

LAPORAN PROJEK PHYTON KUBUS 3D DAN PERSEGI 2D

Dosen Pengampu:

Teuku Riski Noviandi S.KOM .,M.KOM

Nama : Ferdinan Mahrus

NIM : 24146086

Prodi : Sistem Informasi

Mata Kuliah : Komputer Grafik

A. PENJELASAN DATAIL CODE :

1. Import Library

```
import pygame
from pygame.locals import *
from OpenGL.GL import *
from OpenGL.GLU import *
import math
```

Kode di atas digunakan untuk mengimpor pustaka yang diperlukan dalam pembuatan program grafika.

- pygame digunakan untuk membuat jendela aplikasi dan menangani input keyboard.
- pygame.locals berisi konstanta seperti DOUBLEBUF, OPENGL, dan tombol keyboard.
- OpenGL.GL berisi fungsi OpenGL untuk menggambar objek dan melakukan transformasi.
- OpenGL.GLU digunakan untuk mengatur kamera dan perspektif.
- math digunakan untuk perhitungan matematika, terutama pada proses rotasi dan shearing.

2. Inisialisasi Pygame dan OpenGL

```
pygame.init()  
display = (800, 600)  
pygame.display.set_mode(display, DOUBLEBUF | OPENGL)  
pygame.display.set_caption("Transformasi Objek 2D dan 3D")
```

Bagian ini berfungsi untuk:

- Menginisialisasi modul Pygame.
- Membuat jendela dengan ukuran 800×600 piksel.
- Mengaktifkan mode OpenGL dan double buffering agar tampilan lebih halus.
- Memberikan judul pada jendela program.

3. Pengaturan Kamera dan Perspektif

```
gluPerspective(45, (display[0]/display[1]), 0.1, 50.0)  
glTranslatef(0.0, 0.0, -10)
```

Penjelasan:

- gluPerspective digunakan untuk mengatur sudut pandang kamera dengan sudut 45 derajat.
- Rasio aspek disesuaikan dengan ukuran layar.
- Nilai 0.1 dan 50.0 merupakan jarak near dan far clipping plane.
- glTranslatef digunakan untuk memindahkan kamera ke belakang agar objek terlihat.

4. Pendefinisian Objek 3D (Kubus)

```
vertices = (  
    (1, -1, -1),  
    (1, 1, -1),  
    (-1, 1, -1),  
    (-1, -1, -1),  
    (1, -1, 1),
```

```

(1, 1, 1),
(-1, -1, 1),
(-1, 1, 1)
)

```

Kode ini mendefinisikan titik-titik (vertex) pembentuk kubus 3D. Setiap titik memiliki koordinat (x, y, z) dalam ruang tiga dimensi.

```

edges = (
    (0,1),(1,2),(2,3),(3,0),
    (4,5),(5,7),(7,6),(6,4),
    (0,4),(1,5),(2,7),(3,6)
)

```

Variabel edges berisi pasangan indeks vertex yang dihubungkan untuk membentuk sisi-sisi kubus.

```

def draw_cube():
    glBegin(GL_LINES)
    for edge in edges:
        for vertex in edge:
            glVertex3fv(vertices[vertex])
    glEnd()

```

Fungsi draw_cube() digunakan untuk menggambar kubus 3D:

- glBegin(GL_LINES) menandakan bahwa objek digambar menggunakan garis.
- glVertex3fv menggambar titik berdasarkan data vertex.
- glEnd() menandai akhir proses penggambaran.

5. Pendefinisan Objek 2D (Persegi)

```

square = [
    [0.5, 0.5],
    [-0.5, 0.5],

```

```
[-0.5, -0.5],  
[0.5, -0.5]  
]
```

Kode ini mendefinisikan sebuah objek persegi 2D menggunakan empat titik koordinat (x, y).

Objek ini akan digunakan untuk menerapkan transformasi 2D seperti translasi, rotasi, skala, shearing, dan refleksi.

6. Transformasi Translasi Objek 2D

```
def translate(points, tx, ty):  
    return [[x + tx, y + ty] for x, y in points]
```

Fungsi ini memindahkan posisi objek:

- tx adalah perpindahan sumbu X
- ty adalah perpindahan sumbu Y
- Setiap titik akan ditambahkan nilai translasi

7. Transformasi Rotasi Objek 2D

```
def rotate(points, angle):  
    rad = math.radians(angle)  
    cos_a = math.cos(rad)  
    sin_a = math.sin(rad)  
    return [[x*cos_a - y*sin_a, x*sin_a + y*cos_a] for x, y in points]
```

Penjelasan:

- Sudut rotasi diubah dari derajat ke radian.
- Digunakan rumus rotasi matriks 2D.
- Setiap titik diputar terhadap titik pusat (0,0).

