[ ] (globals)
Project Classes Debug main.cpp
Project2 main.cpp
1 // Fungsi untuk menggambar objek kubus
2
3 void drawObject()
4 {
5     // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
6     // fungsiuya agar objek tidak terpengaruh atau mempengaruhi objek lain
7     // misalnya transformasi, ditarik, ditarikbalik, ditarikbalik, ditarikbalik
8     glPushMatrix();
9
10    // operasi transformasi translasi objek
11    // posisi awal, posisi akhir
12    glTranslatef(objectPositionx, objectPositiony, objectPositionz);
13
14    // operasi transformasi scaling objek
15    // memperbesar atau mempercilkan objek
16    // posisi awal, posisi akhir
17    glScalef(objectScalex, objectScaley, objectScalez);
18
19    // operasi transformasi rotasi objek ke arah kanan-kiri
20    // posisi awal, posisi akhir
21    glRotatef(objectAnglex, 0.0f, 1.0f, 0.0f);
22
23    // set warna objek ke warna hijau (0.0f, 1.0f, 0.0f)
24    glColor3f(1.0f, 1.0f, 0.0f);
25
26    // Bila menggambar objek yang dilalu (definisi objek) dan diakhiri dengan glEnd()
27    // berasal dari program yang punya fungsi yang fasih code di diutama, seperti cobaan ini:
28    // glutSolidCube(8); // menggambar objek kubus
29    glutSolidCone(1.0, 2.0, 20, 10);
30
31    glPopMatrix();
32
33    glPopMatrix();
34 }
```

Gambar 4. 6 Fungsi drawObject POST4

Hanya merubah pada bagian drawobject, buat program seperti ini

```
glutSolidCone(1.0, 2.0, 20, 10);
```



The screenshot shows the Dev-C++ IDE interface. The main window displays a 3D rendering of a yellow cone. The code editor window shows the following C++ code:

```
Project2 - [Praktikum] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
Project Classes NAMA / NIM - PRAKTIKUM 4 GRAFIKA KOMPUTER
Project2 main.cpp

85 // Menggambar objek kubah
86 glutSolidCylinder(1.0, 2.0, 20, 10);
87
88 // Menggambar objek kubah
89 glutSolidCone(1.0, 2.0, 20, 10);
90
91 glPopMatrix();
92
93 glPushMatrix();
94
95 glutSolidCylinder(1.0, 2.0, 20, 10);
96
97 glPopMatrix();
98
99 glPopMatrix(); // Mengakhiri objek kubah
100
101 glutSwapBuffers(); // Menggantikan objek kubah dengan objek kubah
102
103 glutPostRedisplay(); // Menggambar objek kubah
104
105 glutMainLoop(); // Memulai loop utama
106
107 }

Line: 70 Col: 9 Set: 32 Lines: 304 Length: 11549 Insert Done parsing in 0.016 seconds
```

The status bar at the bottom indicates "Compilation Time: 2,05s".

Gambar 4. 7 Output POST4

BAB V

PROYEKSI 3D

5.1 PRETEST 5

1. Jelaskan yang dimaksud dengan proyeksi orthogonal!

Proyeksi orthogonal adalah teknik proyeksi dalam grafika komputer di mana semua garis proyeksi tegak lurus terhadap bidang proyeksi. Dalam proyeksi ini, jarak dari objek ke bidang proyeksi tidak mempengaruhi ukuran relatif objek tersebut. Ini berarti bahwa objek yang lebih jauh dari titik pandang tetap memiliki ukuran yang sama seperti objek yang lebih dekat. Proyeksi orthogonal sering digunakan dalam tata letak teknis, seperti denah bangunan, karena mempertahankan proporsi objek yang sebenarnya.

2. Jelaskan yang dimaksud dengan proyeksi perspektif!

Proyeksi perspektif adalah teknik proyeksi dalam grafika komputer di mana garis proyeksi tidak selalu tegak lurus terhadap bidang proyeksi. Pada proyeksi perspektif, jarak dari objek ke bidang proyeksi mempengaruhi ukuran relatif objek tersebut. Objek yang lebih dekat dengan titik pandang akan terlihat lebih besar daripada objek yang lebih jauh. Proyeksi perspektif sering digunakan untuk menciptakan ilusi kedalaman dan realisme dalam gambar atau animasi.

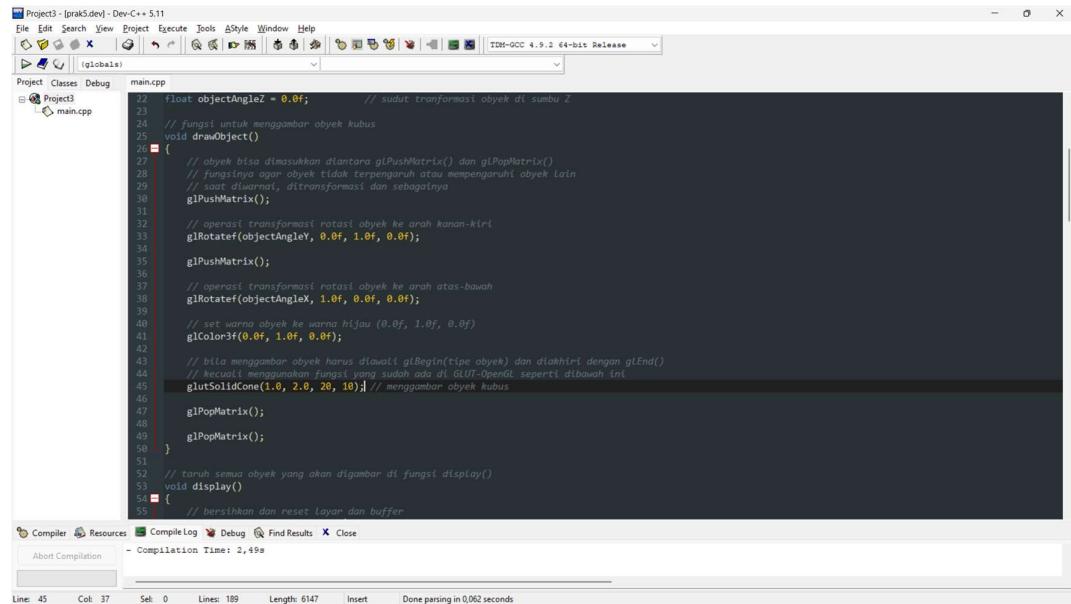
3. Jelaskan perbedaan antara proyeksi orthogonal dengan Prespektif ?

Perbedaan utama antara proyeksi orthogonal dan perspektif dapat dijelaskan sebagai berikut:

- a. Bidang Proyeksi: Pada proyeksi orthogonal, bidang proyeksi tegak lurus terhadap garis pandang, sedangkan pada proyeksi perspektif, bidang proyeksi tidak selalu tegak lurus.
- b. Ukuran Objek: Dalam proyeksi orthogonal, ukuran objek tetap konstan terlepas dari jaraknya dari titik pandang. Sedangkan dalam proyeksi perspektif, ukuran objek berubah tergantung pada jaraknya dari titik pandang.
- c. Ilusi Kedalaman: Proyeksi perspektif menciptakan ilusi kedalaman dan perspektif yang tidak dimiliki oleh proyeksi orthogonal. Hal ini membuat

proyeksi perspektif lebih cocok untuk aplikasi yang membutuhkan representasi visual yang lebih realistik.

5.2 LANGKAH PRAKTIKUM 5

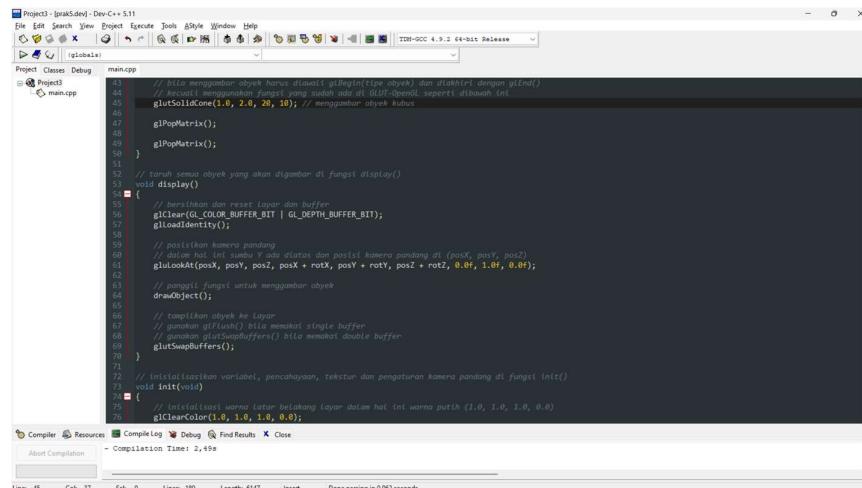


```

Project3 - [prak5.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug main.cpp
Project3 main.cpp
22 float objectAngleZ = 0.0f;           // sudut transformasi objek Zi sumbu Z
23 // fungsi untuk menggambar objek kubus
24 void drawObject()
25 {
26     // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
27     // fungsi agar objek tidak terpengaruh atau mempengaruhi objek lain
28     // saat dimasuk, diltransfromasi dan sebagainya
29     glPushMatrix();
30
31     // operasi transformasi rotasi objek ke arah kanan-kiri
32     glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
33
34     glPushMatrix();
35
36     // operasi transformasi rotasi objek ke arah atas-bawah
37     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
38
39     // set warna objek ke warna hijau (0.0f, 1.0f, 0.0f)
40     glColor3f(0.0f, 1.0f, 0.0f);
41
42     // bila menggambar objek harus diawali glBegin(tipe objek) dan diakhiri dengan glEnd()
43     // kecuali menggunakan fungsi yang sudah ada di GLUT-OpenGL seperti dibawah ini
44     glutSolidCone(1.0, 2.0, 20, 10); // menggambar objek kubus
45
46     glPopMatrix();
47
48     glPopMatrix();
49 }
50
51 // taruh semua objek yang akan digambar di fungsi display()
52 void display()
53 {
54     // bersihkan dan reset layar dan buffer
55 }
```

Gambar 5. 1 Fungsi *drawObject* LP5

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 45 di gunakan untuk membuat objek berbentuk Cone.

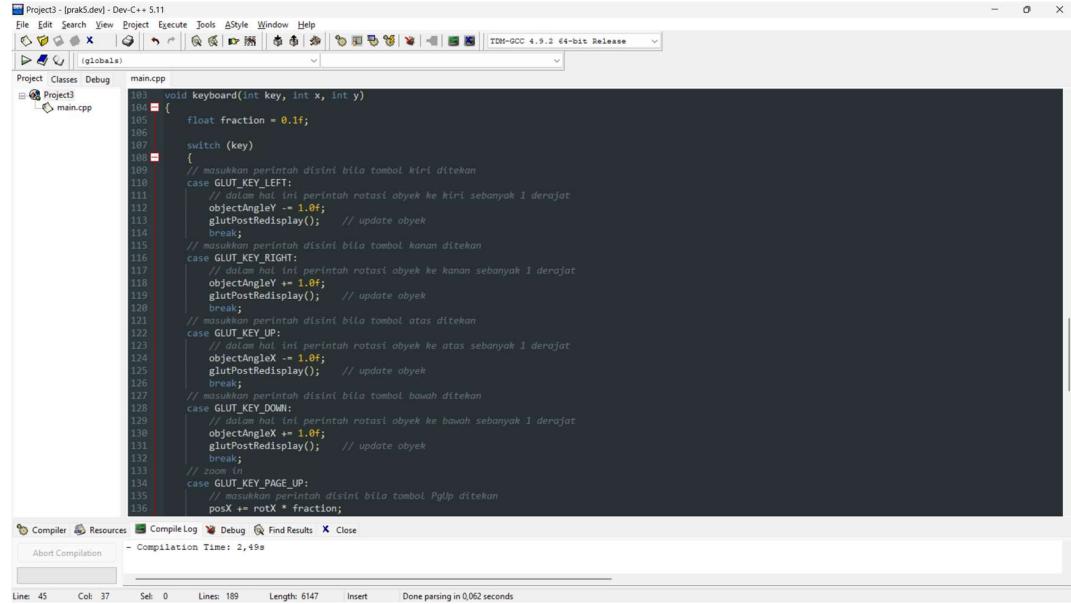


```

Project3 - [prak5.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug main.cpp
Project3 main.cpp
43 // Bila menggambar objek harus diawali glBegin(tipe objek) dan diakhiri dengan glEnd()
44 // kecuali menggunakan fungsi yang sudah ada di GLUT-OpenGL seperti dibawah ini
45 glutSolidCone(1.0, 2.0, 20, 10); // menggambar objek kubus
46
47 glPopMatrix();
48
49 glPopMatrix();
50
51 // taruh semua objek yang akan digambar di fungsi display()
52 void display()
53 {
54     // bersihkan dan reset layar dan buffer
55     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
56     glLoadIdentity();
57
58     // posisikan kamera pandang
59     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
60     glutLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
61
62     // panggil fungsi untuk menggambar objek
63     drawObject();
64
65     // tampilkan objek ke layar
66     // gunakan glFlush() bila memakai single buffer
67     // gunakan glutSwapBuffers() bila memakai double buffer
68     glutSwapBuffers();
69 }
70
71 // Inisialisasi variabel, perekaman, teksur dan pengaturan kamera pandang di fungsi init()
72 void init(void)
73 {
74     // inisialisasi warna layar keputih
75     glClearColor(1.0, 1.0, 1.0, 0.0);
76 }
```

Gambar 5. 2 Fungsi *Display* LP5

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 64 untuk memanggil fungsi drawObject.



```

Project3 - [prak5.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TIK-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug main.cpp
Project3
main.cpp
83 void keyboard(int key, int x, int y)
84 {
85     float fraction = 0.1f;
86
87     switch (key)
88     {
89         // masukkan perintah disini bila tombol kiri ditekan
90         case GLUT_KEY_LEFT:
91             // dalam hal ini perintah rotasi obyek ke kiri sebanyak 1 derajat
92             objectAngleX -= 1.0f;
93             glutPostRedisplay(); // update obyek
94             break;
95         // masukkan perintah disini bila tombol kanan ditekan
96         case GLUT_KEY_RIGHT:
97             // dalam hal ini perintah rotasi obyek ke kanan sebanyak 1 derajat
98             objectAngleX += 1.0f;
99             glutPostRedisplay(); // update obyek
100            break;
101        // masukkan perintah disini bila tombol atas ditekan
102        case GLUT_KEY_UP:
103            // dalam hal ini perintah rotasi obyek ke atas sebanyak 1 derajat
104            objectAngleY -= 1.0f;
105            glutPostRedisplay(); // update obyek
106            break;
107        // masukkan perintah disini bila tombol bawah ditekan
108        case GLUT_KEY_DOWN:
109            // dalam hal ini perintah rotasi obyek ke bawah sebanyak 1 derajat
110            objectAngleY += 1.0f;
111            glutPostRedisplay(); // update obyek
112            break;
113        // zoom in
114        case GLUT_KEY_PAGE_UP:
115            // masukkan perintah disini bila tombol PgUp ditekan
116            posX += rotX * fraction;
117
118        case GLUT_KEY_PAGE_DOWN:
119            // masukkan perintah disini bila tombol PgDn ditekan
120            posX -= rotX * fraction;
121
122        default:
123            break;
124    }
125    glutPostRedisplay(); // update obyek
126}
127
128
129
130
131
132
133
134
135
136

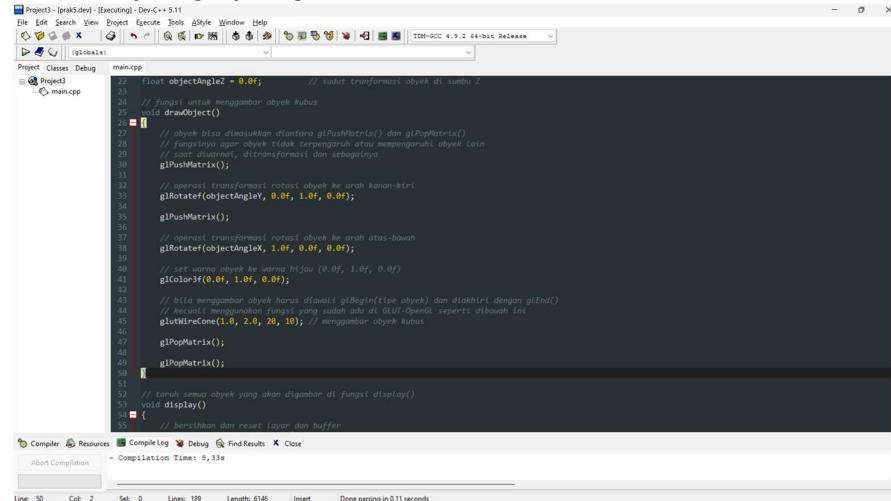
```

Gambar 5. 3 Fungsi keyboard LP5

Fungsi ini di gunakan untuk mengatur pergerekian objek dengan menggunakan keyboard.

5.3 POSTEST 5

1. Membuat jaring – jaring Krucut



```

Project3 - [prak5.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TIK-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug main.cpp
Project3
main.cpp
13 void drawObject()
14 {
15     // set sudut transformasi obyek di sumbu Z
16     float objectAngleZ = 0.0f;
17
18     // fungsi untuk menggambar objek kubus
19     void drawObject()
20     {
21         // obyek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
22         // fungsi ini agar obyek tidak terpengaruh atau mempengaruhi obyek lain
23         // saat diwarna, ditransformasi dan sebagainya
24         glPushMatrix();
25
26         // operasi transformasi rotasi obyek ke arah kanan-kiri
27         glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
28
29         glPushMatrix();
30
31         // operasi transformasi rotasi obyek ke arah atas-bawah
32         glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
33
34         // set warna obyek ke warna hijau (0.0f, 1.0f, 0.0f)
35         glColor3f(0.0f, 1.0f, 0.0f);
36
37         // kita menggambar objek harus diawali glBegin(type obyek) dan diakhiri dengan glEnd()
38         // kecuali menggunakan fungsi yang sudah ada di GLUT_OpenGL seperti dibawah ini
39         glutWireCone(1.0, 2.0, 20, 10); // menggambar objek kubus
40
41         glPopMatrix();
42
43         glPopMatrix();
44
45         // turun semua obyek yang akan digambar di fungsi display()
46     }
47
48     void display()
49     {
50         // bersihkan dan reset layar dan buffer
51
52     }
53
54 }
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
89
90
91
92
93
94
95
96
97
98
99
99
100
101
102
103
104
105
106
107
108
109
109
110
111
112
113
114
115
116
117
118
119
119
120
121
122
123
124
125
126
127
128
129
129
130
131
132
133
134
135
136

```

Gambar 5. 4 drawObject POST

Hanya merubah pada bagian drawobject, buat program seperti ini

