


Introduction

This guide is intended to be used alongside Luke Costello's repository of master's thesis work. The guide will overview the general folder structure, as well as the specific usage with respect to the Cal Poly Wind Turbine. The repository itself is a pared-down version of the full directory of files; those not included were either unfinished programs or files deemed not relevant for others. Regardless, the full directory of files can also be found in the compressed directory "FullRepository.zip".

The main folders and first-level subfolders may be found in the table below.

Folder	Subfolder	Description
Code & Programs	-----	
	Implementation	All code necessary for the implementation of the LifeLine Fault Detection Algorithm. This includes the Python-based algorithms, and MATLAB code used to form text files necessary for the Python implementation.
	MATLAB Algorithms & Validation	The first version of the algorithms developed, and the code used to calculate each algorithm's performance.
	Processing	MATLAB files used to combine Squirrel and LifeLine datasets
	FigureCreation	MATLAB files used to create figures for the final thesis
CAD	-----	
	DesktopSign	A desktop sign for the Cal Poly Research center. Designed for 3D printing; includes a slot for 3 M3 nuts that must be set halfway through printing the "DesktopMount" part.
	MountingPlates	Mechanical mounting for the Raspberry Pi and Nucleo devices in the wind turbine's electrical box.
	FieldSigns	Waterjet profiles for the new signs at the wind turbine
Data	-----	
	Processed	Data processed and ready for use with the MATLAB algorithms, cross-validation study, and for creating training .txt files for the Python implementation
	Raw	(Mostly) raw data collected from the wind turbine. Some minor error-fixing in LifeLine .txt files has been applied.
Figures	-----	Many of the figures created for the thesis, made via the website draw.io. Most other figures were created using code found in the "MATLAB Algorithms & Validation" folder. Files are in .xml format – use of the aforementioned website, or other such viewer, is necessary to export a picture file.

Monitoring Software Usage

In general, the live monitoring software created by the end of Luke Costello's thesis may be run on Raspberry Pi controller by opening the command line (this can be accessed via the  logo in the top left corner of the Raspberry Pi interface), and typing the command:

```
python3 lifeline_FDC_lowpower.py
```

For a more detailed guide on using the Raspberry Pi, please see the QuickStart guide found in the main directory of the repository.

For the latest version of this repository, please see the GitHub repository:
https://github.com/elceenor/LifeLine_FaultDetection

As a part of this thesis, three files have been designed: NSET.py, SPRT.py, and AFFT.py. The NSET.py and SPRT.py files correspond to the NSET+SPRT algorithm, and the AFFT.py file corresponds to the AFFT algorithm. These files require three text files: AFFT_threshold.txt, NSET_inverse.txt, and NSET_memory.txt. They can be found in the *Code > Implementation > Python* folder; in addition, if it is desired for new versions of these files to be created, they can be generated using the program *GenerateTextFiles.mlapp* found in the *Code > Implementation > File Generation Code* folder.

The Python modules were created with the intent of allowing for easy implementation in any live monitoring software; the first version may be found in *Code > Implementation > Python*, in the *lifeline_FDC.py* file.

For the latest version of this repository, please see the GitHub repository:
https://github.com/elceenor/LifeLine_FaultDetection