# Pengembangan Simulasi dan Perhitungan Gerak Parabola Berbasis Python dengan Integrasi Visualisasi Pygame dan Matplotlib

Disusun oleh :
**Majesty Gracia E.R**

# Latar Belakang

Gerak parabola: Fenomena penting dalam fisika.

Siswa sering kesulitan membayangkan konsep ini.

Teknologi memungkinkan pengembangan alat bantu pembelajaran yang lebih interaktif.

Tujuan: Mengembangkan aplikasi berbasis Python yang membantu memahami gerak parabola.

# Rumusan Masalah

- Bagaimana menyajikan informasi gerak parabola secara menarik dan interaktif?
- Apa saja faktor yang memengaruhi jarak dan ketinggian maksimum proyektil?
- Bagaimana efektivitas aplikasi ini dibandingkan metode pembelajaran tradisional?

# Tujuan Penelitian

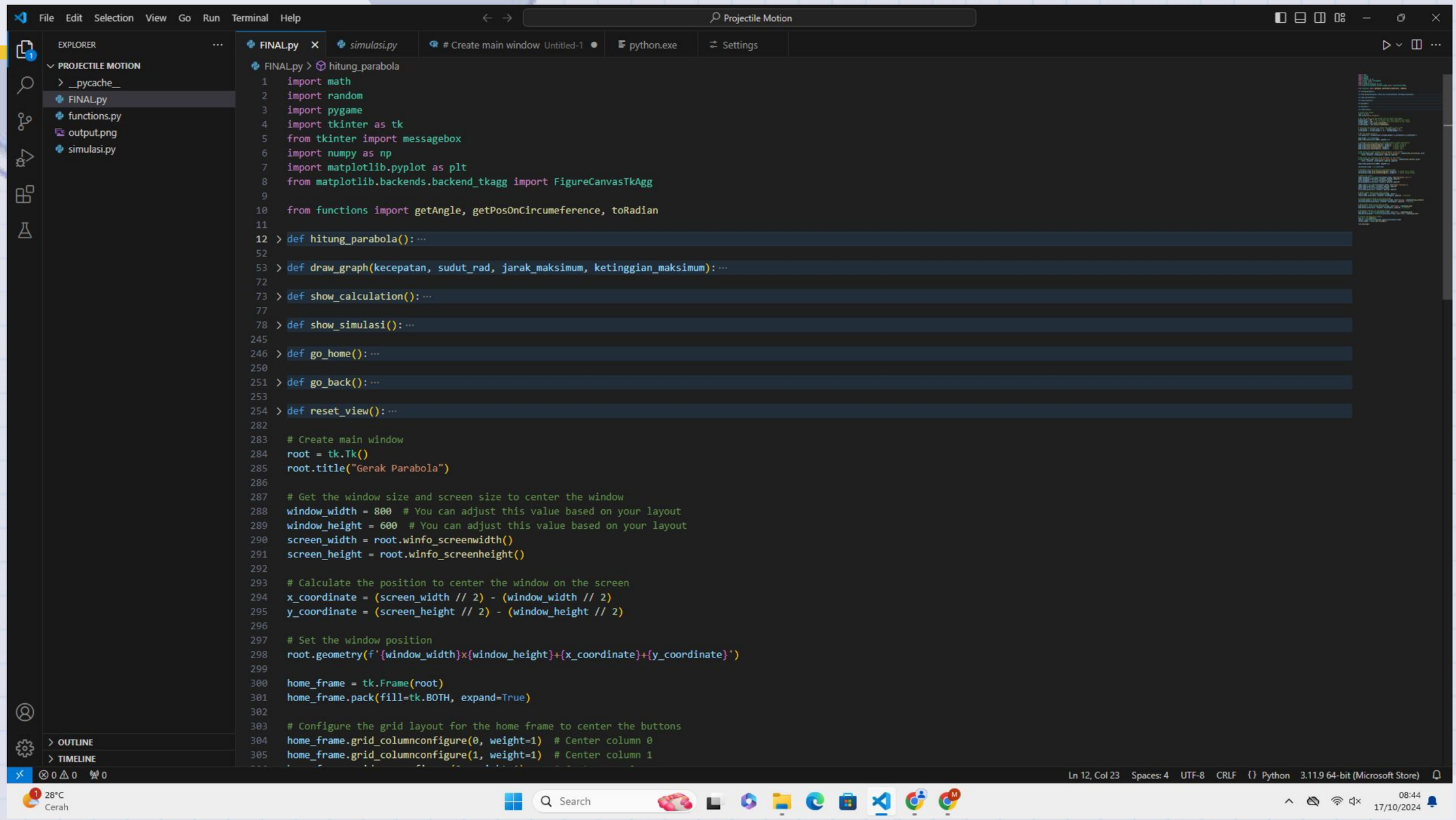
- Menghitung jarak dan ketinggian maksimum proyektil.
- Menyediakan visualisasi interaktif dari gerak parabola.
- Meningkatkan pemahaman siswa tentang gerak parabola.

