Introduction

Welcome to the Interactive Plot Visualization Tool - a powerful Python-based application designed to process HDF5 sensor data and generate interactive HTML reports using Plotly. This manual guides you, the user, through the installation, usage, and understanding of the tool’s workflow and features.

Overview

This tool reads configuration from an XML file and a JSON file to interpret sensor data stored in HDF5 format. It processes the data, performs statistical analyses, and generates interactive HTML visualizations such as scatter plots, histograms, and box plots.

The application is modular, with clear separation between configuration parsing, data processing, and presentation layers.

Required Python packages (see requirements.txt)

Steps

Clone or download the repository.

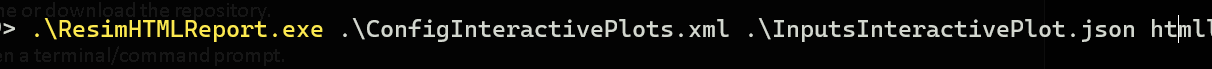
Open a terminal/command prompt.

Navigate to the binary directory.

Usage

Run the tool from the command line as follows:

ResimHtmlReport <config\_file.xml> <input\_plot\_json\_file.json> [output\_directory]



Parameters:

config\_file.xml: Path to the XML configuration file defining HDF5 file type and parsing rules.

input\_plot\_json\_file.json: JSON file specifying input-output mapping for plots.

output\_directory (optional): Directory where HTML reports and logs will be saved. Defaults to html if not provided.

If the output directory is omitted, the tool creates an html folder in the current working directory.

Input Files

XML Configuration File

Defines the HDF5 file type and parsing instructions. And Must be a valid XML file.

JSON Plot Configuration File

Maps inputs to outputs for visualization.Specifies which datasets to plot and how.Must be valid JSON.

Output

HTML Reports: Interactive Plotly-based visualizations saved in the output directory.

Logs: Detailed logs saved as logs.txt inside the output directory.

The tool creates the output directory if it does not exist.

Logging

Logs are written both to the console (INFO level) and to a log file logs.txt (DEBUG level).

Logs include timing information for parsing and processing phases.

Helps in debugging and performance monitoring.

Troubleshooting

Missing files: The tool checks for the existence of the XML and JSON files before running. If missing, it will print an error and exit.

Incorrect usage: If command-line arguments are incorrect, usage instructions will be displayed.

Permission issues: Ensure you have write permissions to the output directory.