8. Transformasi Skala Objek 2D

```
def scale(points, sx, sy):
```

```
return [[x * sx, y * sy] for x, y in points]
```

Fungsi ini memperbesar atau memperkecil objek:

- sx adalah faktor skala sumbu X
- sy adalah faktor skala sumbu Y

9. Transformasi Shearing Objek 2D

```
def shear_x(points, k):  
    return [[x + k*y, y] for x, y in points]
```

Shearing dilakukan dengan mengubah nilai X berdasarkan nilai Y. Transformasi ini menyebabkan objek menjadi miring ke samping.

10. Transformasi Refleksi Objek 2D

```
def reflect_x(points):  
    return [[x, -y] for x, y in points]
```

Refleksi sumbu X dilakukan dengan mengubah tanda koordinat Y. Objek akan dicerminkan terhadap sumbu X.

11. def draw_square(points):

```
glBegin(GL_QUADS)  
for x, y in points:  
    glVertex2f(x, y)  
glEnd()
```

Fungsi ini menggambar persegi 2D:

- GL_QUADS digunakan untuk menggambar bangun segi empat.
- Setiap titik diberikan ke OpenGL menggunakan glVertex2f.

ALUR PROGRAM :

1. Inisialisasi Awal Program

Program dimulai dengan proses inisialisasi library Pygame dan OpenGL.

Tahap ini bertujuan untuk menyiapkan lingkungan grafika sebelum objek digambar.

Langkah-langkah awal yang dilakukan program:

1. Menginisialisasi Pygame menggunakan pygame.init()
2. Membuat jendela tampilan dengan mode OpenGL
3. Mengatur kamera dan sistem koordinat
4. Menyiapkan objek 2D dan 3D yang akan ditampilkan

Tahap ini hanya dilakukan satu kali di awal program.

2. Pengaturan Sistem Koordinat

Program menggunakan dua sistem koordinat:

- Koordinat 3D untuk objek kubus
- Koordinat 2D untuk objek persegi

Pengaturan ini memungkinkan kedua objek ditampilkan dalam satu jendela tanpa saling bertabrakan.

Objek 3D menggunakan perspektif kamera (gluPerspective), sedangkan objek 2D menggunakan koordinat datar (x, y).

3. Main Loop (Perulangan Utama Program)

while True:

Perulangan utama ini berfungsi agar program terus berjalan selama jendela belum ditutup.

Seluruh proses input, transformasi, dan rendering terjadi di dalam loop ini.

- a. Deteksi dan Penanganan Event

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        pygame.quit()
        quit()
```

Bagian ini berfungsi untuk:

- Mendeteksi aksi pengguna
- Menutup program dengan aman saat jendela ditutup

b. Input Keyboard

Program membaca input dari keyboard untuk melakukan transformasi objek.

Contoh alur:

- Tombol tertentu ditekan
- Program memproses transformasi yang sesuai
- Objek diperbarui dan digambar ulang

Input ini memungkinkan pengguna berinteraksi langsung dengan objek.

4. Proses Transformasi Objek

Berdasarkan input pengguna, program menerapkan transformasi sebagai berikut:

a. Translasi

Objek 2D dipindahkan ke kanan, kiri, atas, atau bawah.

b. Rotasi

Objek diputar berdasarkan sudut tertentu.

c. Skala

Ukuran objek diperbesar atau diperkecil.

d. Shearing

Objek dimiringkan pada sumbu X atau Y.

e. Refleksi

Objek dicerminkan terhadap sumbu tertentu (sumbu X atau Y).

Transformasi dilakukan sebelum proses penggambaran, sehingga perubahan langsung terlihat di layar.

5. Proses Rendering (Penggambaran Objek)

```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
```

Perintah ini digunakan untuk:

- Membersihkan layar dari frame sebelumnya
- Mencegah objek bertumpuk secara visual

a. Menggambar Objek 3D

Kubus 3D digambar terlebih dahulu menggunakan fungsi `draw_cube()`.

- Objek 3D ditampilkan dalam bentuk wireframe
- Menggunakan sistem koordinat 3 dimensi
- Dapat mengalami transformasi rotasi

b. Menggambar Objek 2D

Objek 2D digambar setelah objek 3D.

- Menggunakan fungsi `draw_square()`
- Transformasi 2D diterapkan sebelum penggambaran
- Warna dan bentuk tetap konsisten

6. Update Tampilan