**Input Parameter:**
Kecepatan awal, sudut peluncuran.

**Output:**
Jarak maksimum, ketinggian maksimum, grafik lintasan proyektil.

# Metode Penelitian

# Break Down


File Edit Selection View Go Run Terminal Help
Projectile Motion
EXPLORER
PROJECTILE MOTION
__pycache__
FINAL.py
functions.py
output.png
simulasi.py
FINAL.py
simulasi.py
# Create main window Untitled-1
python.exe
Settings
FINAL.py > hitung_parabola
import math
import random
import pygame
import tkinter as tk
from tkinter import messagebox
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from functions import getAngle, getPosOnCircumeference, toRadian
def hitung_parabola():
def draw_graph(kecepatan, sudut_rad, jarak_maksimum, ketinggian_maksimum):
def show_calculation():
def show_simulasi():
def go_home():
def go_back():
def reset_view():
# Create main window
root = tk.Tk()
root.title("Gerak Parabola")
# Get the window size and screen size to center the window
window_width = 800 # You can adjust this value based on your layout
window_height = 600 # You can adjust this value based on your layout
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
# Calculate the position to center the window on the screen
x_coordinate = (screen_width // 2) - (window_width // 2)
y_coordinate = (screen_height // 2) - (window_height // 2)
# Set the window position
root.geometry(f'{window_width}x{window_height}+{x_coordinate}+{y_coordinate}')
home_frame = tk.Frame(root)
home_frame.pack(fill=tk.BOTH, expand=True)
# Configure the grid layout for the home frame to center the buttons
home_frame.grid_columnconfigure(0, weight=1) # Center column 0
home_frame.grid_columnconfigure(1, weight=1) # Center column 1
OUTLINE
TIMELINE
Ln 12, Col 23 Spaces: 4 UTF-8 CRLF Python 3.11.9 64-bit (Microsoft Store)
28°C Cerah
Search
08:44 17/10/2024

Break Do

```python
def hitung_parabola():
    global entry_kecepatan, entry_sudut, result_label, canvas
    try:
        kecepatan = float(entry_kecepatan.get())
        sudut = float(entry_sudut.get())

        # Convert angle to radians
        sudut_rad = np.radians(sudut)

        # Calculate maximum distance and height
        g = 9.81  # gravitational acceleration
        jarak_maksimum = (kecepatan**2 * np.sin(2 * sudut_rad)) / g
        ketinggian_maksimum = (kecepatan**2 * (np.sin(sudut_rad)**2)) / (2 * g)

        # Display results
        result_text = f"Jarak Maksimum: {jarak_maksimum:.2f} m\nKetinggian Maksimum: {ketinggian_maksimum:.2f} m"
        result_label.config(text=result_text)

        # Draw the graph in the same window
        draw_graph(kecepatan, sudut_rad, jarak_maksimum, ketinggian_maksimum)

        # Hide and clear input fields and labels
        entry_kecepatan.delete(0, tk.END)
        entry_sudut.delete(0, tk.END)
        entry_kecepatan.grid_forget()
        entry_sudut.grid_forget()
        label_kecepatan.grid_forget()
        label_sudut.grid_forget()

        # Hide input buttons
        calculate_button.grid_forget()
        home_button.grid_forget()

        # Show navigation buttons and the graph
        back_button.grid(row=6, column=1, padx=5, pady=10)
        home_button_display.grid(row=6, column=0, padx=5, pady=10)
        canvas_widget.grid(row=5, columnspan=2)  # Display the graph

    except ValueError:
        messagebox.showerror("Input Error", "Silakan masukkan nilai yang valid.")
```

```python
# Add button for "Perhitungan Gerak Parabola" in the center
tk.Button(home_frame, text="Perhitungan Gerak Parabola", command=show_calculation).grid(
    row=0, column=0, columnspan=2, padx=10, pady=10)

# Add button for "Simulasi Gerak Parabola" in the center
tk.Button(home_frame, text="Simulasi Gerak Parabola", command=show_simulasi).grid(
    row=1, column=0, columnspan=2, padx=10, pady=10)
```

