Research on Aircraft Noise Model

by Alexander Zhu

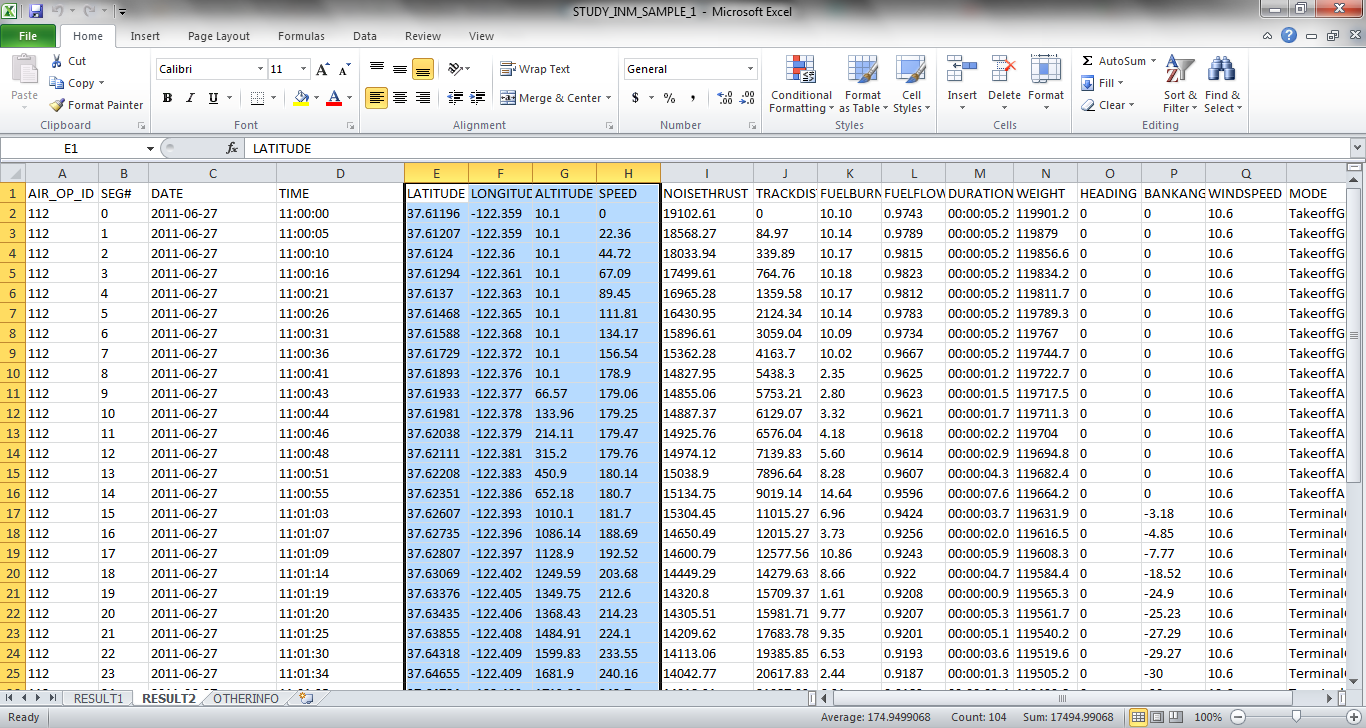
User’s Manual

Purpose of ModifiedSAR

The ModifiedSAR is designed as an extension of the SAR application in NASA WorldWind to simulate the noise model as an aircraft travels along a given trajectory. The program displays a visual representation of the noise levels at a specific set of points along the world map known as receptors. By the conventions in the program, the color of the receptors indicate the amount of noise at that location, with green indicating least noise and red indicating most noise.

