Memoria Proyecto Multidisciplinar

Gabriel Argente Aguilera, Núria Gómez Tronchoni, Arizai Luciana Lira Tapia, y Manuel García Díaz

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Fecha: 3 de junio de 2024

Índice

1. Introducción

A continuación, se desarrollará la memoria del proyecto multidisciplinar del segundo semestre del 1er curso del Grado en Tecnología Digital y Multimedia 2023-2024.

Las actividades asociadas al proyecto multidisciplinar incluyen la investigación de un problema consistente en el desarrollo de una aplicación para medir saltos. Un juego de competición basado en la medición real de la capacidad física en el deporte mediante un Smartphone en la cual se implementarán diferentes conceptos vistos a lo largo del semestre en las asignaturas de programación, física, matemáticas y arquitectura de redes. Para cada asignatura se requieren unos requisitos diferentes que serán descritos en profundidad en el apartado de objetivos.

Además de realizar competencias relacionadas con las asignaturas vistas en clase, se ha realizado un video promocional a modo de anuncio de la aplicación al igual que una presentación acerca del proyecto y los resultados obtenidos.

2. Objetivos

En general los objetivos del proyecto se basan en: integrar la metodología de aprendizaje basada en proyectos (ABP) en el primer curso de GTDM. Otro de los objetivos generales que se pretenden desarrollar en este proyecto es desarrollar una experiencia innovadora compartida entre cuatro asignaturas básicas, que proporcione un enfoque práctico, orientado a las aplicaciones profesionales y que sea capaz de motivar a los alumnos en su aprendizaje; además de, desarrollar un juego de competición basado en la medición de fenómenos reales (medida de la fuerza explosiva de los alumnos al realizar un salto), usando los acelerómetros del Smartphone.

Empezaremos describiendo los objetivos del proyecto de la parte de la asignatura de programación.

Para esta asignatura deberemos diseñar una arquitectura modular para un programa en Python, es decir, dividir el programa en distintas funciones que implementan una sola tarea; además de, implementar este programa en Python con una interfaz de usuario grafica (GUI), prevenir code smells en la implementación, hacer testing adecuado con pytest de todas las funciones del programa y un testing adecuado manual a través de la GUI.

Por otra parte, los objetivos de la asignatura de física consisten en: comprender los fundamentos de la mecánica clásica (dinámica del punto y dinámica de sistemas), aprender a manejar las técnicas de captura y análisis de movimientos mediante fotogrametría y acelerometría mediante el uso del programa de videoanálisis Tracker y la aplicación del móvil acelmov; además de, comprender las bases del método científico y la necesidad de validar los modelos mediante experimentación.

En cuanto a la parte de matemáticas, se combinarán los conceptos adquiridos en los módulos 3 y 5, además de los conocimientos adquiridos durante la realización del mini-proyecto del módulo 5 tales como el tratamiento numérico de datos desde ficheros con formato variado mediante el uso del paquete "pandas" para convertir los datos a formato "numpy", además de hacer las derivadas del desplazamiento y la velocidad vertical, y la aceleración y la velocidad vertical medidas de un Excel, además de detectar hitos (cambios bruscos de tendencia) en las mediciones.

Finalmente, la parte de arquitectura de redes consistirá en realizar un programa que se conecte a un servidor, muy parecido al de una práctica hecha en clase, mediante el uso de sockets TCP/IP desarrollando el cliente de un protocolo de aplicación propietario el cual intercambiará resultados con el servidor, esta parte deberá estar integrada dentro de la aplicación general.

3. Material y métodos

3.1. Modelo teórico empleado en el proyecto

El proyecto se basa en un modelo teórico que combina los fundamentos de la mecánica clásica, la programación, y el análisis de señales para medir y analizar la fuerza explosiva de un salto mediante el uso de acelerómetros en smartphones. El modelo asume que la fuerza explosiva puede ser medida a través de la aceleración vertical registrada durante el salto. Utilizando las leyes de la dinámica del punto y la dinámica de sistemas, los datos obtenidos del acelerómetro se procesan para calcular la fuerza, la velocidad y el desplazamiento.

3.2. Funcionamiento del acelerómetro

El acelerómetro es un sensor que mide la aceleración en tres: A, Y ,y Z. En este proyecto, se utiliza principalmente la aceleración en el eje Z (vertical) para analizar el salto. El acelerómetro del smartphone proporciona datos de aceleración en tiempo real, que pueden ser capturados y procesados para obtener información sobre la fuerza del salto.

3.3. Programas utilizados y procesamiento de la señal

Los programas que han sido empleados para la captación de la señal son Tracker y Acelmov. Tracker es un programa de videoanálisis que permite capturar y analizar movimientos. Se utiliza para validar los datos obtenidos del acelerómetro, y Acelmov es una aplicación móvil que captura los datos de aceleración del smartphone y los almacena en un formato que puede ser procesado posteriormente.

En cuanto al procesamiento de la señal captada, el proceso que hemos realizado ha sido el siguiente:

1. **Capturar de Datos:** Los datos de aceleración son capturados usando la aplicación Acelmov durante el salto.
2. **Preprocesamiento:** Los datos capturados se exportan a un archivo CSV y se cargan en un programa de Python para su procesamiento.
3. **Análisis de Datos:** Donde, primero se utiliza el paquete pandas para manipular los datos y convertidos a formato numpy para así poder procesarlos más eficientemente. A continuación, se calculan las derivadas del desplazamiento y la velocidad vertical para obtener la aceleración y velocidad vertical en cada instante. Y, por último, se detectan hitos o cambios bruscos de tendencia en las mediciones para identificar los puntos clave del salto, como el despegue y el aterrizaje, por ejemplo.

