Linköping University | IDA
Bachelor thesis | Innovative Programming
Spring term 2017 | LIU-IDA/LITH-EX-G--17/010--SE

Visualizing atmospheric data on a mobile platform

Maximilian Bragazzi Ihrén
Henrik Ingbrant Björs

Tutor, Rita Kovordanyi
Examiner, Peter Dalenius

LINKÖPINGS UNIVERSITET

Linköping University | IDA
Bachelor thesis | Innovative Programming
Spring term 2017 | LIU-IDA/LITH-EX-G--17/010--SE

Visualizing atmospheric data on a mobile platform

Maximilian Bragazzi Ihrén
Linköping University
Linköping
maxbr431@student.liu.se

Henrik Ingbrant Björs
Linköping University
Linköping
henbj316@student.liu.se

ABSTRACT
Weather data is important for almost everyone today. The daily weather report, home thermometers, and a lot of other things affect our every day life. In order to develop better and more efficient equipment, tools and algorithms, the people working with this data need to be able to access it in an easily accessible and easy to read format. In this research, methods of visualizing data on mobile platforms are evaluated based on what researchers in the field wants, since their respective fields might want to use very specific visualizations. The implementability of these visualizations are then evaluated, based on the implementations made throughout this paper. The results show that the researchers know what they want, and that what they want is implementable on mobile platforms given some limitations caused by performance.

INTRODUCTION
In this work, we study what workers who use weather data in their professions or hobbies need in terms of data visualization.

Today there are a multitude of professions and hobbies which require knowledge of the local weather. These can be people who work in the military, scientists such as meteorologists and geoscientists, but can also be people who fly with hot air balloons or gliders. Some of the relevant information that can be collected from the atmosphere is the temperature, pressure, wind direction and speed, all at different levels of height relative to sea level.

There exists many tools to measure these different values, such as weather balloons, flying drones, thermometers, and satellites. At Sparv Embedded [1], they develop such devices for measuring the previously mentioned data, and more [2]. They also develop systems to display this data in a multitude of ways to the end user (who at least a few of can count themselves in the previously mentioned crowd). The currently supported platforms are Android devices, but this will be expanded in the future.

The application is capable of doing a lot of different things already, such as connecting to a receiver (which in turn connects to the measuring device), receive data, extrapolate relevant data from the data and show it to the end user through varying graphs and plots.

Since the end users of the company's products are so varied, different methods of presenting the data is a huge advantage. For example, meteorologists might want to use certain graphs to read data that is useful to them, but this data would not at all be useful to, say, a military employee. Even if two groups want the same data, it can still be presented in such ways that is preferable for one group over the other.

The problem at this present moment is simply that the application does not support all different data presentations that the company and the end users wants. Currently the application only supports line and scatter plots (see figure 1) which is limited in expressibility when considering all possible end users.

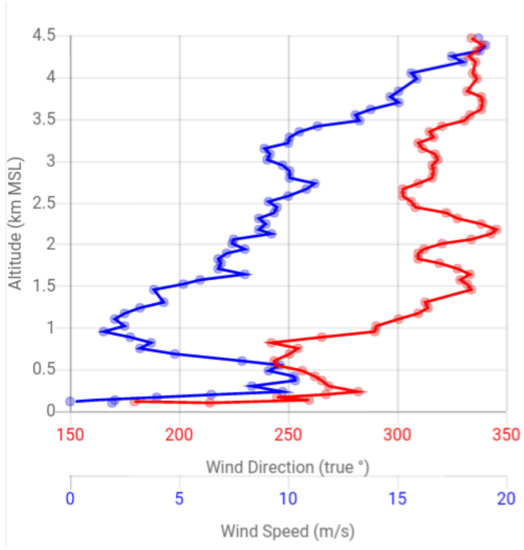


Figure 1. A line plot showing wind direction and wind speed given height.

Purpose
The goal of our work is to explore different ways to visualize the data provided to the application through the measuring devices, and to find which presentations are more preferable than others, for which people in which professions, and in what circumstances.

Research questions
Our first research question is:

- *Which forms of data visualization are the most appropriate for different end users? Do they themselves know what they want?*

When we know this, the following question is:

- *Can these be implemented within the limitations of a mobile device in an easy to use format?*

Limitations

We will limit our end users to the ones that are relevant to Sparv Embedded (military and meteorologists, amongst others), and since we are working on a time limit, we will implement as many visualizations as we have time to produce.

As for the limitations of a mobile device, the main concern that will be focused on is screen size, and how user-friendly the visualizations are (see Theory for specifics).

BACKGROUND

Sparv Embedded is a company that develops a certain type of weather balloon system for measuring weather data. In this system they currently include a program which can represent the data the sonde (the actual machine that measures the data) produces, both in real time and from recorded files.

However, since the current program is written for computers only, they are developing an app that can be used in a more handy fashion, since it can be run on a hand-held device instead of a laptop. When measuring data people often find themselves in hard-to-reach areas, in which case it's very important to have lightweight and easy to use equipment.

The app is developed using the multiplatform framework Cordova, which uses HTML, CSS and JavaScript in order to render the app on multiple different platforms (the target platforms being Android, iOS and browsers).

The app, in its current stage, can only display line and scatter plots over height, which is not enough for what the company's end users want. This is what this project aims to fix. Currently, the graphs are also rendered using a graph plugin, which we are supposed to replace with our own implementation due to licensing costs associated with the currently used library.

Theory

We will begin this chapter by describing the topic of weather data a bit more. We will go through what data that can be collected from the atmosphere, and to whom this can be of interest. Then we will bring up visualizations, and how they can serve to display weather statistics in varying ways, and then sum it up by bringing up some common forms of visualizations. After that, some topics about development for a mobile platform will be brought up, like GUI design and some of the requirements for our visualizations to become a reality.

Atmospheric data

The atmosphere contains many different measurable parameters. These can range from temperature, air pressure, humidity [3], and other data like wind direction and wind speed.

Based on the current weather state, one can give predictions of future weather through several models. For example, the persistence method predicts that tomorrows weather will likely be similar to today's weather [3,4]. One can also predict weather based on trends in data, such as cloud movements, cold fronts and high and low pressure centers [5]. Other, seemingly lesser used models include predicting weather based on long running trends in data (say, for a given date), and running weather simulations in supercomputers [6], which can be done in a multitude of ways [8,9]. There are constantly new models of weather forecasting being developed [23,24], the ones mentioned are just some basic examples.

Predicting weather is mostly the business done by meteorologists. Other professions or hobbies might be interested in such data, but not in the same depth. One such example could be hot air balloon pilots who want to predict what altitude they should rise to in order to go a certain direction (using wind direction) [7]. Gliding works in a similar way. Geo-scientists may also use weather data to research how Earth functions [10].

Data visualization

There is clearly a need for weather data by many target audiences, all of which have different demands and motivations, in terms of what info needs to be highlighted. Thus visualizations of data may look vastly different depending on the end user. Some end users have well known needs.

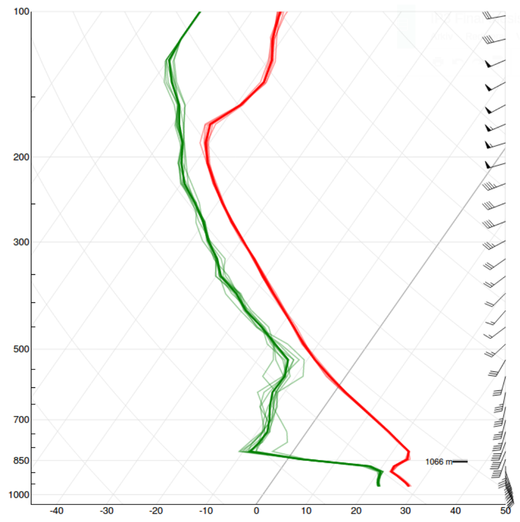


Figure 2. A skew-t graph.

Meteorologists, for example, popularly use skew-t graphs in their field of research [14] (see figure 2). Skew-t graphs are rather complex graphs with a lot of different things being readable, such as atmospheric stability. The leftmost axis represents height using pressure, the bottom represents temperature, and the flags on the right represent wind speed and direction. For more information, see [13,14,25].

Other end users might not have as clear demands. They might need different types of graphs for different purposes. Military personnel might, while trying to launch a plane, want only the latest data in the clearest format, since that is all that matters right before the flight. On the other hand, the military personnel might need to know the full measurement of air pressure over time, so that they can predict turbulence for their planes trajectory.

We can see from these examples that different visualizations serve to emphasize some parameters above others. There exists a plethora of different data visualizations and graphs, shaped in ways most suitable for different needs [11].

