Intuitive Display of Complex Data on iOS

Aaron Sher, Principal Software Engineer

Vanteon Corporation

July 2013

Abstract: This is a case study of the design process for an iOS application. The software in question accepts a high-bandwidth, real-time stream of multidimensional data from a hardware accessory, and must present this data to the user in a form that can be easily understood and manipulated. This paper will examine the decisions made during the design process and the tradeoffs considered, and extract some lessons about how to solve this type of problem within the constraints of a mobile device.

The Problem with Mobile Devices

Ever since the iPhone began allowing third-party applications, it has become almost a requirement for every service, device, and instrument to integrate with the iOS ecosystem. Unfortunately, the small screen and imprecise touch input scheme does not lend itself to clearly presenting complex technical or scientific data, so compromises must be made. This case study examines one such situation, and looks at the tradeoffs and decisions that were made.

One disclaimer: as of this writing, the product has not yet been completed. Design and development is ongoing, and the types of decisions described here are still being made. The final product may bear little resemblance to the current vision of it, but the design process so far will surely have a strong influence on any future design decisions.

The Application

The application in question accepts a data stream from a spectrum analyzer accessory, which is connected to the iOS device via the dock connector. This spectrum analyzer can produce an extremely high-bandwidth data stream, containing far more information than can be presented to the user at any one time.

In addition, the data is multidimensional: fundamentally, it represents a two-dimensional plot of power against frequency, which varies over time. Thus, there is three-dimensional data to be presented on a two-dimensional screen; choices must be made.

This is intended to be a low-cost and easy-to-use device, so it is necessary to keep the interface simple and intuitive. One of the primary purposes for developing this product is to use it as a sales tool and a demonstration of Vanteon's technical capability; this means that the system needs to be simple enough

for a non-engineer (for instance, a salesperson) to use, and intuitive enough for a potential client, who is likely not an RF expert, to understand without training.

Analysis and Presentation

The real question here is how to map the dimensions of the incoming data to the dimensions of the presentation. As discussed in the previous section, the data has three dimensions: power, frequency, and time. The bandwidth and resolution of the data on any of these dimensions can be controlled, within limits, by sending commands to the device. The task of the designer is to determine which dimension(s) of the output will map to each dimension of the input data, and how to scale the incoming data to minimize the load on the device without compromising the user experience.

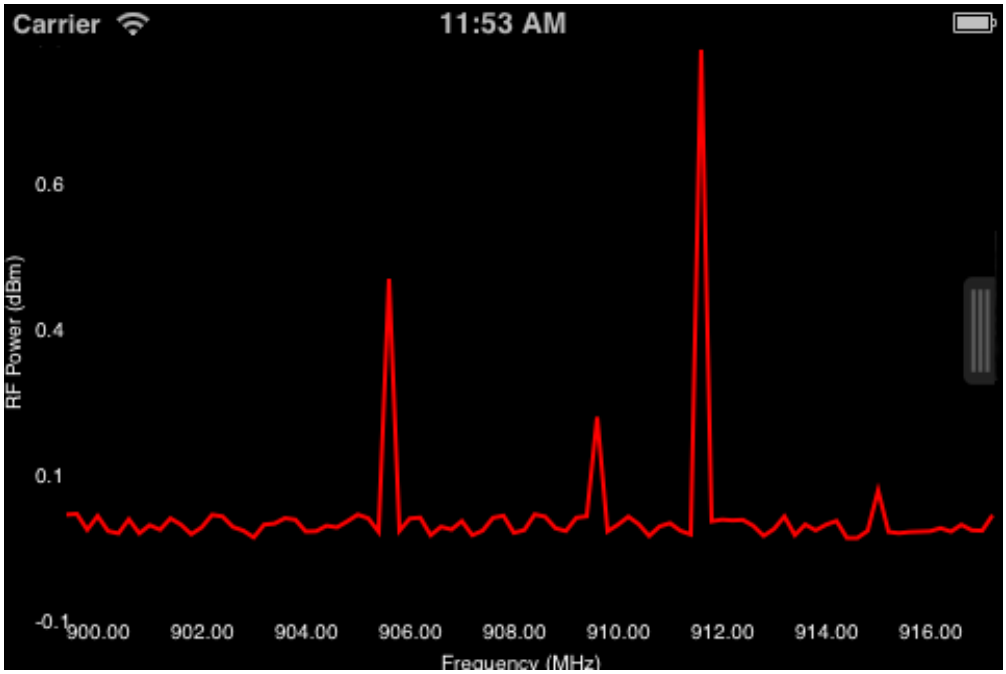
The first step was to examine existing spectrum analyzers to determine how they present their data. Several members of the team have extensive experience with these devices, so this was not difficult. Based on this work, one of the team's RF engineers used his own experience to choose several use cases that he felt were the most fundamental:

- Examining an instantaneous readout of power vs. frequency across a given band (for instance, looking for spikes at particular frequencies).
- Examining the historical trend of power vs. frequency over a period of seconds or minutes (for instance, looking for shifts or dropouts in the signals). This is sometimes referred to as a "waterfall" plot.
- Monitoring the aggregate power level over a given band in a simple way (for example, using the analyzer as a simple power meter).

Each of these use cases involves mapping the three input dimensions to the available output dimensions in a different way.

Instantaneous Power vs. Frequency Plot

In the first case, the team chose to map frequency on the X axis, power on the Y axis, and to display only the most recent reading, like so:



Note that all screenshots are preliminary – these screens are somewhat stark, and serve primarily to illustrate the concepts.

In this mode, there is no need of more spatial resolution than there are pixels on the screen; thus, the software could limit the resolution of the captured data on the frequency and power axes to the size of the display area. Additionally, displaying more than about 30 frames per second adds very little to the user's experience, so the team chose to limit the time resolution to that value.

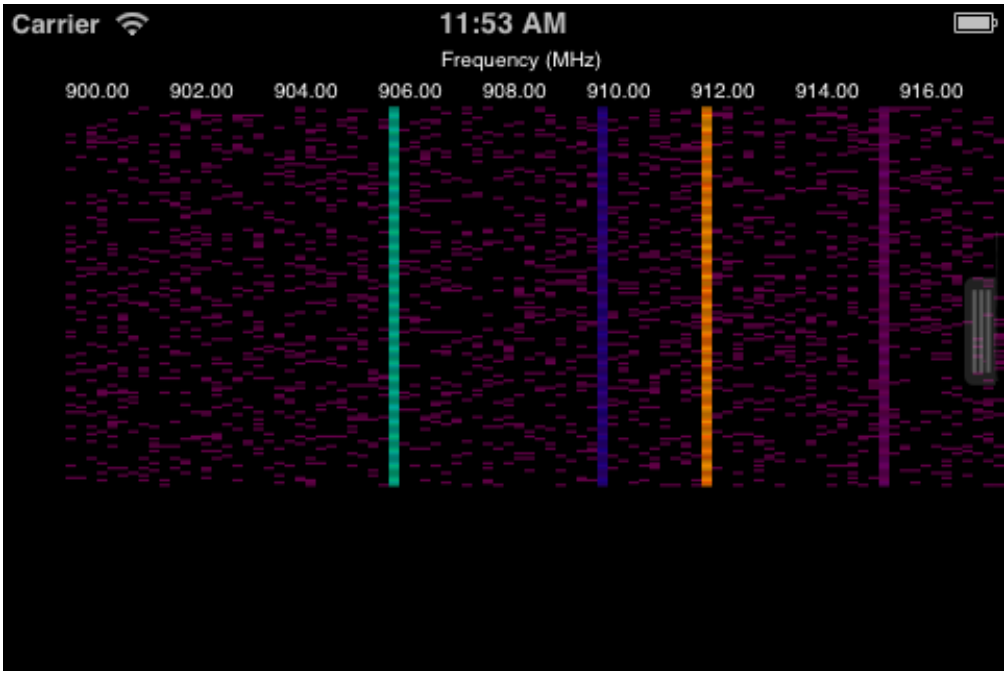
This gives a maximum number of samples of 30 frames per second times the horizontal pixel resolution of the screen, which is 2048 on the highest-resolution iPad – 61440 samples per second. The maximum vertical resolution is 1536 pixels (power levels), so 16 bits per sample is more than the software could ever need. This gives a maximum bandwidth of 122880 bytes per second, which is easily within the capability of even the older 30-pin dock connector standard.

The power readings delivered by the device are on a logarithmic scale, so a 16-bit sample offers much more than a multiple of 2^{16} between the lowest and highest power levels that can be registered. That fact, however, does not change the calculation made above – there are only 1536 pixels in which to display the values, so having the device return more than that many discrete values per sample is useless. The software could, of course, have used 11-bit samples (yielding 2048 possible values), but it was decided that packing and unpacking the values would consume more power than it was worth.