```
glutWireCone(1.0, 2.0, 20, 10);
```

The screenshot shows a C++ development environment with the following details:

- Project Tab:** Shows "Project3 - [prj&dev] - [Executing] - Dev-C++ 5.11".
- Classes Tab:** Shows a file named "main.cp".
- Resources Tab:** Shows a file named "NAMA / NIM - KODE DASAR PRAKTIKUM GRAFIKA KOM...".
- Compiler Log Tab:** Shows the compilation log:
 - Line: 50 Col: 2
 - Sek: 0 Lines: 189 Length: 6146 Insert
 - Compilation Time: 8,33s
 - Done parsing in 0,11 seconds
- Code Editor:** Displays the OpenGL rendering code:

```
49     glutWireCone(1.0, 2.0, 20, 10);
50 }
51 // taruh semua obyek yang akan digambar di fungsi display()
52 void display()
53 {
54     // bersihkan dan reset layar dan buffer
55 }
```
- Output Window:** Shows the rendered wireframe cone.

Gambar 5. 5 Output POST5

BAB VI

REPRESENTASI OBYEK 3D

6.1 PRETEST 6

1. Sebutkan metode representasi obyek 3D yang anda ketahui!

(Prahara et al., n.d.) Beberapa teknik representasi obyek 3D yaitu model wireframe, sweep representation, boundary representation, spatial partitioning representation, constructive solid geometry, dan sebagainya. Pada boundary representation obyek dideskripsikan dari batasan permukaannya yang membedakan obyek bagian dalam dan bagian luar. Hal ini dilakukan dengan menambahkan sisi pada jaring-jaring model obyek.

2. Jelaskan setiap metode representasi obyek 3D yang anda sebutkan di soal nomor 1!
 - a. Wireframe adalah representasi tiga dimensi dari objek yang hanya menggunakan garis untuk menunjukkan bentuk dan struktur dasar objek tersebut. Teknik ini tidak mengisi permukaan objek, sehingga kita dapat melihat melalui objek dan memahami struktur internalnya.
 - b. Sweep representation adalah metode untuk membuat objek 3D dengan menggerakkan bentuk 2D sepanjang lintasan tertentu. Bentuk 2D tersebut disebut sebagai "profile" atau "cross-section", dan lintasan yang dilalui disebut sebagai "path".
 - c. Boundary representation adalah teknik yang menggambarkan bentuk objek 3D dengan menggunakan permukaan batasnya. Objek didefinisikan oleh wajah-wajah (faces), tepi-tepi (edges), dan simpul-simpul (vertices).
 - d. Spatial partitioning representation membagi ruang 3D menjadi bagian-bagian lebih kecil yang lebih mudah diolah dan diakses. Contoh umum dari teknik ini adalah Octree dan BSP (Binary Space Partitioning) Trees.
 - e. Constructive Solid Geometry adalah teknik untuk membuat objek 3D yang kompleks dengan menggabungkan objek-objek sederhana menggunakan operasi boolean seperti union (gabungan), intersection (irisian), dan difference (selisih).

6.2 LANGKAH PRAKTIKUM 6

The screenshot shows the Dev-C++ IDE interface with the code editor open. The project is named 'praktikum6' and the file is 'main.cpp'. The code implements the `drawObject` function, which uses OpenGL commands to draw a 3D object. The code includes calls to `glBegin(GL_TRIANGLES)`, `glVertex3f` for vertices, and `glEnd()`. It also includes color settings like `glColor3f(1.0f, 1.0f, 1.0f);` and camera setup with `gluLookAt`.

```

Project4 - praktikum6 - Dev-C++ 5.1.1
File Edit Search View Project Execute Tools AStyle Window Help
[glutSolidCone] (glutSolidSphere) (glutSolidTeapot) (glutSolidTorus) (glutSolidTorusL) (glutSolidTorusR) (glutSolidTorusH) (glutSolidTorusV)
Project Classes Debug [*] main.cpp
Project4 main.cpp
    void drawObject()
    {
        glBegin(GL_TRIANGLES);
        glVertex3f(0.0f, 1.0f, 1.0f);
        glVertex3f(-1.0f, -1.0f, 1.0f);
        glVertex3f(1.0f, -1.0f, 1.0f);
        glEnd();
        // First we're drawing main body bottom
        glColor3f(1.0f, 1.0f, 0.0f);
        glBegin(GL_TRIANGLES);
        glVertex3f(0.0f, 1.0f, -1.0f);
        glVertex3f(1.0f, 1.0f, -1.0f);
        glVertex3f(-1.0f, 1.0f, -1.0f);
        glEnd();
        // Second we're drawing main body top
        glColor3f(1.0f, 1.0f, 1.0f);
        glBegin(GL_TRIANGLES);
        glVertex3f(0.0f, -1.0f, 1.0f);
        glVertex3f(1.0f, -1.0f, 1.0f);
        glVertex3f(-1.0f, -1.0f, 1.0f);
        glEnd();
        // Third we're drawing main body left
        glColor3f(0.0f, 1.0f, 1.0f);
        glBegin(GL_TRIANGLES);
        glVertex3f(0.0f, 1.0f, -1.0f);
        glVertex3f(1.0f, 1.0f, -1.0f);
        glVertex3f(1.0f, -1.0f, -1.0f);
        glEnd();
        // Fourth we're drawing main body right
        glColor3f(0.0f, 1.0f, -1.0f);
        glBegin(GL_TRIANGLES);
        glVertex3f(0.0f, 1.0f, -1.0f);
        glVertex3f(-1.0f, 1.0f, -1.0f);
        glVertex3f(-1.0f, -1.0f, -1.0f);
        glEnd();
        // Fifth we're drawing main body back
        glColor3f(0.0f, 0.0f, 1.0f);
        glBegin(GL_QUADS);
        glVertex3f(0.0f, 1.0f, 1.0f);
        glVertex3f(1.0f, 1.0f, 1.0f);
        glVertex3f(1.0f, -1.0f, 1.0f);
        glVertex3f(0.0f, -1.0f, 1.0f);
        glEnd();
        // Sixth we're drawing main body front
        glColor3f(0.0f, 0.0f, -1.0f);
        glBegin(GL_QUADS);
        glVertex3f(0.0f, 1.0f, -1.0f);
        glVertex3f(1.0f, 1.0f, -1.0f);
        glVertex3f(1.0f, -1.0f, -1.0f);
        glVertex3f(0.0f, -1.0f, -1.0f);
        glEnd();
    }
    // draw cylinder
    void drawCylinder(float radius, float height, int slices, int stacks)
{
}

```

Compiler Resources Find Results Close
Abort Compilation - Compilation Time: 0,17s

Line: 80 Col: 13 Sel: 0 Lines: 297 Length: 9323 Insert Done parsing in 0.015 seconds

Gambar 6. 1 Fungsi *drawObject* LP6

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 28 - 81 di gunakan untuk membuat objek berbentuk Segitiga sama kaki 3D.

The screenshot shows the Dev-C++ IDE interface with the code editor open. The project is named 'praktikum6' and the file is 'main.cpp'. The code implements the `display` function, which handles the rendering loop. It includes camera setup with `gluLookAt`, clearing buffers with `glClear`, and calling `drawObject` to render objects.

```

Project4 - praktikum6 - Dev-C++ 5.1.1
File Edit Search View Project Execute Tools AStyle Window Help
[glutSolidCone] (glutSolidSphere) (glutSolidTeapot) (glutSolidTorus) (glutSolidTorusL) (glutSolidTorusR) (glutSolidTorusH) (glutSolidTorusV)
Project Classes Debug [*] main.cpp
Project4 main.cpp
    // glutSolidCone(1.0f);
    // glutSolidSphere(1.0f, 50, 50);
    // glutSolidTeapot(1.0f);
    // glutSolidTorus(0.5f, 1.0f, 20, 20);
}

// turunkan semua objek yang akan digambar di fungsi display()
void display()
{
    // bersihkan dan reset layar dan buffer
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    // posisikan kamera pandang
    // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
    gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);

    // panggil fungsi untuk menggambar objek
    drawObject();

    // tampilkan objek ke layar
    // gunakan glFlush() bila memakai single buffer
    // gunakan glutSwapBuffers() bila memakai double buffer
    glutSwapBuffers();
}

// initialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
void init(void)
{
    // initialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glEnable(GL_DEPTH_TEST); // mengaktifkan depth buffer
    glMatrixMode(GL_PROJECTION);

```

Compiler Resources Find Results Close
Abort Compilation - Compilation Time: 0,59s

Line: 152 Col: 7 Sel: 2 Lines: 297 Length: 9324 Insert Done parsing in 0.016 seconds

Gambar 6. 2 Fungsi *display* LP6

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 173 untuk memanggil fungsi `drawObject`.

```

Project4 - [prak6.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools A Style Window Help
(globals)
Project Classes Debug main.cpp
Project4 main.cpp
1 // untuk buat file, nomor, stat, power, play, dan pyon
2 void keyboard(int key, int x, int y)
3 {
4     float fraction = 0.1f;
5
6     switch (key)
7     {
8         case GLUT_KEY_LEFT:
9             // casian perintah distint bila tombol kiri ditekan
10            if (int perintah rotasi objek ke kiri sebanyak 1 derajat)
11                objectangle -= 1.0f;
12            glutPostRedisplay(); // update objek
13            break;
14        // casian perintah distint bila tombol kanan ditekan
15        case GLUT_KEY_RIGHT:
16            if (int perintah rotasi objek ke kanan sebanyak 1 derajat)
17                objectangle += 1.0f;
18            glutPostRedisplay(); // update objek
19            break;
20        // casian perintah distint bila tombol atas ditekan
21        case GLUT_KEY_UP:
22            if (int perintah rotasi objek ke atas sebanyak 1 derajat)
23                objectangle -= 1.0f;
24            glutPostRedisplay(); // update objek
25            break;
26        // casian perintah distint bila tombol bawah ditekan
27        case GLUT_KEY_DOWN:
28            if (int perintah rotasi objek ke bawah sebanyak 1 derajat)
29                objectangle += 1.0f;
30            glutPostRedisplay(); // update objek
31            break;
32        // zoom in
33        case GLUT_KEY_PAGE_UP:
34            posx -= rot1 * fraction;
35            posy -= rot2 * fraction;
36            glutPostRedisplay(); // update objek
37            break;
38        // zoom out
39        case GLUT_KEY_PAGE_DOWN:
40            posx += rot1 * fraction;
41            posy += rot2 * fraction;
42            glutPostRedisplay(); // update objek
43            break;
44    }
45 }

```

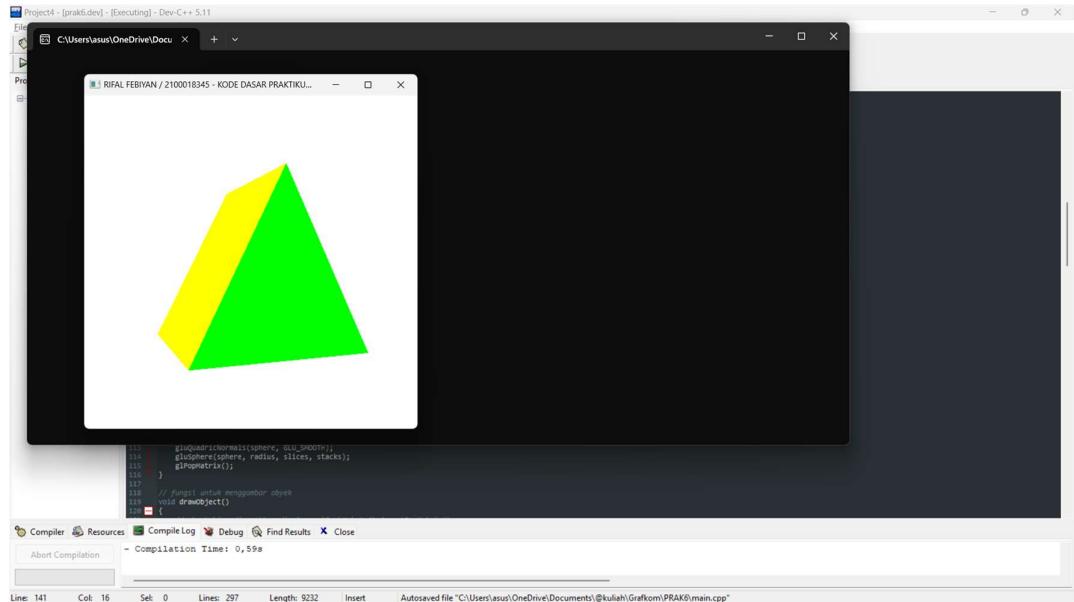
Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,59s

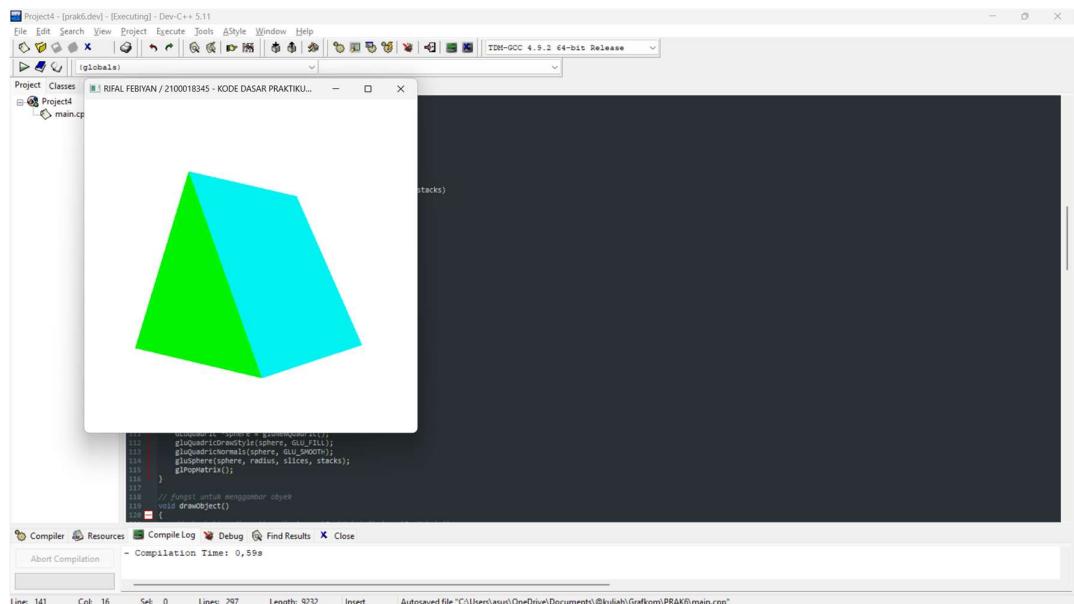
Line: 152 Col: 7 Sel: 2 Lines: 297 Length: 9234 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK6\main.cpp"

Gambar 6. 3 Fungsi keyboard LP6

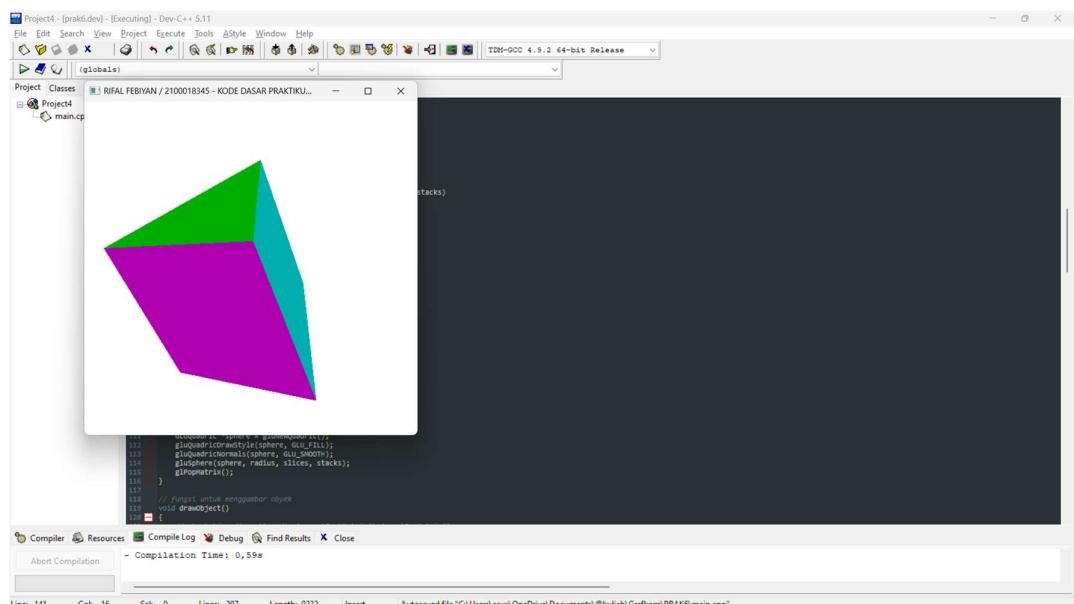
Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.



Gambar 6. 4 output1 LP6



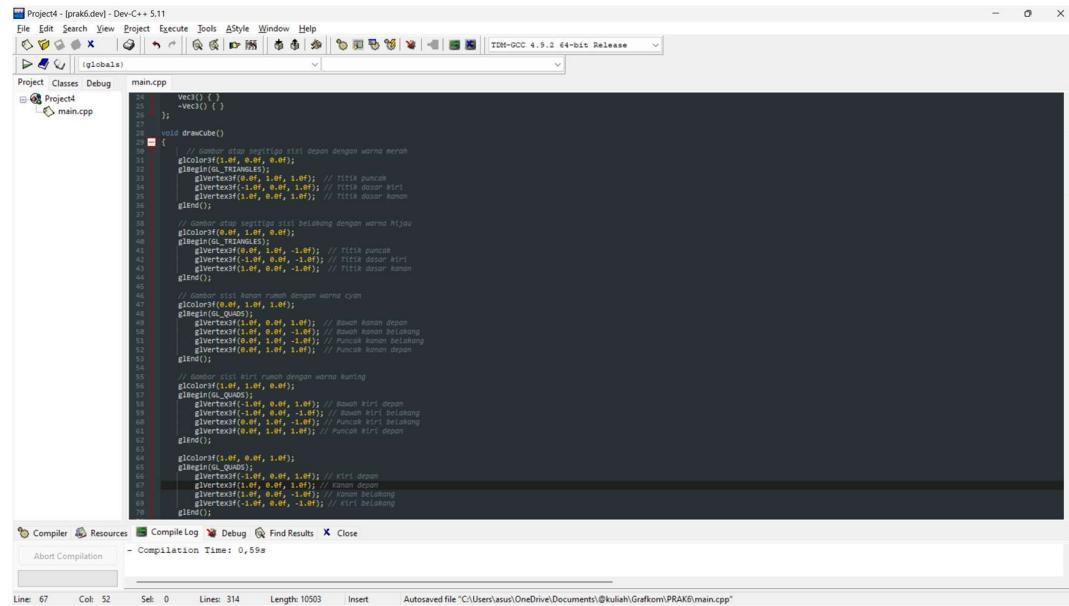
Gambar 6. 5 output2 LP6



Gambar 6. 6 output3 LP6

6.3 POSTEST 6

1. Membuat Rumah BAB 1 Versi 3D