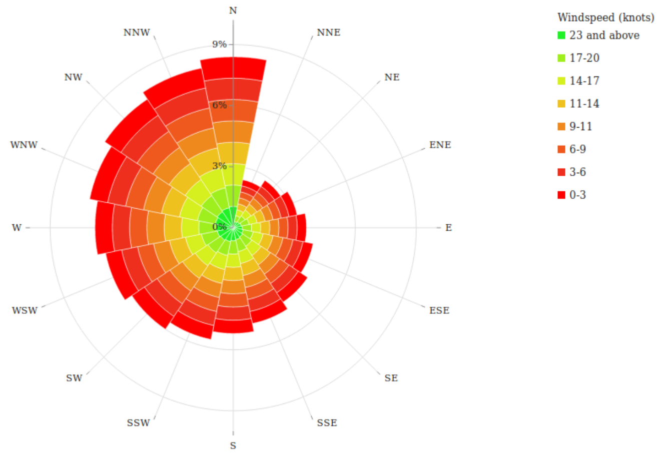


Figure 3. A windrose showing how wind direction and wind speed are distributed given a certain location and time frame (image is not representative of any location or time).

Polar plots (see figure 3) are circular graphs which emphasize directional values, which can be useful for displaying wind directions and speed. There is a special type of polar plot for displaying just wind direction and speed, known as a “windrose”.

Line plots (see figure 1) is the most general form of two-dimensional graphs for displaying a relation of some sorts.

Heat-maps (see figure 4) is a form of three-dimensional graph for visualizing fluctuating data over areas.

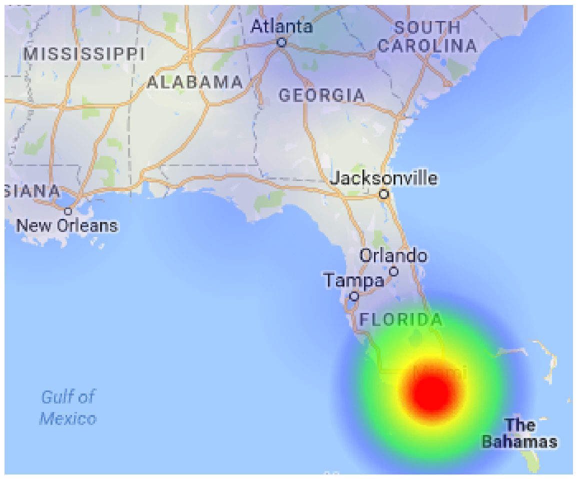


Figure 4. A heat-map showing example data

Another type of circular graph is a radar-like chart displaying directional values, usually over time or height (see figure 5).

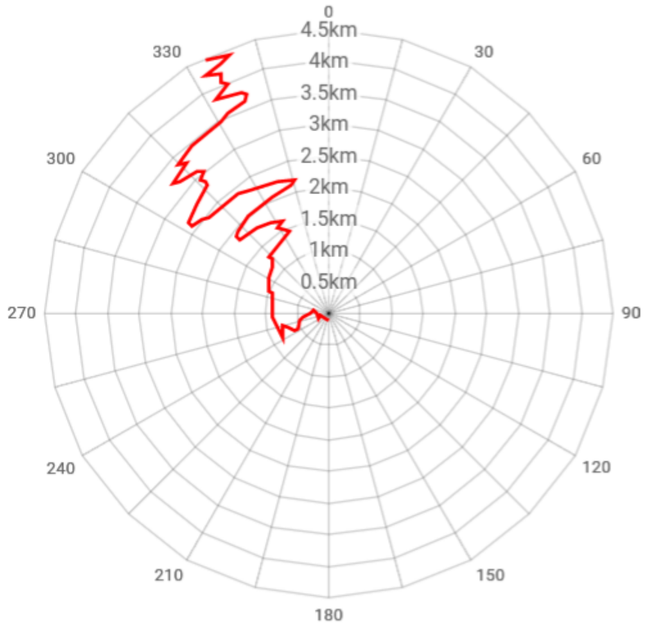


Figure 5. A circular chart showing wind direction over height.

Development for mobile platforms

We will only be developing a small part of the application, the graphs. Had we been implementing graphs for a traditional desktop application, the only obvious challenges would have been the implementation details, as in how to code the graph, what graphics framework (or library) to use, etc. One can safely assume that the graph would fit comfortably inside the entire screen.

Now that we are developing on mobile platforms, things get more complicated, and mainly because of the screen size. Is it really feasible, for example, to take an entire skew-t graph and paste it onto a mobile screen? Will it fit, or would one have to scroll and zoom around to find relevant information, and would that be comfortable? The limited screen size also brings forward the problem of navigation. How can one make switching between different visualizations not feel cumbersome? Will there be any major difference if one decides to use the application in a tablet instead of mobile phone?

There exists some guidelines when it comes to developing graphical user interfaces for mobile platforms [15,16]:

- Performing the main functionality of the app should preferably take as few steps as possible. For us this could mean making transitioning between different graphs as simple as possible, or make graph navigation easy.
- The look and feel of the app should feel consistent, both within the app itself and across multiple platforms. For us this could mean, for example, that navigation should work the same across all types of graphs, and that how one navigates should not be to different across platforms. It could also mean that the colors used in the different graphs should remain similar.
- It should be designed to fit multiple situations. This means that the app should be simple to use regardless of external factors such as brightness, weather, etc.

Though not a unique problem for mobile platforms, performance and responsiveness is also a factor to take into account. It would naturally be undesirable if, for example, pressing some button would present visible results several seconds after having pressed it. It is most desirable if the elements composing the user interface would mimic their physical real life counterparts, in terms of behavior and response.

Survey methodology

One method to find out what the end users prefer using in their respective fields is to conduct surveys, which is fitting if the survey takers are spread out globally, which is the case in this paper. There exists a multitude of ways to conduct surveys [17]. For example, one can have open-ended questions, which are formed so that answers are not black and white. This allows for a varying range of possible answers, which can be great for recording detailed experiences.

One can also asks closed-ended questions, where answers instead are rather black and white, the opposite of an open-ended question. Closed-ended questions are more suitable in a quantitative study, where you just want hard facts (“Did you find X?”, “Did you enjoy Y?”), as opposed to in a qualitative study, where you'd want more in depth answers (“How did you find X?” Did you enjoy Y? If not, why?”).

There is also the option to use standardized surveys. There exists a variety of such [21], for example SUS surveys [18], which are used to gain information about the general usability of a service in an easy to read score-based format. QUIS [20] is another example of a standardized survey, which is a questionnaire regarding system usability along six scales. However, as opposed to SUS, QUIS requires a license (and therefore a license fee). There is also SUMI [22], which contains 50 so called attitude statements, which the user can respond to with either “agree”, “don't know” or “disagree”. An example question is “The software responds too slowly to inputs”. Similar to QUIS, using SUMI also comes with a license fee.

Other than the general type of questions, one also has to think of a multitude of other factors while designing a survey. This can be the ordering of questions, the specific wording of a question, which answer format fits the question best, etc.

The ordering is important since you want to lead the users into the survey to make sure they don't get bored. Therefore, you should not open with the most specific questions, since the survey taker most likely won't have the needed data in mind. That is why you should ease them in with the more simple and open-ended questions first [17].

The wording is important since one needs to make oneself clearly understood. You do not want survey takers to misunderstand the questions, and therefore end up with faulty or useless data. For example, the question “Are you concerned about environmental degradation in your neighborhood?” raise a lot of questions [19]; What does it mean to be concerned more specifically? What is environmental degradation? How does the survey taker define their neighborhood (the street, the block, etc)?

It is therefore important to keep your questions simple, short and clear, and try to avoid to ask more than one question per question [19].

Since we for a final survey need to know if the customers got what they wanted, we need an efficient way to find out both how useful the system is in general and how useful our implementation is, in regard to their use in their field.

There exists tools to measure how much certain parts of an app are being used, like Firebase. We could use this to measure whether or not the customers actually use the features they requested. However due to the limited customer base and time frame, we will not be able to collect enough data this way to draw proper conclusions.

METHOD

This section is split into multiple parts, the first details the original survey for end user visualization preferences, the second part details the implementation, and the third part details the second survey in which we got feedback on our implementation.

In order to conduct the surveys, Google Forms were chosen to be used as the medium, since it is a well used and easy to distribute platform. Since it has been around for a long time, and due to its frequent usage, it has developed an easy to use interface, both for the creator and user.

First Survey

The first step is to figure out what visualizations the end users desire out of the application. The group of end users that were available are spread out across the world, so the means of contact with them is limited to internet services such as Email, Skype, etc.