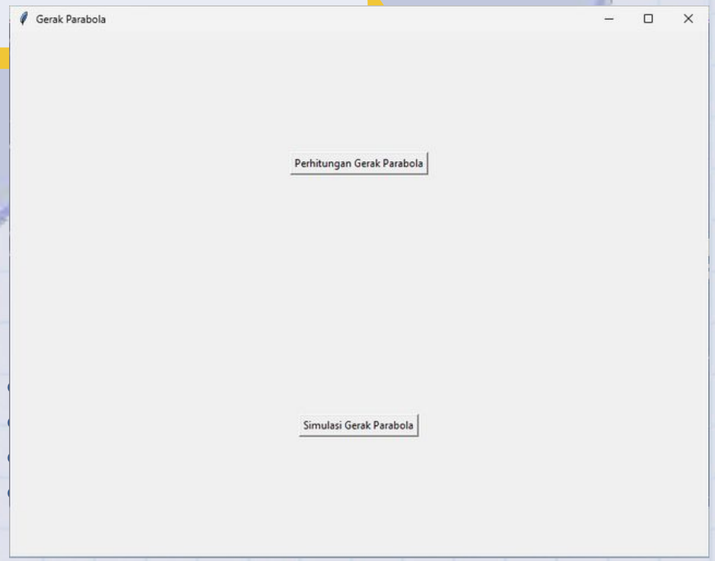
```python
def draw_graph(kecepatan, sudut_rad, jarak_maksimum, ketinggian_maksimum):
    t = np.linspace(0, 2 * jarak_maksimum / kecepatan, num=500)
    x = kecepatan * np.cos(sudut_rad) * t
    y = kecepatan * np.sin(sudut_rad) * t - 0.5 * 9.81 * t**2

    # Clear previous plot
    ax.clear()

    # Plot the graph
    ax.plot(x, y)
    ax.set_title('Grafik Gerak Parabola')
    ax.set_xlabel('Jarak (m)')
    ax.set_ylabel('Ketinggian (m)')
    ax.set_xlim(0, jarak_maksimum * 1.1)
    ax.set_ylim(0, ketinggian_maksimum * 1.1)
    ax.grid()

    # Draw the new figure
    canvas.draw()
```