Text Files in ModifiedSAR

The ModifiedSAR has the option of running the same way the SAR application will run, but the added features of the receptor grid are input using text files. The text files that the ModifiedSAR will accept are pasted in through Microsoft Excel as follows:



Take the columns that correspond to the information that is desired in the text file, highlight them in excel, and then copy paste into the text file. Pictured above, we highlighted the position and speed so we are looking to create a position and speed file. Be sure to note that the labels (ie LATITUDE, LONGITUDE etc. ) are also included in the text file. The program is designed to account for that. However, the program was not designed to account for extra blank lines at the beginning or end of the file. That will cause an ArrayIndexOutOfBounds error. The kinds of text files that are accepted include receptor files, thrust files, and Position & Speed files. To input any of these files, use the respective buttons that are labeled when the program runs. If no files are input, then the program will run on the default files, which correspond to a flight simulation of a plane heading to San Francisco International Airport.

Running the Program

As mentioned, running the ModifiedSAR program the same way as the original SAR will work, as in the same kinds of tracks that can be loaded into the SAR application can also be loaded into the ModifiedSAR. However, ModifiedSAR will also take input of its own. When the program is first run, a JFileChooser window will pop up, allowing the user to choose a file. This is to select a speed file if the user wants to load in a text file with only speeds. There is a default file for this though, so there is no need to enter anything. If the default is desired, then clicking cancel will run the ModifiedSAR as well.

The default files are included in the project, so that the file paths are always the same. Here we list the default files that the program uses if given absolutely no input:

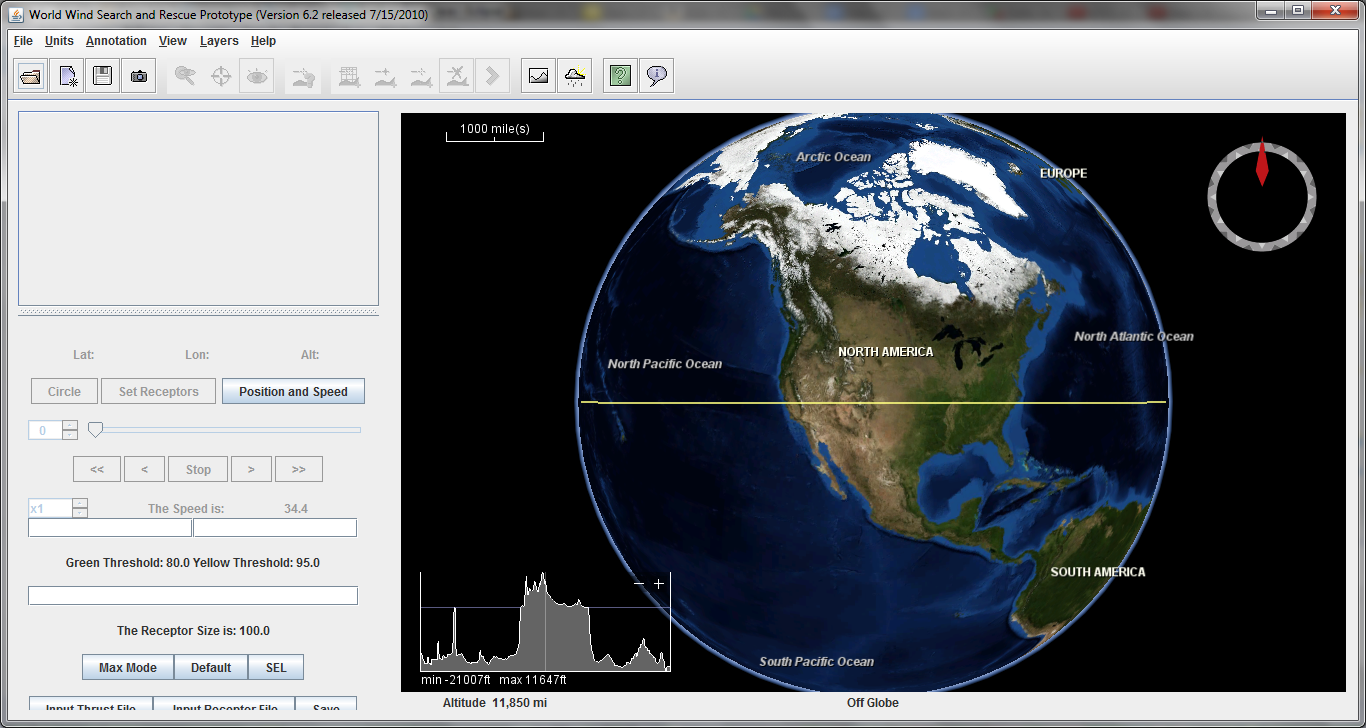
CF565C.txt is the thrust file.