The eighteen end users that received the survey were part of many different groups: hot air balloon pilots and related enthusiasts, university professors and other researchers, and also ammunitions experts.

The questionnaire had a total of 12 questions, with 10 of them being relevant for this research, 4 open-ended questions, and 6 close-ended questions. The 2 irrelevant questions were regarding ways to access the data in general, and were requested by Sparv Embedded, since the survey served as a good opportunity for them to talk to their customers.

The close-ended questions followed a format of first presenting a form of data visualization, and then asking whether it is useful to the person answering or not (see appendix 1). The answers range from one to five, where one is “Not useful at all”, and five is “Very useful”.

One question, for example, was phrased “How useful would the following form of graph be to you? This can represent different measurements at different altitudes”, with an image representing a line graph.

The open-ended questions allowed the survey takers to express their thought on possible changes to the presented visualizations.

Alongside that, some questions were regarding where the weather data should be sent, if not the main app, but those were more relevant for Sparv Embedded and not the research presented in this paper.

Implementation

In the application there already exists some simple graphs, which needed to be re-implement as part of the job.

Since the already existing application is built using the multiplatform framework Cordova, which uses HTML5, CSS3 and JavaScript, HTML5 canvas will be used for drawing graphs. The library Chart.js [12] uses HTML5 canvases for graph rendering, including those that will possibly need to be implemented, and also the possibility to implement graphs unique to this project if needed.

The mobile device used for testing was a Samsung Galaxy Tab Active 4G. It had 1.5 GB RAM, a screen size of 8” with a resolution of 1280x800. The processor was a 1.2GHz Qualcomm Snapdragon 400 with 4 cores.

The metrics that was used for determining whether the visualizations are implementable and easily useable were the following:

1. If the visualization is capable of being interacted with, it should show instant response towards user actions. For example, if a graph is clickable, a clear indication that the click has been made should show up immediately.
2. The visualization should fit within the screen space limitations, and is capable of displaying all possible value ranges one might want to observe.
3. The visualization should not take an excessively long time to render.

Additionally, the amount of interactions required to complete a task was also measured. In this case, the only interaction the app provided was clicking, so this metric can be rephrased as the amount of clicks required to complete a task.

The discussion section will then bring up the feasibility of adding additional functionality to the visualizations, such as zooming, customization, etc, in relation to the constraints of any given platform.

Second survey

After the implementation was completed, the next step was to design the second survey. Like the first one, Google Forms was used. It was sent to the same users as the first survey.

This questionnaire contained questions regarding the implementations of the graphs from the first survey, asking if people found them clear and easy to use. This time, there were 9 closed-ended questions, and 10 open-ended ones, for a total of 19.

The questions served to provide more detailed feedback about whether or not the graphs they asked for in the first survey were really things they'd find useful, or if they just said they wanted the graphs in the first survey because they were asked.

The questions followed this format:

- I found these graphs easy to use (accompanied by images showing one of the implementations).

- Is the contents of the above graphs clear? Is it easy to understand?
- Would you like something changed in the above implementation?

There were also some general questions not directly related to the graphs themselves, but might influence the users usage of the app:

- Overall, I found the user interface easy to use.
- Having two graphs displayable at the same time is useful to me.
- What do you think could be improved about the app?
- What was your overall impression using the app?
- Would you use this app again?

All of the questions were answerable by either a scale ranging from one to five or ten, the higher the number, the more positive of a response, or just free text. Most free text questions were also optional.

RESULTS

The results will be split into the same parts as the method section was split up into. First, the results of the first survey, then some results on how the implementation went, and lastly the results of the second and final survey.

First survey

The visualizations that were presented, and the answer distribution were the following, where one is "Not useful at all" and five is "Very useful":

Type of graph	1	2	3	4	5	Total
Line graph (at height)	0	1	0	2	7	45
Circle graph	0	1	1	5	3	40
Comma-separated values	0	0	0	0	10	50
Line graph (over time)	0	2	1	1	6	41
Current position in text	0	1	1	1	7	44
Skew-t	0	0	1	4	5	44

Table 1. Answer distribution for survey #1.

The answers to the first survey were received in less then a week after it was sent out. Ten people out of eighteen answered the survey. To summarize the answers, people were positive about most visualizations, with some being more preferred than others.

Alongside these answers, people also gave some opinions on how the visualizations could be improved (see appendix 1).

After re-implementation of the already existing line graph was completed, the project moved to the next step: implementing the new visualizations. The work was structured based on these answers, and the order of implementation was decided by implementing the most desired visualizations first, starting with comma-separated values, then the current position in text, circle graph, line graph (over time), and finally skew-t.

Some additional answers arrived a few days after the work was structured, but the new set of answers did not deviate enough from the ones already considered to warrant a restructuring. So the order of graphs to implement did not change.

The choice was made to implement skew-t last, even if the score would suggest otherwise. This is because since it is the most complicated graph to understand and implement out of the ones in the survey, and given the time constraints on the project, there was no guarantee that it would be able to be finished it in time. Instead the graphs that were certain to be implemented were prioritized.

Alongside each visualization there was also a question of if they would like to improve something about the visualization in question, and what that change would be in that case. About half were happy with what was showed them and wanted nothing changed, but there were also smaller requests for implementation changes. Some examples are: a request for a zoom feature, different units of measurement (degrees or military mils for wind direction, amongst others), and a few requests for graphs we hadn't asked about (hodographs, tephigram, etc).

Implementation

The app, as it was presented when work begun, already had support for displaying two line graphs next to each other (see appendix 1, first two figures). They were capable of displaying the parameters the wind measurement devices could capture, such as wind direction and speed, humidity, etc. There was also support for toggling which parameters to show, so that one could customize what to display on the graph, and to disable one of the graphs entirely, to make the other one wider.

The line graph re-implementation (appendix 1, figure 3) managed to behave almost identically to the initial line graphs, with the only difference being having moved the measurement toggle buttons to the main display. This removed the need of first going having to go to a side menu, thus reducing the amount of interactions required for toggling measurements.

The two other line graph variants that was implemented was the graph displaying values over the course of time instead of height (appendix 1, figure 8), and the skew-t graph (appendix 1, figure 9).

Since the skew-t graph is far to complex to implement given the time frame, a choice was made to use an already existing solution for drawing skew-t graphs. A solution using Chart.js was not found, however a multitude of ones implemented in D3.js were found. Since the space overhead of adding an additional library to the code base was insignificant enough (about 150kB) to warrant the usage of one of the previously mentioned implementations.

Given the data sets which the app works with, and the metrics presented in the method section, these three line graph implementations showed the following properties (each number represents a metric presented in the method section).

1. The only interactions they supported was clicking for toggling different measurements, and clicking on the graph to get a detailed view of a measured values at a specific point. One gets response immediately from the system that one indeed have clicked somewhere.
2. As the figures show, the graphs did fit within the screen size. One problem was that there was no way to manually select which portion of a graph to show. An attempt was made to allow zooming in on different sections of the graph, but this proved to be unresponsive, and in general broken to the point that it was nearly unusable.
3. Rendering the graphs took on average slightly less than a second to complete, which was good enough considering the data sets the app works with.

The circle graph (appendix 1, figure 4) ended up showing the exact same properties as the three line graphs described above.

The two text based visualizations (appendix 1, figure 5) was implemented as one singular entity, since all that separates them is some configuration options regarding what measurements to show, and in what fashion. This visualization ended up exhibiting the following properties:

1. This visualization was not capable of being interacted with, other than the different ways to configure it, and doing so proved to be responsive.
2. As figure 5 in appendix 1 shows, the naive implementation where one simply prints every measurement at each time point did not fit within the given screen space, if one draws two visualizations next to each other. The two ways implemented to combat this was being able to zoom in on one specific visualization (appendix 1, figure 7) and being able to select which measurements to show (appendix 1, figure 6). A lot more configuration options is left to be desired, but given the time constraints this was not possible.
3. Rendering times was almost non-existent. This was expected, since any modern computer is usually able to draw lots of text very quick.

Second survey

The second survey showed that the implementations were well received. Most questions were on a scale of 1-5, with an average of 4 both individually and in total (except for comma-separated values, which had an individual average of approximately 5).

Type of graph	1	2	3	4	5	Total
Line graph (at height)	0	0	1	3	2	25
Circle graph	0	0	1	3	2	25
Comma-separated values, and current position in text	0	0	0	1	5	29
Line graph (over time)	0	0	1	3	2	25
Skew-t	0	0	1	2	3	26
Additional questions	**1**	**2**	**3**	**4**	**5**	**Total**
Overall, I found the user interface easy to use	0	0	1	3	2	25
Having two graphs displayable like above is useful (note the zoom feature to only display one graph at a time)	0	0	0	0	6	30
I found this method of choosing different readings to show to be usable	0	0	1	2	3	26

Table 2. Answer distribution for survey #2.