```

Project4 - [prak6.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug main.cpp
Project4
main.cpp
24     vec3() { }
25     ~vec3() { }
26 };
27
28 void drawCube()
29 {
30     // Gambar atap segitiga sisi depan dengan warna merah
31     glColor3f(1.0f, 0.0f, 0.0f);
32     glBegin(GL_TRIANGLES);
33     glVertex3f(1.0f, 1.0f); // Titik puncak
34     glVertex3f(-1.0f, 0.0f); // Titik dasar kiri
35     glVertex3f(1.0f, 0.0f); // Titik dasar kanan
36     glEnd();
37
38     // Gambar atap segitiga sisi belakang dengan warna hijau
39     glColor3f(0.0f, 1.0f, 0.0f);
40     glBegin(GL_TRIANGLES);
41     glVertex3f(0.0f, 1.0f); // Titik puncak
42     glVertex3f(-1.0f, 0.0f); // Titik dasar kiri
43     glVertex3f(1.0f, 0.0f); // Titik dasar kanan
44     glEnd();
45
46     // Gambar sisi kanan rumah dengan warna cyan
47     glColor3f(0.0f, 1.0f, 1.0f);
48     glBegin(GL_QUADS);
49     glVertex3f(1.0f, 0.0f, 1.0f); // Bawah kanan depan
50     glVertex3f(1.0f, 0.0f, -1.0f); // Puncak kanan depan
51     glVertex3f(0.0f, 1.0f, -1.0f); // Puncak kanan belakang
52     glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kanan depan
53     glEnd();
54
55     // Gambar sisi kiri rumah dengan warna kuning
56     glColor3f(1.0f, 1.0f, 0.0f);
57     glBegin(GL_QUADS);
58     glVertex3f(-1.0f, 0.0f, 1.0f); // Bawah kiri depan
59     glVertex3f(-1.0f, 0.0f, -1.0f); // Bawah kiri belakang
60     glVertex3f(0.0f, 1.0f, -1.0f); // Puncak kiri belakang
61     glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kiri depan
62     glEnd();
63
64     // Gambar atap segitiga sisi depan dengan warna hijau
65     glColor3f(0.0f, 1.0f, 0.0f);
66     glBegin(GL_TRIANGLES);
67     glVertex3f(-1.0f, 0.0f, 1.0f); // Kiri depan
68     glVertex3f(1.0f, 0.0f, 1.0f); // Kanan depan
69     glVertex3f(1.0f, 0.0f, -1.0f); // Kanan belakang
70     glVertex3f(-1.0f, 0.0f, -1.0f); // Kiri belakang
71     glEnd();

```

Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Compilation Time: 0.59s
Line: 67 Col: 52 Sek: 0 Lines: 314 Length: 10503 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK8\main.cpp"

Gambar 6. 7 Fungsi drawObject POST6

Hanya merubah pada bagian drawobject, buat program seperti ini

```

glColor3f(1.0f, 0.0f, 0.0f);
glBegin(GL_TRIANGLES);
    glVertex3f(0.0f, 1.0f, 1.0f); // Titik puncak
    glVertex3f(-1.0f, 0.0f, 1.0f); // Titik dasar kiri
    glVertex3f(1.0f, 0.0f, 1.0f); // Titik dasar kanan
glEnd();

// Gambar atap segitiga sisi belakang dengan warna hijau
glColor3f(0.0f, 1.0f, 0.0f);
glBegin(GL_TRIANGLES);
    glVertex3f(0.0f, 1.0f, -1.0f); // Titik puncak
    glVertex3f(-1.0f, 0.0f, -1.0f); // Titik dasar kiri
    glVertex3f(1.0f, 0.0f, -1.0f); // Titik dasar kanan
glEnd();

// Gambar sisi kanan rumah dengan warna cyan
glColor3f(0.0f, 1.0f, 1.0f);
glBegin(GL_QUADS);
    glVertex3f(1.0f, 0.0f, 1.0f); // Bawah kanan depan
    glVertex3f(1.0f, 0.0f, -1.0f); // Bawah kanan belakang
    glVertex3f(0.0f, 1.0f, -1.0f); // Puncak kanan belakang
    glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kanan depan
glEnd();

// Gambar sisi kiri rumah dengan warna kuning
glColor3f(1.0f, 1.0f, 0.0f);
glBegin(GL_QUADS);
    glVertex3f(-1.0f, 0.0f, 1.0f); // Bawah kiri depan
    glVertex3f(-1.0f, 0.0f, -1.0f); // Bawah kiri belakang
    glVertex3f(0.0f, 1.0f, -1.0f); // Puncak kiri belakang
    glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kiri depan
glEnd();

```

```

glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kanan depan
glEnd();

// Gambar sisi kiri rumah dengan warna kuning
glColor3f(1.0f, 1.0f, 0.0f);
glBegin(GL_QUADS);
    glVertex3f(-1.0f, 0.0f, 1.0f); // Bawah kiri depan
    glVertex3f(-1.0f, 0.0f, -1.0f); // Bawah kiri belakang
    glVertex3f(0.0f, 1.0f, -1.0f); // Puncak kiri belakang
    glVertex3f(0.0f, 1.0f, 1.0f); // Puncak kiri depan
glEnd();

glColor3f(1.0f, 0.0f, 1.0f);
glBegin(GL_QUADS);
    glVertex3f(-1.0f, 0.0f, 1.0f); // Kiri depan
    glVertex3f(1.0f, 0.0f, 1.0f); // Kanan depan
    glVertex3f(1.0f, 0.0f, -1.0f); // Kanan belakang
    glVertex3f(-1.0f, 0.0f, -1.0f); // Kiri belakang
glEnd();

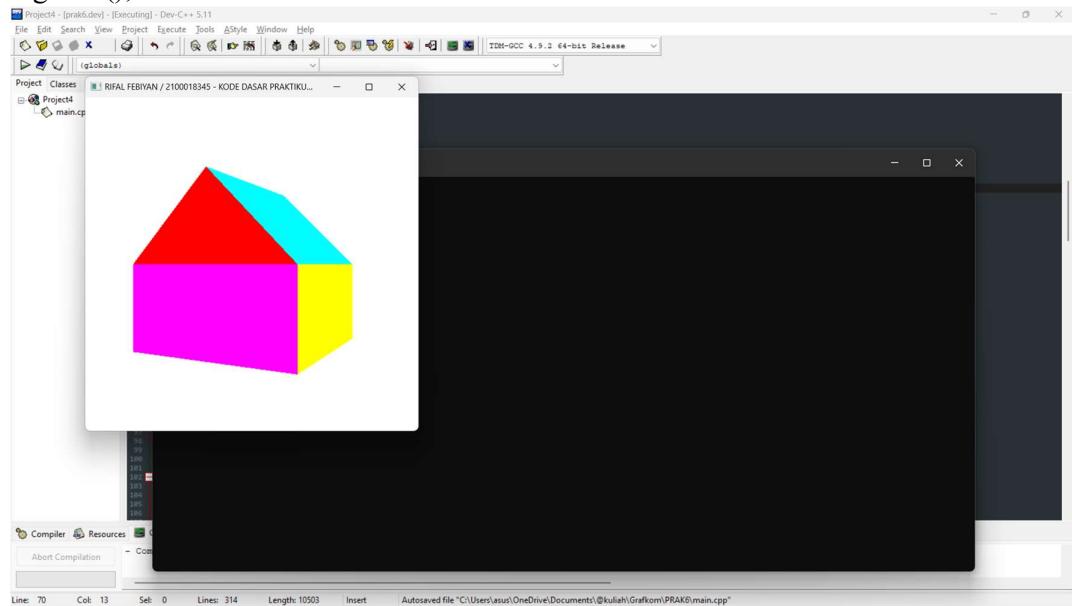
// Gambar badan rumah (depan dan belakang) dengan warna putih
glColor3f(1.0f, 0.0f, 1.0f);
glBegin(GL_QUADS);
    // Sisi depan
    glVertex3f(-1.0f, 0.0f, 1.0f); // Kiri bawah
    glVertex3f(1.0f, 0.0f, 1.0f); // Kanan bawah
    glVertex3f(1.0f, -1.0f, 1.0f); // Kanan atas
    glVertex3f(-1.0f, -1.0f, 1.0f); // Kiri atas
    // Sisi belakang
    glVertex3f(-1.0f, 0.0f, -1.0f); // Kiri bawah
    glVertex3f(1.0f, 0.0f, -1.0f); // Kanan bawah
    glVertex3f(1.0f, -1.0f, -1.0f); // Kanan atas
    glVertex3f(-1.0f, -1.0f, -1.0f); // Kiri atas
    //KIRI
    glColor3f(0.0f, 1.0f, 1.0f);
    glVertex3f(-1.0f, -1.0f, 1.0f);
    glVertex3f(-1.0f, -1.0f, -1.0f);
    glVertex3f(-1.0f, 0.0f, -1.0f);
    glVertex3f(-1.0f, 0.0f, 1.0f);
    //KANAN
    glColor3f(1.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, -1.0f, 1.0f);
    glVertex3f(1.0f, -1.0f, -1.0f);

```

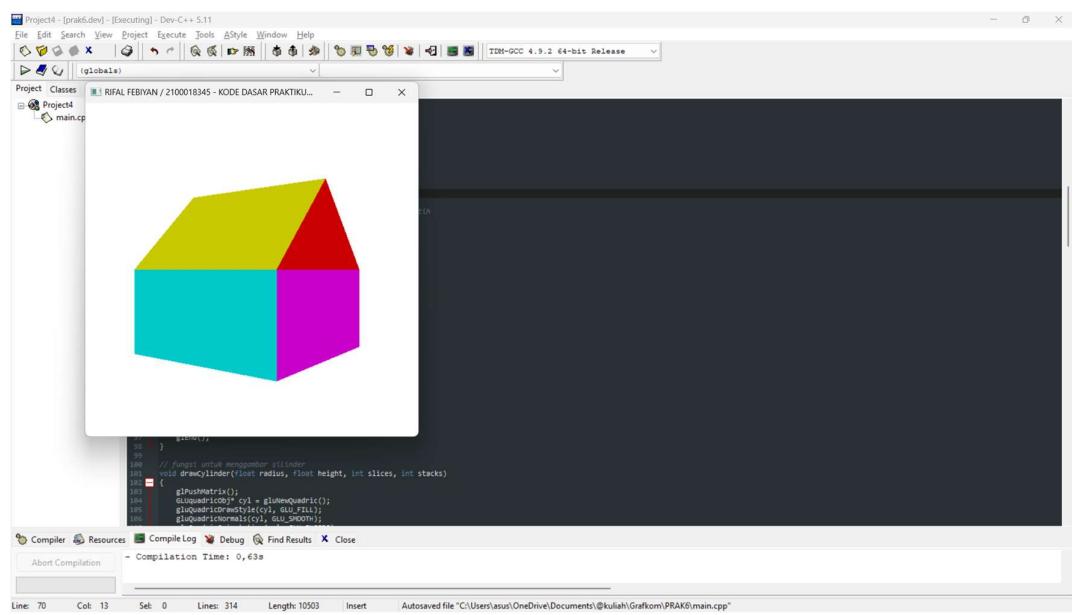
```

glVertex3f( 1.0f, 0.0f,-1.0f);
glVertex3f( 1.0f, 0.0f, 1.0f);
glEnd();

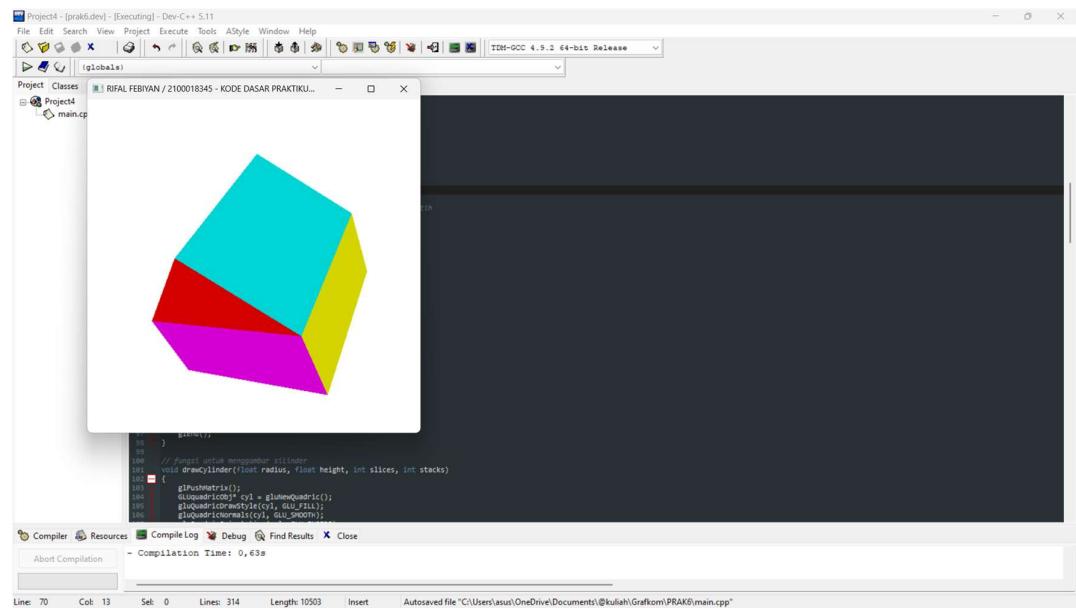
```



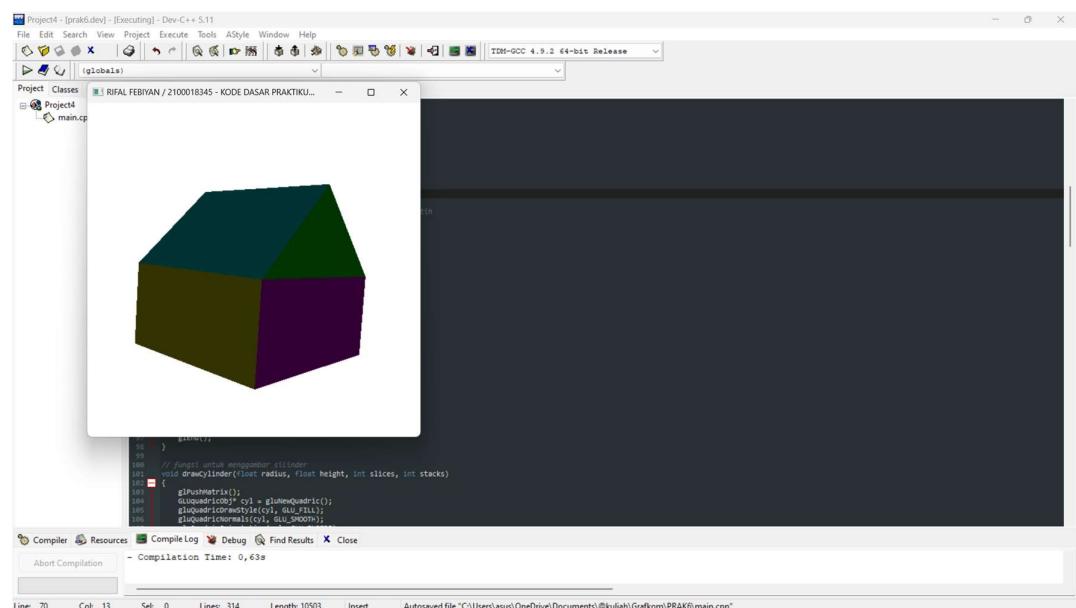
Gambar 6. 8 output1 POST6



Gambar 6. 9 output2 POST6



Gambar 6. 10 output3 POST6



Gambar 6. 11 output4 POST6

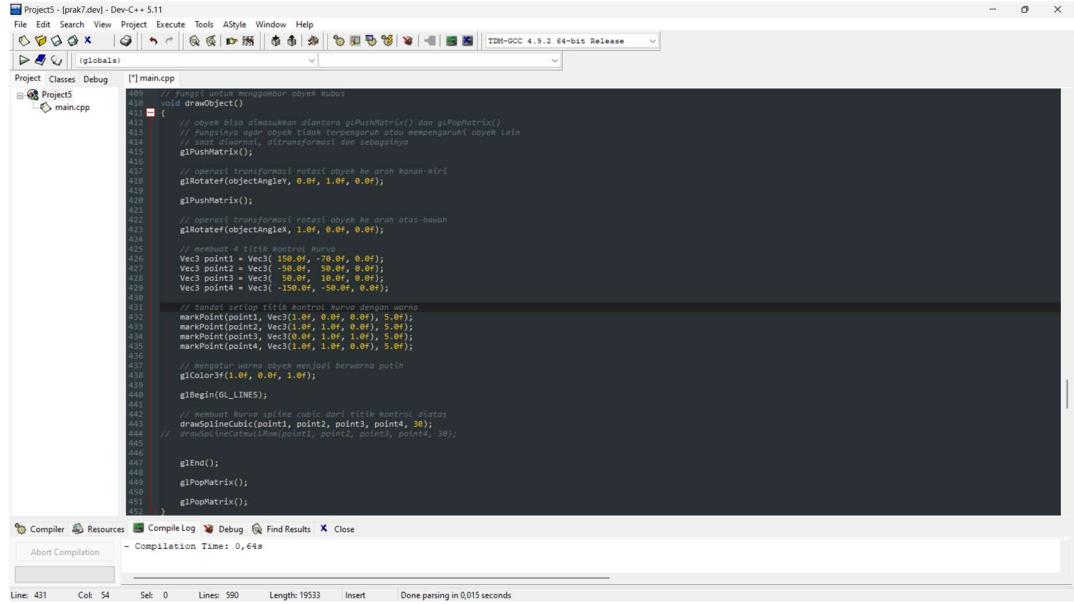
BAB VII

KURVA SPLINE

7.1 PRETEST 7

1. Jelaskan perbedaan antara kurva spline Cubic, Catmull-Rom, Hermit, dan Bezier!
 - a. Spline Cubic
 - Spline cubic adalah jenis kurva yang terbentuk oleh serangkaian segmen spline cubic yang saling terhubung.
 - Spline cubic menggunakan polinomial kubik (tingkat tiga) untuk mendefinisikan kurva di antara titik kontrol.
 - b. Carmull-rom
 - Kurva ini menggunakan teknik interpolasi untuk menghasilkan kurva yang melewati setiap titik kontrol yang diberikan.
 - Salah satu keunggulan Catmull-Rom adalah kemampuannya untuk melalui setiap titik kontrol, sehingga menghasilkan kurva yang lebih akurat dalam menggambarkan kontur objek atau jalur gerakan.
 - c. Hermit
 - Kurva Hermit menggunakan polinomial Hermite untuk menggambarkan kurva.
 - Kurva ini didefinisikan oleh dua titik kontrol dan dua vektor tangen yang terkait dengan masing-masing titik kontrol.
 - d. Bezier
 - Kurva Bezier menggunakan polinomial Bezier untuk menggambarkan kurva.
 - Kurva ini didefinisikan oleh beberapa titik kontrol yang membentuk poligon kendali (control polygon).

7.2 LANGKAH PRAKTIKUM 7