3.4. Cálculos realizados

Los cálculos realizados durante la realización de este proyecto han consistido en el empleo de fórmulas físicas tratadas con anterioridad en las clases de la asignatura de Física.

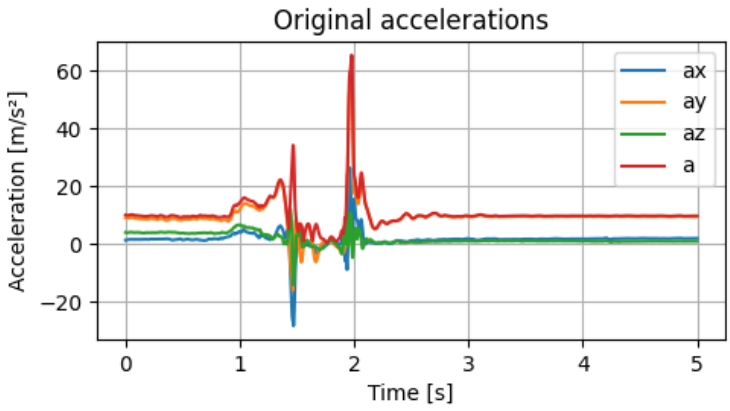
1. **Aceleración original:** Es la aceleración medida con la aplicación Acelmov. Los colores azul, amarillo y verde representan la aceleración medida en cada eje (x, y, z). Por otra parte, el color rojo representa el módulo de las tres aceleraciones, calculado como:

$$a = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

donde:

- a_x es la aceleración en el eje x (color azul),
- a_y es la aceleración en el eje y (color amarillo),

- a_z es la aceleración en el eje z (color verde),
- a es el módulo de la aceleración total (color rojo).

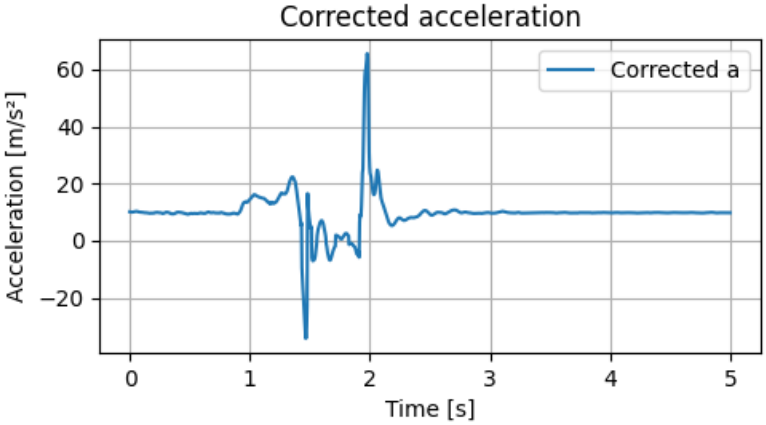


2. **Aceleración corregida:** Es la aceleración corregida, es decir, el módulo de las tres aceleraciones medidas con Acelmov en función del tiempo, calculado como:

$$a_{\text{corregida}} = \sqrt{a_{x,\text{corregida}}^2 + a_{y,\text{corregida}}^2 + a_{z,\text{corregida}}^2}$$

donde:

- $a_{x,\text{corregida}}$ es la aceleración corregida en el eje x,
- $a_{y,\text{corregida}}$ es la aceleración corregida en el eje y,
- $a_{z,\text{corregida}}$ es la aceleración corregida en el eje z,
- $a_{\text{corregida}}$ es el módulo de la aceleración total corregida.



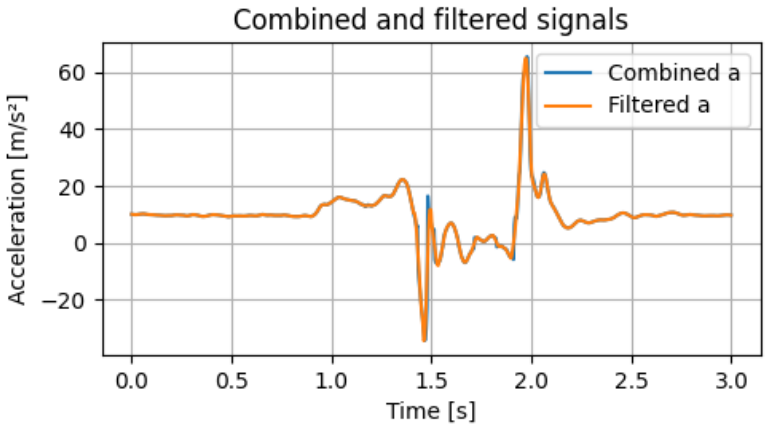
3. **Señales combinadas y filtradas:** Es la combinación de las aceleraciones, es decir, el módulo de estas. Además, estas aceleraciones están filtradas con el filtro de Savyitzky-Golay, calculado como:

$$a = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

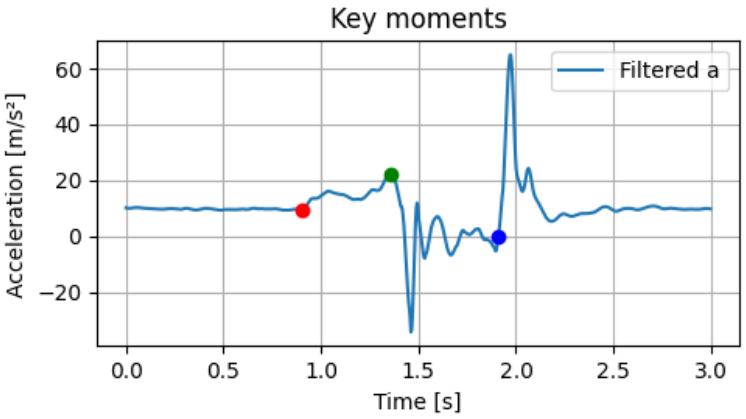
donde:

- a_x es la aceleración en el eje x,

- a_y es la aceleración en el eje y,
- a_z es la aceleración en el eje z,
- a es el módulo de la aceleración total filtrada.