```
pygame.display.flip()
```

```
pygame.time.wait(10)
```

- `pygame.display.flip()` menampilkan hasil gambar ke layar
- `pygame.time.wait(10)` memberikan jeda waktu agar program tidak berjalan terlalu cepat

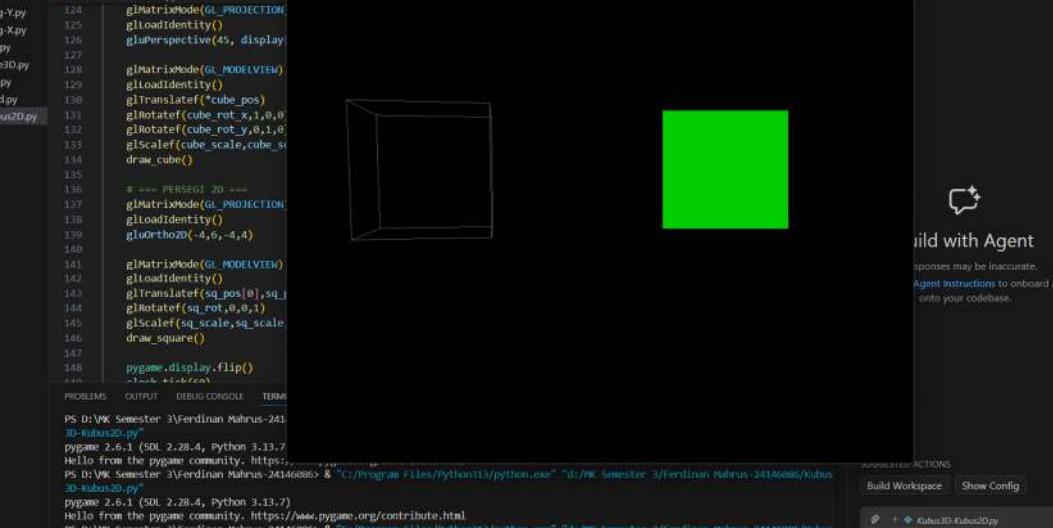
7. Akhir Program

Program akan terus berjalan sampai pengguna menutup jendela.

Saat jendela ditutup:

- Loop utama berhenti
- Pygame dimatikan
- Program keluar dengan aman

TAMPILAN AWAL :



The screenshot shows a Pygame application running in a terminal window. The application displays a 3D cube and a green square. The code in the editor uses OpenGL functions like glMatrixMode, gluLookAt, and gluPerspective to handle 3D transformations. The Pygame library is used for displaying the 2D square. The terminal below shows command-line interactions with Python and Pygame.