```

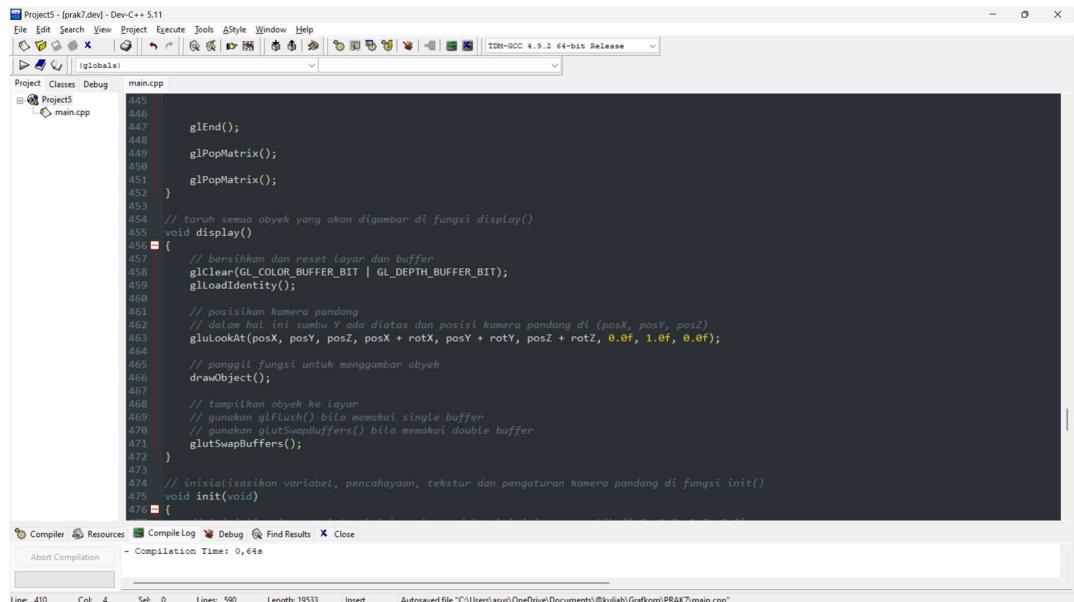
Projects-prak7.dev - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[ ] main.cpp
Project Classes Debug [+] main.cpp
  - t
    // obyek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
    // fungsi agar obyek tidak terpengaruh atau mempengaruhi obyek lain
    // saat diolah, ditransformasi dan sebagainya
    glPushMatrix();
    // operasi transformasi rotasi obyek ke arah manan-kiri
    glRotated(objectangleX, 0.0f, 1.0f, 0.0f);
    glPopMatrix();
    // operasi transformasi rotasi obyek ke arah atas-bawah
    glRotated(objectangleY, 1.0f, 0.0f, 0.0f);
    // menarik 4 titik kontrol kurva
    Vec3 point1 = Vec3( 150.0f, -70.0f, 0.0f);
    Vec3 point2 = Vec3( -80.0f, 50.0f, 0.0f);
    Vec3 point3 = Vec3( 10.0f, 50.0f, 0.0f);
    Vec3 point4 = Vec3( -150.0f, -50.0f, 0.0f);
    // tampil setup titik kontrol kurva dengan warna
    markPoint(point1, Vec3(1.0f, 0.0f, 0.0f), 5.0f);
    markPoint(point2, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
    markPoint(point3, Vec3(0.0f, 1.0f, 0.0f), 5.0f);
    markPoint(point4, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
    // mengisi permukaan objek berbentuk paraf
    glColor3f(1.0f, 0.0f, 1.0f);
    glBegin(GL_LINES);
    // membuat kurva spline cubic dari titik kontrol diatas
    drawSplineCubic(point1, point2, point3, point4, 30);
    // drawSplineCubicWithNormal(point1, point2, point3, point4, 30);
    glEnd();
    glPopMatrix();
    glPopMatrix();
}

```

Line: 431 Col: 54 Sel: 0 Lines: 590 Length: 19533 Insert Done parsing in 0.015 seconds

Gambar 7. 1 Fungsi *drawObject* LP7

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 410 - 452 di gunakan untuk membuat objek berbentuk Paraf.



```

Projects-prak7.dev - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[ ] main.cpp
Project Classes Debug main.cpp
  - t
    glEnd();
    glPopMatrix();
    glPopMatrix();
}
// taruh semua obyek yang akan digambar di fungsi display()
void display()
{
    // bersihkan dan reset layar dan buffer
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    glTranslatef(0, 0, -10);
    // posisikan kamera pandang
    // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
    gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
    // panggil fungsi untuk menggambar obyek
    drawObject();
    // tampilkan obyek ke layar
    // gunakan glutSwapBuffers() bila memakai single buffer
    // gunakan glutDoubleBuffer() bila memakai double buffer
    glutSwapBuffers();
}
// initialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
void init(void)
{

```

Line: 410 Col: 4 Sel: 0 Lines: 590 Length: 19533 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\PRAK7\main.cpp"

Gambar 7. 2 Fungsi *display* LP7

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 173 untuk memanggil fungsi *drawObject*.

```
Project5 - [prak7.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[File|Edit|Search|View|Project|Execute|Tools|AStyle|Window|Help]
[TIM6-GCC 4.9.2 64-bit Release]
Project Classes Debug main.cpp
Project5 main.cpp
main.cpp
56 void keyboard(int key, int x, int y)
57 {
58     float fraction = 0.1f;
59
60     switch (key)
61     {
62         // menambah perintah distini bila tombol kiri ditekan
63         case GLUT_KEY_LEFT:
64             objectAngleX -= 1.0f;
65             glutPostRedisplay(); // update objek
66             break;
67         // menambah perintah distini bila tombol kanan ditekan
68         case GLUT_KEY_RIGHT:
69             objectAngleX += 1.0f;
70             glutPostRedisplay(); // update objek
71             break;
72         // menambah perintah distini bila tombol atas ditekan
73         case GLUT_KEY_UP:
74             // dalam hal ini perintah rotasi objek ke atas sebanyak 1 derajat
75             rotX += 1.0f;
76             glutPostRedisplay(); // update objek
77             break;
78         // menambah perintah distini bila tombol bawah ditekan
79         case GLUT_KEY_DOWN:
80             objectAngleX += 1.0f;
81             glutPostRedisplay(); // update objek
82             break;
83         // zoom out
84         case GLUT_KEY_PAGE_UP:
85             pos -= rotX * fraction;
86             pos -= rotY * fraction;
87             glutPostRedisplay(); // update objek
88             break;
89         // zoom in
90         case GLUT_KEY_PAGE_DOWN:
91             pos += rotX * fraction;
92             pos += rotY * fraction;
93             glutPostRedisplay(); // update objek
94             break;
95     }
96 }
97 
```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,64s

Line: 410 Col: 4 Sel: 0 Lines: 590 Length: 19533 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK7\main.cpp"

Gambar 7. 3 Fungsi keyboard LP7

Fungsi ini di gunakan untuk mengatur pergerekian objek dengan menggunakan keyboard.

```
Project5 - [prak7.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[File|Edit|Search|View|Project|Execute|Tools|AStyle|Window|Help]
[Project5]
C:\Users\asus\OneDrive\Docu
Project5 main.cpp
main.cpp
56 void keyboard(int key, int x, int y)
57 {
58     float fraction = 0.1f;
59
60     switch (key)
61     {
62         // dalam hal ini perintah rotasi objek ke bawah sebanyak 1 derajat
63         case GLUT_KEY_UP:
64             objectAngleX += 1.0f;
65             glutPostRedisplay(); // update objek
66             break;
67         // zoom out
68         case GLUT_KEY_DOWN:
69             objectAngleX -= 1.0f;
70             glutPostRedisplay(); // update objek
71             break;
72     }
73 }
74 
```

Compiler Resources Compile Log Debug Find Results Close

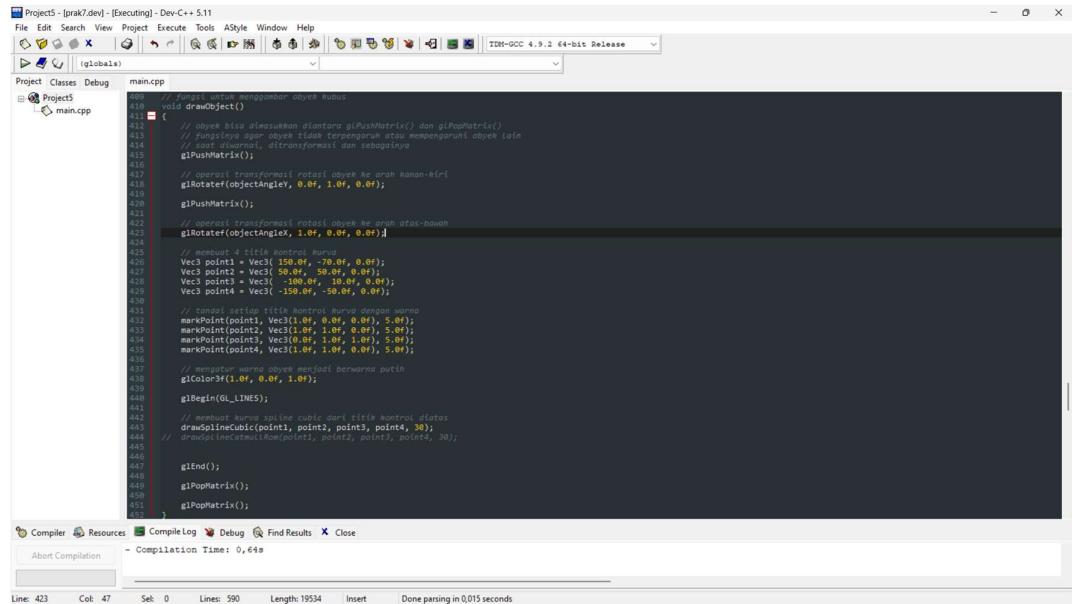
Abort Compilation - Compilation Time: 0,73s

Line: 410 Col: 4 Sel: 0 Lines: 590 Length: 19533 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK7\main.cpp"

Gambar 7. 4 Output LP7

7.3 POSTEST 7

1. Membuat Gunung



The screenshot shows the Dev-C++ IDE interface with the main.cpp file open. The code defines a `drawObject` function that handles matrix operations like pushing and popping matrices, rotating objects, and defining control points for a cubic spline curve. The code is annotated with comments explaining its purpose.

```
Project5 - [prak7.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug main.cpp
Project5
main.cpp
409 // fungsi untuk menggambar objek Kubus
410
411 void drawObject()
412 {
413     // obyek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
414     // Fungsinya agar obyek tidak terpengaruh atau mempengaruhi obyek lain
415     // yang dibuat sebelumnya, dilakukan transformasi dan sebagainya
416     glPushMatrix();
417
418     // operasi transformasi rotasi objek ke arah kanan-kiri
419     glRotatef(objectAngle, 0.0f, 1.0f, 0.0f);
420
421     // operasi transformasi rotasi objek ke arah atas-bawah
422     glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
423
424     // titik kontrol kurva
425     Vec3 point1 = Vec3( 150.0f, -70.0f, 0.0f );
426     Vec3 point2 = Vec3( 50.0f, 50.0f, 0.0f );
427     Vec3 point3 = Vec3( -100.0f, 10.0f, 0.0f );
428     Vec3 point4 = Vec3( -150.0f, -50.0f, 0.0f );
429
430     // tandai setiap titik kontrol kurva dengan warna
431     markPoint(point1, Vec3(1.0f, 0.0f, 0.0f), 5.0f);
432     markPoint(point2, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
433     markPoint(point3, Vec3(0.0f, 1.0f, 1.0f), 5.0f);
434     markPoint(point4, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
435
436     // menutup warna objek menjadi berwarna putih
437     glColor3f(1.0f, 0.0f, 1.0f);
438
439     glBegin(GL_LINES);
440
441     // membuat garis spline cubic dari titik kontrol diatas
442     drawSplineCubic(point1, point2, point3, point4, 30);
443     drawSplineCubic(point4, point3, point2, point1, 20);
444
445     glEnd();
446
447     glPopMatrix();
448
449     glPopMatrix();
450
451 }
```

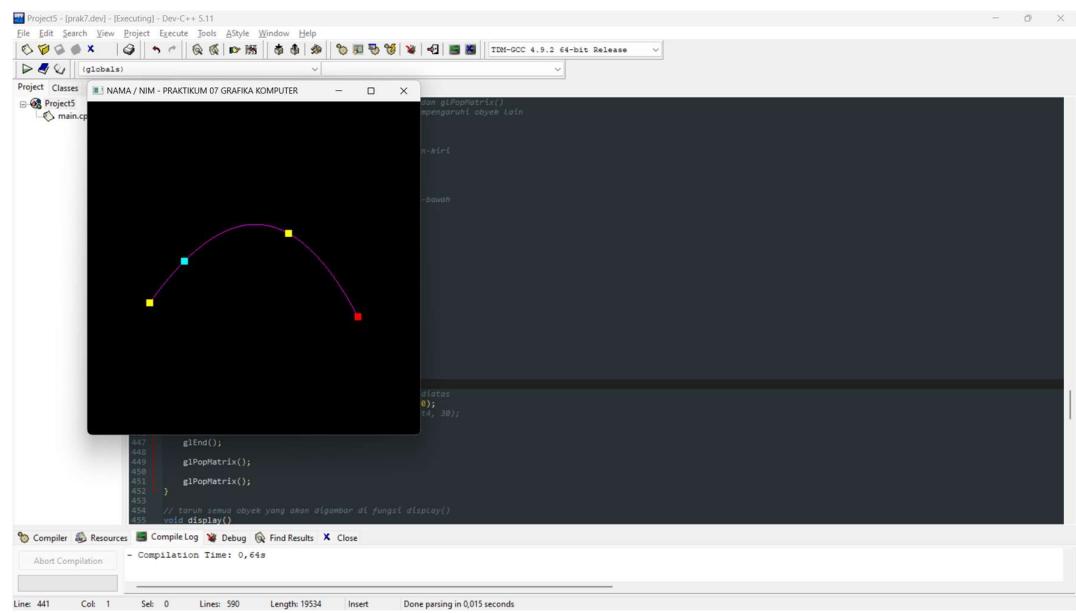
Line: 423 Col: 47 Sel: 0 Lines: 590 Length: 19534 Insert Done parsing in 0.015 seconds

Gambar 7. 5 Fungsi `drawObject` POST7

Hanya merubah pada bagian `drawobject`, buat program seperti ini

```
Vec3 point1 = Vec3( 150.0f, -70.0f, 0.0f);
Vec3 point2 = Vec3( 50.0f, 50.0f, 0.0f);
Vec3 point3 = Vec3( -100.0f, 10.0f, 0.0f);
Vec3 point4 = Vec3( -150.0f, -50.0f, 0.0f);
```