The answer rate was lower this time around, with only six people out of the same eighteen answering. Expectations were not to get a higher answer rate than survey #1, but the 40% relative drop was a bit alarming. However, the responses we got were enough to reach a conclusion.

There were also questions regarding whether the graph content was clear, and whether people thought something could be changed. The answers were varying, and showed a desire for a multitude of different features regarding readability, customizability, etc. People were overall positive about what they received. Details on every answer for the first[1] and the second[2] surveys can be seen online.

DISCUSSION

Results

The results were fairly expected. Seemingly, people were pleased with both the choice of visualizations to implement, and the implementations of said visualizations that were provided.

The only really negative parts are that the minor changes and improvements mentioned under First Survey in the Results section never got worked on. Further graphs to implement weren't possible due to time constraints, but we did try our hand at a zooming feature (amongst other things).

There exists a zooming plugin for Chart.js [26] which we tried to implement. However, given the limitations of the device and the amount of things already happening in the app, the application started to slow down fairly significantly, and the functionality was very hard to control with your fingers (which, on a tablet, is fairly important). The slowdown most likely is due to the fact that with each “zoom tick” (each time the plugin receives a zoom request, which can happen multiple times during a zooming gesture), the plugin tried to redraw the graph in the current zoom level from scratch. If a SVG-based rendering library had been chosen, instead of a HTML5 canvas-based one, this might have been optimizable a lot easier. Because of this and further time constraints, the zoom feature was scrapped.

The different unit measurements that were requested were already a planned feature for the app, so this was left out of our work in favor of this future implementation.

Method for surveys

As the results show for the first survey, the participants seemed to want every visualization presented to them more or less. This is very likely because of the way the questions was presented to them. Being able to convey exactly what one wants in text form is an entire challenge in itself.

The main goal of the first survey was to get insight into what forms of data visualizations the Sparv Embedded target users would want in their day to day usage of the app. Given the way the questions were presented, with already decided upon visualizations, the user taking the survey might have been left with limited options for answering. If one is presented with only a handful of visualizations, why not support as many of them as possible, if nothing says otherwise.

The answer quality could possibly be improved if there, for example, was a bigger emphasis on open-ended questions regarding what the survey participant would find most useful in the daily usage of the app. This could be accompanied with images of example graphs for inspiration purposes. Perhaps a question asking the user to describe what he would want a typical use case of the app to be like, so that one can create an interface adapted for the workflow of the survey participants.

Given the desired changes for the first survey, the resulting second survey would have to be redesigned completely, so that it would make sense given the results acquired from the first survey and the implementation.

Method for implementation

The metrics used for evaluating the implementations was chosen with the goal of having a set of general metrics across all possible different visualizations. This ended up limiting the possible set of metrics to those regarding device input and output, performance, and what value ranges the visualizations allow to display.

Even though these are perfectly valid metrics, one could possibly get more quality results if instead of using a general set of metrics, use separate ones specific for each visualization. This could give more quality results, and one could possibly then derive a general set of conclusions based on these new results, applicable in a broader scale of research.

The process of choosing charting libraries for this research was relatively short. It essentially consisted of searching around for different libraries, and picking the one supporting those graph predicted to be needed at the time. A better choice could have been one optimized for mobile platforms (if such exist). Such a library might have solved some of the problems that occurred, such as the performance hitches when attempting to zoom.

Future work

The future work available to this project can be divided into two; general future work and project-specific future *work*.

In the general case, there is always the option to expand the survey groups to a larger audience. 18 people is a fairly small group to draw any general conclusions from (albeit enough for this specific usage). If you expand to different user bases, you of course need to tailor the graphs asked about to the new set of groups. Once one has changed the survey in such a manner, it's possible one might want to experiment with the survey structure, to see if one can get a clearer response.

In the project-specific case, there's mostly work available in implementing the things the users asked for during survey

[1]https://github.com/B0H95/ExJobb/blob/master/survey1.pdf

[2]https://github.com/B0H95/ExJobb/blob/master/survey2.pdf

#1. The zoom feature, for example, could be implemented properly. One could either implement the skew-t graph in Chart.js, or implement the other graphs in the other added library, D3.js in order to avoid using two separate libraries.

CONCLUSIONS

A multitude of different user groups were asked to fill out the surveys, where they were asked to evaluate how useful a certain type of graph is to them. However, since most of the responses were very positive during the first survey, one can assume that either we asked about visualizations that were too general in usage, or if they just because they've been asked said yes, thinking they wouldn't interfere with their regular usage, but might be useful in very specific instances (a “why not” attitude). In order to draw any more general conclusions, the amount of survey takers would have to be increased dramatically.

For this specific instance, the answers received are enough to warrant a conclusion however. From the results gotten, one can conclude that the users both know what they want, and that the implementations were implemented well enough for their liking.

The implementations themselves are fairly responsive as well, given the limitations of the platform. However, as seen with the zoom feature, it is just barely within the limits, as no new major functionality might be difficult to implement well.

REFERENCES

1. "Sparv Embedded: Passion for Mobile Sensor Systems". [Online]. Available: http://sparvembedded.com/. [Accessed: 16-Feb-2017].

2. "Windsond Product Catalog". [Online]. Available: http://windsond.com/windsond_catalog_Dec2016.pdf. [Accessed: 16-Feb-2017].

3. L. Alter, "Meteorology", *Yale-New Haven Teachers Institute*, 1994. [Online]. Available: http://teachersinstitute.yale.edu/curriculum/units/1994/5/94.05.01.x.html. [Accessed: 28-Feb-2017].

4. "Persistence method: Today equals tomorrow", *University of Illinois*, 1997. [Online]. Available: http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/fcst/mth/prst.rxml. [Accessed: 28-Feb-2017].

5. "Trends method: Using mathematics", *University of Illinois*, 1997. [Online]. Available: http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/fcst/mth/trnd.rxml. [Accessed: 28-Feb-2017].

6. "Other forecasting methods: Climatology, analogue and numerical weather prediction", *University of Illinois*, 1997. [Online]. Available: http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/fcst/mth/oth.rxml. [Accessed: 28-Feb-2017].

7. S. Hasham, S. Majumder, S. Southern, A. Phipps and K. Judkins, "Hot-air ballooning injuries in the United Kingdom (January 1976–January 2004)", *Burns*, vol. 30, no. 8, pp. 856-860, 2004.

8. C. Voyant, M. Muselli, C. Paoli, and M.-L. Nivet, "Numerical weather prediction (NWP) and hybrid ARMA/ANN model to predict global radiation," *Energy*, vol. 39, no. 1, pp. 341–355, Mar. 2012.

9. A. C. Lorenc, "Analysis methods for numerical weather prediction," *Quarterly Journal of the Royal Meteorological Society*, vol. 112, no. 474, pp. 1177–1194, Oct. 1986.

10. "What is a Geoscientist?", *Bucknell University*, 2016. [Online]. Available: http://www.bucknell.edu/arts-and-sciences-college-of/academic-departments/geology-and-environmental-geosciences/what-is-a-geoscientist.html. [Accessed: 28-Feb-2017].

11. M. J. de Smith, "Statistical analysis handbook", 2015. [Online]. Available: http://www.statsref.com/HTML/index.html?graphics.html. [Accessed: 28-Feb-2017].

12. "Chart.js". [Online]. Available: http://www.chartjs.org/. [Accessed: 1-Mar-2017].

13. "Introduction to the SkewT diagram", *NASA*. [Online]. Available: https://airsnrt.jpl.nasa.gov/SkewT_info.html. [Accessed: 01-Mar-2017].

14. "The Skew-T Diagram", *The University of Arizona*, 2012. [Online]. Available: http://www.atmo.arizona.edu/students/courselinks/fall12/atmo336/lectures/sec1/skewt.html. [Accessed: 01-Mar-2017].

15. J. Gong and P. Tarasewich, "Guidelines for handheld mobile device interface design" *In Proceedings of the 2004 DSI Annual Meeting*, 2004.

16. B. Shneiderman, C. Plaisant, and M. Cohen, *Designing the user interface: Strategies for effective human-computer interaction*, 4th ed. Boston: Addison-Wesley Educational Publishers, 2009.

17. N. Mathers, N. Fox and A. Hunn, "Surveys and Questionnaires", *Yorkshire & the Humber: The NIHR RDS for the East Midlands*, 2009.

18. J. Brooke, "SUS-A quick and dirty usability scale", *Usability evaluation in industry*, vol. 189, no. 194, pp. 4-7, 1996.