This approach allows the system to minimize the data transferred without compromising the user experience in any way. This is the first critical consideration when choosing a data mapping approach on a mobile device – **don't do any work that doesn't directly contribute to the user experience**.

"Waterfall" Plot

In the second mode, the designers chose to map frequency on the X axis and time on the Y axis, with the newest reading at the top and older readings below. This left us no spatial dimension for power, so the design uses color instead. Technically, color is a three-dimensional quantity itself – one could in principle map hue, saturation, and brightness to three separate dimensions of the input data. However, in practice the human eye is poor at picking out subtle gradations of saturation and brightness, so the decision was to use hue and brightness together to represent the input power. The appearance of the waterfall plot is shown below (note that this is displaying the same data as in the previous image).



Again, the frequency resolution can be held to the maximum number of pixels on the screen (2048). The power resolution can be capped at the number of colors distinguishable by the human eye; in principle, this would be on the order of 2^{24}, but in practice this is far more than necessary; one or two thousand colors is more than enough, so the 16-bit samples from the previous case will be plenty.

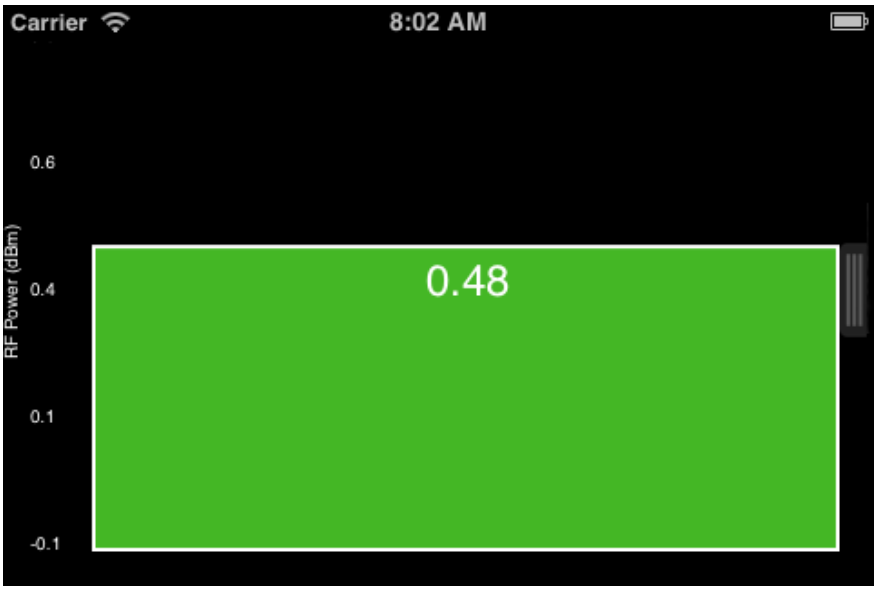
The question of time resolution is more interesting. The software could continue to use 30 frames per second, but even if each is displayed at 1 pixel high, the maximum time period displayable would be only 52 seconds (1536 pixels divided by 30 pixels per second). This would be less than optimal in other ways

as well – on a smaller device such as an iPhone 4 with its 640-pixel-high screen, the software would be limited to displaying only 21 seconds of history, and one-pixel-high samples would be nearly indistinguishable on a 326 ppi "retina" screen such as the iPhone 5's.

The team chose to arbitrarily limit the time resolution to 2 frames per second, and to scale the height of the frames to the period chosen by the user (that is, each frame is represented by a number of pixels equal to the vertical resolution divided by twice the number of seconds chosen). This allows the software to display up to 768 seconds (12.8 minutes) of history on a retina iPad, albeit at a microscopically small scale, while still comfortably showing several minutes of history even on a comparatively low-resolution screen.

