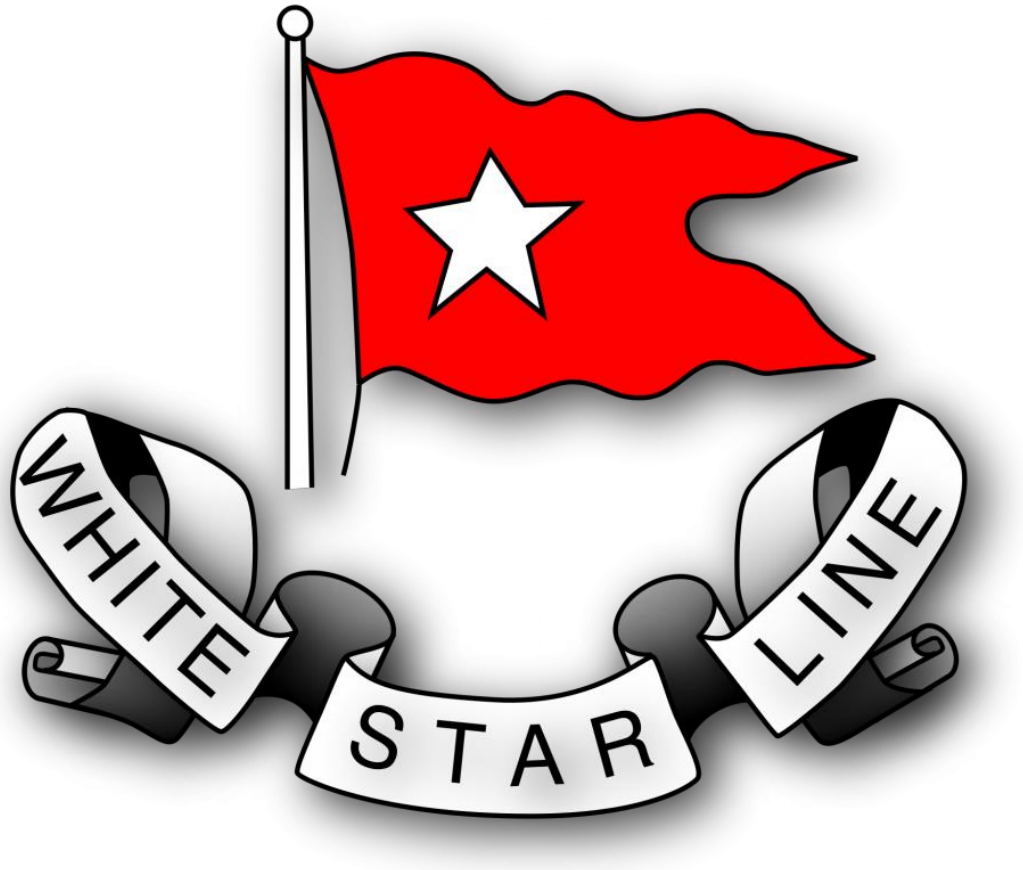
White Start Line Software Development Guide



Javid Yousaf

GEOG5003M Assignment 2

Contents

Introduction

The White Star Line Company provides transatlantic voyages and cruises for passengers. To improve safety and mitigate risk to customers and the company, it has been decided to send an iceberg towing vessel with each journey to assist in removing icebergs out of the shipping lane where they pose a potential safety risk.

To assist with this requirement a software tool is needed that can assist employees of the White Star Line company to detect and identify icebergs along the route and determine if they are towable or if the route needs to be modified to prevent a collision.

This will be achieved by the software tool by analysing radar and lidar data for a given are of the seas.

Requirements

The following requirements have been extracted from the provided specification document

Required:

1. Create a GUI to contain the functionality of the application.
2. Read radar data file - each pixel is a m^2 grid is 300 x 300.
3. Read Lidar data file - each pixel is a m^2 grid is 300 x 300.
4. Display Lidar data.
5. Display Radar data.
6. File chooser to select loadable files.
7. Process button – executes the default process to read the files for a single berg and analyse.
8. Asses what is ice from radar data.
9. Asses height from lidar data.
10. Calculate total mass above sea level.
11. Calculate total mass.
12. Calculate volume of ice.
13. Display total mass.
14. Display total volume.
15. Display if iceberg can be towed.
16. Save information to file.

Additional (for extra marks):

17. Read radar and lidar files for multiple bergs and process data.
18. Identify individual bergs in the area and process.
19. Highlight the towable bergs in green and the non-towable in red.