Receptor\_Grid01\_SFO.txt is the default receptor grid file.

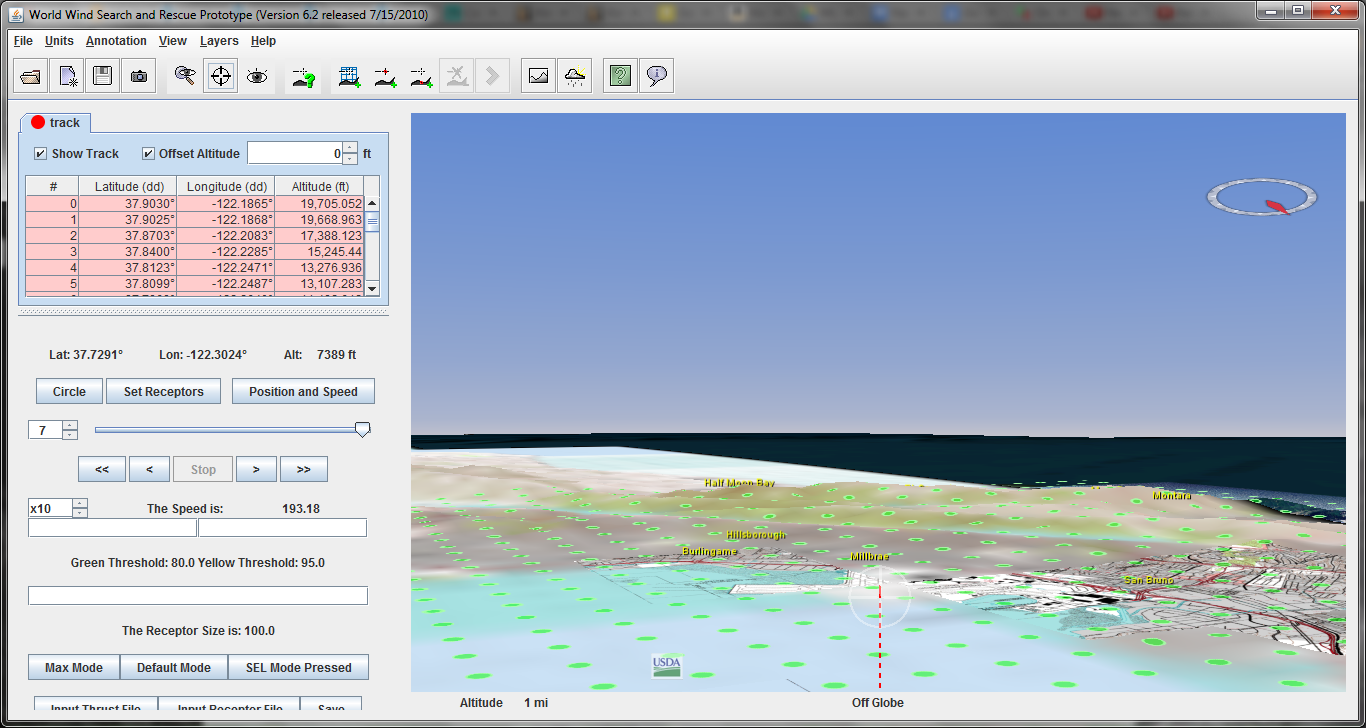
SFOSpeeds.txt is the default speed file.