```
// tandai setiap titik kontrol kurva dengan warna
markPoint(point1, Vec3(1.0f, 0.0f, 0.0f), 5.0f);
markPoint(point2, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
markPoint(point3, Vec3(0.0f, 1.0f, 1.0f), 5.0f);
markPoint(point4, Vec3(1.0f, 1.0f, 0.0f), 5.0f);
```



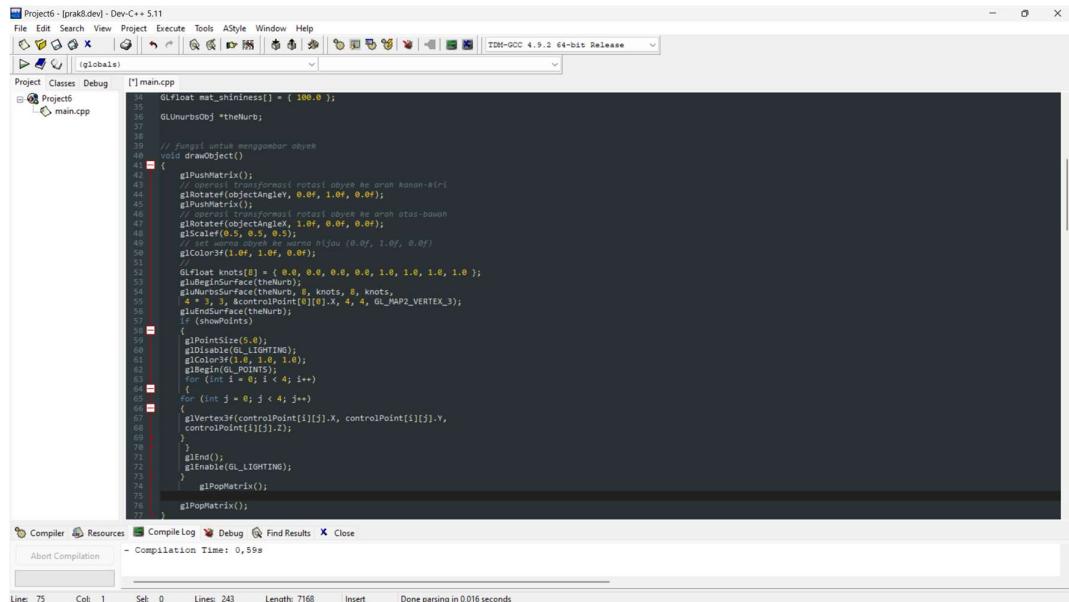
Gambar 7. 6 output POST7

BAB VIII TEKNIK REPRESENTASI PERMUKAAN

8.1 PRETEST 8

1. Sebutkan teknik representasi permukaan yang anda ketahui!
 - a. Triangulasi Poligon
 - b. Subdivisi permukaan poligon
 - c. lofting
2. Jelaskan setiap teknik representasi permukaan yang anda sebutkan di soal nomor 1!
 - a. Triangulasi poligon adalah proses mengubah poligon dengan lebih dari tiga sisi menjadi kumpulan segitiga.
 - b. Subdivisi permukaan poligon adalah proses membagi setiap poligon dalam model 3D menjadi poligon-poligon yang lebih kecil.
 - c. Lofting adalah teknik yang digunakan untuk membuat bentuk 3D dengan menggabungkan dua atau lebih kurva.

8.2 LANGKAH PRAKTIKUM 8

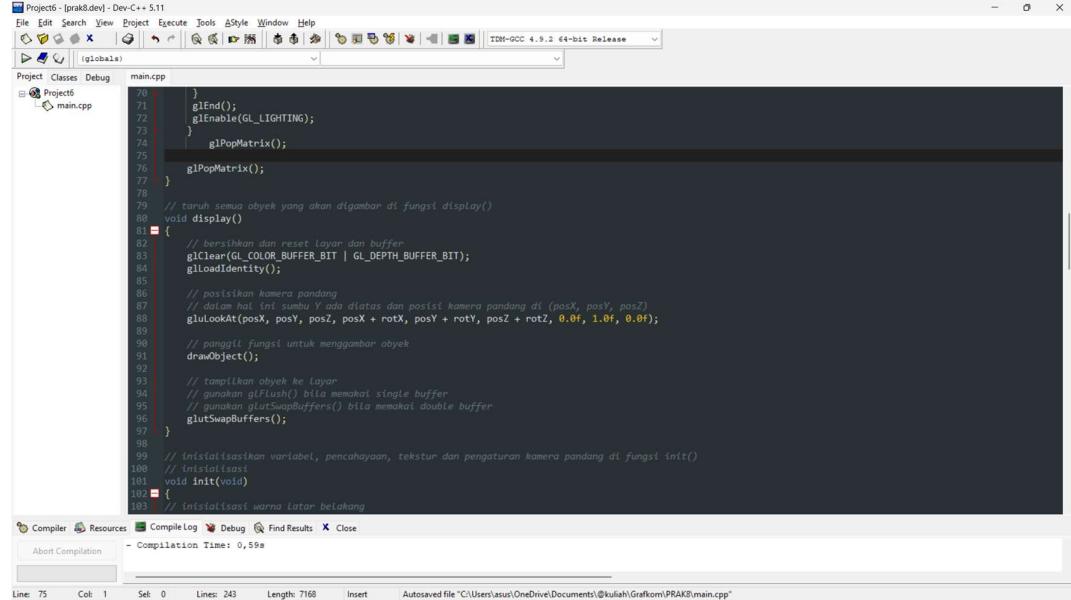


The screenshot shows the Dev-C++ IDE interface with the project 'Project6 - [prak8dev]' open. The main window displays the 'main.cpp' file containing C++ code for rendering a NURBS surface. The code includes matrix operations like pushing and popping matrices, setting up the NURBS object, and defining control points and knots. It also uses OpenGL functions like glBegin(GL_POINTS) and glVertex3f to draw the surface. The status bar at the bottom shows compilation details: Line: 75, Col: 1, Sel: 0, Lines: 243, Length: 7168, Insert, Done parsing in 0.016 seconds.

```
31 //Float mat_shininess[] = { 100.0 };
32
33 GLUnurbsObj *theNurb;
34
35 // fungsi untuk menggambar objek
36
37 void drawObject()
38 {
39     glPushMatrix();
40     //matriks transformasi miring objek ke arah kanan-kiri
41     glRotatef(45.0, 0.0, 1.0, 0.0);
42     glPushMatrix();
43     //matriks transformasi rotasi objek ke arah atas-bawah
44     glRotatef(0.0, 0.0, 0.0, 1.0);
45     glPushMatrix();
46     //matriks transformasi rotasi objek ke arah depan-belakang
47     glRotatef(0.0, 1.0, 0.0, 0.0);
48     glScalef(0.5, 0.5, 0.5);
49     // set warna objek ke warna objek (0.0f, 1.0f, 0.0f)
50     glColor3f(1.0, 1.0, 0.0);
51
52     GLfloat knots[] = { 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0 };
53     glBegin(GL_POINTS);
54     gluNurbsSurface(theNurb, 6, knots, 8, knots,
55     4, 3, &controlPoint[0][0], 4, 4, GL_MAP2_VERTEX_3);
56     gluEndSurface(theNurb);
57     glDisable(GL_DEPTH_TEST);
58     glPointSize(5.0);
59     glEnable(GL_POINT_SMOOTH);
60     glColor3f(1.0, 1.0, 1.0);
61     glBegin(GL_POINTS);
62     for (int i = 0; i < 4; i++)
63     {
64         for (int j = 0; j < 4; j++)
65         {
66             glVertex3f(controlPoint[i][j].x, controlPoint[i][j].y,
67                         controlPoint[i][j].z);
68         }
69     }
70     glEnd();
71     glEnable(GL_LIGHTING);
72     glPopMatrix();
73     glPopMatrix();
74 }
75
76
77 }
```

Gambar 8. 1 Fungsi drawObject LP8

Fungsi ini digunakan untuk menggambar objek yang akan ditampilkan nanti. Line 40 - 77 di gunakan untuk membuat objek Cembung



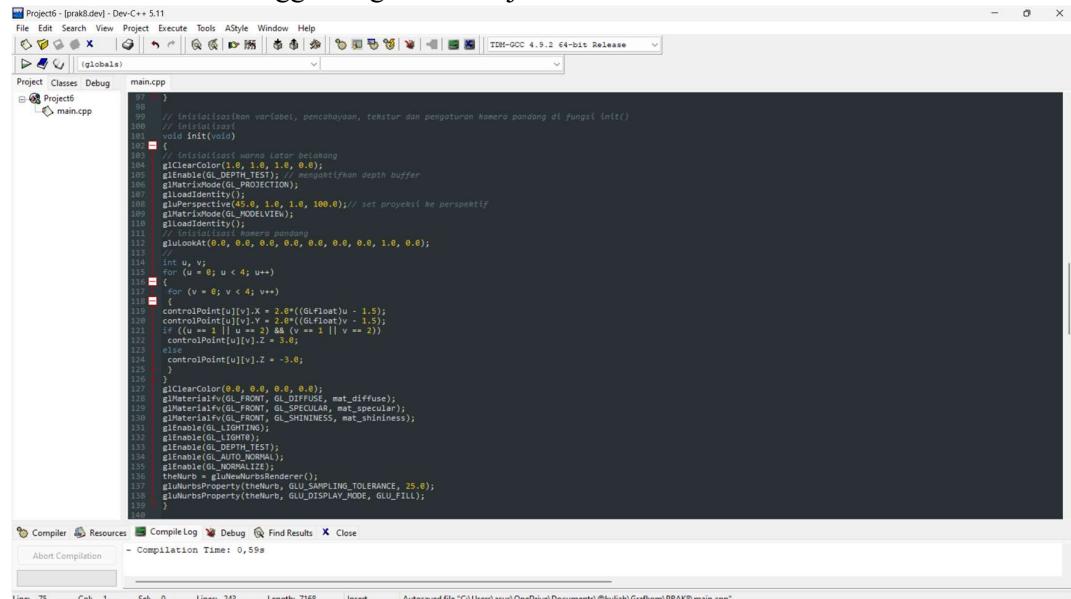
```

Project6 - [prak8&dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TIK-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug main.cpp
Project6
main.cpp
78 }
79 // taruh semua obyek yang akan digambar di fungsi display()
80 void display()
81 {
82     // bersihkan dan reset layar dan buffer
83     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
84     glLoadIdentity();
85
86     // posisikan kamera pandang
87     // dalam hal int sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posz)
88     gluLookAt(posx, posY, posz, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
89
90     // panggil fungsi untuk menggambar obyek
91     drawObject();
92
93     // tampilkan obyek ke layar
94     // gunakan glFlush() bila memakai single buffer
95     // gunakan glutSwapBuffers() bila memakai double buffer
96     glutSwapBuffers();
97 }
98
99 //inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
100 //inisialisasi
101 void init(void)
102 {
103     //inisialisasi warna Latar belakang

```

Gambar 8. 2 Fungsi display LP8

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 91 untuk memanggil fungsi drawObject.



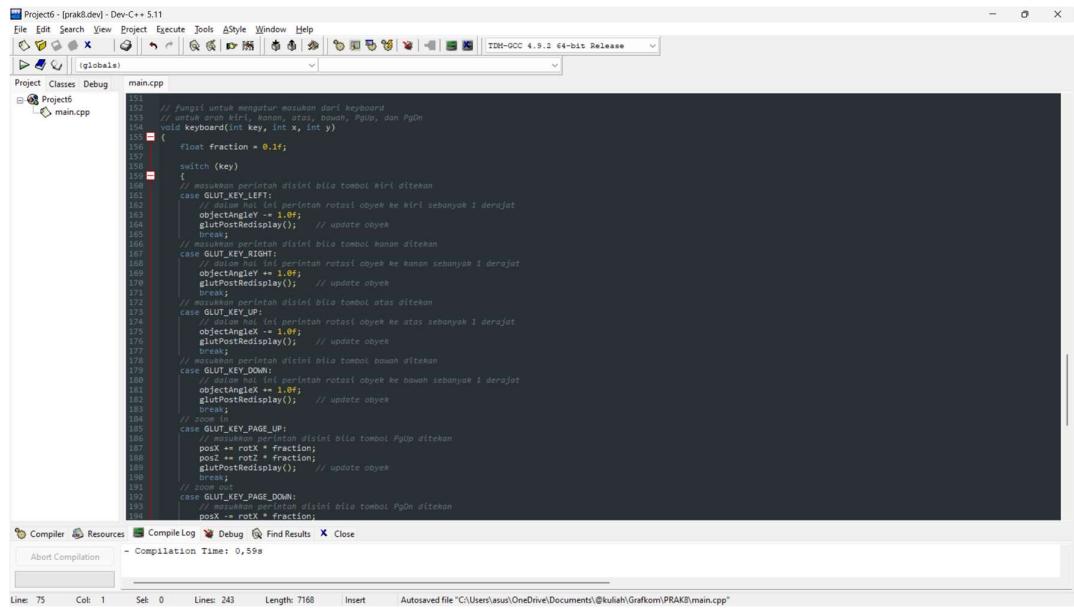
```

Project6 - [prak8&dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TIK-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug main.cpp
Project6
main.cpp
97 }
98 //inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
99 //inisialisasi
100 void init(void)
101 {
102     //inisialisasi warna latar belakang
103     glClearColor(1.0, 1.0, 1.0, 0.0);
104     glEnable(GL_DEPTH_TEST); //mengaktifkan depth buffer
105     glMatrixMode(GL_PROJECTION);
106     glLoadIdentity();
107     gluPerspective(45.0, 1.0, 1.0, 100.0); // set proyeksi ke perspektif
108     glMatrixMode(GL_MODELVIEW);
109     glLoadIdentity();
110     //inisialisasi posisi dan orientasi pandang
111     gluLookAt(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
112
113     //int u, v;
114     // (u = 0; u < 4; u++)
115     //{
116         // for (v = 0; v < 4; v++)
117         //{
118             controlPoint[u][v].X = 2.0*((GLfloat)u - 1.5);
119             controlPoint[u][v].Y = 2.0*((GLfloat)v - 1.5);
120             if ((u == 0 || u == 3) && (v == 0 || v == 3))
121                 controlPoint[u][v].Z = 3.0;
122             else
123                 controlPoint[u][v].Z = -3.0;
124         //}
125     //}
126
127     glClearColor(0.0, 0.0, 0.0, 0.0);
128     glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
129     glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
130     glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
131     glEnable(GL_LIGHTING);
132     glEnable(GL_LIGHT0);
133     glEnable(GL_DEPTH_TEST);
134     glDepthFunc(GL_LESS);
135     glEnable(GL_NORMALIZE);
136     theNurs = glGenNurbsRender();
137     glNurbsProperty(theNurs, GL_SAMPLING_TOLERANCE, 25.0);
138     glNurbsProperty(theNurs, GL_DISPLAY_MODE, GLU_FILL);
139 }


```

Gambar 8. 3 Fungsi init LP8

Fungsi init di gunakan untuk menginisialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init().



```

Project6 - [prak8.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[ ] (globals)
Project Classes Debug main.cpp
Project6 main.cpp
151 // fungsi untuk mengatur masukan dari keyboard
152 // untuk antara PgUp, PgDn, Atas, Bawah, PgUp, dan PgDn
153 void keyboard(int key, int x, int y)
154 {
155     float fraction = 0.1f;
156
157     switch (key)
158     {
159         // masukan periton distel bila tombol kiri ditekan
160         case GLUT_KEY_LEFT:
161             // dalam hal ini periton rotasi objek ke kiri sebanyak 1 derajat
162             objectAngleX += 1.0f;
163             glutPostRedisplay(); // update objek
164             break;
165         // masukan periton distel bila tombol kanan ditekan
166         case GLUT_KEY_RIGHT:
167             // dalam hal ini periton rotasi objek ke kanan sebanyak 1 derajat
168             objectAngleX -= 1.0f;
169             glutPostRedisplay(); // update objek
170             break;
171         // masukan periton distel bila tombol atas ditekan
172         case GLUT_KEY_UP:
173             // dalam hal ini periton rotasi objek ke atas sebanyak 1 derajat
174             objectAngleY -= 1.0f;
175             glutPostRedisplay(); // update objek
176             break;
177         // masukan periton distel bila tombol bawah ditekan
178         case GLUT_KEY_DOWN:
179             // dalam hal ini periton rotasi objek ke bawah sebanyak 1 derajat
180             objectAngleY += 1.0f;
181             glutPostRedisplay(); // update objek
182             break;
183         // zoom in
184         case GLUT_KEY_PAGE_UP:
185             // dalam hal ini periton distel bila tombol PgUp ditekan
186             posx += rotX * fraction;
187             posy += rotY * fraction;
188             posz += rotZ * fraction;
189             glutPostRedisplay(); // update objek
190             break;
191         // zoom out
192         case GLUT_KEY_PAGE_DOWN:
193             // dalam hal ini periton distel bila tombol PgDn ditekan
194             posx -= rotX * fraction;
195             posy -= rotY * fraction;
196             posz -= rotZ * fraction;
197             glutPostRedisplay(); // update objek
198     }
199 }

```

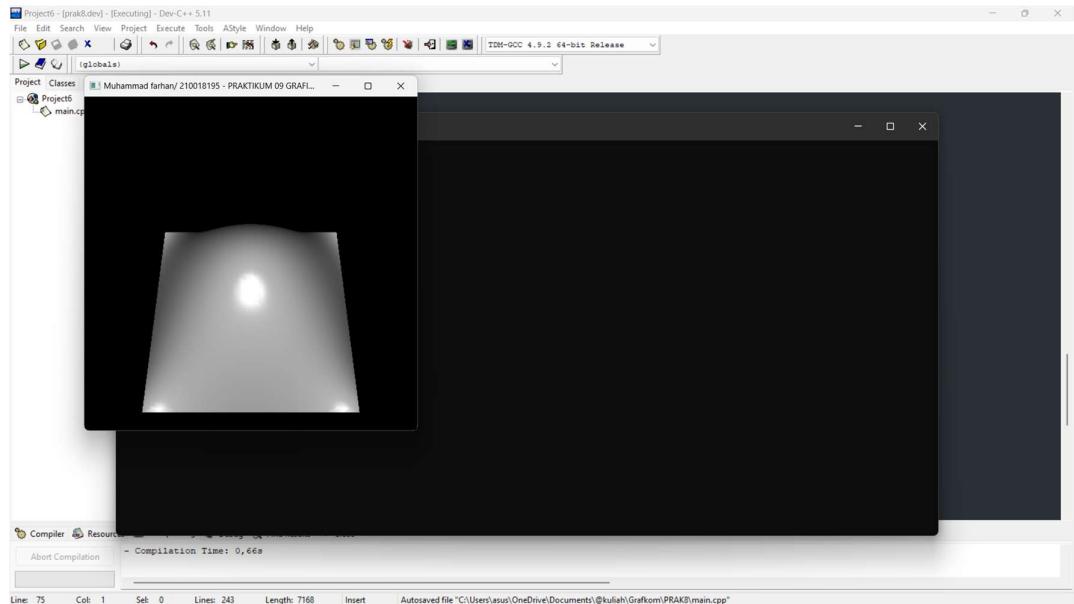
Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,59s

Line: 75 Col: 1 Sel: 0 Lines: 243 Length: 7168 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK8\main.cpp"

Gambar 8. 4 Fungsi keyboard LP8

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.



Gambar 8. 5 Fungsi output LP8

8.3 POSTEST 8

```

Project6 - prak8dev [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools Style Window Help
| D | C | X | (global)
Project Classes Debug main.cpp Makefile.win
Project6 main.cpp
87 // Inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
88 void init(void)
89 {
90     glClearColor(1.0, 1.0, 1.0, 0.0);
91     glEnable(GL_DEPTH_TEST); // Mengaktifkan depth buffer
92     glMatrixMode(GL_PROJECTION);
93     glLoadIdentity();
94     gluPerspective(45.0, 1.0, 1.0, 100.0); // Set proyeksi ke perspektif
95     glMatrixMode(GL_MODELVIEW);
96     glLoadIdentity();
97     gluLookAt(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
98
99     // Mengatur control points untuk permukaan NURBS
100    int u, v;
101    for (u = 0; u < 4; u++)
102    {
103        for (v = 0; v < 4; v++)
104        {
105            controlPoint[u][v].X = 2.0 * ((GLfloat)u - 1.5);
106            controlPoint[u][v].Y = 2.0 * ((GLfloat)v - 1.5);
107            if ((u == 1 || u == 2) && (v == 1 || v == 2))
108                controlPoint[u][v].Z = 10.0;
109            else
110                controlPoint[u][v].Z = -10.0;
111        }
112    }
113
114

```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation - Compilation Time: 0,61s

Line: 109 Col: 42 Sel: 0 Lines: 202 Length: 5953 Insert Done parsing in 0 seconds

Gambar 8. 6 Fungsi init POST8

Hanya merubah pada bagian init, buat program seperti ini

for (u = 0; u < 4; u++)

