Telemetry Windowing Project Description

Version 1.0

# Purpose

The purpose of this project is to develop a C# function that will accept a date range, an integer, and a list of SensorData classes that contain DateTime attributes. The function will take the list of items and window them to a new list of SensorData classes of the size passed in the ninteger.

# Function Signature and Interface

Below is the function signature that will be used to interface with the existing code base. The function developed will have to match this signature and the input/output types.

List<SensorData> telemetryWindow(List<SensorData> inData, DateTime startDate, DateTime endDate, int dataCount);

inData: The list of SensorData classes passed in for windowing.

startDate: The start of the DateTime range for windowing.

endDate: The end of the DateTime range for windowing.

dataCount: The number of SensorData classes requested back.

Below is the declaration for the SensorData class.

public class SensorData

{

public string deviceID;

public string sensorID;

public string data;

public DateTime timeStamp;

public int errorcount;

public SensorData()

{

}

}

# Windowing Description

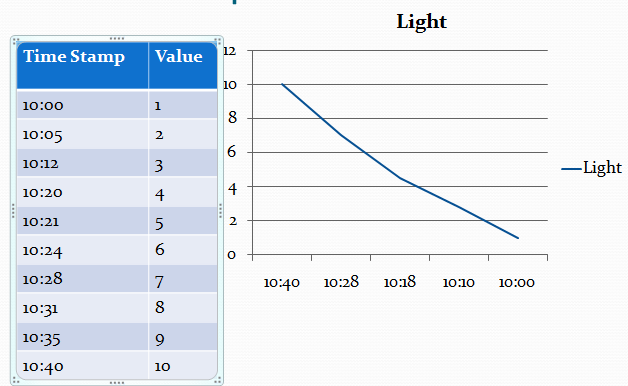
Windowing a collection of data is the process of taking a DateTime start, DateTime finish, dataCount, and a list of data. Using the data the function will create a list of data that will imitate a timeline of data with a number of points equal to the dataCount. The function may have to fill in missing data using averages or use averages to condense multiple points into one. Data may not have consistent time intervals between data points so the function must be able to account for both situations for a set of data.

In our example the SensorData class represents our data points, to create the windowed list of SensorData points the SensorData class will have to be created and filled with data from the list of SensorData classes passed in. The only difference will be in the data attribute and the timeStamp attribute. These will be modified before being inserted into the new list and returned. The data attribute will be usually being an average to represent a missing data point or many condensed data points. The timeStamp data point will represent the time interval that point represents.

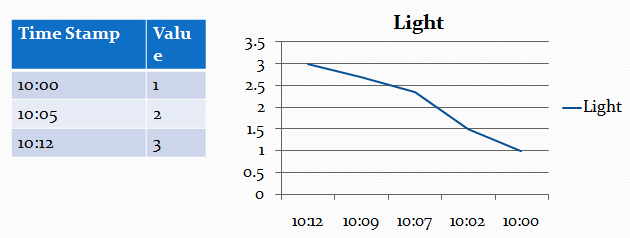
# Examples of Windowing

These are rough examples of what the functions inputs and outputs will look like. The function will not be responsible for graphing, only returning the List<SensorData> data structure.

Please remember these are rough examples and are NOT using accurate numbers. The real windowing function will use better mathematics than this.



In this example we have too many points and must trim them down and condense those using averages. The function must take 10 points and using averages trim them down to 5 points along the DateTime range.



In this example we don’t have enough points to fill 5 points. In this example the function would take 3 points and use averages to create artificial points across the DateTime range.