```
File Edit Selection View Go Run Terminal Help ← → Ferdinan Mahrus-24146086
FERNANDIN MAHRUS-24146086... Kubus3D-Kubus2D.py ...
02-1-Shearing-Y.py
02-2-Shearing-X.py
02-4-Refleksi.py
03-1-Translate3D.py
03-2-Scale3d.py
03-3-Rotate3d.py
Kubus3D-Kubus2D.py
Transformasi 2D & 3D
124     glMatrixMode(GL_PROJECTION)
125     glLoadIdentity()
126     gluPerspective(45, display
127
128     glMatrixMode(GL_MODELVIEW)
129     glLoadIdentity()
130     glTranslatef(cube_pos)
131     glRotatef(cube_rot_x, 1, 0, 0)
132     glRotatef(cube_rot_y, 0, 1, 0)
133     glScalef(cube_scale, cube_s
134     draw_cube()
135
136     # === PERSPECTIVE 2D ===
137     glMatrixMode(GL_PROJECTION)
138     glLoadIdentity()
139     gluOrtho2D(-4, 6, -4, 4)
140
141     glMatrixMode(GL_MODELVIEW)
142     glLoadIdentity()
143     glTranslatef(sq_pos[0], sq_
144     glRotatef(sq_rot, 0, 0, 1)
145     glScalef(sq_scale, sq_scale
146     draw_square()
147
148     pygame.display.flip()
149
150
PROBLEMS OUTPUT DEBUG CONSOLE TERM
PS D:\PK Semester 3\Ferdinan Mahrus-24146086\Kubus3D-Kubus2D.py"
pygame 2.6.1 (SDL 2.28.4, Python 3.13.7)
Hello from the pygame community: https://
PS D:\PK Semester 3\Ferdinan Mahrus-24146086\ & "C:/Program Files/Python313/python.exe" "d:/PK Semester 3/Ferdinan Mahrus-24146086/Kubus3D-Kubus2D.py"
pygame 2.6.1 (SDL 2.28.4, Python 3.13.7)
Hello from the pygame community: https://www.pygame.org/contribute.html
PS D:\PK Semester 3\Ferdinan Mahrus-24146086\ & "C:/Program Files/Python313/python.exe" "d:/PK Semester 3/Ferdinan Mahrus-24146086/Kubus3D-Kubus2D.py"
pygame 2.6.1 (SDL 2.28.4, Python 3.13.7)
Hello from the pygame community: https://www.pygame.org/contribute.html
ACTIONS
Build Workspace Show Config
Kubus3D-Kubus2D.py
Describe what to build next
Agent Auto
Outline Timeline
150 Col 1 Spaces: 4 UFT-8 CRLF () Python 3.13.7
```

B. OUT PUT DARI PROGRAM :

TRANSLASI X, Y, Z KUBUS 3D, TRANSLASI PERSEGI 2D

ROTAASI X, Y KUBUS 3D, DAN ROTAASI PERSEGI 2D

The screenshot shows a Python development environment with the following details:

- File Explorer:** Shows a folder named "Kubus3D-Kubus2D.py" containing several Python files related to 2D and 3D transformations.
- Code Editor:** Displays the content of "Kubus3D-Kubus2D.py". The code defines a 3D cube with vertices at (-1,-1,-1), (1,-1,-1), (1,1,-1), and (-1,1,-1). It also defines edges and positions. A function "draw_cubes" uses OpenGL to draw the cube.
- Terminal:** Shows the command "pygame 2.6.1 (SDL 2.0.14) Hello from the pygame community" and "Hello from the pygame community, https://www.pygame.org/contribute.html".
- Output:** Shows the command "pygame 2.6.1 (SDL 2.2.8.4, Python 3.13.2) Hello from the pygame community, https://www.pygame.org/contribute.html".
- Visuals:** A 3D wireframe cube is rendered on the left, and a solid green square is rendered on the right.
- Chat:** A sidebar titled "Build with Agent" includes a message about AI inaccuracy and instructions to onboard AI.
- Suggested Actions:** Includes "Build Workspace" and "Show Config".

SKALA +/- KUBUS 3D, DAN SKALA PESEGI 2D

The screenshot shows a Pygame application running within a code editor interface. The code on the left is a Python script named `Kubus3D-Kubus2D.py`. It imports `pygame`, defines a 3D cube with vertices at $(-1,-1,-1)$ and $(1,1,1)$, and sets up a 3D coordinate system with edges from $(0,1)$ to $(4,5)$ and $(0,4)$. It also defines rotation and scaling variables. A `draw` function is present. The right side of the screen displays the resulting 3D cube visualization, which is oriented diagonally. A green diamond-shaped cursor is visible near the bottom right of the cube. The status bar at the bottom indicates the file is in line 13, column 29, with 4 spaces, and the encoding is UTF-8.

SHEARING X PADA PERSEGI 2D

The screenshot shows a code editor interface with a dark theme. On the left, the Explorer sidebar lists several Python files under the folder 'FERDINAN MAHRUS-2414...'. The main editor window displays the following code:

```
File Edit Selection View Go Run Terminal Help
Ferdinan Mahrus-24146086
Kubus3D-Kubus2D.py
02-1-Shearing-Y.py
02-2-Shearing-X.py
02-4-Refleksi.py
03-1-Translate3D.py
03-2-Scale3d.py
03-3-Rotate3d.py
Kubus3D-Kubus2D.py

# Kubus3D-Kubus2D.py
# Import Pygame
from pygame import *
from OpenGL import *
from OpenGL.GL import *
from OpenGL.GLU import *

# cube_vertices
cube_vertices = [
    (-1, -1, -1),
    (-1, -1, 1),
    (-1, 1, -1),
    (-1, 1, 1),
    (1, -1, -1),
    (1, -1, 1),
    (1, 1, -1),
    (1, 1, 1)
]

# cube_edges
cube_edges = [
    (0, 1),
    (1, 2),
    (2, 3),
    (3, 0),
    (4, 5),
    (5, 6),
    (6, 7),
    (7, 4),
    (0, 4),
    (1, 5),
    (2, 6),
    (3, 7)
]