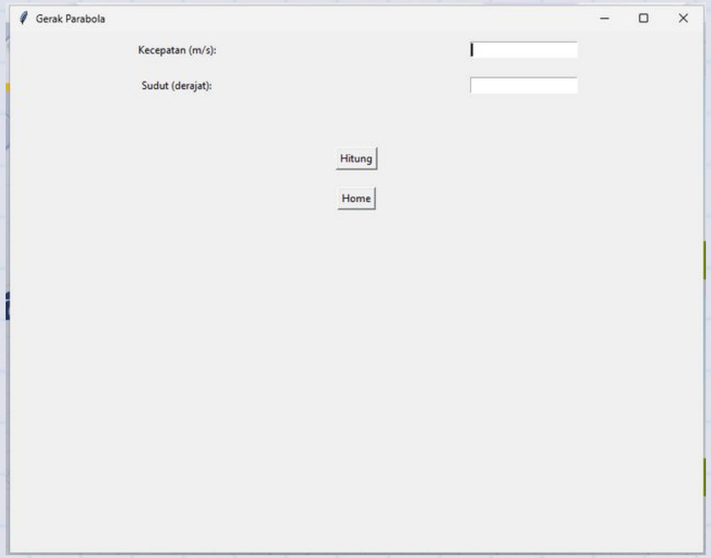
# Break Down

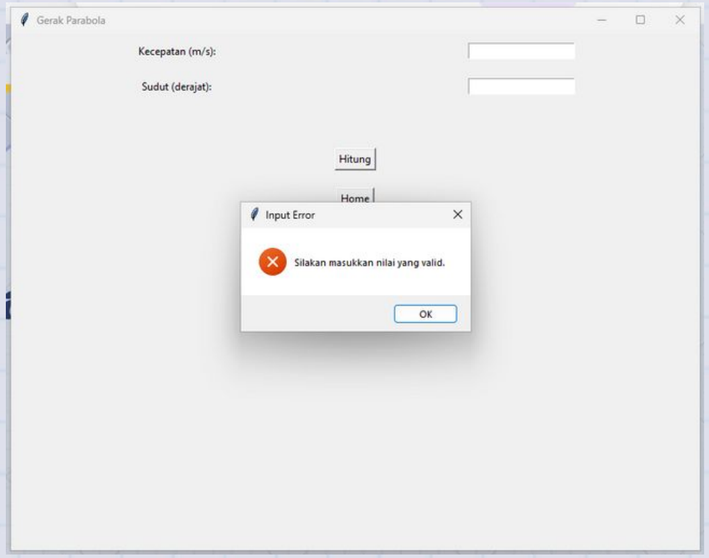
```python
    running = True
    while running:
        win.fill(BLACK)

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running = False

            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE or event.key == pygame.K_q:
                    running = False

                if event.key == pygame.K_r:
                    projectile_group.empty()
                    currentp = None

            if event.type == pygame.MOUSEBUTTONDOWN:
                clicked = True

            if event.type == pygame.MOUSEBUTTONUP:
                clicked = False

                pos = event.pos
                theta = getAngle(pos, origin)
                if -90 < theta <= 0:
                    projectile = Projectile(u, theta)
                    projectile_group.add(projectile)
                    currentp = projectile

            if event.type == pygame.MOUSEMOTION:
                if clicked:
                    pos = event.pos
                    theta = getAngle(pos, origin)
                    if -90 < theta <= 0:
                        end = getPosOnCircumeference(theta, origin)
                        arct = toRadian(theta)
```

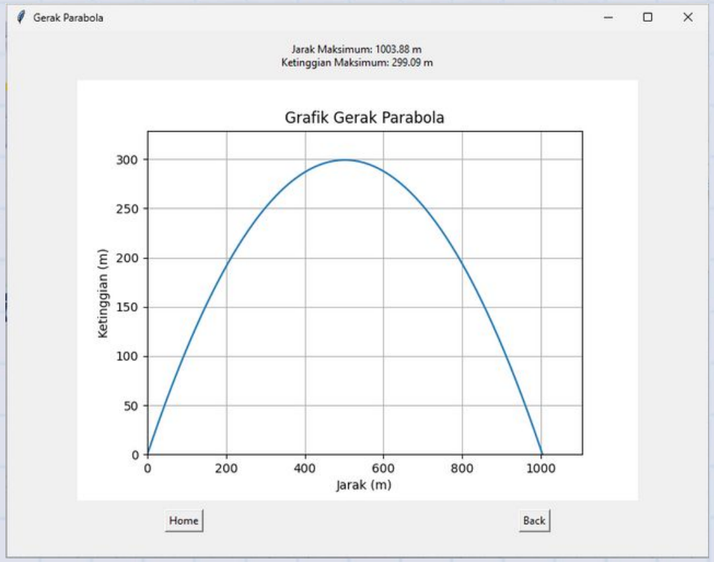
```python
        projectile_group.update()

        # Info ***************************************************************
        title = font.render("Projectile Motion", True, WHITE)
        fpstext = font.render(f"FPS : {int(clock.get_fps())}", True, WHITE)
        thetatext = font.render(f"Angle : {int(abs(theta))}", True, WHITE)
        degreetext = font.render(f"{int(abs(theta))}°", True, YELLOW)
        win.blit(title, (80, 30))
        win.blit(fpstext, (20, 400))
        win.blit(thetatext, (20, 420))
        win.blit(degreetext, (origin[0]+38, origin[1]-20))

        if currentp:
            veltext = font.render(f"Velocity : {currentp.u}m/s", True, WHITE)
            timetext = font.render(f"Time : {currentp.timeOfFlight()}s", True, WHITE)
            rangetext = font.render(f"Range : {currentp.getRange()}m", True, WHITE)
            heighttext = font.render(f"Max Height : {currentp.getMaxHeight()}m", True, WHITE)
            win.blit(veltext, (WIDTH-150, 400))
            win.blit(timetext, (WIDTH-150, 420))
            win.blit(rangetext, (WIDTH-150, 440))
            win.blit(heighttext, (WIDTH-150, 460))

        pygame.draw.rect(win, (0,0,0), (0, 0, WIDTH, HEIGHT), 5)
        clock.tick(FPS)
        pygame.display.update()

    pygame.quit()
```