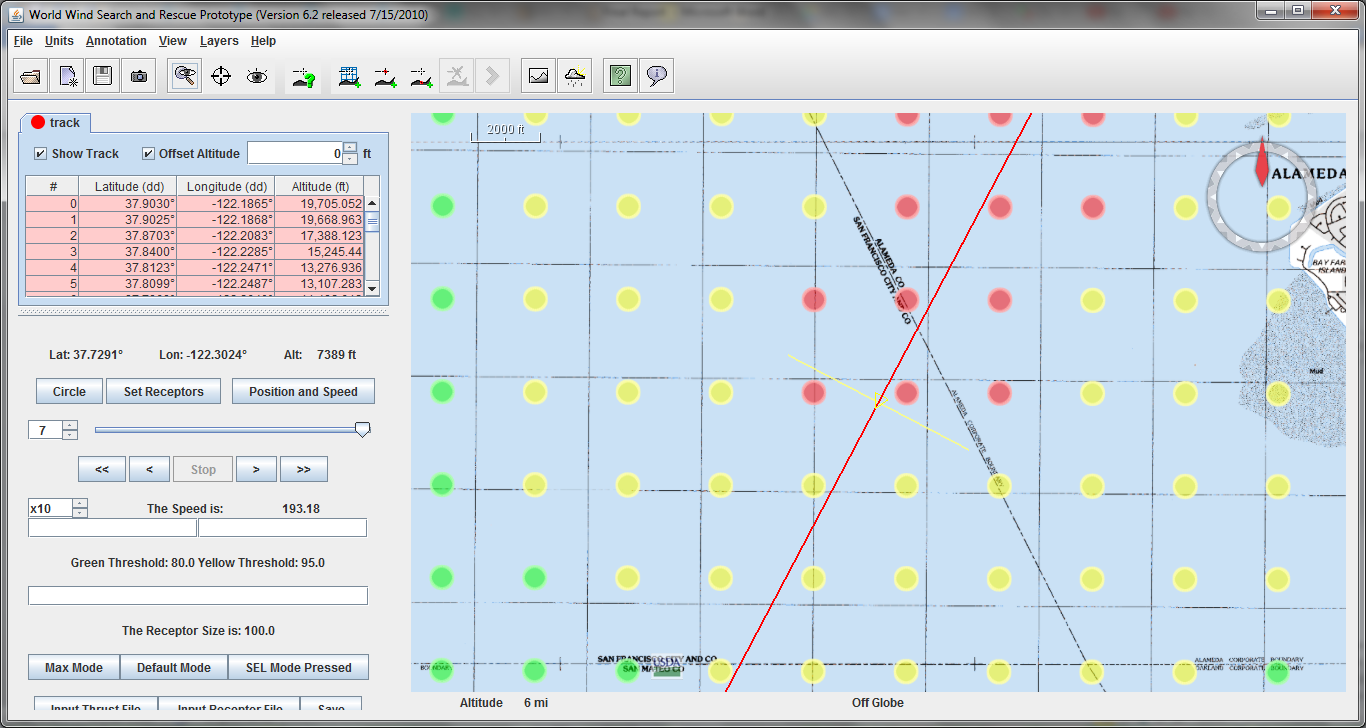
There is no default Position and Speed file, but it can be created using the default receptor grid file and default speed file.

Once the program starts running, the WorldWind globe will pop up, and the user can enter in a file with positions and speeds by clicking the button labeled “Position and Speed”. The operations are similar with the receptor and thrust files. The save button will also open up a JFileChooser window, but the purpose is to input the file in which the user desires to save the receptor locations and noise levels in. Note that the noise levels saved by clicking the save file will correspond to the current mode that the program is running in.

