**README - Proyecto Phygital Human Bone**

**Descripción General**

**Este proyecto se centra en el análisis automatizado de imágenes de microtomografía CT para la detección y análisis de canales de Havers en tejido óseo cortical. Utilizando técnicas de visión por computadora y aprendizaje automático, el software permite identificar estructuras microanatómicas del hueso y analizar su distribución espacial.**

**Componentes Principales**

**El proyecto consta de dos scripts principales:**

1. **fixed-phygital-code.py: Aplicación para la detección automática de canales de Havers utilizando un modelo YOLO entrenado.**
2. **cuadrantes-analyzer.py: Herramienta complementaria que analiza la distribución espacial de los canales dividiendo la imagen en nueve cuadrantes.**

**Tecnologías Utilizadas**

* **Python como lenguaje de programación principal**
* **OpenCV para el procesamiento de imágenes**
* **YOLO (You Only Look Once) para la detección de objetos**
* **Pandas y NumPy para el análisis de datos**
* **Matplotlib para la visualización de resultados**
* **Tkinter para la interfaz gráfica de usuario**

**Funcionalidades**

**fixed-phygital-code.py**

* **Carga y procesamiento de imágenes de microtomografía CT**
* **División de imágenes en segmentos más pequeños para análisis detallado**
* **Detección automática de canales de Havers mediante modelo YOLO**
* **Cálculo de parámetros relevantes: posición, área y distribución de canales**
* **Generación de visualizaciones: mapa de coordenadas y mapa de calor**
* **Interfaz gráfica para interacción con el usuario**

**cuadrantes-analyzer.py**

* **División de la imagen analizada en nueve cuadrantes (matriz 3x3)**
* **Análisis de la distribución de canales por cuadrante**
* **Identificación del cuadrante con mayor densidad de canales**
* **Visualización de resultados con marcado de cuadrantes**
* **Exportación de datos a formato Excel**

**Proceso de Análisis**

1. **Selección de imagen: El usuario carga una imagen de microtomografía CT**
2. **Procesamiento: La imagen se divide en segmentos más pequeños**
3. **Detección: Se aplica el modelo YOLO para identificar canales de Havers**
4. **Análisis: Se calculan estadísticas relevantes (conteo, áreas, distancias)**
5. **Visualización: Se generan gráficos y mapas de calor para facilitar la interpretación**
6. **Análisis espacial: Se estudia la distribución por cuadrantes (opcional)**

**Configuración y Requisitos**

**El proyecto requiere las siguientes bibliotecas de Python:**

* **opencv-python (cv2)**
* **numpy**
* **pandas**
* **matplotlib**
* **pillow (PIL)**
* **ultralytics (YOLO)**
* **torch**

**Modelo de Machine Learning**

**El sistema utiliza un modelo YOLO entrenado específicamente para la detección de canales de Havers en imágenes histológicas. El entrenamiento se realizó con los siguientes parámetros:**

* **Modelo base: YOLOv8n**
* **Epochs: 100**
* **Batch size: 16**
* **Tamaño de imagen: 640x640**
* **Threshold de confianza: 0.4**

**Datos de Entrada/Salida**

**Entrada:**

* **Imágenes de microtomografía CT en formato JPG, JPEG o PNG**

**Salida:**

* **Archivo Excel con coordenadas y áreas de los canales detectados**
* **Visualizaciones gráficas (mapa de coordenadas, mapa de calor)**
* **Imagen con cuadrantes analizados (opcional)**
* **Estadísticas por cuadrante (opcional)**

**Limitaciones**

* **El rendimiento óptimo se obtiene con imágenes de microtomografía CT de alta calidad**
* **Imágenes muy grandes pueden requerir redimensionamiento**

**Uso Recomendado**

**Este software está diseñado para:**

* **Investigación en biomecánica y estructura ósea**
* **Análisis histológico cuantitativo**
* **Estudios de distribución espacial de canales de Havers**
* **Desarrollo de modelos biomiméticos de tejido óseo**

**Resumen Detallado de fixed-phygital-code.py**

**Descripción General**

Este script analiza imágenes de microtomografía CT para detectar canales de Havers en el hueso cortical mediante técnicas de aprendizaje automático. El programa procesa la imagen, identifica los canales, y proporciona análisis estadísticos y visuales de los resultados.