"Power Meter" Mode

The final mode presented a somewhat different challenge than the other two. In this case, only a single data point is displayed at a time. However, this data point must be presented in such a way as to be easily visible and distinguishable at a distance and without the user's full attention, because they are likely using the power meter while working on something else. Therefore, it was decided to map the power to *both* the Y axis and the color, using the entire X axis to make the display large:



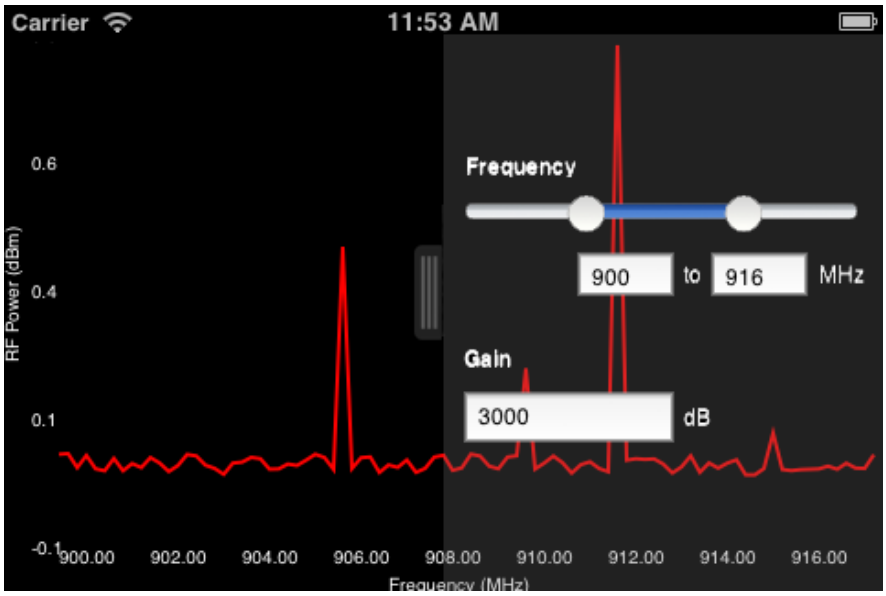
The color of the bar changes based on the reading – higher readings result in brighter, more "energetic" colors (in practice, this translates as brighter, more saturated, and closer to the red end of the spectrum). This allows the user to give most of their attention to something else – the controls of their transmitter, for example – while still tracking the power output from the corner of their eye.

Adding Control

Once the data presentation was worked out, there remained the matter of control. The three modes were carefully designed to make maximum use of all available screen area, meaning there was none left over for controls. Therefore, the team chose to place the controls in a pull-out drawer.

As shown in the previous screen images, the pull handle is on the right side of the screen, in the middle; this was the location that the team decided was the least likely to contain valuable data, since in the frequency plot and waterfall modes it's at the upper edge of the frequency band, and in the waterfall mode it's at least a few seconds into the past. In addition, the handle is semitransparent, meaning that it won't obscure the data much even if it does happen to impinge on the useful area.

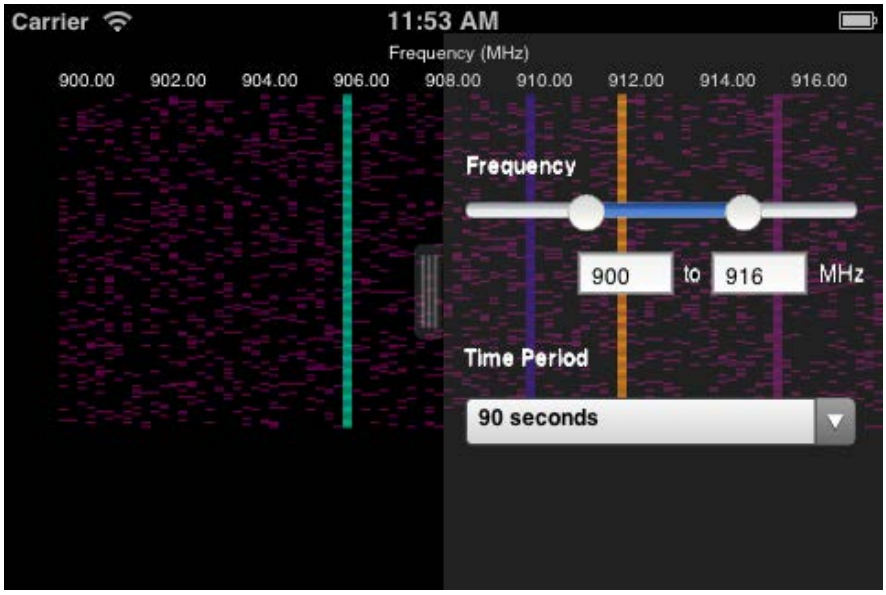
In the frequency plot mode, the drawer appears as shown:



The selection of the frequency range to be displayed is done using a two-thumb slider control, with units of 10 KHz (the minimum frequency resolution offered by the device). This permits a single-control presentation of a quantity that the user thinks of as a single value, even though to the software the minimum and maximum limits are independent. In addition to the slider, numeric fields are shown for the minimum and maximum frequencies; this permits precise control when required, and also gives live feedback while the user adjusts the slider.

