Tugas MK. Grafika Komputer Terapan TA 2023-2024

Tugas Besar

Transformasi Objek 2D - Tempat Pemakaman Umum

Ditulis Oleh

Muhammad Yazid – 152022192
Deden Fahrul Roziqin - 152022182

Kelompok : 15

Tgl. Penugasan : Rabu, 6 Desember 2023
Tgl. Penyerahan : Rabu, 24 Desember 2023

Prodi Informatika
Fakultas Teknologi Industri
Institut Teknologi Nasional
Bandung

2023

KATA PENGANTAR

Puji syukur kehadirat Allah SWT, Tuhan Yang Maha Kuasa, atas limpahan rahmat, taufik, dan hidayah-Nya yang telah menyertai penulis dalam menyelesaikan penyusunan laporan ini. Laporan ini disusun sebagai bagian dari pemenuhan tugas Mata Kuliah IFB-201 Grafika Komputer Terapan, yang mengambil fokus pada penjelasan mengenai Transformasi Objek 2 Dimensi. Penyusunan laporan ini bertujuan utama untuk mendalamkan pemahaman mengenai Transformasi Objek 2 Dimensi. Dengan merinci setiap aspeknya, diharapkan pembaca dapat memperoleh wawasan yang lebih komprehensif mengenai bagaimana Transformasi Objek 2 Dimensi bekerja. Melalui analisis yang mendalam, laporan ini bertujuan memberikan kontribusi pada pemahaman yang lebih luas tentang konsep tersebut.

Tentunya, ucapan terima kasih sebesar-besarnya kami sampaikan kepada Ibu Theta Dinnarwaty Putri S.Kom. M.T., selaku dosen Mata Kuliah Grafika Komputer Terapan, atas arahan, bimbingan, dan kesempatan yang diberikan sehingga penulis dapat mengeksplorasi dan memahami lebih dalam mengenai Transformasi Objek 2 Dimensi.

Semoga laporan ini dapat memberikan manfaat dan pemahaman yang lebih mendalam mengenai Transformasi Objek 2 Dimensi. Akhir kata, Kami sadar akan kemungkinan kesalahan dan kekurangan dalam laporan ini, oleh karena itu, kami memohon maaf atas segala kekhilafan. Kritik dan saran dari pembaca sangat diharapkan guna perbaikan di masa mendatang.

Bandung, 24 Desember 2023

Deden Fahrul

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1. Latar belakang

Transformasi objek, sebagai konsep fundamental dalam dunia pemrograman, mencakup perubahan posisi, rotasi, dan skala suatu objek dalam ruang dua atau tiga dimensi. Kemampuan ini memberikan kekuatan kepada pengembang untuk menciptakan efek visual yang dinamis, meningkatkan interaktivitas pengguna, dan memberikan sentuhan artistik pada aplikasi atau perangkat lunak yang dikembangkan. Dengan menggunakan prinsip-prinsip matematis, terutama matriks transformasi, transformasi objek memungkinkan pengubahan koordinat objek untuk menciptakan perubahan yang diinginkan, baik secara estetis maupun fungsional.

Pentingnya konsep matematis dalam implementasi transformasi objek tidak bisa diabaikan. Pemahaman mendalam tentang aljabar linear dan geometri komputasional menjadi landasan untuk merancang dan menyusun transformasi objek secara efisien. Meskipun proyek ini penuh potensi, tantangan-tantangan mungkin muncul dalam menghadapi kompleksitas perhitungan matematis atau memastikan optimalisasi performa. Strategi solusi yang efektif diterapkan untuk mengatasi hambatan-hambatan ini akan menjadi sorotan dalam laporan ini.

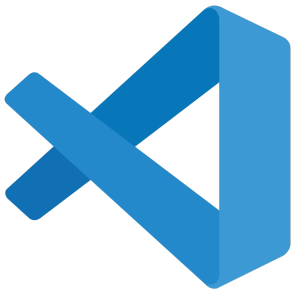
Proyek ini juga menggali potensi penerapan hasilnya dalam berbagai konteks pemrograman, dari aplikasi desktop hingga pengembangan web. Melalui pemahaman dan eksplorasi mendalam terhadap transformasi objek dalam pemrograman, laporan ini bertujuan untuk memberikan wawasan yang komprehensif dan mendalam tentang konsep tersebut serta memberikan kontribusi terhadap pengembangan aplikasi yang lebih baik di masa depan.