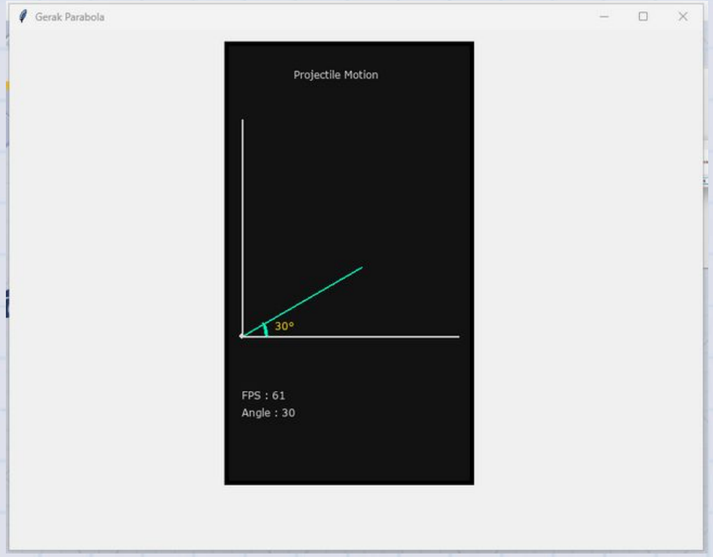
# Hasil

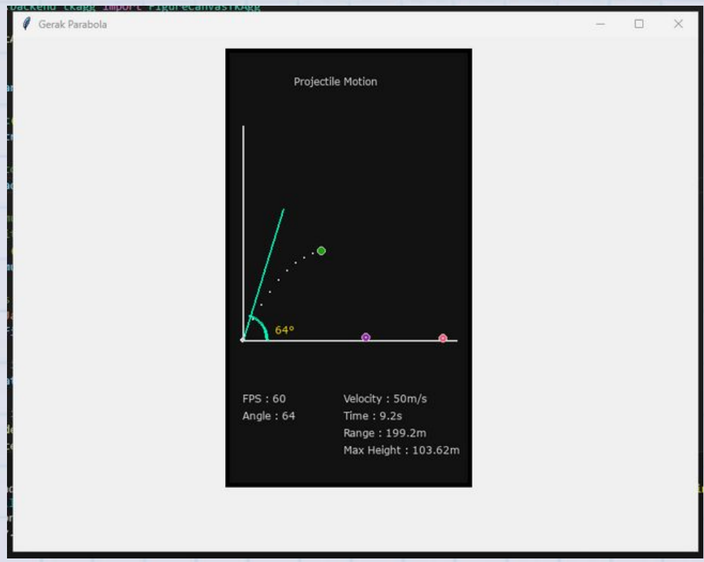

Gerak Parabola
Perhitungan Gerak Parabola
Simulasi Gerak Parabola

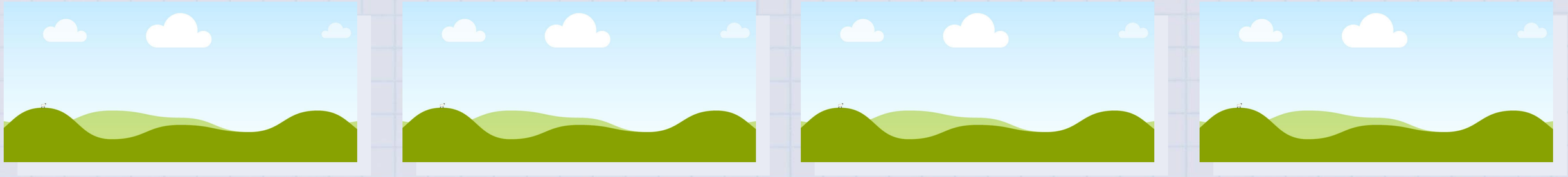

Gerak Parabola
Kecepatan (m/s):
Sudut (derajat):
Hitung
Home


Gerak Parabola
Kecepatan (m/s):
Sudut (derajat):
Hitung
Home
Input Error
Silakan masukkan nilai yang valid.
OK


Gerak Parabola
Jarak Maksimum: 1003.88 m
Ketinggian Maksimum: 299.09 m
Grafik Gerak Parabola
Ketinggian (m)
Jarak (m)
0
50
100
150
200
250
300
0
200
400
600
800
1000
Home
Back


Gerak Parabola
Projectile Motion
30°
FPS : 61
Angle : 30


Gerak Parabola
Projectile Motion
64°
FPS : 60
Angle : 64
Velocity : 50m/s
Time : 9.2s
Range : 199.2m
Max Height : 103.62m



# Kesimpulan

Aplikasi simulasi gerak parabola berhasil dikembangkan dengan akurasi tinggi.

Simulasi interaktif membantu meningkatkan pemahaman siswa.

Aplikasi ini dapat digunakan sebagai alat bantu pembelajaran fisika.

# Terima Kasih

**Majesty Gracia E.R**

*Pengembangan Simulasi dan Perhitungan Gerak Parabola Berbasis Python dengan Integrasi Visualisasi Pygame dan Matplotlib*