```

{
    for (v = 0; v < 4; v++)
    {
        controlPoint[u][v].X = 2.0 * ((GLfloat)u - 1.5);
        controlPoint[u][v].Y = 2.0 * ((GLfloat)v - 1.5);
        if ((u == 1 || u == 2) && (v == 1 || v == 2))
            controlPoint[u][v].Z = 10.0;
        else
            controlPoint[u][v].Z = -10.0;
    }
}

```

The screenshot shows the Dev-C++ IDE interface. The main window displays a OpenGL rendering of a yellow cone centered on a black background. The code editor on the right contains C++ code for rendering this cone. The code includes calculations for lighting and shading, such as:

```
.0 * ((GLfloat)u - 1.5);
.0 * ((GLfloat)v - 1.5);
& (v == 1 || v == 2))
= 10.0;

= -10.0;

E, mat_diffuse);
AR, mat_specular);
ESS, mat_shininess);

MPLING_TOLERANCE, 25.0);
SPLAY_MODE, GLU_FILL);

gIVViewport(0, 0, (GLsizei)w, (GLsizei)h);
```

Below the code editor, the status bar shows: Line: 113, Col: 6, Sel: 386, Lines: 202, Length: 5953, Insert, Done parsing in 0 seconds.

Gambar 8. 7 output POST8

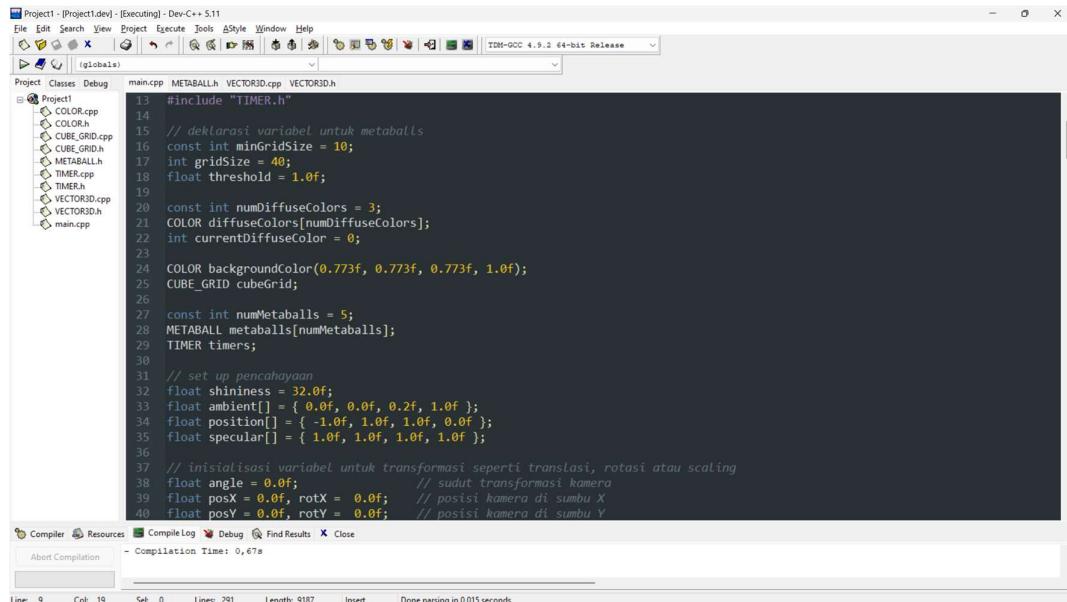
BAB IX TEKNIK

PEMODELAN OBYEK 3D

9.1 PRETEST 9

1. Sebutkan teknik pemodelan 3D yang anda ketahui!
 - a. Triangulasi Poligon
 - b. Subdivisi permukaan poligon
 - c. lofting
2. Jelaskan setiap teknik pemodelan 3D yang anda sebutkan di soal nomor 1!
 - a. Triangulasi poligon adalah proses mengubah poligon dengan lebih dari tiga sisi menjadi kumpulan segitiga.
 - b. Subdivisi permukaan poligon adalah proses membagi setiap poligon dalam model 3D menjadi poligon-poligon yang lebih kecil.
 - c. Lofting adalah teknik yang digunakan untuk membuat bentuk 3D dengan menggabungkan dua atau lebih kurva.

9.2 LANGKAH PRAKTIKUM 9



The screenshot shows the Dev-C++ IDE interface with the following details:

- Project:** Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
- File menu:** File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, Help
- Toolbars:** Standard, Editor, Compiler, Resources, Find Result
- Compiler toolbar:** Compiler, Resources, Compile Log, Debug, Find Result, Close
- Code Editor:** Shows C++ code for initializing metaballs and setting up lighting. The code includes includes for TIMER.h, METABALL.h, COLOR.h, and VECTOR3D.h. It defines variables like minGridSize (10), gridSize (40), threshold (1.0f), numDiffuseColors (3), diffuseColors (array of 3 colors), currentDiffuseColor (0), backgroundColor (0.773f, 0.773f, 0.773f, 1.0f), and cubeGrid. It also initializes numMetaballs (5) and creates an array of METABALL metaballs. The code then sets up lighting parameters: shininess (32.0f), ambient (0.0f, 0.0f, 0.2f, 1.0f), position (-1.0f, 1.0f, 1.0f, 0.0f), and specular (1.0f, 1.0f, 1.0f, 1.0f).

```
#include "TIMER.h"
#include "METABALL.h"
#include "VECTOR3D.h"
#include "COLOR.h"

const int minGridSize = 10;
int gridSize = 40;
float threshold = 1.0f;

const int numDiffuseColors = 3;
COLOR diffuseColors[numDiffuseColors];
int currentDiffuseColor = 0;

COLOR backgroundColor(0.773f, 0.773f, 0.773f, 1.0f);
CUBE_GRID cubeGrid;

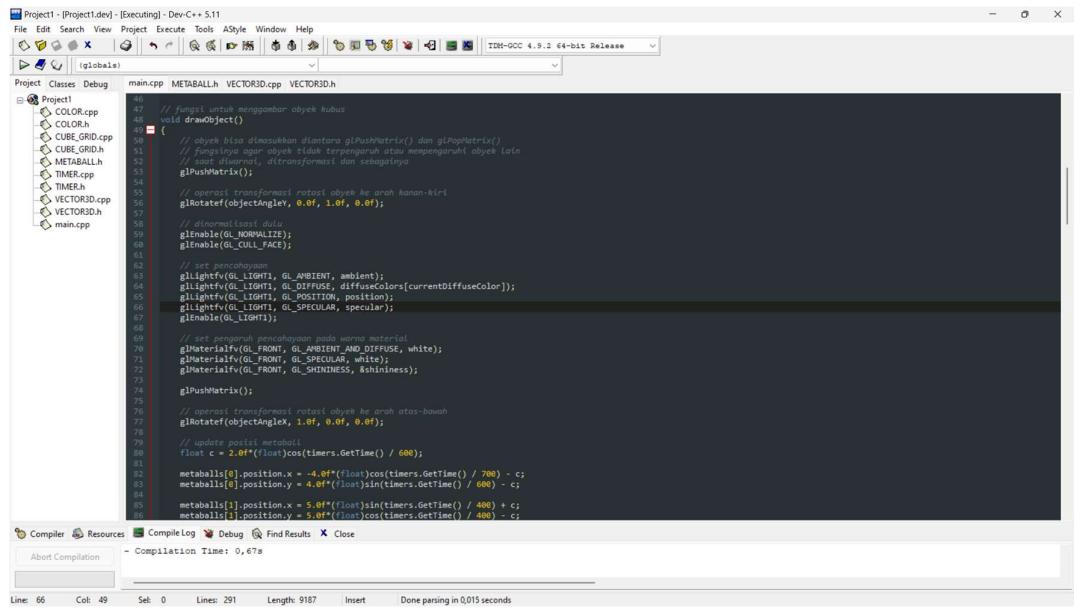
const int numMetaballs = 5;
METABALL metaballs[numMetaballs];
TIMER timers;

float shininess = 32.0f;
float ambient[] = { 0.0f, 0.0f, 0.2f, 1.0f };
float position[] = { -1.0f, 1.0f, 1.0f, 0.0f };
float specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };

// inisialisasi variabel untuk transformasi seperti translasi, rotasi atau scaling
float angle = 0.0f; // sudut transformasi kamera
float posX = 0.0f, rotX = 0.0f; // posisi kamera di sumbu X
float posY = 0.0f, rotY = 0.0f; // posisi kamera di sumbu Y
```
- Compiler Status:** Compilation Time: 0,67s
- Bottom status bar:** Line: 9 Col: 19 Sel: 0 Lines: 291 Length: 9187 Insert Done parsing in 0,015 seconds

Gambar 9. 1 Variable inisiasi Metaball dan cahaya

Line 27 – 35 untuk menginisiasi jumlah metaBall dan Pencahayaan.



```

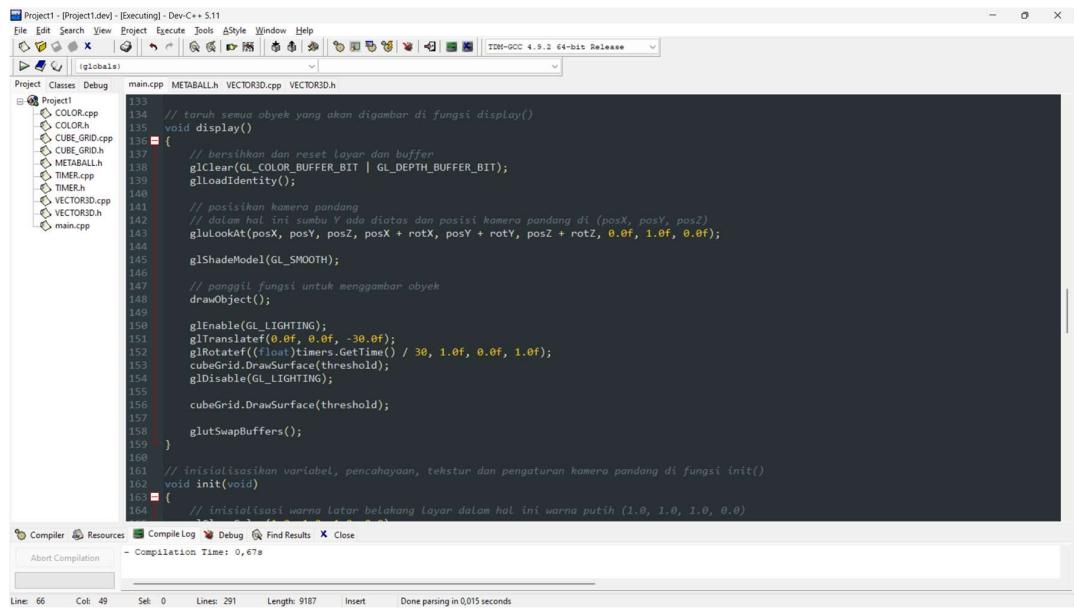
Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glballs]
Project Classes Debug main.cpp METABALLh VECTOR3D.cpp VECTOR3D.h
Project1
  COLOR.cpp
  COLOR.h
  CUBE_GRID.cpp
  CUBE_GRID.h
  METABALLh
  METABALL.h
  TIMER.cpp
  TIMER.h
  VECTOR3D.cpp
  VECTOR3D.h
  main.cpp

46 // fungsi untuk menggambar objek kubus
47 void drawObject()
48 {
49     // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
50     // fungsi ini agar objek tidak terpengaruh atau mempengaruhi objek lain
51     // yang dibuat sebelumnya, ditransformasi dan sebagainya
52     glPushMatrix();
53
54     // operasi transformasi rotasi objek ke arah kanan-kiri
55     glRotatef(objectAngleX, 0.0f, 1.0f, 0.0f);
56
57     // dinormalisasi dulu
58     glEnable(GL_NORMALIZE);
59     glEnable(GL_CULL_FACE);
60
61     // set pencahayaan
62     glLightfv(GL_LIGHT1, GL_AMBIENT, ambient);
63     glLightfv(GL_LIGHT1, GL_DIFFUSE, diffuse);
64     glLightfv(GL_LIGHT1, GL_SPECULAR, specular);
65     glLightfv(GL_LIGHT1, GL_POSITION, position);
66
67     // set pencahayaan pada warna material
68     glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, white);
69     glMaterialfv(GL_FRONT, GL_SPECULAR, white);
70     glMaterialfv(GL_FRONT, GL_SHININESS, shininess);
71
72     glPushMatrix();
73
74     // set pencahayaan pada warna material
75     glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, white);
76     glMaterialfv(GL_FRONT, GL_SPECULAR, white);
77     glMaterialfv(GL_FRONT, GL_SHININESS, shininess);
78
79     // update posisi metaball
80     float c = 0.0f*(float)cos(timers.getTime() / 600);
81
82     metaballs[0].position.x = -4.0f*(float)cos(timers.getTime() / 700) - c;
83     metaballs[0].position.y = 4.0f*(float)sin(timers.getTime() / 600) - c;
84
85     metaballs[1].position.x = 5.0f*(float)sin(timers.getTime() / 400) + c;
86     metaballs[1].position.y = 5.0f*(float)cos(timers.getTime() / 400) + c;
87
88     glPopMatrix();
89
90     // operasi transformasi posisi objek ke arah atas-bawah
91     glRotatef(objectAngleY, 1.0f, 0.0f, 0.0f);
92
93     // update posisi metaball
94     float c = 2.0f*(float)cos(timers.getTime() / 600);
95
96     metaballs[0].position.x = -4.0f*(float)cos(timers.getTime() / 700) - c;
97     metaballs[0].position.y = 4.0f*(float)sin(timers.getTime() / 600) - c;
98
99     metaballs[1].position.x = 5.0f*(float)sin(timers.getTime() / 400) + c;
100    metaballs[1].position.y = 5.0f*(float)cos(timers.getTime() / 400) + c;
101
102    glPopMatrix();
103
104    // taruh semua objek yang akan digambar di fungsi display()
105    void display()
106    {
107        // bersihkan dan reset layar dan buffer
108        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
109        glLoadIdentity();
110
111        // posisikan kamera pandang
112        // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
113        glTranslatef(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
114
115        glShadeModel(GL_SMOOTH);
116
117        // panggil fungsi untuk menggambar objek
118        drawObject();
119
120        glEnable(GL_LIGHTING);
121        glTranslatef(0.0f, 0.0f, -30.0f);
122        glRotatef((float)timers.getTime() / 30, 1.0f, 0.0f, 1.0f);
123        cubeGrid.DrawSurface(threshold);
124        glBindable(GL_LIGHTING);
125
126        cubeGrid.DrawSurface(threshold);
127
128        glutSwapBuffers();
129    }
130
131    // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
132    void init(void)
133    {
134        // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
135    }

```

Gambar 9. 2 Fungsi *drawObject* LP9

Fungsi ini di gunakan untuk menggambar objek yang akan di tampilkan nanti. Line 48 - 132 di gunakan untuk membuat objek.



```

Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glballs]
Project Classes Debug main.cpp METABALLh VECTOR3D.cpp VECTOR3D.h
Project1
  COLOR.cpp
  COLOR.h
  CUBE_GRID.cpp
  CUBE_GRID.h
  METABALLh
  METABALL.h
  TIMER.cpp
  TIMER.h
  VECTOR3D.cpp
  VECTOR3D.h
  main.cpp

133 // taruh semua objek yang akan digambar di fungsi display()
134 void display()
135 {
136     // bersihkan dan reset layar dan buffer
137     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
138     glLoadIdentity();
139
140     // posisikan kamera pandang
141     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
142     glTranslatef(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
143
144     glShadeModel(GL_SMOOTH);
145
146     // panggil fungsi untuk menggambar objek
147     drawObject();
148
149     glEnable(GL_LIGHTING);
150     glTranslatef(0.0f, 0.0f, -30.0f);
151     glRotatef((float)timers.getTime() / 30, 1.0f, 0.0f, 1.0f);
152     cubeGrid.DrawSurface(threshold);
153     glBindable(GL_LIGHTING);
154
155     cubeGrid.DrawSurface(threshold);
156
157     glutSwapBuffers();
158 }
159
160 // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
161 void init(void)
162 {
163     // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
164 }

```

Gambar 9. 3 Fungsi *display* LP9

Fungsi ini digunakan untuk menampilkan semua objek yang di gambar ke layar. Line 148 untuk memanggil fungsi *drawObject*.

```

Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1] [global]
Project Classes Debug main.cpp METABALLh VECTOR3D.cpp VECTOR3D.h
Project1
  COLOR.cpp
  COLOR.h
  CUBE_GRID.cpp
  CUBE_GRID.h
  METABALLh
  METABALL.h
  TIMER.cpp
  TIMER.h
  VECTOR3D.cpp
  VECTOR3D.h
  main.cpp

160 //inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
161
162 void init(void)
163 {
164     //inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
165     glClearColor(1.0, 1.0, 1.0, 0.0);
166     glEnable(GL_DEPTH_TEST); // mengaktifkan depth buffer
167     glMatrixMode(GL_PROJECTION);
168     gluLookAt();
169     gluPerspective(45.0, 1.0, 1.0, 100.0); // set proyeksi ke perspektif
170     glMatrixMode(GL_MODELVIEW);
171     gluLookAt();
172
173     //inisialisasi kamera pandang
174     gluLookAt(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
175
176     //set up grid
177     if (!cubeGrid.CreateMemory())
178         return;
179     if (!cubeGrid.Init(gridSize))
180         return;
181
182     //set up metaballs
183     for (int i = 0; i < numMetaballs; i++)
184         metaballs[i].Init(VECTOR3D(0.0f, 0.0f, 0.0f), 5.0f + float(i));
185
186     //set up warnanya
187     diffuseColors[0].Set(0.345f, 0.843f, 0.902f, 1.0f);
188     diffuseColors[1].Set(0.047f, 0.339f, 0.271f, 1.0f);
189     diffuseColors[2].Set(0.976f, 0.213f, 0.847f, 1.0f);
190
191     timers.Reset();

```

Compiler Resources Compile Log Debug Find Results Close

About Compilation - Compilation Time: 0,67s

Line: 66 Col: 49 Sel: 0 Lines: 291 Length: 9187 Insert Done parsing in 0.015 seconds

Gambar 9. 4 Fungsi init LP9

Fungsi init di gunakan untuk menginisialisasikan variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init().