Design

The following design decisions were made after consideration of the requirements above:

1. The specification document dictates that the application is to be developed in Python with a GUI interface therefore it will be developed for a desktop platform in its first iteration. This will be primarily for Linux, Microsoft Windows and Apple Mackintosh Mac OS platforms that can support a Python environment.
2. For the sake of simplicity, it seems appropriate to have all the controls and analysis on one GUI windows as the desktop environment has the required screen real estate.
3. The system will have a default processing feature to automatically load the radar and lidar files and display the data in charts and complete processing.
4. Option to manually load the radar and lidar files by selecting them from the host computers file system. This will allow different data to be processed by the application.
5. Calculate the total mass above sea level (M_{asl}) of the iceberg. The radar data will give information about the texture of the berg – a value > 99 is ice.
6. Calculate the total volume above sea level of the iceberg. The lidar data will give information about the height – a value of 10 is equivalent of 1 metre.
7. Calculate total mass (M_{tot}) of iceberg from the knowledge that 10% is above water. $M_{tot} = M_{asl} \times 10$.
8. Calculate the total volume of the iceberg if we k now that 900 Kg/m^3 is density of ice. Volume (m^3) = M_{tot} / 900.
9. Calculate if the iceberg is towable. The tug can tow the berg if its total mass (M_{tot}) < 36 million kgs.
10. The results of the analysis will be displayed in the user interface.
11. The results panel will also show if the iceberg can be towed by the accompany towing tug. This will be coloured green if the iceberg is towable and red if it is not.
12. Export button to write the analysis results to a file that can be saved on the local file system.
13. Clear button to reset the application so that new data can be loaded. This will clear the loaded files, charts and results.

Prototype

The following prototypes were created to experiment with screen layout of the required components as specified in the previous section.

Fig. a. Chart panel on right hand side.

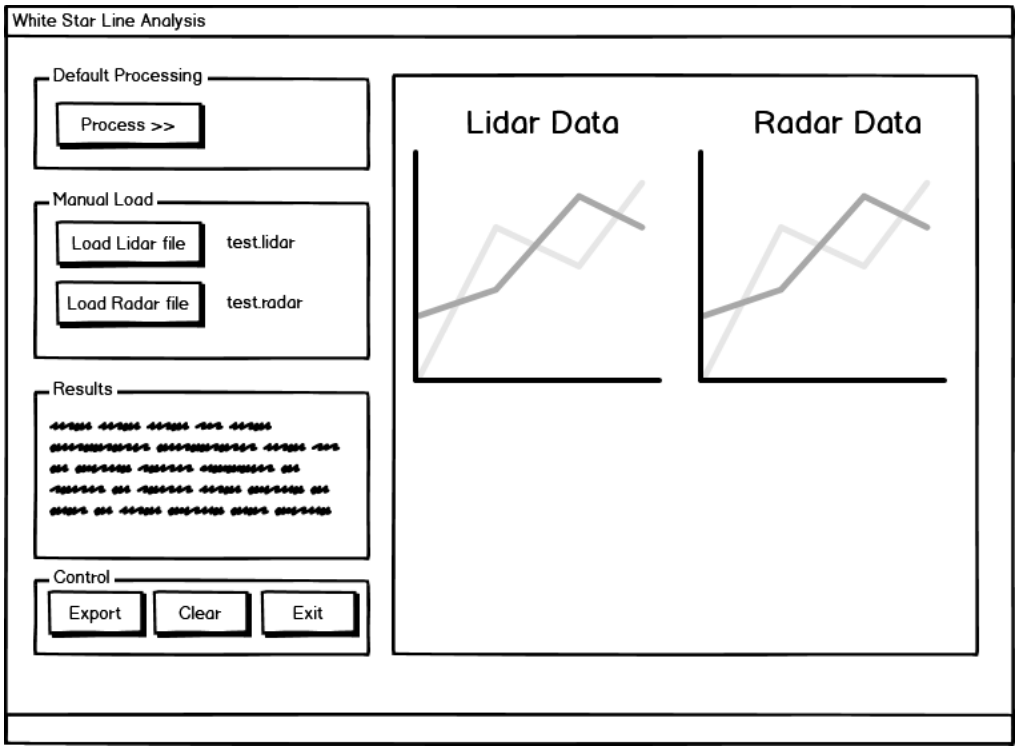
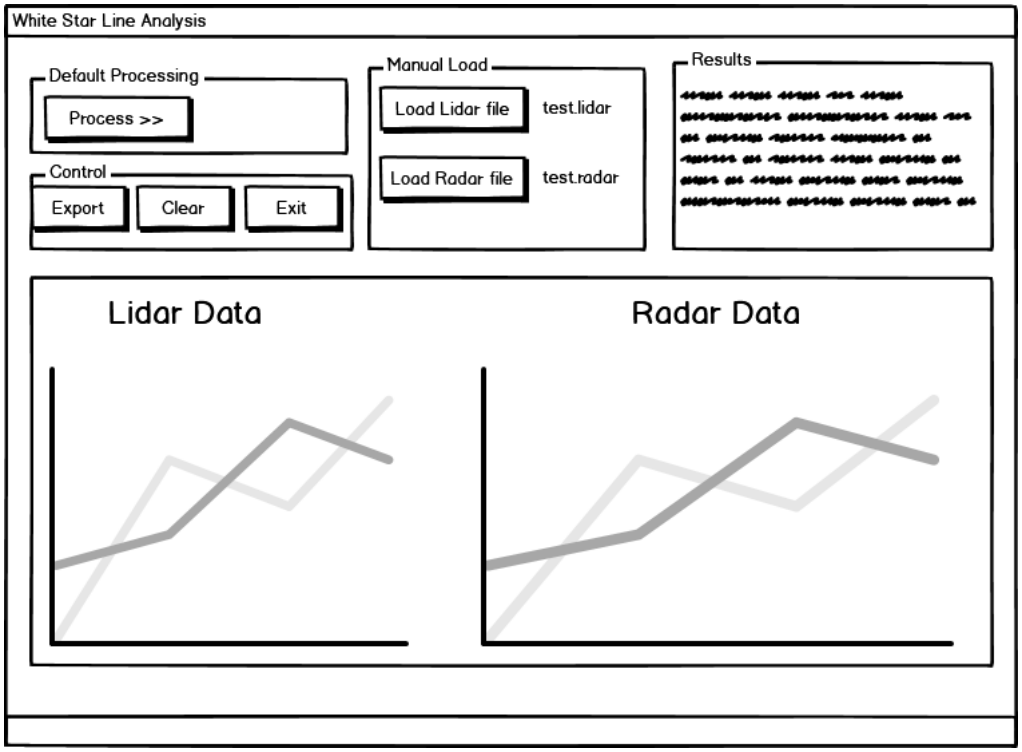