BAB II

PEMBAHASAN

2.1. Software yang digunakan

Visual Studio Code (VS Code) adalah editor sumber terbuka yang dikembangkan oleh Microsoft. Sebagai editor kode, VS Code dirancang untuk memberikan pengalaman pengembangan yang efisien dan fleksibel. Kelebihan utamanya meliputi ringan dan responsif, membuatnya dapat berjalan lancar di berbagai sistem, serta mendukung banyak bahasa pemrograman seperti JavaScript, Python, Java, dan lainnya. Salah satu fitur unggulannya adalah ekosistem ekstensi yang kaya, memungkinkan pengguna menyesuaikan fungsionalitas dan penampilan editor sesuai kebutuhan proyek. Dukungan terintegrasi untuk Git mempermudah manajemen versi, sementara alat debugging terintegrasi membantu pengembang melacak dan memperbaiki bug. Terminal terintegrasi memungkinkan pengguna menjalankan perintah shell langsung dari editor, meningkatkan produktivitas. VS Code juga menyediakan linting dan pemeriksaan kode untuk membantu pengembang mengidentifikasi potensi masalah dalam kode mereka. Dengan integrasi yang baik dengan layanan cloud, seperti Azure, serta ketersediaan pada berbagai platform, Visual Studio Code terus menjadi pilihan utama bagi pengembang perangkat lunak yang mencari kombinasi kinerja, kegunaan, dan fleksibilitas.



Gambar 3.1.1 Visual Studio Code.

2.2. Bahasa Pemrograman

Python adalah bahasa pemrograman tingkat tinggi yang terkenal karena sintaksisnya yang bersih, mudah dibaca, dan mudah dipelajari. Diciptakan oleh Guido van Rossum pada tahun 1991, Python dirancang untuk mempromosikan kode yang ekspresif dan efisien. Keunikan Python terletak pada penggunaan indentasi sebagai metode untuk menandai blok kode, yang memberikan kejelasan visual dan mengurangi kebutuhan akan tanda kurung atau kurawal tambahan. Bahasa ini mendukung tipe data dinamis dan abstraksi tingkat tinggi, menghilangkan keharusan deklarasi tipe variabel secara eksplisit. Python juga dikenal karena serbagunanya, dengan kemampuan digunakan dalam berbagai konteks seperti pengembangan perangkat lunak, web, kecerdasan buatan, dan analisis data. Ekosistem Python yang kaya, termasuk perpustakaan dan kerangka kerja seperti NumPy, Pandas, Flask, dan TensorFlow, memperkaya fungsionalitasnya dan memungkinkan pengembang untuk dengan cepat membangun solusi kompleks. Selain itu, Python memiliki portabilitas tinggi dan dapat dijalankan di berbagai platform. Dengan komunitas pengembang yang besar dan aktif, Python menjadi pilihan utama baik untuk pemula yang baru memasuki dunia pemrograman maupun untuk profesional yang mencari efisiensi dan fleksibilitas dalam pengembangan perangkat lunak.

Gambar 3.2.1 Python.

BAB III

IMPLEMENTASI

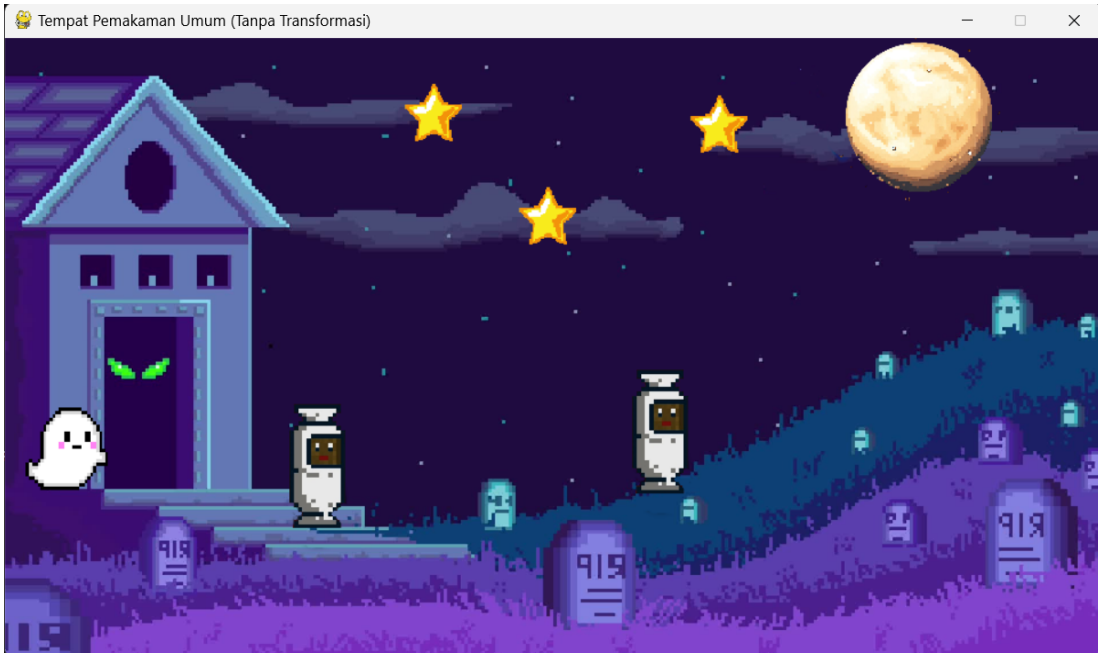
3.1. Penjelasan source code sebelum transformasi

Kode	Penjelasan
`import pygame`	Mengimpor modul `pygame`, yang digunakan untuk pengembangan game dengan Python.
`import sys`	Mengimpor modul `sys`, yang menyediakan akses ke beberapa variabel dan fungsi yang memengaruhi interpreter Python.
`pygame.init()`	Menginisialisasi modul pygame.
`window_size = (960, 540)`	Menetapkan ukuran jendela permainan.
`screen = pygame.display.set_mode(window_size)`	Membuat jendela permainan dengan ukuran yang telah ditentukan.
`pygame.display.set_caption("Tempat Pemakaman Umum (Tanpa Transformasi)")`	Menetapkan judul jendela permainan.
`def load_background(background_image)`	Fungsi untuk memuat gambar latar belakang dan menyesuaikannya dengan ukuran jendela.
`def load_bintang(bintang_image)`	Fungsi untuk memuat gambar bintang dan menyesuaikan ukurannya.

