A System for Visual Analysis of Radio Signal Data

Tarik Crnovrsanin, *Student Member, IEEE*, Chris Muelder, and Kwan-Liu Ma, *Fellow, IEEE*

Abstract—Analysis of radio transmissions is vital for military defense as it provides valuable information about enemy communication and infrastructure. One challenge to the data analysis task is that there are far too many signals for analysts to go through by hand. Even typical signal meta data (such as frequency band, duration,and geographic location) can be overwhelming. In this paper, we present a system for exploring and analyzing such radio signal meta-data. Our system incorporates several visual representations for signal data, designed for readability and ease of comparison, as well as novel algorithms for extracting and classifying consistent signal patterns. We demonstrate the effectiveness of our system using data collected from real missions with an airborne sensor platform.

Index Terms—Intelligence Analysis, Coordinated and Multiple Views, Time-varying data, Geographic/Geospatial Visualization

1 Introduction

The radio frequency spectrum is complex and dense, with thousands of events occurring simultaneously every second in a typical suburban environment. These events can include both authorized and unauthorized frequency usage from the Federal Communications Commission (FCC) perspective, potentially criminal activity from a legal perspective, or even naturally occurring noise phenomena. Differentiation between signals of interest (SOI) and non-signals of interest (NSOI) is important not only for domestic radio frequency use management, but also for military intelligence gathering. For military applications in particular, rapid analysis of such data is vital. However, this classification task is resource intensive because of the wide variety of signaling systems that are both in use now and expected to become available in the future. The wide variance in signals has led to the development of Signal Intelligence (SIGINT) sensors which can capture the information from these signals in real-time.

The growing use of SIGINT sensors in today's military enterprise dramatically increases the amount of data flowing into Intelligence, Surveillance, and Reconnaissance (ISR) data processing centers. Modern SIGINT sensors ingest nearly all signals in the environment simultaneously and thus produce vast amounts of signal data at an incredible rate. Traditional tools found in ISR processing centers can easily overwhelm an operator who is inundated by the volume of incoming data.

There are also constraints on the amount of processing that can be done ahead of time. In many situations, signal information is best collected from the air with specialized antennae as a high vantage point reduces line-of-sight blockage and eases moving the signal collection platform to the required location. However, the airborne collection of Radio Frequency (RF) signals puts severe constraints on the size, weight and power of the equipment, and thus the capabilities that can be installed. Dedicated streaming hardware is used to continuously sample measurements of the external characteristics of transmissions, including frequency, signal-to-noise ratio (SNR), bandwidth, up time of the signal, off-time of the signal, and, in a multi-antenna collection system, the direction from which the transmission originates. However, these per-signal measurements are instantaneous, and signals need to be comprised of many such samples. So the additional hardware is often dedicated to compiling these samples into meaningful transmissions. This still leaves the operators with having to sift through innumerable signals in order to find particular higher order patterns (such as communications) that they might be interested in.

Analytics can be applied to pull out specific features, but there are numerous analytic methods that could be applied, and the potential for future development of any number of situational analytics. Visual analytics approaches would provide a framework to manage such analytics, allowing analysts to focus on the more important high level tasks. We have developed a system that provides a visual workflow to manage a suite of such analytics by providing summarizations of potentially interesting signal traffic patterns, while still exposing the underlying analytics to extract specific patterns that they might be interested, and enabling the development or application of situationally specific analytic processes. In this manner, our system aims to aid in deriving solid intelligence in near real-time, providing the ability for operators to quickly identify combatants and potential opportunists while discounting allies in time-critical situations.

Due to the scale of the data and the complex relationships between transmissions, understanding signal data is a nontrivial task. To our knowledge, the visualization community has not substantially explored this type of data. We have developed an interactive visual analysis system to support this task by working closely with expert analysts to obtain constant feedback and to guide the design and development of our system. When talking to analysts, we found that they were particularly interested in answering the following questions:

- Can we identify communications in the pattern of the signals?
- Is it possible to discover and locate signal repeaters?
- How can we guide analysts to signals of interest?
- What general discovery can we make about the data?

In this paper, we present a system designed to explore and analyze radio signal data. This system consists of a combination of several visualizations and algorithms that help analysts answer important questions. The contributions of our work are:

- A new system for visual representations of radio signal data.
- An interface to manage the workflow of radio signal analytics.
- A novel algorithm for finding repetitive (digital) signal patterns.
- Demonstrations of identifying repeaters, communications, and repetitive patterns.

Most importantly, we have created an effective visual analytics solution to a very important application.