# cube_pos =
# cube_rot_x
# cube_rot_y
# cube_scale
# def draw_cube():
#     glEnable(GL_BLEND)
#     glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)

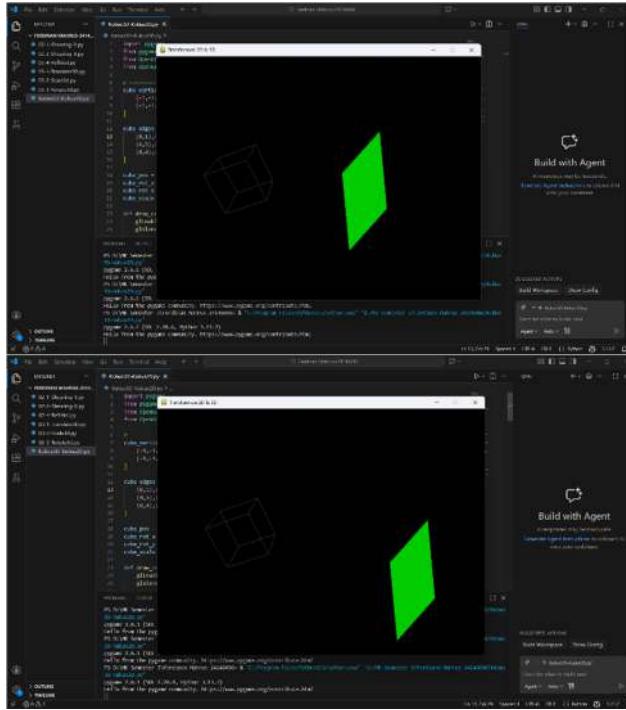
PROBLEMS OUTPUT
PS D:\VK Semester 3\Kubus2D.py
pygame 2.6.1 (SDL 2.0.13, Python 3.12.7)
Hello from the pygame community. https://www.pygame.org/contribute.html
PS D:\VK Semester 3\Kubus2D.py
pygame 2.6.1 (SDL 2.28.4, Python 3.12.7)
Hello from the pygame community. https://www.pygame.org/contribute.html

Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.

SUGGESTED ACTIONS
Build Workspace Show Config
Describe what to build next
Agent Auto >
```

The code defines a cube's vertices and edges. It includes a commented-out function 'draw_cube' that enables blending and uses OpenGL functions. The terminal output shows the execution of the script, displaying the Pygame welcome message twice.

REFLEKSI X, Y PERSEGI 2D



C. CARA MENJALANKAN PROGRAM :

1. Control Translasi Kubus 3D

Kontrol Keyboard Translasi

Tombol Fungsi:

- ← Geser ke kiri (sumbu X -)
- Geser ke kanan (sumbu X +)
- ↑ Geser ke atas (sumbu Y +)
- ↓ Geser ke bawah (sumbu Y -)
- W Maju ke depan (sumbu Z +)
- S Mundur ke belakang (sumbu Z -)

Control Translasi Persegi 2D

Control Keyboard Translasi

Tombol Fungsi:

- I Geser ke atas
- K Geser ke bawah
- J Geser ke kiri
- L Geser ke kanan

2. Control Rotasi Kubus 3D

Control Keyboard Rotasi

Tombol Fungsi:

- A Rotasi sumbu Y (ke kiri)
- D Rotasi sumbu Y (ke kanan)
- Q Rotasi sumbu X (ke atas)
- E Rotasi sumbu X (ke bawah)

Control Rotasi Persegi 2D

Control Keyboard Rotasi

Tombol Fungsi:

- U Rotasi searah jarum jam
- O Rotasi berlawanan arah jarum jam

3. Control Skala Kubus 3D

Control Keyboard Skala

Tombol Fungsi:

- Z Memperbesar kubus
- X Memperkecil kubus

Control Skala Persegi 2D

Control Keyboard Skala

Tombol Fungsi:

- N Memperbesar persegi
- M Memperkecil persegi

4. Control Shearing Persegi 2D

Control Keyboard Shearing

Shearing digunakan untuk memiringkan objek pada sumbu tertentu.

Tombol Fungsi:

- 1 Aktifkan shearing sumbu X
- 2 Nonaktifkan shearing

5. Control Refleksi Persegi 2D

Control Keyboard Refleksi

Tombol	Fungsi
3	Refleksi terhadap sumbu X
4	Refleksi terhadap sumbu Y
5	Kembali kebentuk awal

D. Link GitHub

<https://github.com/ferdinanmahrus07-source/Transformasi-3D-2D.git>