Kode	Penjelasan
`def load_pocong(pocong_image)`	Fungsi untuk memuat gambar pocong dan menyesuaikan ukurannya.
`def load_pocong_rf(pocong_image)`	Fungsi untuk memuat gambar pocong dan melakukan refleksi sumbu X.
`def load_jurig(jurig_image)`	Fungsi untuk memuat gambar jurig dan menyesuaikan ukurannya.
`def load_bulan(bulan_image)`	Fungsi untuk memuat gambar bulan dan menyesuaikan ukurannya.
`def game_loop()`	Fungsi utama yang mengandung logika permainan.
`jurig_speed = 0`	Menentukan kecepatan translasi objek hantu. Diatur dengan nilai 0 agar tidak terjadi translasi.
`while True:`	Loop utama permainan yang berjalan selama True (permainan berjalan).
`for event in pygame.event.get():`	Loop untuk menangani peristiwa yang terjadi selama permainan (misalnya, menutup jendela).
`screen.blit(background, (0, 0))`	Menampilkan latar belakang pada jendela permainan.
`screen.blit(bintang, (600, 50))`	Menampilkan bintang pada posisi tertentu pada jendela.
`screen.blit(pocong, (200, 300))`	Menampilkan pocong pada posisi tertentu pada jendela.

Kode	Penjelasan
`screen.blit(jurig, (jurig_x, 300))`	Menampilkan jurig pada posisi tertentu pada jendela.
`screen.blit(rotated_bulan, bulan_rect.topleft)`	Menampilkan bulan pada posisi tertentu pada jendela.
`angle += 0`	Sudut rotasi bulan = 0 agar tidak terjadi transformasi pada objek bulan
`pygame.display.flip()`	Memperbarui layar permainan.
`clock.tick(30)`	Menetapkan batas FPS (Frame Per Second) permainan.
`if __name__ == "__main__":`	Mengeksekusi fungsi `game_loop()` jika file dieksekusi langsung (bukan diimpor sebagai modul).
`game_loop()`	Memanggil fungsi utama untuk menjalankan permainan.

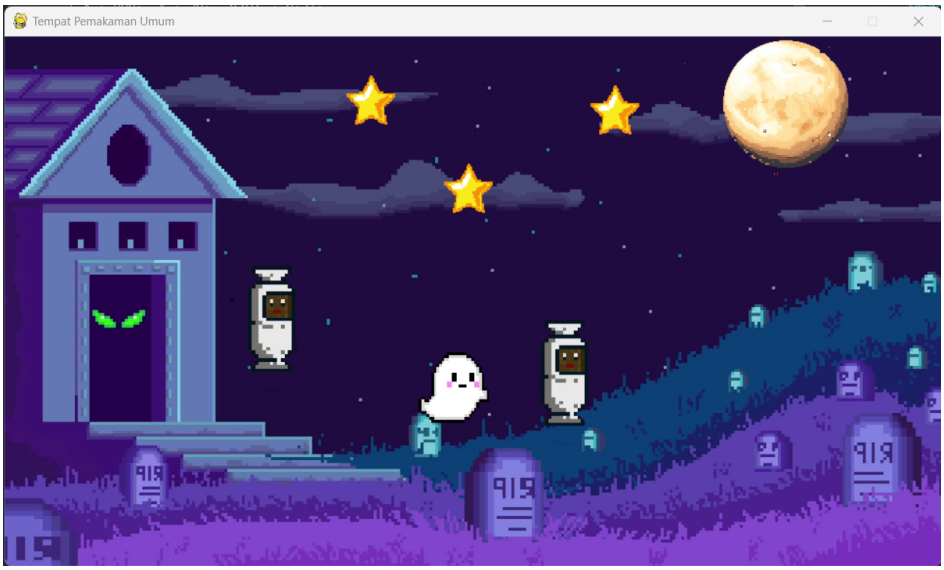
3.2. Output hasil sebelum transformasi



Gambar 3.2.1 Objek sebelum Transformasi.

3.3. Penjelasan source code setelah transformasi

3.3.1. Translasi



Gambar 3.3.1.1 Translasi objek pocong dan hantu.