4. **Instantes clave:** En esta gráfica vemos la aceleración filtrada en función del tiempo; además, el punto rojo representa el inicio del impulso, el verde representa la máxima aceleración y el azul representa el momento en el que el saltador toca el suelo.

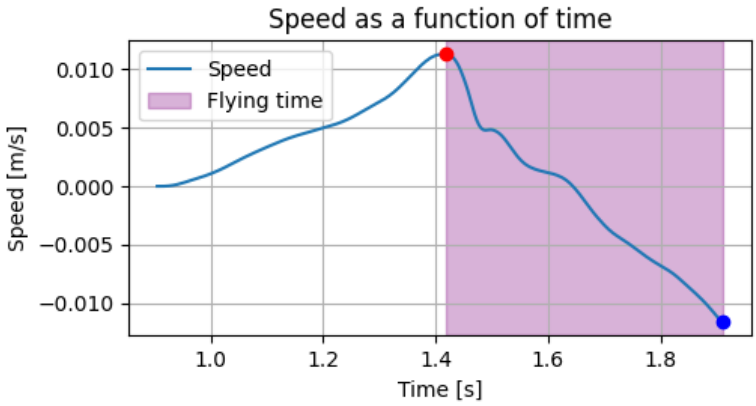


5. **Cálculo de la Velocidad:** La gráfica representa las oscilaciones del móvil durante el vuelo acortada hasta el punto de velocidad mínimo (representado por el punto azul). Para calcular la velocidad en función del tiempo $v(t)$ utilizamos la fórmula:

$$v(t) = \int a(t)\, dt$$

donde:

- $a(t)$ es la aceleración en función del tiempo,
- $v(t)$ es la velocidad en función del tiempo.

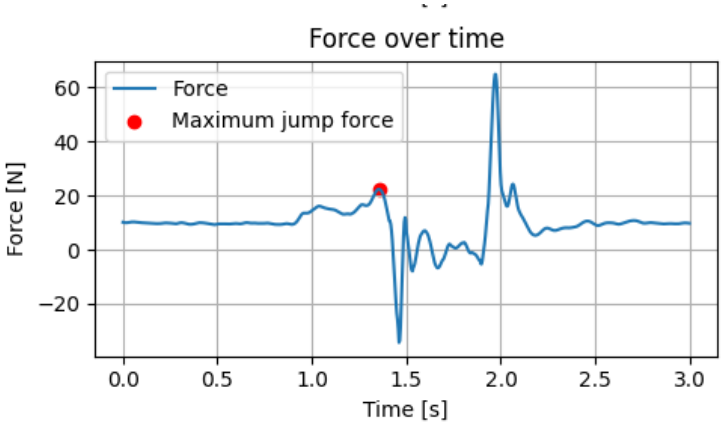


6. **Fuerza en función del tiempo:** La fuerza en función del tiempo se calcula utilizando la segunda ley de Newton:

$$F(t) = m \cdot a(t)$$

donde:

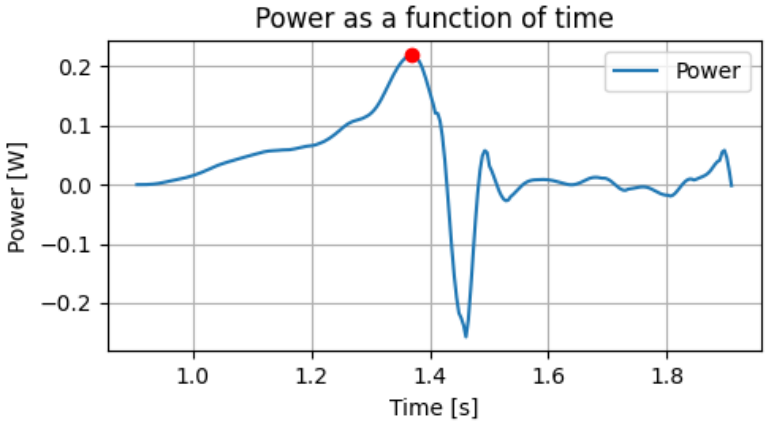
- $F(t)$ es la fuerza en función del tiempo,
- m es la masa del objeto,
- $a(t)$ es la aceleración en función del tiempo.



7. **Potencia en función del tiempo:** Una vez tenemos la fuerza $F(t)$ y la velocidad $v(t)$, la potencia se calcula como:

$$P(t) = F(t) \cdot v(t)$$

El punto rojo en la gráfica representa el punto donde la potencia es máxima; sin embargo, puede ser un valor erróneo asociado a la oscilación del móvil sobre la piel. El mínimo al final es real y corresponde al frenado del cuerpo al tomar contacto con el suelo (fuerza hacia arriba y velocidad hacia abajo, resultando en una potencia negativa).