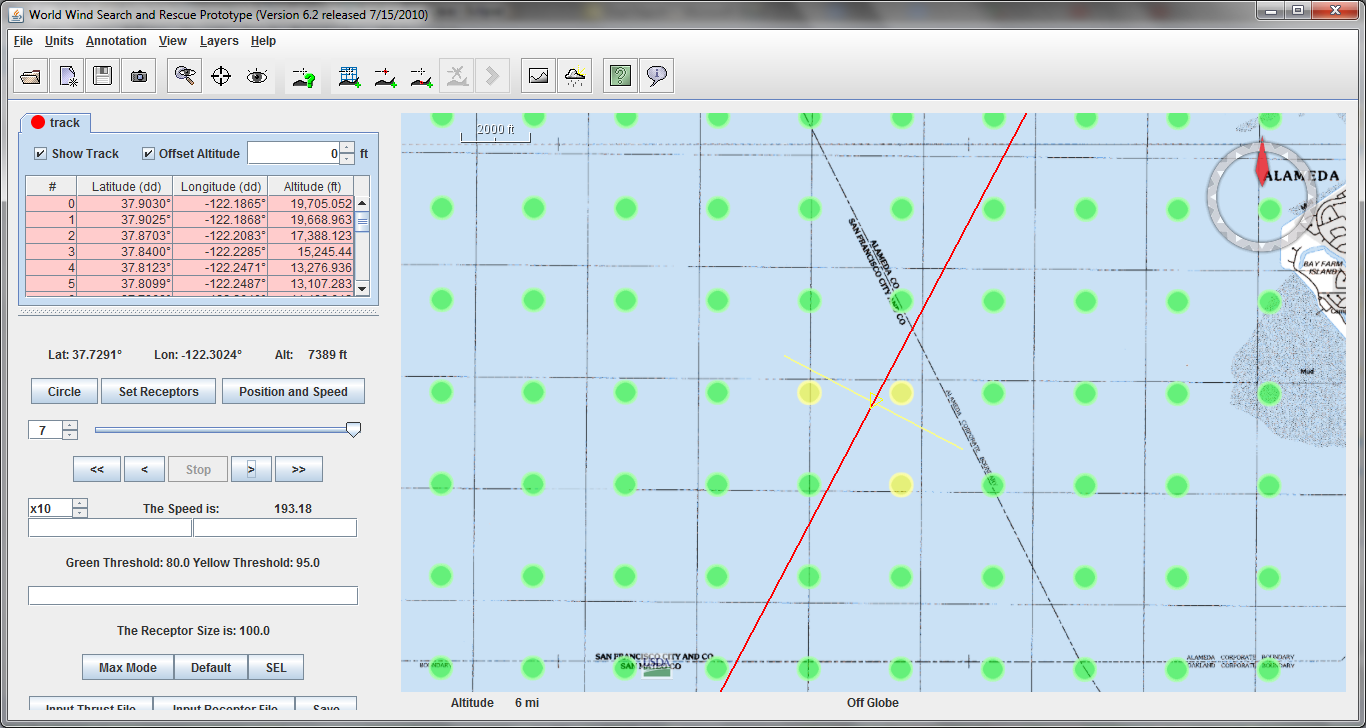


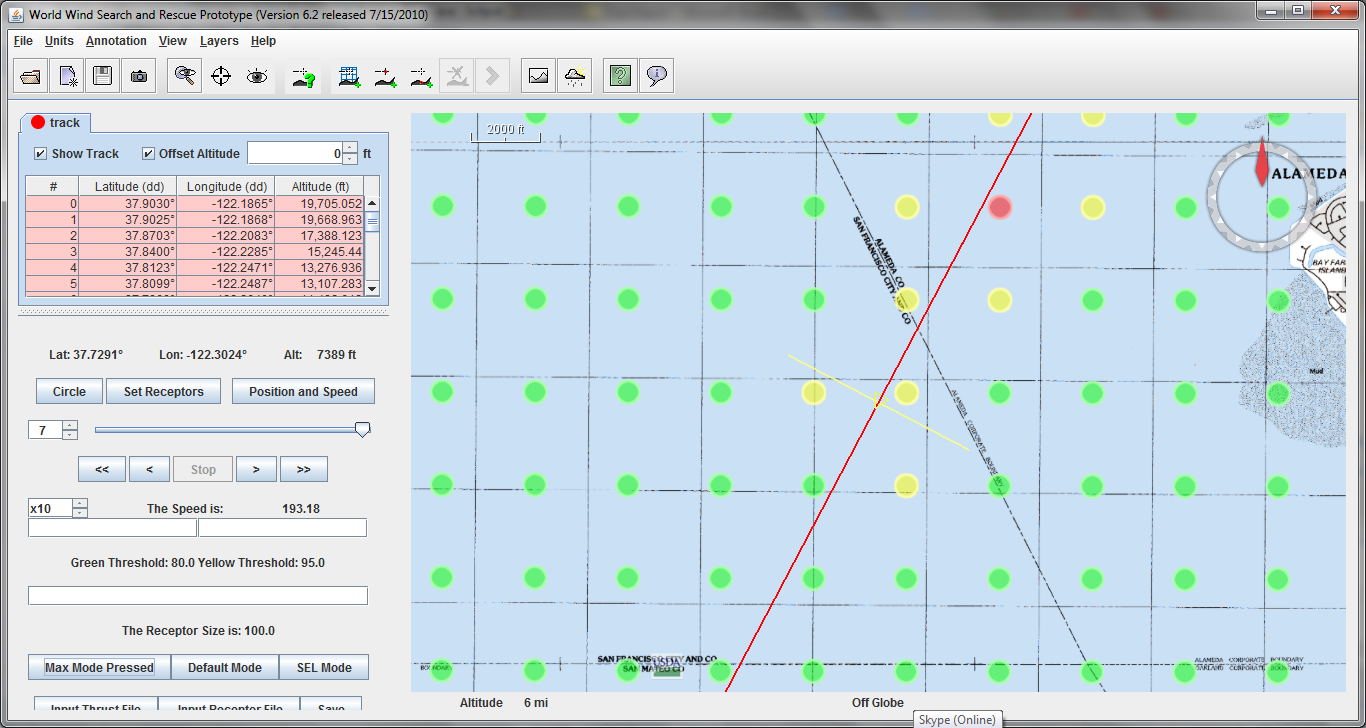
Note that the three buttons at the top that are to the right of the camera button are used to adjust the view.