```

Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Project1] [global]
Project Classes Debug main.cpp METABALLh VECTOR3D.cpp VECTOR3D.h
Project1
  COLOR.cpp
  COLOR.h
  CUBE_GRID.cpp
  CUBE_GRID.h
  METABALLh
  METABALL.h
  TIMER.cpp
  TIMER.h
  VECTOR3D.cpp
  VECTOR3D.h
  main.cpp

285 //mengatur pergerakan objek dengan menggunakan PgUp, PgDn, dan PgMn
286 void keyboard(int key, int x, int y)
287 {
288     float fraction = 0.1f;
289
290     switch (key)
291     {
292         //masukin perintah distin bila tombol kiri ditekan
293         case GLUT_KEY_LEFT:
294             //dalam hal int perintah rotasi objek ke kiri sebanyak 1 derajat
295             objectAngleY -= 1.0f;
296             glutPostRedisplay(); // update objek
297             break;
298
299         //masukin perintah distin bila tombol kanan ditekan
300         case GLUT_KEY_RIGHT:
301             //masukin perintah int rotasi objek ke kanan sebanyak 1 derajat
302             objectAngleY += 1.0f;
303             glutPostRedisplay(); // update objek
304             break;
305
306         //masukin perintah distin bila tombol atas ditekan
307         case GLUT_KEY_UP:
308             //dalam hal int perintah rotasi objek ke atas sebanyak 1 derajat
309             objectAngleX -= 1.0f;
310             glutPostRedisplay(); // update objek
311             break;
312
313         //masukin perintah distin bila tombol bawah ditekan
314         case GLUT_KEY_DOWN:
315             //masukin perintah rotasi objek ke bawah sebanyak 1 derajat
316             objectAngleX += 1.0f;
317             glutPostRedisplay(); // update objek
318             break;
319
320         case GLUT_KEY_PAGE_UP:
321             //masukin perintah distin bila tombol PgUp ditekan
322             pos2 -= rot * fraction;
323             pos2 += rot * fraction;
324             glutPostRedisplay(); // update objek
325             break;
326
327         case GLUT_KEY_PAGE_DOWN:
328             //masukin perintah distin bila tombol PgDn ditekan
329             pos2 += rot * fraction;
330             pos2 -= rot * fraction;
331             glutPostRedisplay(); // update objek
332             break;
333     }
334 }

```

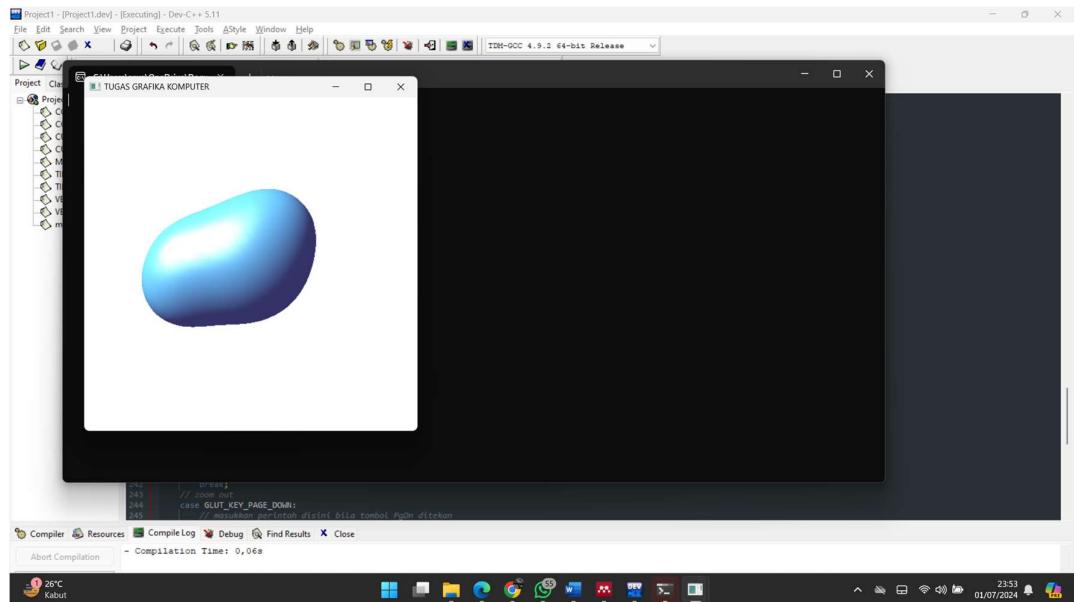
Compiler Resources Compile Log Debug Find Results Close

About Compilation - Compilation Time: 0,67s

Line: 66 Col: 49 Sel: 0 Lines: 291 Length: 9187 Insert Done parsing in 0.015 seconds

Gambar 9. 5 Fungsi keyboard LP9

Fungsi ini di gunakan untuk mengatur pergerakan objek dengan menggunakan keyboard.



Gambar 9. 6 Output LP9

9.3 PRETEST

1. Membuat 2 bola

```

Project1 - [Project1.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[Icons] File Project Classes Debug
Project1
  COLOR.cpp
  COLOR.h
  COLORR.cpp
  COLORR.h
  CUBE3D.cpp
  CUBE3D.h
  CUBE_GRID.cpp
  CUBE_GRID.h
  METABALL.cpp
  METABALL.h
  TIMER.cpp
  TIMER.h
  VECTOR3D.cpp
  VECTOR3D.h
  main.cpp

main.cpp METABALL VECTOR3D.cpp VECTOR3D.h
9 #include "COLOR.h"
10 #include "CUBE_GRID.h"
11 #include "METABALL.h"
12 #include "VECTOR3D.h"
13 #include "TIMER.h"
14
15 // deklarasi const untuk metaballs
16 const int minGridSize = 16;
17 int gridSize = 48;
18 float threshold = 1.0f;
19
20 const int numDiffuseColors = 3;
21 COLOR diffuseColors[numDiffuseColors];
22 int currentDiffuseColor = 0;
23
24 COLOR backgroundColor(0.773f, 0.773f, 0.773f, 1.0f);
25 CUBE_GRID cubeGrid;
26
27 const int numMetaballs = 2;
28 METABALL metaballs[numMetaballs];
29 TIMER timers;
30
31 // set up pencarian
32 float shininess = 32.0f;
33 float ambient[] = { 1.0f, 0.0f, 0.2f, 1.0f };
34 float position[] = { -1.0f, 1.0f, 1.0f, 0.0f };
35 float specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };
36
37 // Inisiasi posisi objek untuk transformasi seperti translasi, rotasi atau scaling
38 float angle = 0.0f; // sudut transformasi objek di sumbu X
39 float posX = 0.0f, rotX = 0.0f; // posisi kamera di sumbu X
40 float posY = 0.0f, rotY = 0.0f; // posisi kamera di sumbu Y
41 float posZ = 5.0f, rotZ = -1.0f; // posisi kamera di sumbu Z
42
43 float objectAnglex = 0.0f; // sudut transformasi objek di sumbu X
44 float objectAngley = 0.0f; // sudut transformasi objek di sumbu Y
45 float objectAnglez = 0.0f; // sudut transformasi objek di sumbu Z
46
47 // fungsi untuk menggambar objek kubus
48 void drawObject()
49 {

```

Gambar 9. 7 Variable inisiasi Metaball dan cahaya POST9

Hanya merubah pada bagan init, buat program seperti ini

```

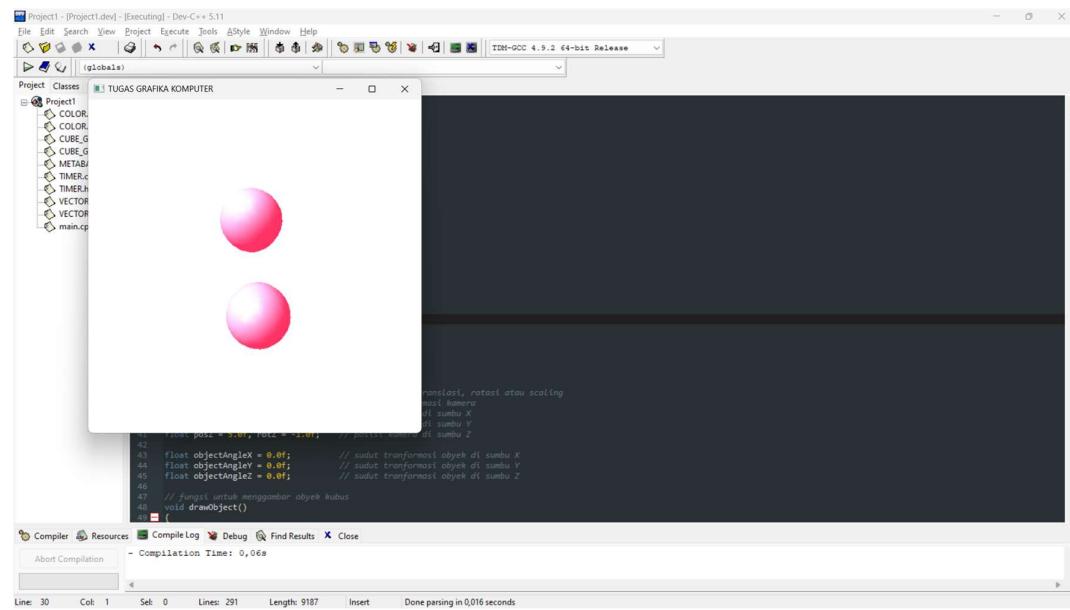
const int numMetaballs = 2;

METABALL metaballs[numMetaballs];

TIMER timers;

// set up pencahayaan
float shininess = 32.0f;
float ambient[] = { 1.0f, 0.0f, 0.2f, 1.0f };
float position[] = { -1.0f, 1.0f, 1.0f, 0.0f };
float specular[] = { 1.0f, 1.0f, 1.0f, 1.0f };

```



Gambar 9. 8 Output POST9

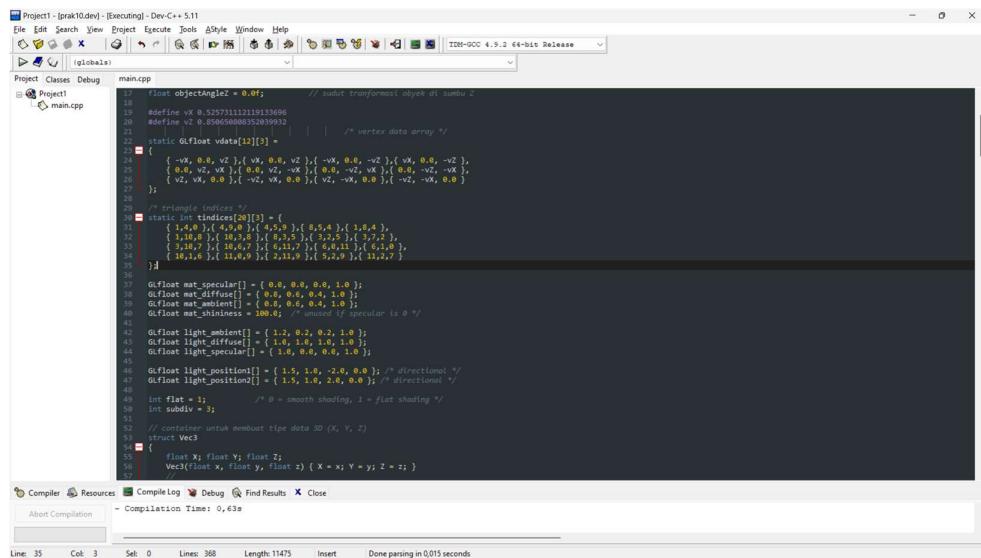
BAB X

TEKNIK SUBDIVISI

10.1 PRETEST 10

1. Bagaimana cara menerapkan subdivisi 2x pada permukaan segitiga?
 - a. Pertama, membagi setiap sisi segitiga menjadi dua segmen dengan menyambungkan titik tengah setiap sisi. Ini akan menciptakan tiga segitiga baru di dalam segitiga asli.
 - b. perluasan segitiga sejajar dengan sisi segitiga awal harus ditambahkan. Untuk melakukan ini, hubungkan setiap titik tengah sisi segitiga baru dengan titik yang sesuai pada sisi segitiga asli.
 - c. mengganti segmen tengah setiap segitiga baru dengan garis yang menghubungkan titik-titik tengah segmen sisi yang bersebelahan. Ini akan membentuk segitiga baru di tengah segitiga yang lebih kecil.
 - d. hapus segitiga asli dan segmen sisinya. Yang tersisa adalah subdivisi 2x dari segitiga awal, yang terdiri dari segitiga-segitiga kecil dengan lebih banyak segmen sisi.

10.2 LANGKAH PRAKTIKUM 10



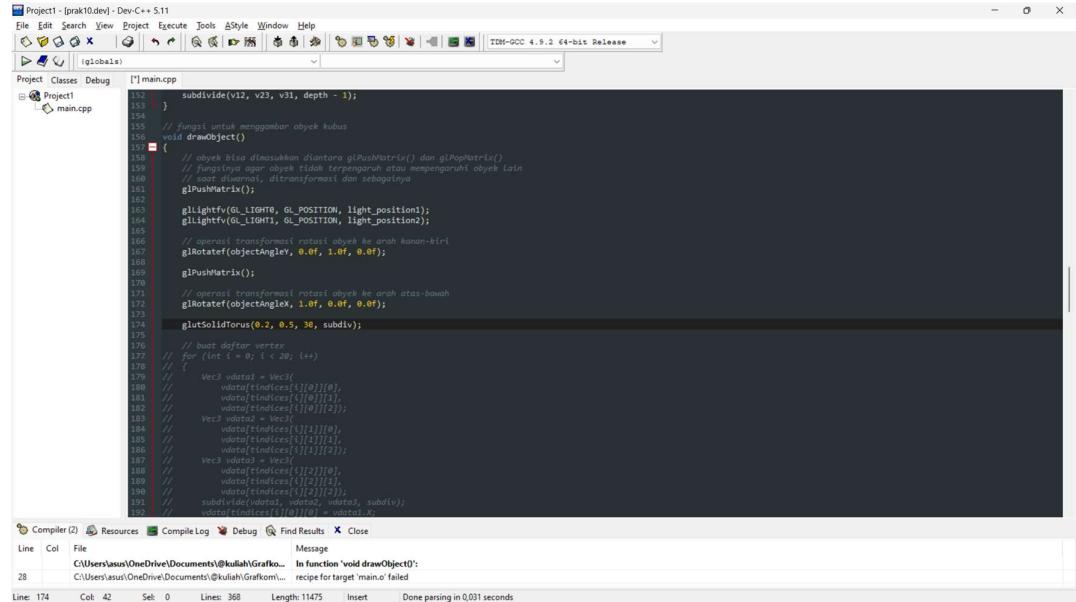
The screenshot shows the Dev-C++ IDE interface with the following details:

- Project:** I-praktik10dev
- File:** main.cpp
- Code Content:**

```
Project: I-praktik10dev [Execution] - Dev-C++ 5.11
File Edit Search View Project Execute Tools Help Window Help
File Project Class Debug main.cpp
Project I-praktik10dev
main.cpp
17 float objectAngleZ = 0.0f; // sudut transformasi objek di sumbu Z
18 #define vX 0.52573112119133696
19 #define vZ 0.85650808852639932
20 static GLfloat vdata[12][3] = { /* vertex data array */
21 {
22     { -vX, 0.0, vZ }, { vX, 0.0, vZ }, { -vX, 0.0, -vZ }, { vX, 0.0, -vZ },
23     { 0.0, vZ, vX }, { 0.0, vZ, -vX }, { 0.0, -vZ, vX }, { 0.0, -vZ, -vX },
24     { vZ, vX, 0.0 }, { -vZ, vX, 0.0 }, { vZ, -vX, 0.0 }, { -vZ, -vX, 0.0 }
25 };
26
27 /* triangle indices */
28 static GLuint indices[10] = {
29     { 1,4,0 }, { 4,1,0 }, { 4,5,0 }, { 8,5,4 }, { 1,8,4 },
30     { 5,10,8 }, { 10,3,8 }, { 6,3,5 }, { 3,2,5 }, { 3,7,2 },
31     { 5,9,7 }, { 10,6,7 }, { 6,11,7 }, { 6,4,11 }, { 6,4,0 },
32     { 10,5,6 }, { 11,8,5 }, { 5,11,9 }, { 5,2,9 }, { 3,11,2 }
33 };
34
35
36 GLfloat mat_specular[] = { 0.0, 0.0, 0.0, 1.0 };
37 GLfloat mat_diffuse[] = { 0.0, 0.0, 0.4, 1.0 };
38 GLfloat mat_ambient[] = { 0.0, 0.0, 0.4, 1.0 };
39 GLfloat mat_shininess = 100.0; /* unused if specular is 0 */
40
41 GLfloat light_ambient[] = { 1.0, 0.2, 0.2, 1.0 };
42 GLfloat light_diffuse[] = { 1.0, 0.2, 0.2, 1.0 };
43 GLfloat light_specular[] = { 1.0, 0.0, 0.0, 1.0 };
44
45 GLfloat light_position[] = { 1.5, 1.0, -2.0, 0.0 }; /* directional */
46 GLfloat light_position2[] = { 1.5, 1.0, 2.0, 0.0 }; /* directional */
47
48 int flat = 1; /* 0 = smooth shading, 1 = flat shading */
49
50 int subdiv = 3;
51
52 // controller untuk membuat type data 3D (X, Y, Z)
53 struct Vec3
54 {
55     float X; float Y; float Z;
56     Vec3(float x, float y, float z) { X = x; Y = y; Z = z; }
57 }
```
- Compiler Status:** Compilation Time: 0,63s
- Text at Bottom:** Line: 35 Col: 3 Sel: 0 Lines: 368 Length: 11475 Insert Done parsing in 0.015 seconds

Gambar 10. 1 Variable inisiasi warna dan jumlah subdiv

Di Gunakan untuk menginisiasi cahaya, warna dan jumlah subdiv awal.

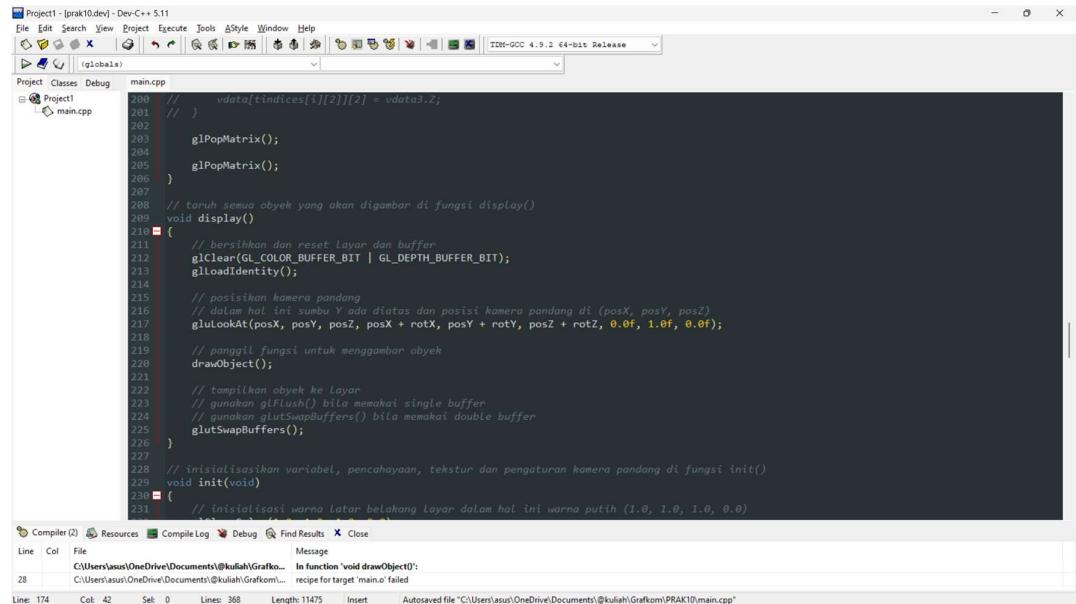