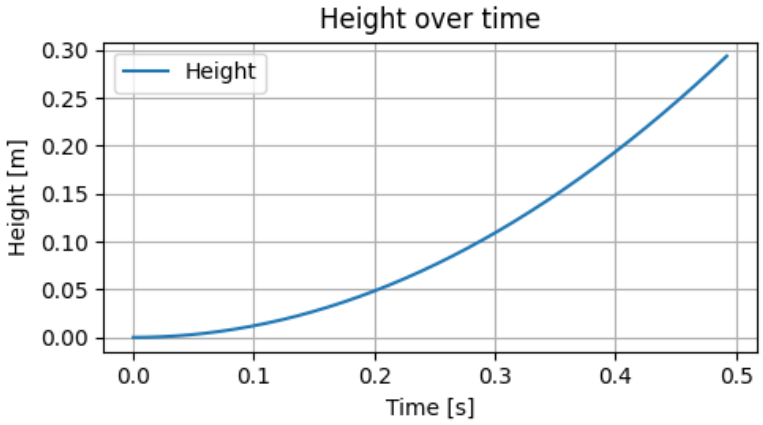
8. **Altura en función del tiempo:** La altura en función del tiempo se calcula integrando la velocidad con respecto al tiempo. Para obtener la altura $h(t)$, utilizamos la fórmula:

$$h(t) = \int v(t)\, dt$$

donde:

- $v(t)$ es la velocidad en función del tiempo,
- $h(t)$ es la altura en función del tiempo.

Al integrar la velocidad obtenemos la posición (altura) del objeto en función del tiempo. La gráfica representa la variación de la altura del objeto a lo largo del tiempo.



3.5. Estructura de los programas [Arquitectura del software]

1. **Arquitectura modular:** El programa se divide en módulos específicos para cada tarea: captura de datos, preprocesamiento, análisis y visualización.
2. **Interfaz gráfica de usuario (GUI)** Desarrollada en Python utilizando bibliotecas como Tkinter o PyQt, la GUI permite a los usuarios cargar los datos, ejecutar el análisis y visualizar los resultados.
3. **Funciones Clave**
   - Captura de Datos: Funciones para importar y leer archivos de datos
   - Preposicionamiento: Funciones para limpiar y preparar los datos
   - Análisis: Algoritmos para calular la fuerza, velocidad y desplazamiento, y detectar hitos
   - Visualización: herramientas para graficar los resultados y mostrar informes.

3.6. Comprobaciones experimentales

Para verificar las hipótesis y comparar soluciones, se realizaron las siguientes comprobaciones experimentales:

- **Validación con Tracker:** Se compararon los datos obtenidos del acelerómetro con los resultados de análisis de video en Tracker para asegurar la precisión de las mediciones.
- **Pruebas Repetidas:** Se realizaron múltiples saltos y se analizaron los datos para verificar la consistencia y repetibilidad de los resultados.
- **Análisis Comparativo:** Se compararon los resultados obtenidos de diferentes métodos de procesamiento para determinar el más efectivo y preciso.

Estas comprobaciones aseguran que el modelo y los métodos utilizados son válidos y confiables para medir y analizar la fuerza explosiva de un salto utilizando acelerómetros en smartphones.

4. Arquitectura del software

Una arquitectura de software bien diseñada debe facilitar la escalabilidad, la robustez y la eficiencia, a la vez que mantiene la flexibilidad para adaptarse a los cambios y evolución de las necesidades del programa. Es una representación que nos permite comprender cómo interactuarán los diferentes elementos del sistema, y cómo estas interacciones cumplirán con los requerimientos funcionales y no funcionales del proyecto.

En este apartado, exploraremos los fundamentos más importantes de la arquitectura de software de nuestro proyecto. Desglosaremos los componentes clave, detallaremos cómo y por qué interactúan, como lo hacen. A través de esta exploración, proporcionaremos una comprensión clara del diseño del sistema, preparando el escenario para una implementación eficaz y una futura mantenibilidad.

4.1. Explicación de las funciones

En este apartado explicaremos una a una todas las funciones que constituyen el programa, teniendo en cuenta su funcionamiento, requisitos y fallas.

- **center_window(app, width, height):** Centra la ventana de la aplicación app en la pantalla principal del usuario. Calcula las coordenadas x y y necesarias para posicionar la ventana en el centro horizontal y vertical de la pantalla, utilizando las dimensiones proporcionadas width (ancho) y height (alto). Luego, ajusta la geometría de la ventana tkinter para ubicarla en estas coordenadas.
- **load_image(canvas, path, dimensiones):** Abre una imagen desde la ruta path usando PIL (Python Imaging Library), la redimensiona a las dimensiones especificadas en dimensiones, y la carga en un canvas de tkinter. También evita que la imagen sea recolectada por el garbage collector de Python almacenándola como atributo del canvas.
- **clear_box(box):** Elimina todos los widgets hijos de un box proporcionado. Esto se logra iterando sobre los hijos del box y llamando al método destroy en cada widget, limpiando así el box para su reutilización.
- **aplicar_tema(tema):** Configura la aplicación con los colores y recursos definidos en el diccionario tema. Esto incluye colores de fondo, texto, títulos, sidebar, botones y fondos de diversas pantallas, así como las rutas para la música y fondos de pantalla específicos.
- **insertar_salto_linea_en_punto(texto):** Encuentra el primer punto (.) en el texto y añade dos saltos de línea justo después de este punto. Si no se encuentra ningún punto, retorna el texto sin cambios. Este formato es útil para mejorar la legibilidad del texto en la interfaz.