19. S. McLafferty, "Conducting questionnaire surveys", *Key methods in geography*, pp. 87-100, 2003.

20. B. Harper and K. Norman, "Improving user satisfaction: the questionnaire for user interaction satisfaction version 5.5", *Proceedings of the 1st Annual Mid-Atlantic Human Factors Conference*, pp. 224-228, 1993.

21. J. Lewis and J. Sauro, *Excel and R Companion to "Quantifying the User Experience"*, 1st ed. Denver, Col.: Create Space Publ., 2012.

22. J. Kirakowski, "The software usability measurement inventory: background and usage.", *Usability evaluation in industry*, pp. 169-178, 1996.

23. Michalakes, S. Chen, J. Dudhia, L. Hart, J. Klemp, J. Middlecoff and W. Skamarock, "Development of a next generation regional weather research and forecast model", *Developments in Teracomputing: Proceedings of the Ninth ECMWF Workshop on the use of high performance computing in meteorology*, vol. 1, pp. 269-276, 2001.

24. M. Jacobson, "GATOR-GCMM: A global- through urban-scale air pollution and weather forecast model: 1. Model design and treatment of subgrid soil, vegetation, roads, rooftops, water, sea ice, and snow", *Journal of Geophysical Research: Atmospheres*, vol. 106, no. 6, pp. 5385-5401, 2001.

25. G. Stipanuk, "Algorithms for Generating a SKEW-T, log p Diagram and Computing Selected Meteorological Quantities", *Army Electronics Command Fort Monmouth NJ*, No. ECOM-5515, 1973.

Appendix 1

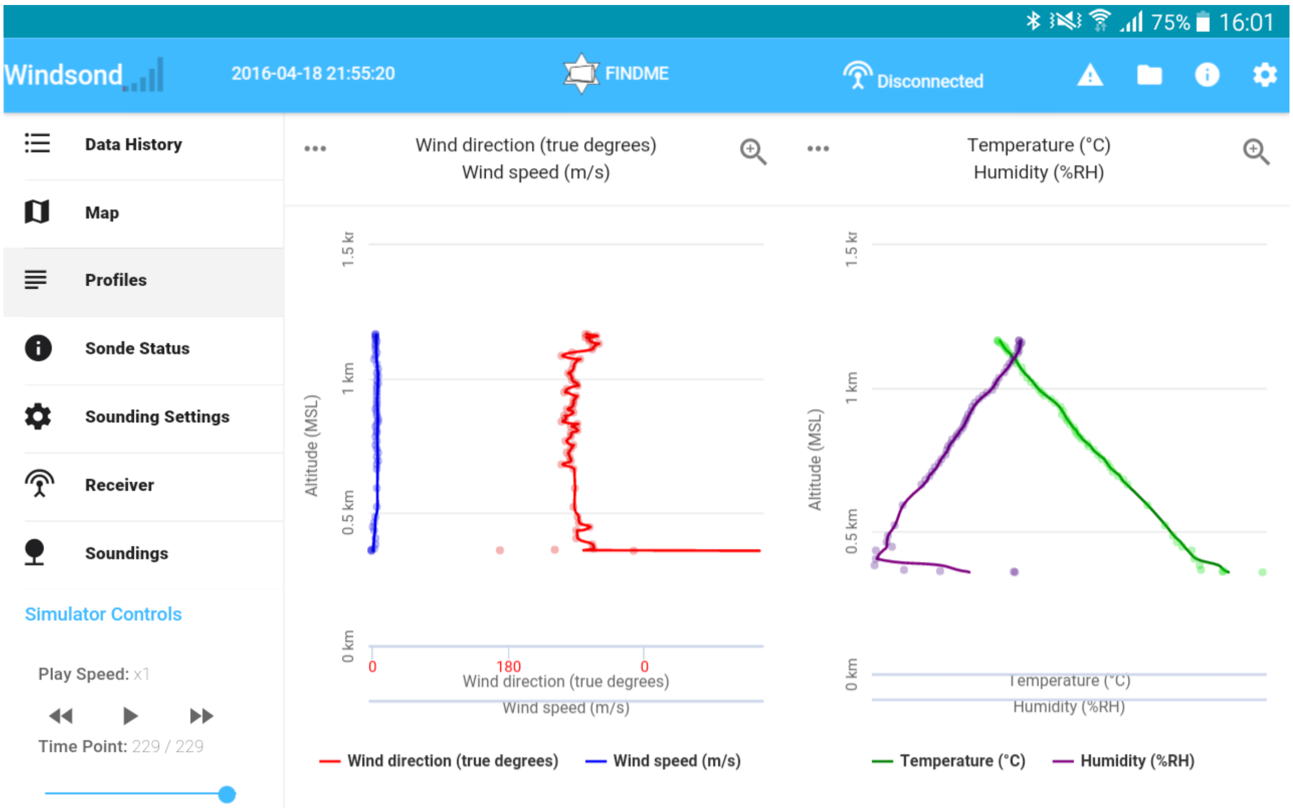


Figure 1. The line graphs already present in the app when work started. Note the zoom button to the top right of each graph, which allows you to let that one graph cover the entire screen.

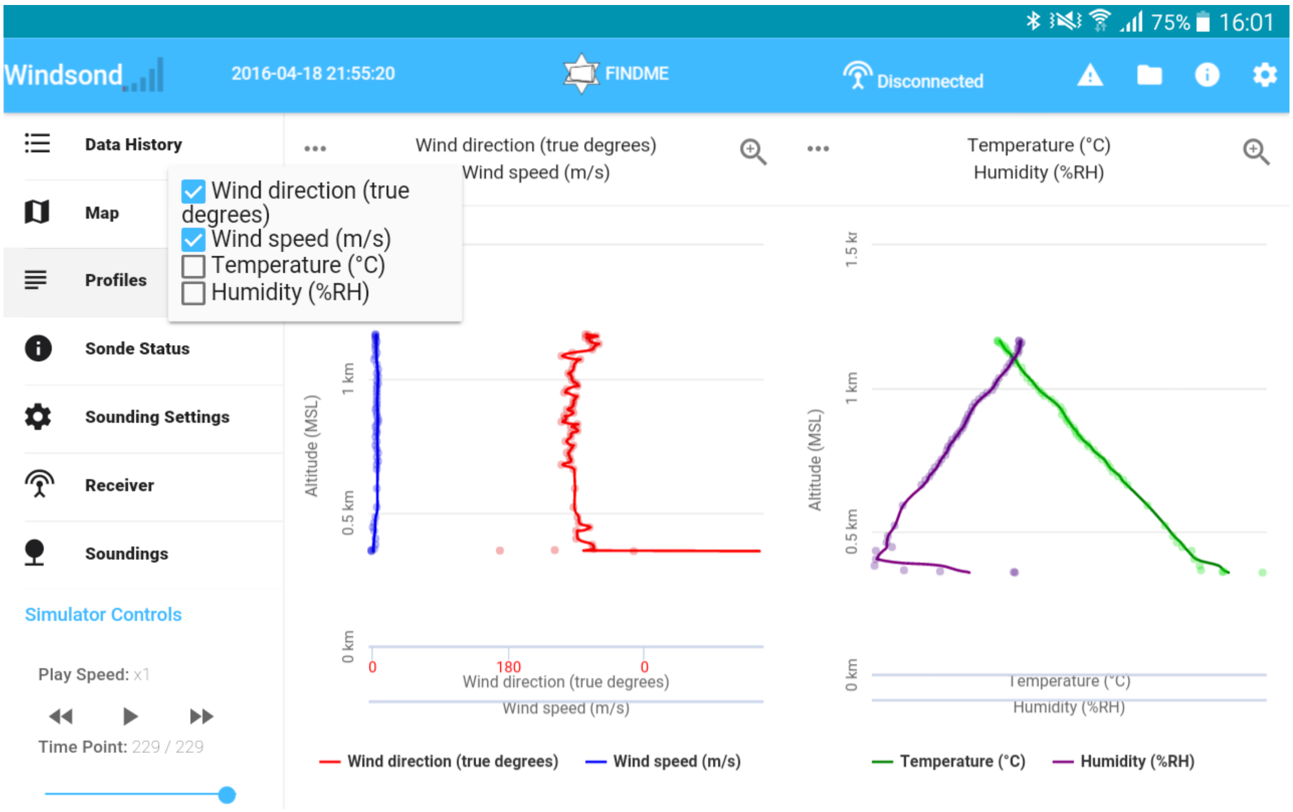


Figure 2. The line graphs already present in the app when work started, including the menu in which you can select which measurements to show.