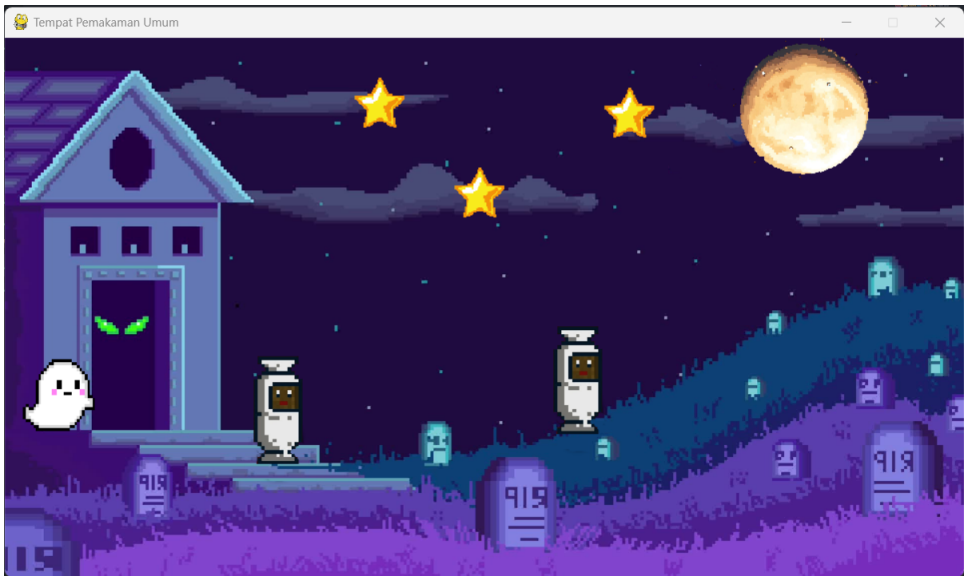
Disini kami mengaplikasikan transformasi translasi pada objek hantu (bergerak ke kanan) dan pocong (loncat). Berikut merupakan tabel penjelasan codingan yang kami buat.

Kode	Penjelasan
`screen.blit(pocong, (pocong_x, pocong_y))`	Menaruh (menggambar) gambar yang direpresentasikan oleh `pocong` pada koordinat yang ditentukan `(pocong_x, pocong_y)` di layar.
`if jumping_up:`	Memeriksa apakah karakter saat ini sedang dalam fase meloncat ke atas.
`pocong_y -= pocong_speed`	Mengurangi nilai koordinat y karakter untuk membuatnya bergerak ke atas.
`if pocong_y <= 200:`	Memeriksa apakah karakter telah mencapai puncak loncatan (koordinat y kurang dari atau sama dengan 200).
`jumping_up = False`	Menetapkan `jumping_up` menjadi False, menunjukkan bahwa karakter sekarang berada dalam fase jatuh.
`else:`	Dijalankan jika karakter sedang dalam fase jatuh.
`pocong_y += pocong_speed`	Menambah nilai koordinat y karakter untuk membuatnya bergerak ke bawah.
`if pocong_y >= 300:`	Memeriksa apakah karakter telah mencapai bagian bawah loncatan (koordinat y lebih dari atau sama dengan 300).
`jumping_up = True`	Menetapkan `jumping_up` menjadi True, menunjukkan bahwa karakter sekarang berada dalam fase meloncat ke atas.
`def load_jurig(jurig_image):`	Fungsi untuk memuat gambar jurig. Parameter `jurig_image` adalah path gambar jurig.

`jurig = pygame.image.load(jurig_image)`	Memuat gambar jurig menggunakan Pygame.
`jurig = pygame.transform.scale(jurig, (100, 100))`	Mengubah ukuran gambar jurig menjadi 100x100 piksel.
`return jurig`	Mengembalikan gambar jurig yang sudah dimuat dan diubah ukurannya.
`jurig_x = 0`	Inisialisasi posisi horizontal awal jurig.
`jurig_reset_pos = window_size[0] + 50`	Inisialisasi posisi reset jurig_x setelah melewati batas. `window_size[0]` adalah lebar layar.
`jurig_speed = 3`	Inisialisasi kecepatan translasi jurig.
`screen.blit(jurig, (jurig_x, 400))`	Menampilkan gambar jurig pada layar di posisi `(jurig_x, 400)`.
`jurig_x += jurig_speed`	Menambahkan kecepatan translasi pada posisi horizontal jurig.
`if jurig_x > jurig_reset_pos:`	Pengecekan apakah jurig melewati batas posisi reset.

`jurig_x = -jurig.get_width()`	Jika ya, reset posisi horizontal jurig ke posisi awal sebelum layar.