- **load_translations(file_path):** Carga un archivo JSON de traducciones desde la ruta file_path y lo retorna como un diccionario. Este archivo contiene las traducciones necesarias para soportar múltiples idiomas en la aplicación.
- **get_translation(translations, lang, key):** Devuelve la traducción correspondiente a key para el idioma lang desde el diccionario de translations. Si la clave no se encuentra, retorna la key como valor por defecto.
- **importar_datos(fichero:** Importa datos desde un archivo Excel especificado en fichero usando pandas, corrige comas decimales y extrae las columnas t, a, ax, ay y az como arrays de floats. Maneja posibles errores de conversión de datos.
- **corregir_aceleracion(a, ay):** Ajusta el valor absoluto de la aceleración a utilizando el signo de la componente ay. Esto corrige la aceleración medida para tener en cuenta la orientación del sensor.
- **recortar_combinar_datos(t, a_fixed):** Recorta los datos de aceleración a_fixed en un intervalo específico del tiempo t, y los combina para formar un nuevo conjunto de datos ajustado. Esto permite analizar sólo la porción relevante del salto.
- **aplicar_filtro_savgol(a_combinada):** Aplica un filtro de Savitzky-Golay a la señal a_combinada para suavizarla, utilizando una ventana de longitud 11 y un polinomio de orden 3.
- **obtener_instantes_clave(t_combinada, a_filtrada):** Identifica y retorna los instantes clave (inicio de impulso, máxima aceleración e impacto en el suelo) en la señal a_filtrada utilizando derivadas y puntos de interés específicos.
- **calcular_fuerza_salto(a_filtrada, g_medida, m):** Calcula la fuerza del salto F utilizando la aceleración filtrada a_filtrada, la gravedad medida g_medida y la masa m del saltador, aplicando la fórmula $F = m * (a\_salt + 9{,}81)$.
- **calcular_velocidad_salto(t_combinada, a_filtrada, g_medida, start_index, end_index):** Calcula la velocidad del salto integrando la aceleración filtrada a_filtrada, ajustada por la gravedad medida g_medida, sobre el intervalo de tiempo especificado por start_index y end_index.
- **calcular_potencia_salto(F, v_recortada, t1_index, t3_index, t_combinada):** Calcula la potencia del salto multiplicando la fuerza F por la velocidad v_recortada en el intervalo entre t1_index y t3_index.
- **calcular_altura_salto(v_max, g_medida, t_vuelo):** Calcula la altura del salto usando dos métodos: (a) la fórmula cinemática basada en la velocidad máxima v_max y (b) la fórmula basada en el tiempo de vuelo t_vuelo y la gravedad medida g_medida.
- **calcular_datos_salto(fichero, masa):** Realiza el procesamiento completo de los datos del salto desde la importación de datos hasta el cálculo de altura, pasando por corrección de aceleración, filtrado, identificación de instantes clave, y cálculos de fuerza, velocidad y potencia. Retorna la altura calculada y un diccionario con todos los datos intermedios y finales para su representación.
- **representar_todos_los_datos(datos):** Dibuja múltiples gráficos relevantes para el análisis del salto utilizando los datos proporcionados en datos. Incluye gráficos de aceleración original y corregida, señales combinadas y filtradas, instantes clave, velocidad, y potencia.
- **salir(app):** Gestiona la salida de la aplicación. Si el usuario está conectado a un servidor, muestra una confirmación y cierra la sesión de forma segura. Luego, muestra la pantalla de login.
- **ajustar_texto(texto, max_length):** Ajusta el texto para que no exceda max_length caracteres, añadiendo puntos suspensivos si es necesario. Útil para mantener la consistencia visual en la interfaz de usuario.
- **estilizar_boton(boton, colores):** Aplica un estilo específico a un boton de tkinter utilizando un conjunto de colores para el fondo, texto y efectos de hover. También configura el comportamiento del botón al interactuar con el mouse.
- **on_hover(boton, color='white'):** Cambia el cursor y el color de fondo de un boton cuando el mouse se posiciona sobre él, indicando que el botón es interactivo.

- **on_leave(boton, color):** Restaura el color de fondo original del boton y elimina el cursor de mano cuando el mouse deja de estar sobre él.
- **ignore_event(event):** Ignora un evento específico en tkinter, utilizado para suprimir ciertos tipos de interacción con widgets.
- **on_button_release(event):** Restaura el color del botón al soltarlo y activa la acción asociada al botón.
- **seleccionar_archivo(archivo_btn):** Abre un diálogo para seleccionar un archivo y actualiza el texto del botón archivotexttt_btn con el nombre del archivo seleccionado. Guarda la ruta del archivo seleccionado en una variable global para su posterior uso.
- **cambiar_tema(nuevo_tema):** Cambia el tema visual de la aplicación a nuevo_tema, aplica los nuevos colores y recursos, y guarda el ajuste en un archivo de configuración.
- **guardar_ajustes(tipo, ajuste):** Guarda los ajustes de tema o idioma en un archivo de configuración ajustes_guardados.txt. Si el archivo ya existe, actualiza el ajuste correspondiente; de lo contrario, crea un nuevo archivo con los ajustes.
- **cargar_ajustes(temas):** Carga los ajustes guardados de tema e idioma desde un archivo de configuración y aplica el tema correspondiente. Actualiza las variables globales con los valores cargados.
- **cambiar_volumen(volumen):** Ajusta el volumen de la música en la aplicación según el valor proporcionado volumen.
- **reproducir_sonido_boton(ruta):** Reproduce un sonido específico almacenado en ruta cuando se presiona un botón. El volumen del sonido se ajusta según el volumen actual de la aplicación.
- **reproducir_musica():** Reproduce una música desde la ruta ruta en bucle con un tiempo de fundido de entrada fadeintexttt_duration. El parámetro bucle determina si la música debe repetirse indefinidamente o no.
- **detener_musica(fadeout_duration=1000):** Detiene la música con una duración de desvanecimiento fadeout_duration, creando un efecto de salida suave.
- **cambiar_pista_despues_de_tiempo(tiempo, siguiente_pista, fadein_duration=5000):** Cambia la pista de música a siguiente_pista después de un tiempo especificado tiempo con un fundido de entrada fadein_duration.
- **cambiar_idioma(idioma):** Cambia el idioma de la aplicación a idioma, actualiza la interfaz para reflejar el nuevo idioma y guarda el ajuste en un archivo de configuración.
- **crear_imagen_texto(texto, width, height, radio, color_fondo, color_texto, ruta_fuente_titulos=None, tamano_fuente_titulos=16):** Crea una imagen con el texto proporcionado, con dimensiones width y height, y un fondo redondeado con radio radio. El texto se dibuja con color_texto y se puede especificar una fuente personalizada ruta_fuente_titulos y su tamaño tamano_fuente_titulos. Retorna un objeto ImageTk.PhotoImage.
- **comprobar_campos_salto(main_content, masaTxTBox, nombreTxTBox, grupoTxTBox):** Verifica que todos los campos necesarios para el registro de un salto (masa, nombre, grupo, archivo seleccionado) estén completos y sean válidos. Muestra mensajes de error si algún campo es inválido.
- **guardar_datos(datos, main_content):** Guarda los datos del salto. Si la aplicación está en modo online, envía los datos al servidor. Si está en modo offline, los guarda localmente. Muestra mensajes de confirmación o error según corresponda.
- **guardar_datos_locales(datos_ranking, ruta="datos_locales_guardado/top_saltos.txt"):** Guarda los datos del salto en un archivo local especificado por ruta. Si el archivo no existe, lo crea con las cabeceras necesarias.
- **cargar_datos_locales(ruta="datos_locales_guardado/top_saltos.txt"):** Carga los datos de saltos almacenados localmente desde el archivo especificado por ruta. Ordena los datos por altura en orden descendente.

