Ommo Summer 2024
Software Engineer Internship Project

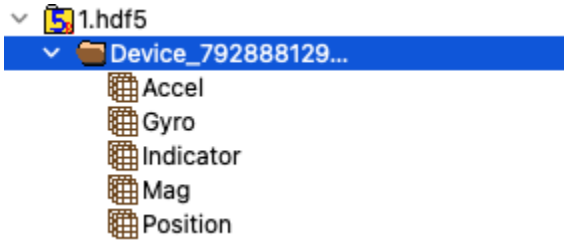
Create a command line program to process and aggregate data files logged from Ommo's tracking system software.

Data File Description

Ommo's tracking system outputs logged data files in the HDF5 format. A useful library to reference is the Python H5py library: https://docs.h5py.org/en/stable/quick.html#quick.

The root of the file contains a list of Groups, where each Group contains the logged data for an Ommo tracking device.

For example, the following image shows a file named "1.hdf5" containing one tracking device called Device_792888129_0.



Under each Group that represents a tracking device, there are a number of datasets that were logged into the file. **For this project, you only need to consider the "Position" dataset that represents the 3D position outputs of the sensors on the device.**

The Position dataset has the following dimensions: [samples, sensor #, xyz]
For example, a Position dataset with a shape of [1000, 2, 3] means this data set has 1000 samples and 2 sensors.
In Python, accessing all samples of the 2nd sensor (index 1) looks like the following:
position_data = dataset[:, 1, :]

Some sample data files have been provided as part of this project.

Objectives & Requirements

1. The program should take 2 arguments
    a. The first argument is the folder where the data files to be processed are
    b. The second argument is the output folder where the processed results will be output to
    c. The program must correctly ignore files that are not HDF5 data files in the input folder, including HDF5 files that are not properly formatted
    d. The program must handle errors in user input correctly (e.g. no valid data files in the input folder, input folder doesn't exist, output folder doesn't exist)

2. The program should output 2 CSV files to the output folder
    a. File 1 - average position output
        i. The program should compute the average position for each sensor on each device in all data files in the input folder.
        ii. The output files should be a CSV file with the following requirements
            1. Each row correspond to one data file
            2. The first column of a row should be the data file name
            3. For each row, there should be 3 columns corresponding to the average x,y,z position for each sensor in that data file
                a. For example, if a file has 2 devices with 3 sensors in each device, there will be a total of 18 columns, excluding the first column with the data file name
            4. Each column should represent the same sensor data from the same device across different data files
                a. This means that for a column, the column cannot contain data from two different sensors across different data files
                b. The program should correctly handle any sensors that are missing from a data file by skipping the columns corresponding to the sensor for the row representing that data file
    b. File 2 - max distance output
        i. The program should compute the maximum euclidean distance of the samples for each sensor on each device in all data files in the input folder
        ii. The format and requirements for file 2 shall be the same as the requirements of File 1, except the following
            1. Instead of 3 columns for each sensor's XYZ data, there will be only 1 column for each sensor, representing the maximum euclidean of its samples in each data file
    c. The columns in each output file should be identifiable with which sensor the data came from, such as the device name and a sensor index
3. The program should be efficient in handling a large number of files
4. Write unit tests to verify the correctness of your implementation, including any edge cases and error handling
5. The project can be implemented in a language of your choice, but we must be able to compile, if necessary, and run your code as a command line program
    a. If external libraries or environments are required, you must provide instructions for building or setting up the runtime environment

You will be evaluated on the correctness of your implementation and the overall design.