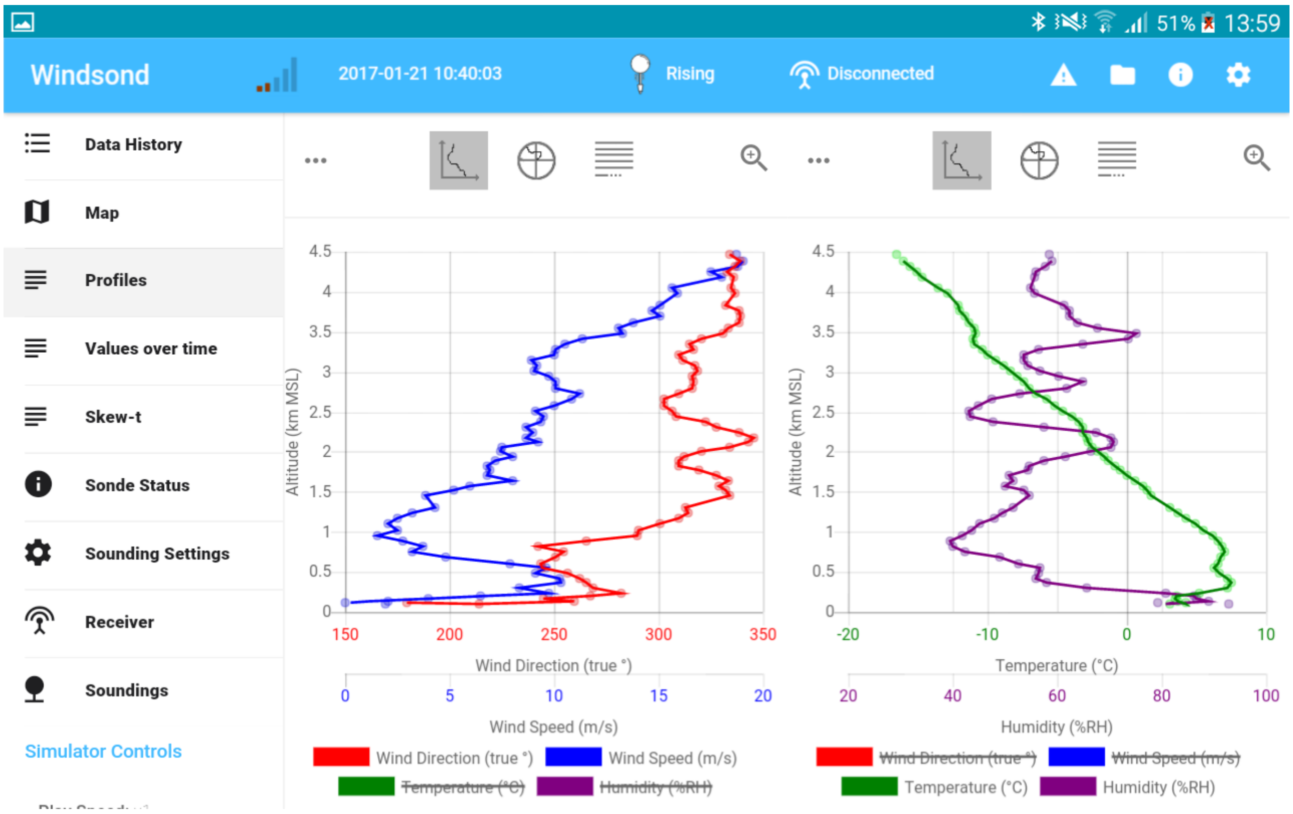


Figure 3. Our re-implementation of the line graphs. Note the three buttons above each graph which allows you to switch between visualizations. Also note the additional menu items to the right.