3.3.2. Rotasi



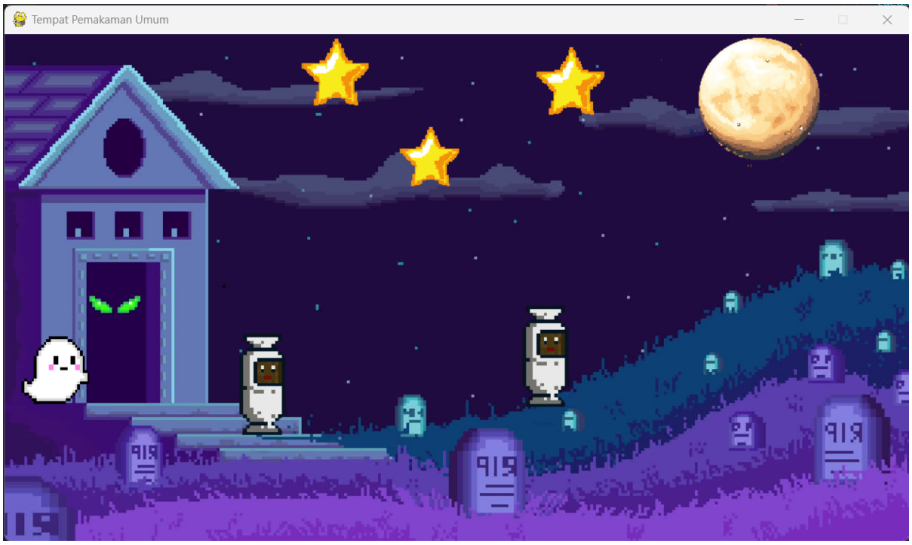
Gambar 3.3.2.1 Rotasi pada objek bulan.

Untuk transformasi rotasi, kami mengaplikasikannya pada objek bulan (rotasi berlawanan arah jarum jam). Berikut merupakan tabel penjelasan codingan yang kami buat.

Kode	Penjelasan
`def load_bulan(bulan_image):`	Deklarasi fungsi `load_bulan` dengan satu parameter yaitu `bulan_image`, yang berisi path gambar bulan.
`bulan = pygame.image.load(bulan_image)`	Memuat gambar bulan dari path yang diberikan menggunakan fungsi `pygame.image.load` dan menyimpannya dalam variabel `bulan`.

`bulan = pygame.transform.scale(bulan, (200, 200))`	Mengubah skala gambar bulan menjadi (200, 200) menggunakan fungsi `pygame.transform.scale` dan menyimpannya kembali dalam variabel `bulan`.
`return bulan`	Mengembalikan gambar bulan yang telah dimodifikasi (dengan skala yang diubah).
`bulan_image = "bulan.png"`	Variabel `bulan_image` menetapkan path atau nama file gambar bulan ("bulan.png").
`bulan = load_bulan(bulan_image)`	Memanggil fungsi `load_bulan` dengan menggunakan variabel `bulan_image` sebagai argumen untuk memuat dan memodifikasi gambar bulan.
`rotated_bulan = pygame.transform.rotate(bulan, angle)`	Menghasilkan gambar bulan yang telah dirotasi sejauh `angle` derajat menggunakan fungsi `pygame.transform.rotate`.
`bulan_rect = rotated_bulan.get_rect(center=(800, 70))`	Mendapatkan objek Rect yang digunakan untuk menentukan posisi dan ukuran gambar bulan setelah dirotasi.
`screen.blit(rotated_bulan, bulan_rect.topleft)`	Menampilkan gambar bulan yang telah dirotasi pada posisi yang dihitung berdasarkan objek Rect `bulan_rect`.
`angle += 1`	Menambah nilai `angle` sebesar 1 untuk merotasi gambar bulan pada iterasi selanjutnya.

3.3.3. Skala



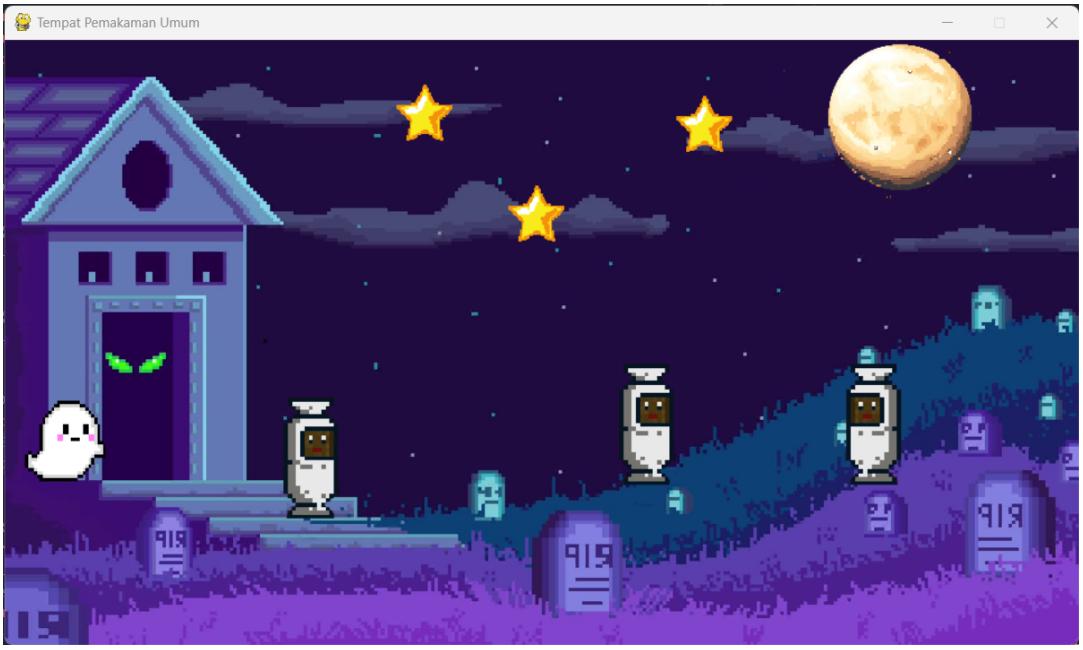
Gambar 3.3.3.1 Skala pada objek bintang.

