Documentación del Código:

Simulación de Caída de Bomba

Introducción

Este código implementa una simulación de la caída de una bomba desde un avión que se mueve sobre un terreno con un cráter. La simulación utiliza las bibliotecas `matplotlib` para la visualización y `tkinter` para la interfaz gráfica de usuario. El objetivo es simular la trayectoria de una bomba en un escenario en el que existe un cráter y determinar dónde impactará.

Descripción de las Bibliotecas Utilizadas

- **matplotlib: Se utiliza para graficar el movimiento del avión y la trayectoria de la bomba, así como para visualizar el cráter y el suelo.**
- **tkinter: Se utiliza para crear una interfaz gráfica que permite al usuario ingresar los parámetros necesarios para la simulación, como la altura del avión y la velocidad inicial de la bomba.**
- **numpy: Se usa para manejar cálculos matemáticos como la generación de datos de posiciones y el cálculo de ángulos en radianes.**

Archivo `requirements.txt`

Para asegurar que el programa funcione correctamente en cualquier entorno, debes crear un archivo `requirements.txt` que incluya todas las librerías utilizadas en el proyecto. En este caso, el archivo debería contener:

matplotlib

numpy

Esto permite que los usuarios instalen todas las dependencias necesarias ejecutando el siguiente comando en la línea de comandos:

Parámetros de Entrada

El código requiere varios parámetros de entrada para realizar la simulación, que el usuario puede ingresar mediante la interfaz gráfica:

1. **Altura inicial del avión (ha): La altura desde la cual el avión comienza su vuelo (en metros).**
2. **Altura del fondo del cráter (hc): La profundidad del cráter desde la altura del suelo (en metros).**
3. **Longitud horizontal del cráter (L): La distancia horizontal del cráter (en metros).**
4. **Ángulo de las paredes del cráter (α): El ángulo de inclinación de las paredes del cráter (en grados).**
5. **Distancia inicial del avión desde el borde del cráter (d): La distancia desde el punto de lanzamiento hasta el borde del cráter (en metros).**
6. **Velocidad inicial de la bomba (v0): La velocidad con la que se lanza la bomba (en m/s).**
7. **Posición del cráter (pos_crater): La distancia desde el origen hasta el comienzo del cráter (en metros).**

Descripción del Cráter

El cráter se modela como un trapecio invertido con una altura de 100 metros. Las coordenadas del cráter se calculan utilizando la posición inicial del cráter y el ángulo de inclinación de las paredes.

Descripción de la Trayectoria de la Bomba

La función `trayectoria` calcula la altura `y` de la bomba en función de la distancia horizontal `x`. La ecuación de la trayectoria toma en cuenta la aceleración debida a la gravedad y está dada por:

donde `g` es la constante de gravedad (9.81 m/s^2) y `t` es el tiempo que la bomba ha estado en el aire.

Proceso de Simulación

1. **Interfaz Gráfica de Usuario: El código crea una ventana mediante `tkinter` donde el usuario puede ingresar los valores iniciales de los parámetros.**
2. **Visualización del Terreno: La simulación grafica el suelo, las paredes del cráter, y la trayectoria del avión.**
3. **Movimiento del Avión: El avión se mueve horizontalmente de izquierda a derecha y se representa con una imagen cargada desde el archivo `avion.png`.**
4. **Lanzamiento de la Bomba: La bomba se lanza cuando el avión alcanza una distancia `d`. La trayectoria de la bomba se actualiza en cada iteración para mostrar el descenso, y se verifica si impacta el cráter o el suelo.**
5. **Colisiones: La simulación tiene en cuenta posibles colisiones:**
    - **Con la pared izquierda o derecha del cráter.**
    - **Con el fondo del cráter.**
    - **Con el suelo si la bomba cae fuera del cráter.**

Código Gráfico y Animación

- La figura se genera con `matplotlib` y se utilizan `AnnotationBbox` y `OffsetImage` para representar la posición del avión.
- La bomba y su trayectoria se muestran como puntos rojos y una línea naranja respectivamente.
- La simulación se ejecuta en un bucle donde se actualizan la posición del avión y la trayectoria de la bomba en cada iteración.

Manejador de Señales

El manejador de señales (`signal_handler`) permite cerrar el programa de forma segura si el usuario interrumpe la ejecución (Ctrl+C).