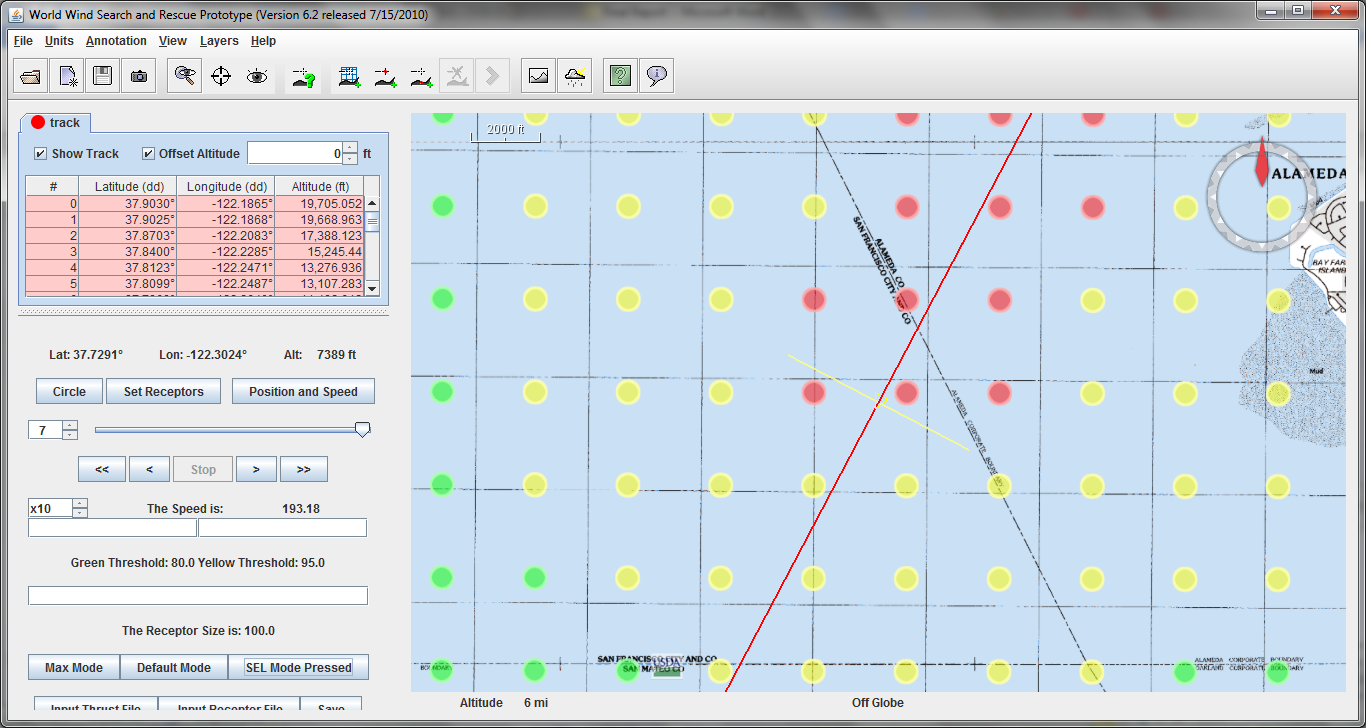




There are 3 modes in the ModifiedSAR that are meant to display different information. The default mode will display colors of receptors in terms of what they are at the time the airplane is at its current location along the trajectory. The Max mode will display the receptor’s maximum noise level when the airplane is at the current point along the track. The SEL mode will display the SEL number of each receptor when the plane is at its current location.