Untuk transformasi skala, kami mengaplikasikannya pada objek bintang (skala membesar dan mengecil). Berikut merupakan tabel penjelasan codingan yang kami buat.

Kode	Penjelasan
`stars = [ {"x": 600, "y": 50, "scale": 0.5}, {"x": 450, "y": 130, "scale": 0.5}, {"x": 350, "y": 40, "scale": 0.5} ]`	Menentukan koordinat bintang yang akan ditransformasi.
`for star in stars:`	Melakukan iterasi untuk setiap elemen `star` dalam list `stars`.

Kode	Penjelasan
`scaled_bintang = pygame.transform.scale(bintang, (int(40 * star["scale"]), int(40 * star["scale"])))`	Menghasilkan gambar bintang yang telah diubah skala berdasarkan nilai skala yang disimpan dalam atribut "scale" di setiap elemen `star`.
`bintang_rect = scaled_bintang.get_rect(center=(star["x"], star["y"]))`	Mendapatkan objek Rect yang digunakan untuk menentukan posisi dan ukuran gambar bintang setelah diubah skala.
`screen.blit(scaled_bintang, bintang_rect.topleft)`	Menampilkan gambar bintang yang telah diubah skala pada posisi yang dihitung berdasarkan objek Rect `bintang_rect`.
`star["scale"] += 0.01 * scaling_direction`	Mengubah nilai skala pada setiap iterasi loop dengan menambahkan sejumlah kecil berdasarkan `scaling_direction`.
`if star["scale"] >= 2.0 or star["scale"] <= 0.5:`	Memeriksa apakah nilai skala melebihi batas atas (2.0) atau batas bawah (0.5). Jika ya, mengubah arah `scaling_direction` menjadi sebaliknya untuk menghasilkan efek pulsasi atau perubahan skala berulang.

3.3.4. Refleksi



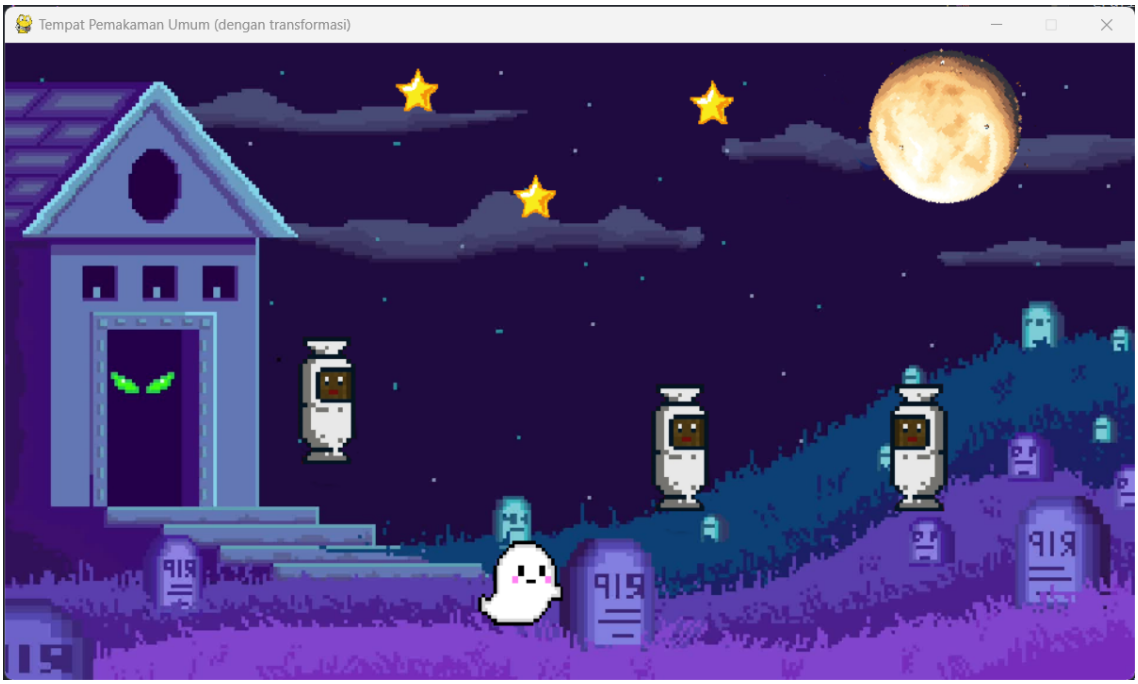
Gambar 3.3.4.1 Refleksi pada objek pocong di bukit.

Untuk transformasi refleksi, kami mengaplikasikannya pada objek pocong(refleksi terhadap sumbu x). Berikut merupakan tabel penjelasan codingan yang kami buat.