- **mostrar_pantalla_realizarSalto(main_content):** Muestra la interfaz para registrar los datos de un nuevo salto, incluyendo campos para masa, nombre, grupo, y selección de archivo. Proporciona botones estilizados y valida los datos antes de permitir el registro.
- **mostrar_pantalla_resultados(main_content, masaRecipiente, nombreRecipiente, grupoRecipiente):** Muestra la interfaz para registrar los datos de un nuevo salto, incluyendo campos para masa, nombre, grupo, y selección de archivo. Proporciona botones estilizados y valida los datos antes de permitir el registro.
- **mostrar_pantalla_inicio(main_content):** Muestra la pantalla de inicio con información y tutoriales sobre el uso de la aplicación, organizados en secciones claras y estilizadas.
- **mostrar_pantalla_configuracion(main_content):** Muestra la pantalla de configuración donde el usuario puede ajustar el tema, volumen e idioma de la aplicación, utilizando menús desplegables y sliders.
- **mostrar_pantalla_ranking(main_content):** Muestra el ranking de los mejores saltos registrados. Si está en modo online, recupera los datos del servidor; si está en modo offline, carga los datos almacenados localmente.
- **mostrar_pantalla_principal(app):** Muestra la pantalla principal de la aplicación con un menú lateral que permite acceder a diferentes secciones (inicio, realizar salto, ranking, configuración). Configura el comportamiento y estilo de los botones del menú.
- **enviar_datos_login(app, userTxTBox, passTxTBox):** Envía los datos de inicio de sesión (usuario y contraseña) al servidor. Si el inicio de sesión es exitoso, muestra la pantalla principal; si falla, muestra un mensaje de error.
- **mostrar_pantalla_login(app):** Muestra la pantalla de inicio de sesión donde el usuario puede ingresar sus credenciales o iniciar sesión en modo offline. Proporciona una interfaz estilizada y botones funcionales.
- **iniciar_offline(app):** Cambia el modo de la aplicación a .ºfflinez muestra la pantalla principal, permitiendo al usuario operar sin conexión al servidor.

4.2. Principales variables

Además de funciones, el código cuenta con diversas variables, a continuación explicaremos las principales:

'tema_actual'	(En la función 'cambiar_tema(nuevo_tema)')Esta variable muestra cual es el tema que hay actualmente, si no hay nada por defecto se pone el tema 'claro'
'idioma_actual'	(En la función 'cambiar_idioma(idioma)') Esta variable muestra cual es el idioma que hay actualmente, si no hay nada por defecto se pone el tema 'es'
'archivo_seleccionado'	(En la función 'seleccionar_archivo(archivo_btn)') Esta variable muestra cual es el archivo que se ha seleccionado para calcular el salto, por defecto no hay ninguno.
'server'	Esta variable tiene como objetivo principal guardar el estado actual del servidor, es decir: esta variable representa el socket y su conexión con el servidor,la cual tiene que utilizarse en muchas ocasiones, sobre todo en funciones que realizan acciones sobre el servidor.
'modo'	Esta variable tiene como objetivo principal guardar otro estado del servidor, es decir: esta variable si está conectado o no el usuario con el servidor,la cual tiene que utilizarse en muchas ocasiones, se usa principalmente para iniciar el inicio del programa
'temas'	Esta variable guarda principalmente todos los temas empleados en la app, es decir: esta variable guarda un formato JSON de los temas para posteriormente, poder ser utilizados y llamados por las funciones que cambián el estado o las variables del tema (fondos, música etc.)
'textos'	Con una estructura parecida a los temas, esta variable guardar todos los textos, empleados en la aplicación, en diferentes idiomas (Inglés, Español etc.) para posteriormente, poder cargar las traducciones de cada página.