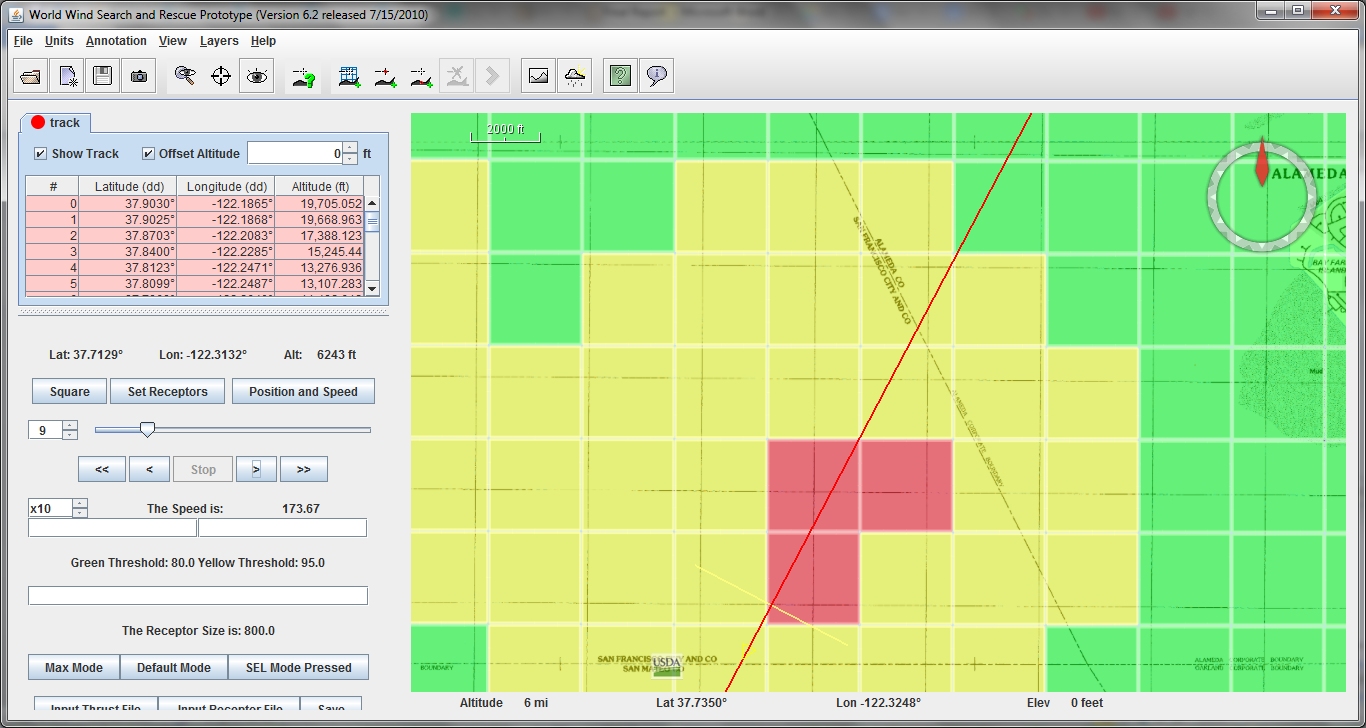




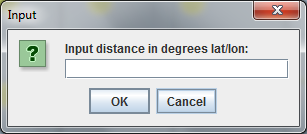


Directly above the labels for the Green Threshold and the Yellow Threshold, there are two text boxes. The Green Threshold allows the user to decide how many decibels are required below which the receptor is green. The Yellow Threshold allows the user to decide how many decibels are required, above which the receptor will turn red. Note that these text fields will only accept decimal numbers as input.

The shape of the receptors can be changed between circles and squares via the button left of the Set Receptors Button that is initially labeled “Circle” for the default shape. Also the size of the receptors has a default of radius 100m, but can be adjusted via the textbox directly above the label for the receptor size. Here we show a screenshot of a receptor grid that has been adjusted to side-length 800m squares.



The user is also allowed to create a receptor grid via the Set Receptors button. This button will prompt the user for the distance, which is actually in degrees latitude/longitude, but it is recommended that a distance of 0.1 be used. Then the user is prompted for the latitude/longitude coordinates of the center of the receptor grid. Finally the user is prompted for how many receptors will go along the length and then the width of the receptor grid.



Program Structure

The ModifiedSAR program starts with the SAR application, but there were changes in 2 main classes. The AnalysisPanel and the TrackViewPanel. The AnalysisPanel controls the run of the program, whereas the TrackViewPanel controls the buttons and input to the program. The buttons and user inputs were all added to the TrackViewPanel, and then were read in in the AnalysisPanel so that the animation could be updated as desired. The updateView method in the AnalysisPanel is what performs all the animation, so at the end of the updateView method, the methods to draw the receptors were added in. What happens in the updateView method is that it will redraw the entire frame that the user sees, and it is called many times in order to give the animation effect. Hence the code to draw the receptors was added into the updateView method, and the receptors are redrawn every time the frame gets redrawn, giving the impression that the receptors are changing color as the plane from the SAR application moves along its own trajectory.