**Funcionalidades Principales**

**configure\_window**

Esta función establece la apariencia básica de la ventana de la aplicación:

* Define el título de la ventana
* Establece el tamaño a 1000x1000 píxeles
* Configura un fondo azul oscuro (#001f3f)

**configure\_button**

Personaliza la apariencia de los botones:

* Color verde para los botones (#4CAF50)
* Texto en color blanco
* Fuente Helvetica de tamaño 12
* Añade efectos visuales cuando el ratón pasa por encima (cambia ligeramente el tono del verde)

**select\_image**

Permite al usuario seleccionar una imagen para analizar:

* Muestra una ventana con el título "PhygitalBone 2.0"
* Presenta un botón "Load Image" para abrir el explorador de archivos
* Guarda la ruta de la imagen seleccionada para su uso posterior

**divide\_and\_save\_image**

Divide la imagen original en segmentos más pequeños para facilitar el análisis:

* Corta la imagen en 150 piezas (organizadas en 15 columnas y 10 filas)
* Guarda cada fragmento como un archivo PNG separado
* Registra las posiciones de cada segmento para poder reconstruir la imagen completa después

**calculate\_box\_centers\_and\_areas**

Calcula información de cada canal de Havers detectado:

* Determina las coordenadas del centro de cada canal
* Calcula el área de cada canal (tratándolo como una elipse)
* Asocia cada canal con el segmento de imagen donde fue detectado

**plot\_centers**

Crea un gráfico de los canales detectados:

* Muestra la imagen original como fondo
* Coloca puntos azules donde se ha detectado cada canal
* Etiqueta el gráfico como "Mapa de coordenadas de canales de Havers"

**plot\_heatmap**

Genera un mapa de calor que muestra la densidad de canales:

* Las zonas con mayor concentración de canales aparecen en tonos más cálidos
* Superpone el mapa de calor sobre la imagen original
* Permite visualizar fácilmente patrones de distribución

**calculate\_distance\_matrix**

Calcula la distancia media entre los canales detectados:

* Mide la distancia entre cada par de canales
* Calcula el promedio de todas las distancias
* Este dato es útil para analizar la distribución espacial de los canales

**display\_results**

Muestra los resultados del análisis en una interfaz con pestañas:

1. **Coordinates**: Datos numéricos de los centros y áreas de los canales
2. **Plot**: Visualización gráfica de los canales sobre la imagen original
3. **Analysis**: Resumen con número total de canales, área promedio y distancia media
4. **Heatmap**: Mapa de densidad de canales

**main**

Función principal que orquesta todo el proceso:

1. Muestra la interfaz inicial para seleccionar la imagen
2. Verifica y redimensiona la imagen si es necesario
3. Divide la imagen en segmentos
4. Carga el modelo YOLO entrenado para detectar canales de Havers
5. Analiza cada segmento y recopila los datos
6. Genera visualizaciones y resultados estadísticos
7. Muestra los resultados al usuario

**Flujo de Trabajo**

1. El usuario selecciona una imagen de microtomografía CT
2. El programa muestra un mensaje de "Processing, please wait..."
3. La imagen se divide en 150 segmentos más pequeños
4. Cada segmento se analiza con el modelo YOLO para detectar canales de Havers
5. Se recopilan y procesan los datos de todos los canales detectados
6. Se generan visualizaciones (gráfica de puntos y mapa de calor)
7. Se presentan los resultados en una interfaz con pestañas
8. El usuario puede guardar los gráficos o explorar los datos numéricos

**Resumen Detallado de cuadrantes-analyzer.py**

**Descripción General**

Esta aplicación complementa el análisis de canales de Havers dividiendo la imagen en nueve cuadrantes para estudiar la distribución espacial de los canales y determinar qué zonas presentan mayor densidad.

**Funcionalidades Principales**

**configure\_window y configure\_button**

Similares a las funciones del script anterior, establecen la apariencia visual de la ventana y los botones.

**reconstruir\_imagen\_con\_detecciones**

Toma la imagen original y los datos de detecciones para crear una visualización:

* Lee la imagen original y el archivo Excel con las coordenadas
* Dibuja un círculo verde alrededor de cada canal detectado
* El tamaño de cada círculo se basa en el área calculada del canal
* Guarda la imagen resultante para su posterior análisis

**analizar\_cuadrantes**

Divide la imagen en 9 cuadrantes (3x3) y analiza la distribución de canales:

* Calcula las dimensiones de cada cuadrante
* Clasifica cada canal dentro del cuadrante correspondiente
* Suma el área total de canales por cuadrante
* Identifica el cuadrante con mayor densidad de canales
* Dibuja líneas blancas para delimitar los cuadrantes
* Resalta el cuadrante de mayor densidad con un rectángulo rojo semitransparente
* Añade información numérica sobre cada cuadrante (área total y número de canales)

**visualizar\_resultados\_cuadrantes**

Presenta los resultados del análisis por cuadrantes en una interfaz gráfica:

* Muestra la imagen con los cuadrantes marcados
* Proporciona una pestaña con datos detallados por cuadrante
* Permite guardar la imagen del análisis
* Ofrece la opción de exportar los datos a un archivo Excel

**main**

Función principal que coordina todo el proceso:

1. Presenta la ventana inicial con una descripción del programa
2. Permite al usuario seleccionar la imagen original y el archivo Excel con las detecciones
3. Muestra una ventana de progreso durante el análisis
4. Reconstruye la imagen con las detecciones marcadas
5. Realiza el análisis por cuadrantes
6. Visualiza los resultados en una interfaz gráfica