5. Manual de uso

- **Introducción:**

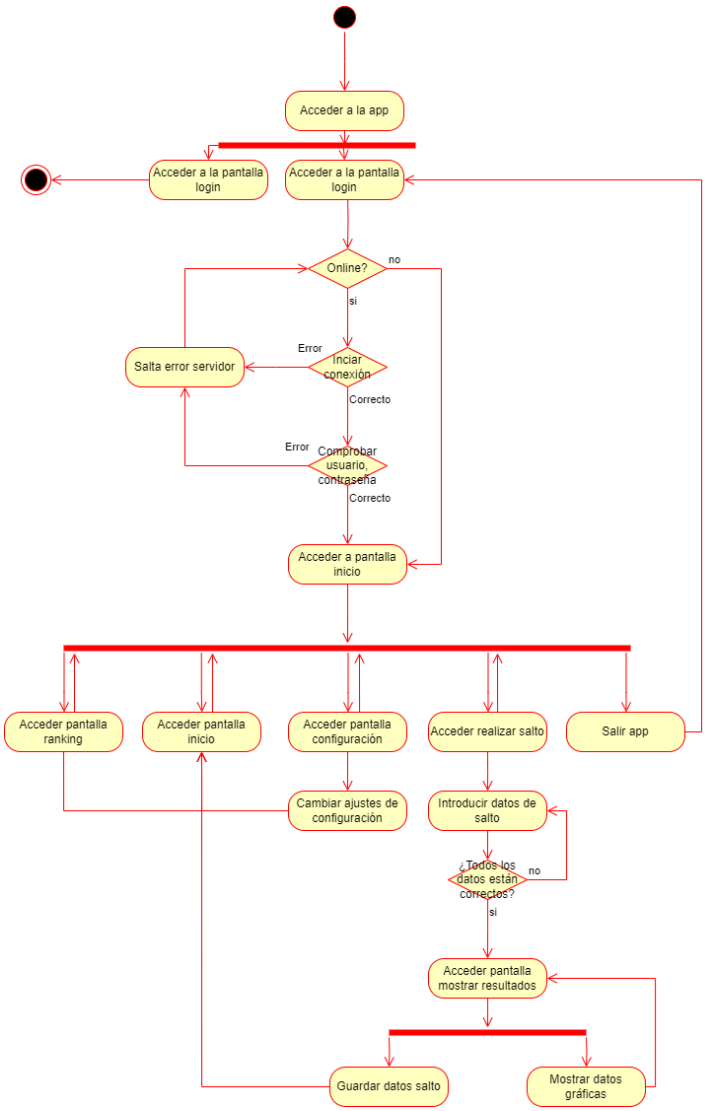
  Este manual proporciona instrucciones para utilizar el programa de análisis de saltos. El objetivo del programa es calcular la altura del salto y otros parámetros relevantes a partir de datos de aceleración.

- **Requisitos previos:**

  - Python 3.x instalado.
  - Bibliotecas necesarias: pandas, numpy, scipy, matplotlib, tkinter, guizero, pygame.
  - Archivo de datos en formato Excel que contenga las columnas: t (tiempo), a (aceleración total), ax, ay, az (componentes de la aceleración).

- **Diagrama de flujo**

  Aquí podemos ver un pequeño diagrama de flujo del funcionamiento del programa el cual vamos a analizar paso a paso:



1. **Iniciar programa:** Es el comienzo del flujo

2. **Inicio de conexión:** Dependiendo de la elección del usuario podrá acceder a la aplicación conectandose al servidor o de forma local, esto podrá hacerlo en la pantalla de login.

3. **En el menú, el usuario tiene cinco opciones principales:**

   - **Acceder a la pantalla de inicio:**

     Aquí el usuario tendrá una pequeña guía para hacer uso de la aplicación.

   - **Acceder a la página de ranking:**

     El usuario al elegir esta opción podrá cargar los datos de la tabla de clasificación del servidor en caso de estar conectado a este, o del archivo de guardado local en caso de no iniciar la conexión.

   - **Acceder a la pantalla de configuración**

     El usuario podrá configurar diferentes parametros siendo estos:
     ◇ **El tema**, el cual cambiara los colores de la aplición y la música

- ◇ **El volumen**, puediendo bajar o subir este a su gusto
- ◇ **El idioma**, podrá elegir entre ingles o español, pero con nuestros archivo de traducciónes que hemos implementado solamente con introducir las claves traducidas tendrá la disponibilidad en cualquier idioma.

- ○ **Acceder a la pantalla de realizar salto:**

  En este el usuario aportara los datos necesarios para realizar los cálculos, siendo estos un archivo excel con los datos del salto, la masa del saltador, nombre y grupo al que pertenece. Una vez todo aportado seremos enviados a la siguiente página.

- ○ **Acceder a la pantalla de resultados:**

  Una vez el usuario entre se realizarán los cálculos con los datos aportados anteriormente, de esta forma diciendo que tipo de pikmin eres.

  También podrás guardar los datos obtenidos en la tabla de clasificación y ver la representación de los datos en diferentes gráficas.

4. **Finalización del programa:**

   En la pantalla de login, arriba a la derecha se encuentra un botón con el cual se podrá realizar la terminación del programa.

6. Mantenimiento preventivo

En este apartado se detallan las medidas tomadas durante el desarrollo del juego para prevenir posibles problemas y mantener su funcionalidad a largo plazo. Aquí dejamos listados los puntos principales a considerar:

- **Comentarios y documentación:**

  Hemos incluido comentarios detallados en el código para facilitar su comprensión y mantenimiento futuro. Además, hemos generado documentación adicional sobre la estructura del programa, las funciones y su interacción. Así mismo diagramas que ejemplifican el funcionamiento del programa.

- **Pruebas unitarias:**

  Hemos desarrollado pruebas unitarias para verificar el correcto funcionamiento de las diferentes partes del programa utilizando **pytest**. Estas pruebas están diseñadas de forma que perdúren mucho tiempo aún realizando cambios significativos en el código.