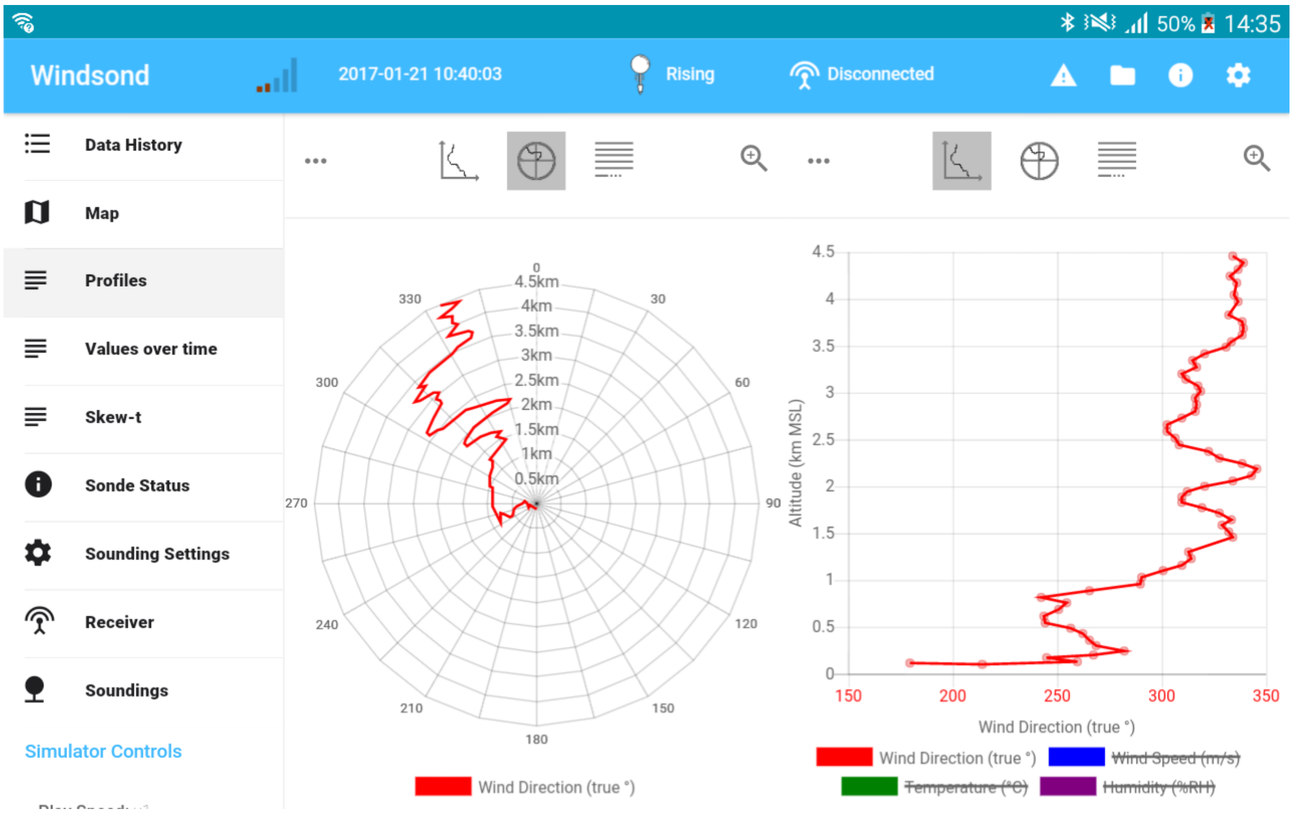
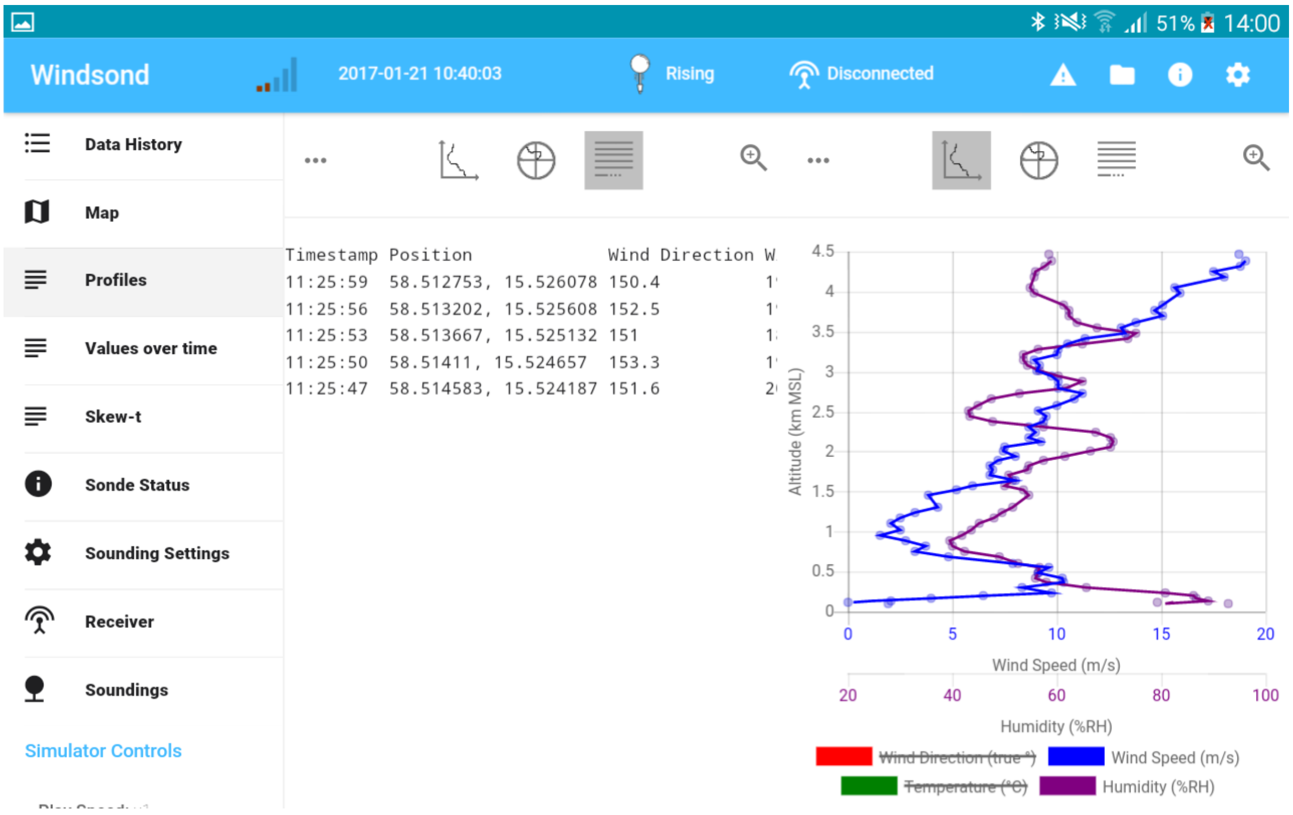
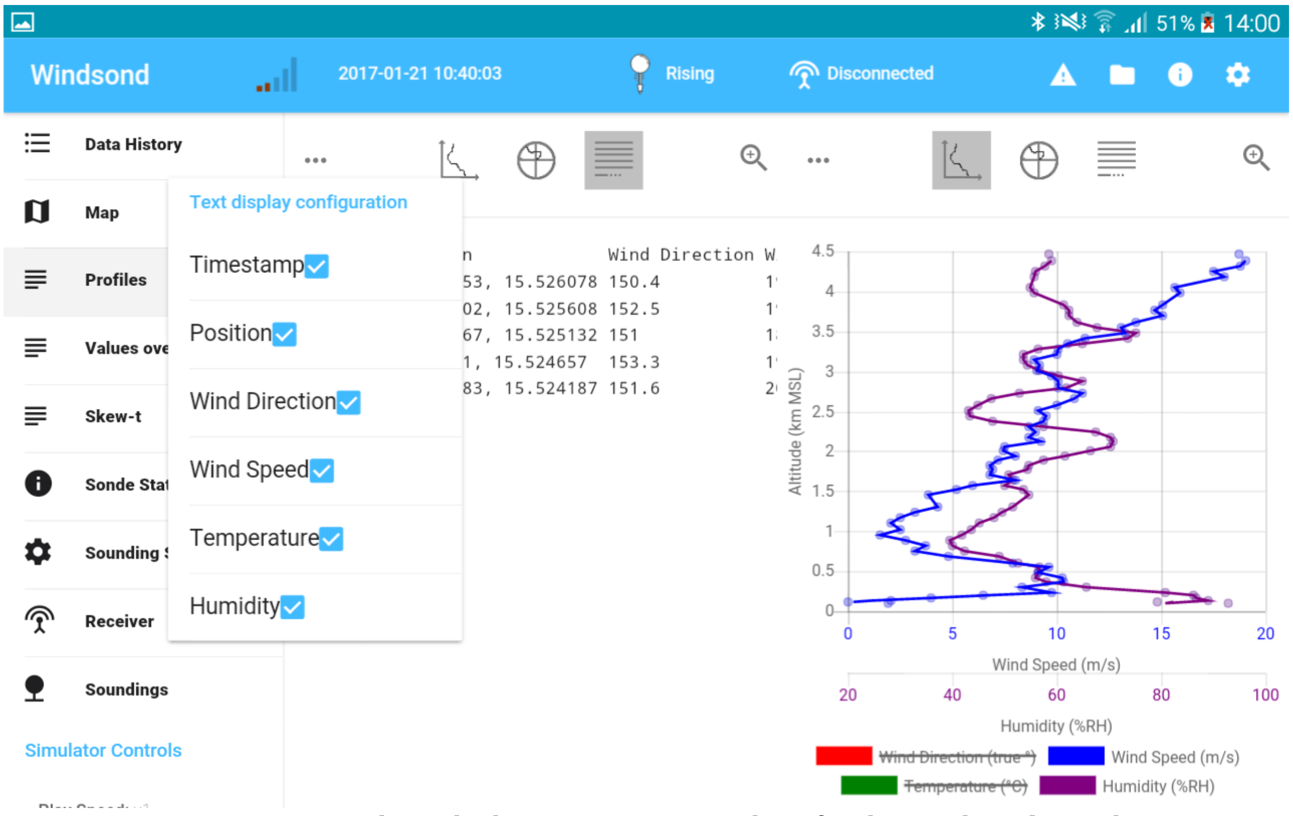


Figure 4. The circle graph shown to the left. The right shows a line graph with some measurements disabled.



```
Timestamp Position             Wind Direction W
11:25:59  58.512753, 15.526078 150.4          1
11:25:56  58.513202, 15.525608 152.5          1
11:25:53  58.513667, 15.525132 151            1
11:25:50  58.51411, 15.524657  153.3          1
11:25:47  58.514583, 15.524187 151.6          2
```

Figure 5. The text-based visualization.



Text display configuration

Timestamp ✓

Position ✓

Wind Direction ✓

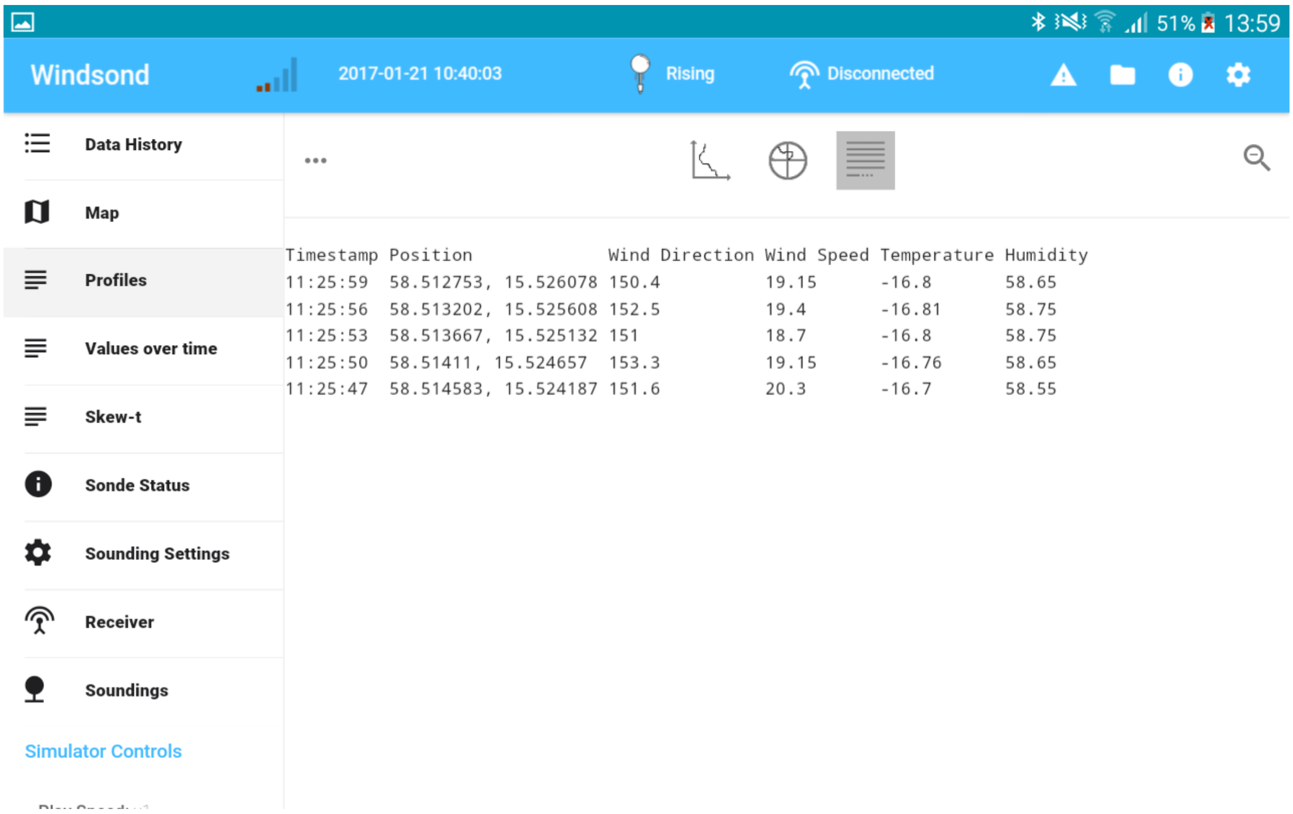
Wind Speed ✓

Temperature ✓

Humidity ✓

```
n                    Wind Direction W
53, 15.526078 150.4          1
02, 15.525608 152.5          1
67, 15.525132 151            1
1, 15.524657  153.3          1
83, 15.524187 151.6          2
```