In the TrackViewPanel, a lot of buttons were added in to take in user inputs. These user inputs were then taken from the TrackViewPanel and given to the AnalysisPanel, hence getting receptor information to the AnalysisPanel. The AnalysisPanel had methods and fields written to respond to these different user inputs, and then it would adjust the animation accordingly.

Also, the Receptor class was created as an addition to the original SAR application in order to store information that each receptor had to keep. The receptor class stores the information about the individual receptor that the program can reference when drawing the receptors. Information such as the location and color of the receptor, the current decibel level, the maximum decibel level, and the SEL were all values stored in the Receptor class. These values were updated in the AnalysisPanel as the animation progressed, but were then stored in the Receptors. That way the colors could all be read and updated in the AnalysisPanel.

Conclusion and Future Work

This is the extent of the development over the past 10 weeks of the noise model in NASA WorldWind. While many important features were implemented, there were also many features that could not be implemented in such a short time span, but that would still be very useful. The noise model could still be made more accurate, as we used a simplified algorithm accounting for only speed, thrust, and distance, but there are other factors that could affect the noise. Another important feature that was unimplemented in this model was the animation of multiple aircraft. As it is now, the model can only animate one aircraft at a time, but in the short time allowed for this model, we were unable to perform the animation for numerous aircraft. For the most part, we did not have other factors that we wanted to add to this model, but what was left unimplemented is expected to be very complicated if implemented.