Kode	Penjelasan
`def load_pocong_rf(pocong_image):`	Deklarasi fungsi `load_pocong_rf` dengan satu parameter yaitu `pocong_image`, yang merupakan path gambar pocong.
`pocong = pygame.image.load(pocong_image)`	Memuat gambar pocong dari path yang diberikan menggunakan fungsi `pygame.image.load` dan menyimpannya dalam variabel `pocong`.
`pocong = pygame.transform.scale(pocong, (150, 150))`	Mengubah skala gambar pocong menjadi (150, 150) menggunakan fungsi `pygame.transform.scale` dan menyimpannya kembali dalam variabel `pocong`.

`pocong = pygame.transform.flip(pocong, True, False)`	Melakukan refleksi sumbu X terhadap gambar pocong menggunakan fungsi `pygame.transform.flip`, dengan parameter `True` untuk sumbu X dan `False` untuk sumbu Y.
`return pocong`	Mengembalikan gambar pocong yang telah diubah skala dan direfleksi.
`pocong_rf = load_pocong_rf(pocong_image)`	Memanggil fungsi `load_pocong_rf` dengan argumen `pocong_image`, yang kemudian mengembalikan gambar pocong yang telah dimodifikasi (skala dan refleksi sumbu X).
`screen.blit(pocong_rf, (700, 270))`	Menampilkan gambar pocong yang telah dimodifikasi pada posisi (700, 270) pada jendela permainan menggunakan fungsi `screen.blit`.

3.4. Output hasil setelah transformasi



Gambar 3.4.1 Objek Setelah Transformasi.

Pada **Gambar 3.4.1** Objek pocong dan hantu akan melakukan transformasi translasi, pocong yang ada pada bukit akan melakukan refleksi terhadap sumbu x, kemudian bintang melakukan translasi skala dan bulan melakukan transformasi rotasi.

3.5. Full source code

```
import pygame
import sys

# Inisialisasi Pygame
pygame.init()

# Ukuran jendela permainan
window_size = (960, 540)
```

```
# Membuat jendela permainan
screen = pygame.display.set_mode(window_size)
pygame.display.set_caption("Game dengan Latar Belakang Kustom")

def load_background(background_image):
    # Memuat gambar latar belakang dan menyesuaikannya dengan ukuran jendela
    background = pygame.image.load(background_image)
    background = pygame.transform.scale(background, window_size)
    return background

def load_bintang(bintang_image):
    # Memuat gambar tambahan (bintang)
    bintang = pygame.image.load(bintang_image)
    # Sesuaikan ukuran gambar sesuai kebutuhan
    bintang = pygame.transform.scale(bintang, (50, 50))  # Ganti ukuran
    return bintang

def load_pocong(pocong_image):
    # Memuat gambar tambahan (pocong)
    pocong = pygame.image.load(pocong_image)
    pocong = pygame.transform.scale(pocong, (150, 150))  # Ganti ukuran
    return pocong

def load_pocong_rf(pocong_image):
    pocong = pygame.image.load(pocong_image)
    pocong = pygame.transform.scale(pocong, (150, 150))
    # REFLEKSI SUMBU X
    pocong = pygame.transform.flip(pocong, True, False)
    return pocong


def load_jurig(jurig_image):
    # Memuat gambar tambahan (bintang)
    jurig = pygame.image.load(jurig_image)
    jurig = pygame.transform.scale(jurig, (100, 100))  # Ganti dengan ukuran
yang sesuai
    return jurig

def load_bulan(bulan_image):
    # Memuat gambar tambahan (bintang)
    bulan = pygame.image.load(bulan_image)
    bulan = pygame.transform.scale(bulan, (200, 200))  # Ganti dengan ukuran
yang sesuai
    return bulan
```

```
def game_loop():
    clock = pygame.time.Clock()

    # File Gambar
    background_image = "background.png"
    bintang_image = "star.png"
    pocong_image = "pocong.png"
    jurig_image = "jurig.png"
    bulan_image = "bulan.png"

    background = load_background(background_image)
    bintang = load_bintang(bintang_image)
    pocong = load_pocong(pocong_image)
    pocong_rf = load_pocong_rf(pocong_image)
    jurig = load_jurig(jurig_image)
    bulan = load_bulan(bulan_image)

    # BINTANG
    angle = 0
    scaling_direction = 2 # kecepatan skala bintang

    # JURIG
    jurig_x = 0
    jurig_reset_pos = window_size[0] + 50  # Posisi reset jurig_x setelah
melewati batas
    # JURIG GERAK
    jurig_speed = 3  # Kecepatan 0 untuk berhenti

    # POCONG
    pocong_x = 200
    pocong_y = 200
    pocong_speed = 5  # Kecepatan translasi pocong
    jumping_up = True  # Status loncatan pocong

    # Daftar bintang dengan koordinat dan skala
    stars = [
        {"x": 600, "y": 50, "scale": 0.5},
        {"x": 450, "y": 130, "scale": 0.5},
        {"x": 350, "y": 40, "scale": 0.5}
    ]


    while True:
        for event in pygame.event.get():
```

```
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()

        # Menampilkan latar belakang pada jendela
        screen.blit(background, (0, 0))


        # # OBJEK TANPA TRANSFORMASI
        # screen.blit(bintang, (600, 50))
        # screen.blit(bintang, (500, 200))
        # screen.blit(bintang, (350, 40))
        # screen.blit(pocong, (200, 300))



        # TRANSLASI POCONG ke atas dan ke bawah LONCAT
        screen.blit(pocong, (pocong_x, pocong_y))
        if jumping_up:
            pocong_y -= pocong_speed
            if pocong_y <= 200:  # Posisi puncak loncatan
                jumping_up = False
        else:
            pocong_y += pocong_speed
            if pocong_y >= 300:  # Posisi bawah loncatan
                jumping_up = True


        # REFLEKSI POCONG
        screen.blit(pocong, (500, 270))
        # Yang di REFLEKSI
        screen.blit(pocong_rf, (700, 270))


        # TRANSLASI JURIG ke kanan
        screen.blit(jurig, (jurig_x, 400))
        jurig_x += jurig_speed  # Ubah sesuai kecepatan translasi yang
diinginkan
        # Jika jurig melewati batas, reset posisinya
        if jurig_x > jurig_reset_pos:
            jurig_x = -jurig.get_width()


        # ROTASI BULAN
        rotated_bulan = pygame.transform.rotate(bulan, angle)
```

```
        bulan_rect = rotated_bulan.get_rect(center=(800, 70))
        screen.blit(rotated_bulan, bulan_rect.topleft)
        # BERHENTI ROTASI ISIKAN 0
        angle += 1  # Ubah sesuai kecepatan rotasi yang diinginkan

        # SKALA BINTANG
        for star in stars:
            scaled_bintang = pygame.transform.scale(bintang, (int(40 *
star["scale"]), int(40 * star["scale"])))
            bintang_rect = scaled_bintang.get_rect(center=(star["x"],
star["y"]))
            screen.blit(scaled_bintang, bintang_rect.topleft)
            star["scale"] += 0.01 * scaling_direction
            if star["scale"] >= 2.0 or star["scale"] <= 0.5: # menentukan
ukuran skala maks dan min
                scaling_direction *= -1




        # Update layar
        pygame.display.flip()

        # Menetapkan batas FPS
        clock.tick(30)

if __name__ == "__main__":
    game_loop()
```