Figure 6. Options menu to select which measurements to show for the text-based visualization.



```
Timestamp Position             Wind Direction Wind Speed Temperature Humidity
11:25:59  58.512753, 15.526078 150.4          19.15      -16.8       58.65
11:25:56  58.513202, 15.525608 152.5          19.4       -16.81      58.75
11:25:53  58.513667, 15.525132 151            18.7       -16.8       58.75
11:25:50  58.51411, 15.524657  153.3          19.15      -16.76      58.65
11:25:47  58.514583, 15.524187 151.6          20.3       -16.7       58.55
```

Figure 7. The text-based visualization, zoomed in.

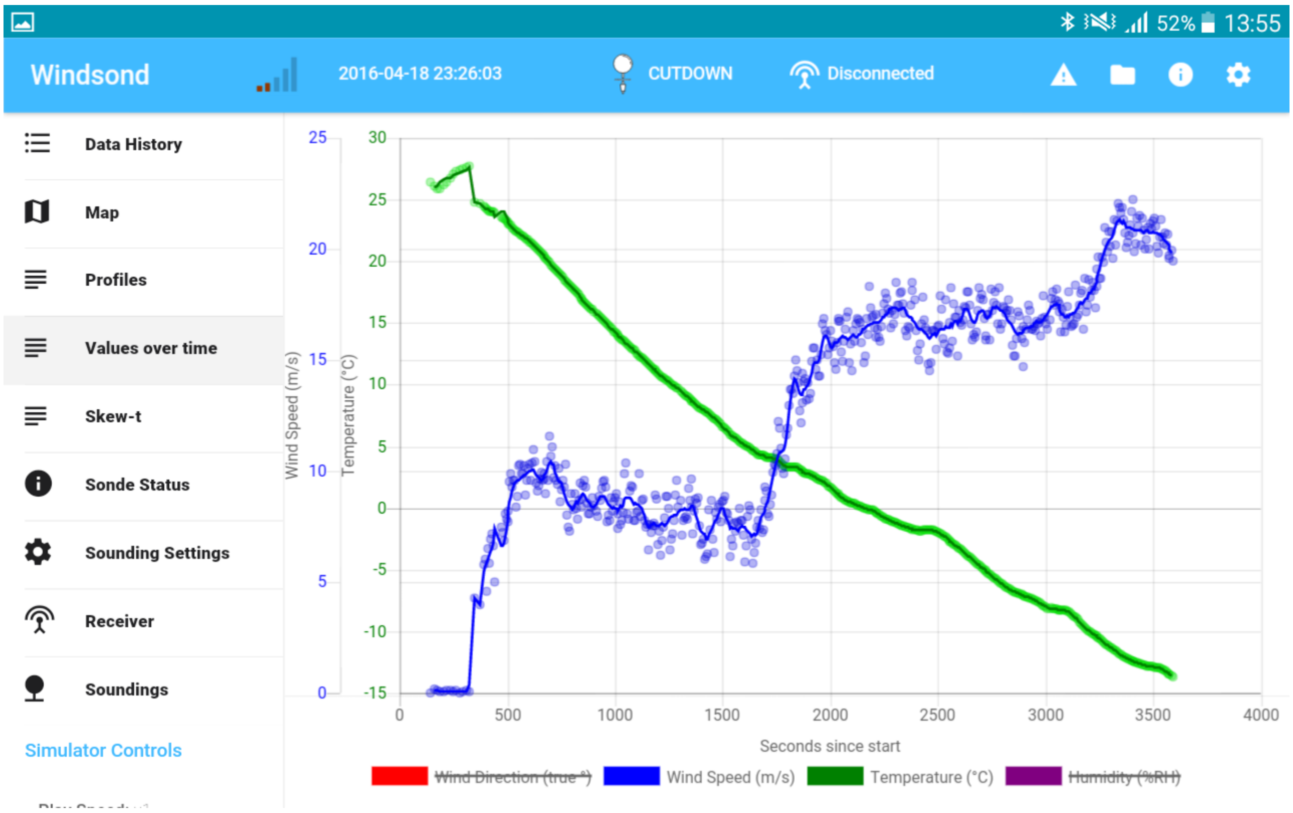


Figure 8. The line graph displaying measurements over time.

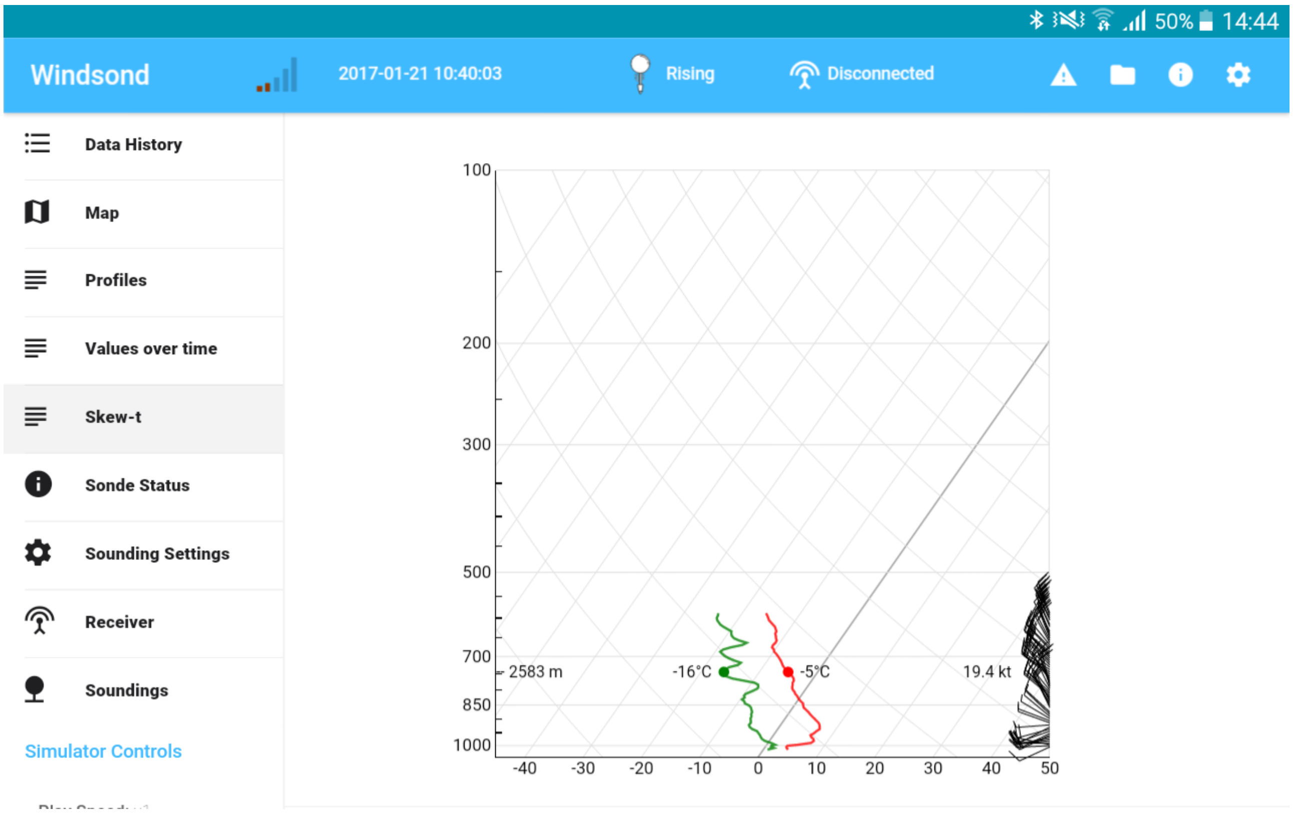


Figure 9. The skew-t graph.