```

Project - prak10.dev - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
File Project Resources Compile Log Debug Find Results Close
Project1 | (globals)
main.cpp | [-] main.cpp
152     } subdivde(v12, v23, v31, depth - 1);
153
154     // fungsi untuk menggambar objek kubus
155     void drawObject()
156     {
157         // obyek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
158         // fungsi ini agar obyek tidak terpengaruh atau mempengaruhi obyek lain
159         // saat dimasuk, ditransformasi dan sebagainya
160         glPushMatrix();
161
162         glLightfv(GL_LIGHT0, GL_POSITION, light_position1);
163         glLightfv(GL_LIGHT1, GL_POSITION, light_position2);
164
165         // operasi transformasi rotasi obyek ke arah kanan-kiri
166         glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
167
168         glPushMatrix();
169
170         // operasi transformasi rotasi obyek ke arah atas-bawah
171         glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
172
173         glutSolidTorus(0.2, 0.5, 30, subdiv);
174
175         // buat daftar vertex
176         for (int i = 0; i < 20; ++i)
177         {
178             Vec3 vdata0 = Vec3(
179                 vdata[tIndices[i][0]][0],
180                 vdata[tIndices[i][0]][1],
181                 vdata[tIndices[i][0]][2]);
182
183             Vec3 vdata2 = Vec3(
184                 vdata[tIndices[i][1]][0],
185                 vdata[tIndices[i][1]][1],
186                 vdata[tIndices[i][1]][2]);
187
188             Vec3 vdata3 = Vec3(
189                 vdata[tIndices[i][2]][0],
190                 vdata[tIndices[i][2]][1],
191                 vdata[tIndices[i][2]][2]);
192
193             subdivide(vdata0, vdata2, vdata3, subdiv);
194             vdata[tIndices[i][0]][0] = vdata3.Z;
195         }
196     }
197
198     // taruh semua obyek yang akan digambar di fungsi display()
199     void display()
200     {
201         // kembalikan dan reset Layar dan buffer
202         glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
203         glLoadIdentity();
204
205         glPopMatrix();
206     }
207
208     // panggil fungsi untuk menggambar obyek
209     drawObject();
210
211     // tampilkan obyek ke layar
212     // gunakan glutFlush() bila memakai single buffer
213     // gunakan glutSwapBuffers() bila memakai double buffer
214     glutSwapBuffers();
215
216     // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
217     void init(void)
218     {
219         // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
220     }
221
222     // jalankan fungsi drawObject()
223     void main()
224     {
225         glutInitDisplayMode(GL_RGB | GL_DEPTH);
226         glutInitWindowSize(800, 600);
227         glutCreateWindow("Praktikum PRAK10");
228         glutDisplayFunc(display);
229         glutIdleFunc(drawObject);
230         glutMainLoop();
231     }

```

Gambar 10. 2 Fungsi drawObject LP10



```

Project - prak10.dev - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
File Project Resources Compile Log Debug Find Results Close
Project1 | (globals)
main.cpp | [-] main.cpp
200     // vdata[tIndices[i][2]][2] = vdata3.Z;
201
202     glPopMatrix();
203
204     glPopMatrix();
205 }
206
207 // taruh semua obyek yang akan digambar di fungsi display()
208 void display()
209 {
210     // kembalikan dan reset Layar dan buffer
211     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
212     glLoadIdentity();
213
214     // posisikan kamera pandang
215     // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
216     // gunakan glutLookAt() bila memakai single buffer
217     // gunakan glutSwapBuffers() bila memakai double buffer
218     glutLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
219
220     // panggil fungsi untuk menggambar obyek
221     drawObject();
222
223     // tampilkan obyek ke layar
224     // gunakan glutFlush() bila memakai single buffer
225     // gunakan glutSwapBuffers() bila memakai double buffer
226     glutSwapBuffers();
227
228     // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
229     void init(void)
230     {
231         // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
232     }
233
234     // jalankan fungsi drawObject()
235     void main()
236     {
237         glutInitDisplayMode(GL_RGB | GL_DEPTH);
238         glutInitWindowSize(800, 600);
239         glutCreateWindow("Praktikum PRAK10");
240         glutDisplayFunc(display);
241         glutIdleFunc(drawObject);
242         glutMainLoop();
243     }

```

Gambar 10. 3 fungsi display LP10

```

Project1 - [prak10.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools A Style Window Help
[Project1 | main.cpp | (globals)]
Project Classes Debug main.cpp
200 // vData[tIndices[i][j][2]] = vData3.z;
201 // }
202 glPopMatrix();
203 glPopMatrix();
204 }
205 // taruh semua objek yang akan digambar di fungsi display()
206 void display()
207 {
208 // bersihkan dan reset layar dan buffer
209 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
210 gluLookAt(posX, posY, posZ, posX + rotX, posY + rotY, posZ + rotZ, 0.0f, 1.0f, 0.0f);
211 // posisikan kamera pandang
212 // dalam hal ini sumbu Y ada diatas dan posisi kamera pandang di (posX, posY, posZ)
213 // panggil fungsi untuk menggambar objek
214 drawObject();
215 // tampilkan objek ke layar
216 // gunakan glFlush() bila memakai single buffer
217 // gunakan glutSwapBuffers() bila memakai double buffer
218 glutSwapBuffers();
219 }
220 // inisialisasi variabel, pencahayaan, tekstur dan pengaturan kamera pandang di fungsi init()
221 void init(void)
222 {
223 // inisialisasi warna latar belakang layar dalam hal ini warna putih (1.0, 1.0, 1.0, 0.0)
224

```

Compiler (2) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... In function 'void drawObject()';
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... recipe for target main.o failed

Line 174 Col: 42 Sek: 0 Lines: 368 Length: 11475 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK10\main.cpp"

Gambar 10. 4 Fungsi init LP10

```

Project1 - [prak10.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools A Style Window Help
[Project1 | main.cpp | (globals)]
Project Classes Debug main.cpp
722 // fungsi untuk menerima masukan dari keyboard
723 // untuk arah kiri, kanan, atas, bawah, PgUp, dan PgDn
724 void keyboard(int key, int x, int y)
725 {
726 float fraction = 0.1f;
727 switch (key)
728 {
729 // masukkan perintah disini bila tombol kiri ditekan
730 case GLUT_KEY_LEFT:
731 // dalam hal ini perintah rotasi objek ke kiri sebanyak 1 derajat
732 objectAngleY -= 1.0f;
733 glutPostRedisplay(); // update objek
734 break;
735 // masukkan perintah disini bila tombol kanan ditekan
736 case GLUT_KEY_RIGHT:
737 // dalam hal ini perintah rotasi objek ke kanan sebanyak 1 derajat
738 objectAngleY += 1.0f;
739 glutPostRedisplay(); // update objek
740 break;
741 // masukkan perintah disini bila tombol atas ditekan
742 case GLUT_KEY_UP:
743 // dalam hal ini perintah rotasi objek ke atas sebanyak 1 derajat
744 objectAngleX -= 1.0f;
745 glutPostRedisplay(); // update objek
746 break;
747 // masukkan perintah disini bila tombol bawah ditekan
748 case GLUT_KEY_DOWN:
749 // dalam hal ini perintah rotasi objek ke bawah sebanyak 1 derajat
750 objectAngleX += 1.0f;
751 glutPostRedisplay(); // update objek
752 break;

```

Compiler (2) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... In function 'void drawObject()';
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... recipe for target main.o failed

Line 174 Col: 42 Sek: 0 Lines: 368 Length: 11475 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK10\main.cpp"

Gambar 10. 5 Fungsi Keyboard LP10

```

Project1 - [prak10.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glut] [globals]
Project Classes NAMA / NIM - PRAKTIKUM 10 GRAFIKA KOMPUTER
Project1 main.cpp
Keyboard PgUp, dan PgDn

    col_kiri ditekan
    obyek ke kiri sebanyak 1 derajat
    > obyek

    col_kanan ditekan
    obyek ke kanan sebanyak 1 derajat
    > obyek

    col_atas ditekan
    obyek ke atas sebanyak 1 derajat
    > obyek

    col_bawah ditekan
    // masukkan perintah disini bila tombol bawah ditekan
    case GLUT_KEY_DOWN:
        // dalam hal ini perintah rotasi obyek ke bawah sebanyak 1 derajat
        objectAngleX += 1.0f;
        glutPostRedisplay(); // update obyek
    break;

Compiler (3) Resources Compile Log Debug Find Result Close
Line Col File Message
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... [Error] Id returned 1 exit status
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... recipe for target 'prak10.exe' failed
25 Line: 174 Col: 42 Sek: 0 Lines: 368 Length: 11475 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK10\main.cpp"

```

Gambar 10. 6 Output1 LP10

```

Project1 - [prak10.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
[glut] [globals]
Project Classes NAMA / NIM - PRAKTIKUM 10 GRAFIKA KOMPUTER
Project1 main.cpp
Keyboard PgUp ditekan
Keyboard PgUp ditekan

    obyek ke kanan sebanyak 1 derajat
    > obyek

    col_atas ditekan
    obyek ke atas sebanyak 1 derajat
    > obyek

    col_bawah ditekan
    obyek ke bawah sebanyak 1 derajat
    > obyek

    tombol PgUp ditekan
    tombol PgUp ditekan

    obyek
    tombol PgDn ditekan

    // masukkan perintah disini bila tombol PgDn ditekan
    case GLUT_KEY_UP:
        posX -= rotX * fraction;
        posZ -= rotZ * fraction;
        glutPostRedisplay(); // update obyek
    break;

Compiler (3) Resources Compile Log Debug Find Result Close
Line Col File Message
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... [Error] Id returned 1 exit status
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... recipe for target 'prak10.exe' failed
25 Line: 174 Col: 42 Sek: 0 Lines: 368 Length: 11475 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK10\main.cpp"

```

Gambar 10. 7 Output2 LP10

```

Project1 - [prak10.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes main.cpp
NAMA / NIM - PRAKTIKUM 10 GRAFIKA KOMPUTER
Project1
objek ke kanan sebanyak 1 derajat
objek
objek atas ditekan
objek ke atas sebanyak 1 derajat
objek
objek bawah ditekan
objek ke bawah sebanyak 1 derajat
objek
tombol PgUp ditekan
tombol PgUp ditekan
objek
tombol PgDn ditekan
masukkan perintah disini bila tombol PgDn ditekan
rotX = rotX * fraction;
rotZ = rotZ * fraction;
glutPostRedisplay(); // update objek
case GLUT_KEY_F1:
    subdivs++;
}

155 // fungsi untuk menggambar objek kubus
156 void drawObject()
157 {
    // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
    // fungisinya agar objek tidak terpengaruh atau mempengaruhi objek lain
    // saat diwartakan, ditransformasi dan sebagainya
    glPushMatrix();
    glLightfv(GL_LIGHT0, GL_POSITION, light_position1);
    glLightfv(GL_LIGHT1, GL_POSITION, light_position2);
    // operasi transformasi rotasi objek ke arah kanan-kiri
    glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
    glPushMatrix();
    // operasi transformasi rotasi objek ke arah atas-bawah
    glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
    glutSolidCone(1.5, 1.0, 30, subdiv);
    // buat daftar vertex
    for (int i = 0; i < 20; i++)
    {
        Vec3 vdata0 = Vec3(
            vdata[tindices[i][0]][0],
            vdata[tindices[i][0]][1],
            vdata[tindices[i][0]][2]);
        Vec3 vdata02 = Vec3(
            vdata[tindices[i][1][0]],
            vdata[tindices[i][1][1]],
            vdata[tindices[i][1][2]]);
    }
}

```

Compiler (3) Resources Compile Log Debug Find Results Close

Line Col File Message

C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... [Error] Id returned 1 exit status
C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom... recipe for target 'prak10.exe' failed

Line: 302 Col: 14 Sel: 0 Lines: 368 Length: 11475 Insert Autosaved file "C:\Users\asus\OneDrive\Documents\@kuliah\Grafkom\PRAK10\main.cpp"

Gambar 10. 8 Output3 LP10

10.3 POSTEST 10

1. Buat Caping

```

Project1 - [prak10.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug main.cpp
Project1
155 // fungsi untuk menggambar objek kubus
156 void drawObject()
157 {
    // objek bisa dimasukkan diantara glPushMatrix() dan glPopMatrix()
    // fungisinya agar objek tidak terpengaruh atau mempengaruhi objek lain
    // saat diwartakan, ditransformasi dan sebagainya
    glPushMatrix();
    glLightfv(GL_LIGHT0, GL_POSITION, light_position1);
    glLightfv(GL_LIGHT1, GL_POSITION, light_position2);
    // operasi transformasi rotasi objek ke arah kanan-kiri
    glRotatef(objectAngleY, 0.0f, 1.0f, 0.0f);
    glPushMatrix();
    // operasi transformasi rotasi objek ke arah atas-bawah
    glRotatef(objectAngleX, 1.0f, 0.0f, 0.0f);
    glutSolidCone(1.5, 1.0, 30, subdiv);
    // buat daftar vertex
    for (int i = 0; i < 20; i++)
    {
        Vec3 vdata0 = Vec3(
            vdata[tindices[i][0]][0],
            vdata[tindices[i][0]][1],
            vdata[tindices[i][0]][2]);
        Vec3 vdata02 = Vec3(
            vdata[tindices[i][1][0]],
            vdata[tindices[i][1][1]],
            vdata[tindices[i][1][2]]);
    }
}

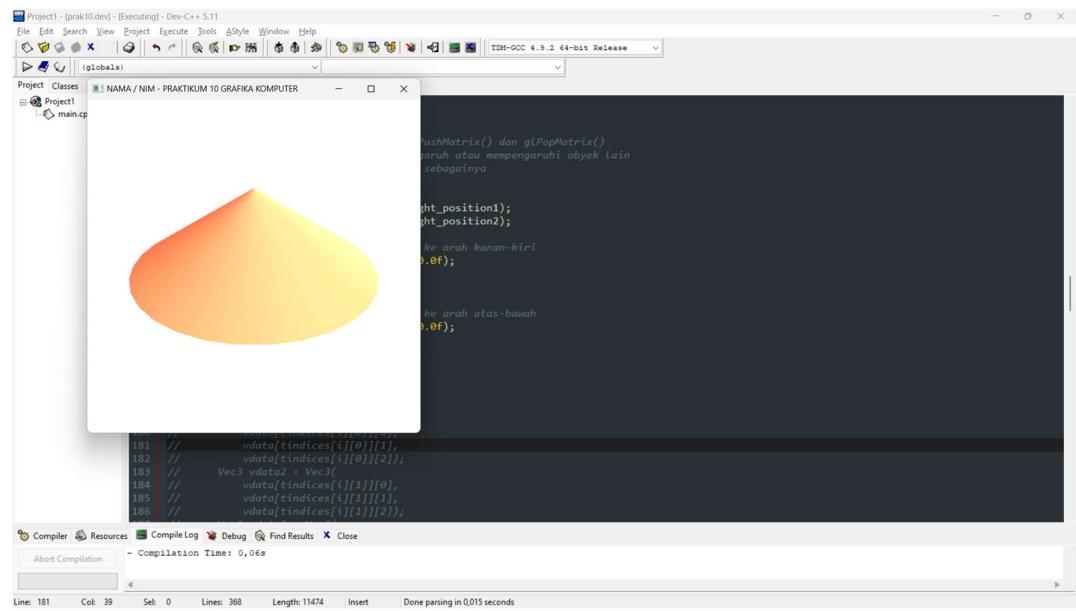
```

Compiler Resources Compile Log Debug Find Results Close

Aabort Compilation - Compilation Time: 0,06s

Line: 181 Col: 39 Sel: 0 Lines: 368 Length: 11474 Insert Done parsing in 0,015 seconds

Gambar 10. 9 Fungsi drawObject POST10



Gambar 10. 10 Output POST10

DAFTAR PUSTAKAN

- Karisma, H. (2013). Pengenalan OpenGL. *Unikom Repository*.
[https://repository.unikom.ac.id/41687/1/Pengenalan OpenGL di DEVC.pdf](https://repository.unikom.ac.id/41687/1/Pengenalan%20OpenGL%20di%20DEVC.pdf)
- Prahara, A., Si, S., Cs, M., Azhari, A., Kom, S., Eng, M., Si, S., Kom, M., Rahani, F. F., Si, S., & Cs, M. (n.d.). *Grafika komputer*.
- S.P. Baviskar, "DDA Line Drawing Algorithm Using MATLAB," International Journal of Innovative Research in Science, Engineering and Technology, vol. 4, no. 5, 2015.
- K. R. Anvekar, "Drawing lines using DDA algorithm: A graphical analysis," Journal of Computer Applications, vol. 2, no. 2, 2018.
- Smith, J., & Johnson, A. (2016). "Linear interpolation techniques in data analysis." Journal of Scientific Computing, 25(2), 123-135.
- Brown, R., & Lee, C. (2017). "Application of linear interpolation in financial modeling." Journal of Finance and Economics, 12(4), 321-333.
- Miller, D., & Wilson, P. (2019). "Applications of cubic spline interpolation in engineering." Engineering Applications, 15(1), 81-95.
- <https://www.opengl.org/resources/libraries/glut/spec3/node83.html>
- https://www.khronos.org/opengl/wiki/Polygon_Offset_and_Point_and_Lines