Fig. b. Chart panel on bottom.



Although both prototype layout designs would be adequate for the development and would allow the data to be loaded and results to be displayed clearly, the design with the chart panel on the right-hand side (Figure a.) was chosen as the preferred layout after consulting with stakeholders.

Development

The application was developed using the Python programming language version 3.x.x.

An interface was created using the *Tkinter* module as this is a popular environment and there are many resources for a novice developer to use. *Tkinter* also integrates well with the three main platforms (Microsoft Windows, Linux, Apple Mac OS) that this application is required to work with.

Visual Studio Code with the Microsoft Python extension was used to write, debug and execute the code.

The following classes were identified:

A main execution class that loads the GUI and controls all interaction and processing within the application.

An element class that can store the lidar and radar and derived/calculated information about each 1m x 1m area of the sea.

An iceberg class to store the processed and calculated data for the icebergs identified.

Testing

The application was tested using unit testing where each of the individual components was test individually:

1. Application window
    a. Open and close application multiple times to observe behaviour.
    b. Attempt to open multiple instances of the application.
    c.
2. Buttons
    a. Process button
        i. Label text correct and readable.
        ii. Button runs the process when clicked.
        iii. Button re-runs the process when clicked multiple times.
    b. Load Lidar and radar buttons
        i. Label text correct and readable.
        ii. Buttons open the select file dialog box.
        iii. Only allow selection or .lidar or .radar files respectively.
        iv. Processing occurs correctly when file selected.
    c. Exit button
        i. Closes the application and releases/ends the python application process successfully.
    d. Clear Data button
        i. Clears all the loaded data in the system.
        ii. Clears the charts in the chart panel.
        iii. Resets the labels in the interface correctly.

        iv. Has no adverse effect when pressed multiple times.
3. Labels
    a. Check all label text appears correctly and without typos.
    b. Check that label text and styles are reset to default when clear is pressed.
4. Canvas/charts
    a. Check that the chart data is displayed in the interface correctly.
    b. Check that the axes have the correct range ticks displayed.
    c. Check that the colour bar is showing the correct range for each chart.
5. Export file
    a. Open the exported file and check that the correct data is displayed and matched what is on the screen in the application.
    b. Check that the date time stamp in the chart is correct.

Limitation and Future Development

The scope of the brief was to develop an application that could detect a single iceberg in the sample area, covered by the lidar and radar files provided, and this has been achieved by this phase of development.

For additional marks it was indicated that the application could load a file that contained multiple icebergs and process each one to individually indicate if they could be towed. Some attempt was made to work out an algorithm to do this, however this was not achieved in this development.

There was a degree of complexity to do this as it would mean creating an algorithm to detect the edges of each discrete iceberg by examining every element of the lidar and radar data and using something similar to a a modified D8 algorithm (https://www.mathworks.com/matlabcentral/fileexchange/21682-d8-algorithm-for-hydrological-models) that is used for flow direction or some sort of edge detection such as https://datacarpentry.org/image-processing/08-edge-detection/. This will require further research.

For future development the application could to be refactored to handle multiple berg data as this would be more useful.

A workaround using this application would be to split up the multiple berg data into single bergs and then load each set of files into the application and process them one at a time. The application has the ability to load other files rather than the default ones provided so this could be a solution in the interim.

References

Numerous web references were used to assist with Python coding techniques and knowledge:

1. https://stackoverflow.com/questions
2. https://docs.python.org/3.8/
3. https://realpython.com/python-gui-tkinter/

Attributions

1. White Star Line logo. Whistlerpro / CC BY-SA (https://creativecommons.org/licenses/by-sa/3.0)

Revision

Date	Comment	Version	Author
15th May 2020	Initial version	1.0	Javid Yousaf