The displayed power gain will not affect the samples returned from the device, but it will allow the user to “focus in” on lower-level spikes. This is presented as a numeric field, in order to offer precise control. This could be done with a slider, but that type of control can more easily be done through direct interaction with the screen (see below).

In the “waterfall” mode, the appearance is very similar:



The frequency range selection is identical to that in the frequency plot mode. However, in this mode, the user must also select the time period for the historical data. The user is unlikely to care about the precise period (e.g., the difference between 37 and 38 seconds), but more the general scope of the display, so it was decided to present this option as a drop list with geometrically-increasing periods: 5 seconds, 10 seconds, 20 seconds, 30 seconds, 45 seconds, etc.

In all modes, it was decided *not* to allow the user control over the displayed colors, fonts, axis labels, etc. This is the second vital concept for the presentation of complex data on a mobile device – **if it doesn't add directly to the functionality of the system, leave it out**. The software could certainly have permitted the user to customize every aspect of the display, but that would have made the control drawer so crowded as to be nearly unusable. By limiting the user's options to the bare minimum, the overall usability of the system was significantly increased.

Direct Interaction

Finally, the third and last lesson on presenting complex data on a mobile device: **allow the user to interact with everything they see**. Users on touchscreen devices are used to reaching out and “grabbing” things and manipulating them. Too many mobile interfaces of scientific or technical systems are simply transplants of interfaces designed for a display next to a button panel, or a mouse-and-keyboard interface. Neither is a good fit for a mobile interface.

In the case of this system, the designers wanted to offer the user direct control over as many axes of the data as possible. In the frequency plot and “waterfall” modes, the software has two-dimensional data

presentations, so the team considered the effect of scrolling and zooming on each axis in each mode. In either case, a horizontal scroll or pinch should clearly alter the displayed frequency band.

In the frequency plot, a vertical pinch should change the gain; it's not clear that a vertical scroll should allow the user to “navigate away” from the noise floor, but it was decided to allow it in order to maintain consistency of interaction between the modes, and to allow the user to “zoom in” on a peak. In the “waterfall” mode, a vertical pinch can change the displayed time interval, and a vertical scroll can navigate through the time axis (subject to the maximum amount of data that can be stored – the software obviously shouldn’t try to store every sample back to the beginning of time).

In order to maintain the apparent behavior of the “waterfall” mode, it was decided to tie the view position to a *relative* time offset from “now”; that is, the user can scroll back 10 seconds, but the data will continue to progress across the screen just as it would have had they been watching 10 seconds ago.

There is one additional interaction that the team decided to incorporate. In the frequency plot and “power meter” modes, the user has no way to directly interact with the time dimension. Since this dimension is not mapped to a spatial axis on the screen, pinch and scroll gestures don't map intuitively to the time dimension. Instead, it was decided that a simple tap would “pause” the display, and another would “resume” (jumping to the current moment; since in these modes there is no historical display, it doesn't make sense to try to pick up where they left off.)

Summary

The interface of a mobile device is severely constrained when compared to a standard computer or a custom-designed device with a purpose-built button panel. It is necessary to consider carefully the possible ways to map the input data to the possible dimensions of output available. This mapping has dramatic effects on the usability of the final system, and many different combinations were explored before the specific modes described here were chosen.

All the decisions discussed here were made based on a single set of guiding principles: do nothing that doesn’t directly help the user, and let the user interact with everything. This lets the system maximize the utility of the minimal interface available, and gives the user the most intuitive experience possible.

About the Author

Aaron Sher has been programming professionally for over 25 years on platforms including MS-DOS, Windows, Macintosh, Linux, and Android. He joined Vanteon (then Millennium Computer Corporation) in 1993, developing multimedia applications on Macintosh System 7. Since then, he has taken part in many dozens of projects including e-learning, image processing systems, consumer software, enterprise Web development, device drivers, and many others. He has been a developer, a software architect, a project lead, and a project manager, and is currently working as a senior software engineer and project lead.