BAB IV
KESIMPULAN

Dalam proyek transformasi objek pada implementasi pemrograman grafika komputer, telah dilakukan eksplorasi dan pengimplementasian berbagai konsep transformasi pada objek-objek grafis. Transformasi ini melibatkan operasi translasi, rotasi, skala, dan refleksi yang diaplikasikan pada elemen-elemen seperti bintang, pocong, jurig, dan bulan dalam suatu representasi tempat pemakaman umum. Melalui implementasi transformasi objek dua dimensi, proyek ini berhasil menciptakan efek visual yang menarik dan dinamis dalam sebuah simulasi grafis. Penggunaan variasi transformasi seperti translasi pocong yang memberikan efek loncatan, refleksi pocong terhadap sumbu X, rotasi bulan, dan skala bintang secara dinamis, menciptakan pengalaman visual yang menarik bagi pengguna.

Penerapan transformasi pada proyek ini memberikan pemahaman yang lebih dalam tentang konsep dasar grafika komputer. Selain itu, proyek ini memberikan peluang untuk memahami pengaruh dan interaksi antar berbagai transformasi terhadap elemen-elemen grafis, serta bagaimana transformasi tersebut dapat diaplikasikan dalam menciptakan efek-efek visual yang kompleks.

Dengan demikian, proyek transformasi objek ini berhasil mencapai tujuannya untuk memahamkan dan mengimplementasikan konsep transformasi objek pada bidang grafika komputer. Pengalaman ini dapat menjadi dasar untuk pengembangan lebih lanjut dalam pemrograman grafika komputer dan aplikasi visualisasi lainnya.

DAFTAR PUSTAKA

Wikipedia (2023), *Visual Studio Code,*
https://en.wikipedia.org/wiki/Visual_Studio_Code

Wikipedia (2023), *Python (programming languange),*
https://en.wikipedia.org/wiki/Python_(programming_language)

OpenSea (2018), *Gambar Pocong,*
https://images.app.goo.gl/yqC43kFMdxHvbWZs8

ArtStation (2021), *Gambar Kuburan,*
ArtStation - pixel graveyard

PNGWING (2023), *Gambar Hantu,*
Pixel art YouTube Ghost Drawing, youtube, text, rectangle, bead png | PNGWing

PNGTREE (2023), *Gambar Bulan,*
Bulan Seni Piksel, Bulan, Pixel Art, Langit PNG Transparan dan Clipart untuk Unduhan Gratis (pngtree.com)

Freepik (2023), *Gambar Bintang,*
Premium Vector | Art illustration draw artwork pixel character icon symbol design concept video game set of star (freepik.com)