Recomendaciones de Uso

- Asegúrate de que la imagen del avión (`avion.png`) esté en el mismo directorio que el script para que se cargue correctamente.
- La interfaz de `tkinter` permite modificar fácilmente los parámetros, lo cual es útil para observar los efectos de cambiar la altura del avión, la velocidad de la bomba o el tamaño del cráter.

Metodología Utilizada

La metodología del proyecto se centra en la implementación de una simulación visual basada en parámetros físicos y matemáticos. La simulación se desarrolla siguiendo estos pasos:

- Entrada de Parámetros: A través de una interfaz gráfica creada con Tkinter, el usuario ingresa parámetros clave como la altura inicial del avión, la altura del cráter, la longitud del cráter, el ángulo de las paredes, la velocidad de la bomba, y la posición del cráter.
- Cálculo de Trayectoria: Se utiliza una función para calcular la trayectoria de la bomba basada en las ecuaciones del movimiento parabólico. Esto permite simular la caída bajo el efecto de la gravedad.
- Visualización de la Simulación: La trayectoria y el movimiento del avión se visualizan utilizando Matplotlib, lo cual ofrece una representación gráfica dinámica.
- Gestión de Colisiones: Se implementa una lógica para determinar cuándo la bomba colisiona con el suelo o las paredes del cráter, momento en el cual la simulación se detiene.

Desarrollo Matemático:

El cálculo de la trayectoria de la bomba se basa en las ecuaciones del movimiento parabólico. La función `trayectoria(x, x0, y0, v0x, v0y)` se utiliza para calcular la posición vertical `y` en función de la posición horizontal `x`, la posición inicial `(x0, y0)`, y la velocidad inicial `v0x`. La fórmula utilizada es:

y = y0 - 0.5 * g * t^2

Donde `g` es la aceleración debido a la gravedad (9.81 m/s²), y `t` es el tiempo que la bomba ha estado en movimiento, calculado en función de `x` y `v0x`.

Diagrama de Flujo / Pseudocódigo:

- **Inicio**
- **Configurar parámetros iniciales ingresados por el usuario**
- **Crear la visualización con Matplotlib**
- **Iniciar la animación del avión**
- **Si el avión alcanza la distancia indicada, lanzar la bomba**
- ***Calcular la trayectoria de la bomba***
- **Si la bomba colisiona con el suelo o las paredes del cráter, detener la simulación**
- **Mostrar la gráfica resultante**
- **Fin**

Detalles Importantes de la Implementación:

- **Interfaz Gráfica: Se utilizó Tkinter para permitir al usuario ingresar los valores de los parámetros de la simulación de manera intuitiva.**
- **Colisiones: La bomba se detiene al colisionar con el cráter o el suelo. Esta lógica está implementada en un bucle que actualiza continuamente la posición de la bomba hasta que ocurre una colisión.**
- **Visualización: El avión y la trayectoria de la bomba se dibujan utilizando Matplotlib. La imagen del avión se mueve de manera continua y la trayectoria de la bomba se dibuja como una curva naranja.**

Resultados Obtenidos y Conclusiones

Resultados:

- **La simulación permite visualizar cómo la bomba sigue una trayectoria parabólica desde el avión hasta el impacto con el suelo o las paredes del cráter. La representación gráfica muestra claramente cómo cambian las trayectorias al modificar los parámetros iniciales como la velocidad de lanzamiento, la altura del avión, o el ángulo del cráter.**
- **Se observó que la precisión de la trayectoria depende directamente de los parámetros ingresados, y que el usuario puede interactuar con diferentes**

configuraciones para analizar cómo los diferentes factores afectan el comportamiento del proyectil.

Conclusión

- **Interactividad y Educación: La simulación proporciona una herramienta interactiva útil para la enseñanza de conceptos relacionados con el movimiento de proyectiles. Al permitir al usuario modificar los parámetros, se fomenta una mejor comprensión del efecto de la gravedad y la velocidad inicial en la trayectoria de un proyectil.**
- **Aplicaciones Futuras: El programa podría ampliarse para incluir otros factores, como el viento o diferentes tipos de terreno, para hacer la simulación más realista.**
- **Limitaciones: El modelo actual no tiene en cuenta factores como la resistencia del aire o efectos externos que podrían afectar la trayectoria. Además, la representación del cráter está simplificada como un trapecio, lo cual puede no ser totalmente realista en todas las situaciones.**