- **Gestión de versiones:**

  Hemos utilizado un sistema de control de versiones, como GitHub, para mantener un historial de cambios en el código fuente y facilitar la colaboración entre los miembros del equipo. Se han realizado los commits debidos y cada uno se ha documentado en cada lanzamiento.

- **Optimización de código:**

  Hemos realizado esfuerzos para optimizar el rendimiento del programa, identificando y corrigiendo posibles cuellos de botella en el código. Se han aplicado buenas prácticas de

programación para mejorar la legibilidad y la eficiencia del código, y así mismo para mejorar la experiencia del usuario.

7. Mantenimiento correctivo

En este apartado describiremos las acciones tomadas o por tomar para corregir errores y problemas que han surgido después del desarrollo del programa. Algunos aspectos a considerar son:

- **Gestión de errores:**

  Tenemos pensado implementar a futuro un sistema de registro de errores para capturar y registrar los errores que experimentan los usuarios durante el uso del programa. Se ha establecido un procedimiento para priorizar y abordar estos errores de manera oportuna. Por ahora solamente tenemos implementados controladores de errores para a posterior poder implementar esta funcionalidad.

- **Actualizaciones y parches:**

  Se ha pensado en lanzar actualizaciones y parches periódicos para corregir errores conocidos y mejorar la estabilidad y la seguridad del programa. Estas actualizaciones se distribuyen de manera eficiente a través canales como descargas en línea o actualizaciones automáticas.

- **Retroalimentación de los usuarios:**

  Hemos recopilado la retroalimentación de los usuarios (otros estudiantes) para identificar áreas problemáticas y solicitar nuevas funciones o mejoras. Se ha establecido un proceso para incorporar esta retroalimentación en futuras versiones del programa.

- **Monitorización de rendimiento**

  Se ha pensado implementar un sistema de monitorización del rendimiento para detectar y resolver problemas de rendimiento, para posteriormente realizar ajustes en el programa según los datos recopilados por este sistema.

8. Discusión

En este apartado vamos a discutir la solución que hemos hecho, tanto sus ventajas como sus desventajas:

8.1. Ventajas

El programa que hemos creado es un programa muy sencillo e intuitivo, no es complicado de usar y es muy fácil encontrar todos los apartados, además es agradable a la vista y tiene diferentes opciones para poder modificar el programa al gusto del usuario.

8.2. Desventajas

Al haber extraído los datos del acelerómetro del teléfono y en el proceso de los cálculos haber integrales estos cálculos tienen un margen de error por lo que los resultados no son precisos y no se ve la realidad del todo refelejada en ellos.

9. Conclusión

Del desarrollo de este proyecto destacamos las siguientes aportaciones:

En primer lugar, con la implementación de este proyecto hemos tenido la oportunidad de aplicar conocimientos teóricos en un contexto real y con ello se demuestra la eficacia del aprendizaje basado en proyectos (ABP) al proporcionar una experiencia de aprendizaje activa y práctica.

También, al colaborar cuatro asignaturas fundamentales: programación, física, matemáticas y arquitectura de redes, hemos podido ver cómo los diferentes campos de estudio se interrelacionan y complementan, lo cual nos nos es útil para saber abordar problemas complejos desde múltiples perspectivas

Y por último, consideramos que hemos adquirido y perfeccionado habilidades técnicas esenciales para nuestro campo de trabajo como pueden ser: la programación Modular y GUI en Python con el diseño e implementación de una arquitectura modular y una interfaz gráfica de usuario, el análisis de Datoos mediante el uso de herramientas como pandas y numpy para el tratamiento de datos y la aplicación de técnicas matemáticas avanzadas para el análisis de señales, las técnicas de Medición y Validación, de acelerómetros y programas de videoanálisis como Tracker para capturar y validar datos de manera precisa, y la conexión a Servidores mediante Sockets TCP/IP gracias al desarrollo de un cliente de protocolo de aplicación propietario hemos reforzando sus habilidades en redes y comunicación de datos

En resumen, este proyecto multidisciplinar nos ha enriquecido en conocimiento técnico y en el desarrollo de habilidades de colaboración, resolución de problemas y aplicación práctica de la teoría, y consideramos que estas habilidades son fundamentales para nuestro desarrollo profesional y personal, y que nos has servido para prepararnos mejor para los desafíos del mundo laboral en el campo de la tecnología digital y multimedia.

Referencias

[1] Overleaf, *Documentación y Tutoriales de Overleaf*. https://www.overleaf.com/learn.

[2] Matplotlib, *Documentación de Matplotlib*. https://matplotlib.org/stable/contents.html.

[3] NumPy, *Documentación de NumPy*. https://numpy.org/doc/.

[4] Python, *Documentación Oficial de Python*. https://docs.python.org/3/.

[5] Stack Overflow, *Preguntas y Respuestas sobre Programación*. https://stackoverflow.com/.

[6] Datetime, *Documentación de Datetime*. https://docs.python.org/3/library/datetime.html.

[7] JSON , *Documentación de JSON*. https://docs.python.org/3/library/json.html.

[8] Tempfile, *Documentación de Tempfile*. https://docs.python.org/3/library/tempfile.html.

[9] TK, *Documentación de Tkinter*. https://docs.python.org/3/library/tk.html.

[10] PIL, *Documentación de Pillow*. https://pillow.readthedocs.io/en/stable/.

[11] GUI , *Documentación de Guizero*. https://lawsie.github.io/guizero/book/.

[12] Pandas, *Documentación de Pandas*. https://pillow.readthedocs.io/en/stable/.

[13] SciPy, *Documentación de SciPy*. https://docs.scipy.org/doc/scipy/.

[14] Pygame, *Documentación de Pygame*. https://www.pygame.org/docs/.