- *Tarik Crnovrsanin is with VIDi @ U. C. Davis: tecrnovr@ucdavis.edu.*
- *Chris Muelder is with VIDi @ U. C. Davis: cwmuelder@ucdavis.edu*
- *Kwan-Liu Ma is with VIDi @ U. C. Davis: ma@cs.ucdavis.edu.*

Manuscript received 31 March 2007; accepted 1 August 2007; posted online 2 November 2007.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

2 RELATED WORK

Our work draws from a variety of existing research, including wavelet visualization, communication visualization, geospatial visualization, parallel computation system analysis, and the coefficient of variation.

Wavelets analysis Wavelet analysis has a wide range of uses in computer science [10]. In this section, we focus on its application to signal processing [22], which is directly applicable to our work. Miller et al. [19] applied wavelet transformations to custom digital signals constructed from words within a document. The resulting wavelets are used to analyze the characteristics of the narrative flow in the frequency domain, such as theme changes. Faith et al. [9] applies wavelets to optical wireless signals and then runs PCA on the resulting wavelets to create a scatterplot visualization. Barford et al. [3] used wavelets for anomaly detection. The pseudo-spline filter can expose distinct characteristics of each class of anomalies: outages, flash crowds, attacks and measurement failures. Muelder et al. [20] applies wavelet scalograms to network scan patterns. The resulting wavelets are used to create a graph in which a node is a scan and an edge exists between two nodes if they are highly similar. Our wavelet use is inspired by these works, but differentiated by the discrete nature of the signal data to which our system is tailored to.

Communication detection There is extensive research on the duration of gaps, pauses and overlaps in conversations [8, 13, 12] which focuses on person to person communication. Walkie talkie conversations do not share the same characteristics as person to person conversations, but the research provides some good general rules that are applicable to our work. While some existing works discuss analysis of the contents of communication records [4, 5, 7], the data we were working with only consisted of the signals' metadata. However, in situations where the signal contents are available, our system could work in concert with such approaches as our metadata analysis by extracting conversations that such content-based techniques could be applied to.

Geospatial visualization Visualization of geospatial data (or 'Geovisualization') has been an ongoing research topic for many years [17], which has produced many geospatial visualizations and analytics [1, 2]. Some approaches focus on the particulars of radio signal data. Han et al. [11] presented a visual analytics system for the development of signal fingerprinting-based localization systems. Though their approach works with radio signals, it depends on the intricacies of precise, indoor signals, and would not scale up to the size or uncertainty of our data. Wood et al. [27] apply graph/matrix based techniques to the analysis of pairs of discrete origin and destination locations, such as a communication network could form. However, our data is too noisy for such a discrete technique. Rather, we borrow from techniques for geospatial that can handle uncertainty such as splatting and heat map techniques, as in the works of MacEachren or Thomson [18, 25].

Parallel computing systems Our signal data visualization problem shares surprising similarities with parallel computing visualization. Each processor, like a frequency band, has a start time and duration for each task, similar to a signal. For instance, Gantt charts [26] look very similar to frequency versus time plots (which can be seen on the bottom of Figure 3). Many scalable performance visualizations [29, 23, 21] use techniques that show system resource utilization, the timings and durations of parallel events, or the application executions. These use the aforementioned Gantt charts and other applicable visualizations such as histograms and Kiviat diagrams. Spear et al. [24] and Landge et al.[16] focused on node-link diagrams or matrices to display the communication topology. Communication between processes is important as it has direct impact on performance in a parallel environment. Many toolkits combine both types of visualizations.

Coefficient of variation The coefficient of variation is used in probability theory and statistics as a way to show the variability of the mean in relation to the standard deviation. Xu et al. [28] proposed a new similarity metric called variation coefficient similarity based on an extension of the Dice and cosine similarity measures. They demonstrated the effectiveness of their metric by comparing it to three other prompt similarity metrics. Though the metric shares the same name as coefficient of variation, it is not the same. Their metric works on a set of vectors and relies on an alpha value.

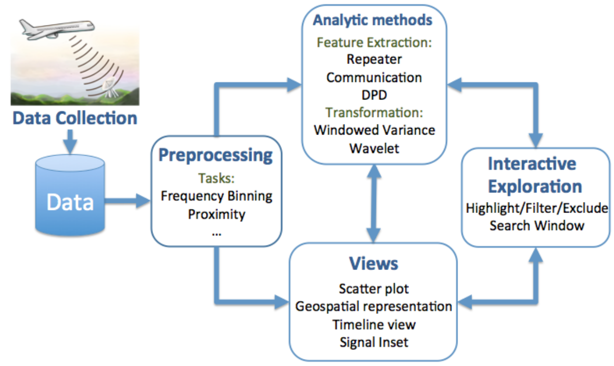

Fig. 1. As the data is loaded, several preprocessing steps are applied. Afterwards, the data is saved in a binary format with the included preprocessing results. The user can either start with the geospatial representation, timeline or calculate one of the analytic methods. The interactive process allows the user to define the scope of his alteration, by specify which views are affected by the interaction.

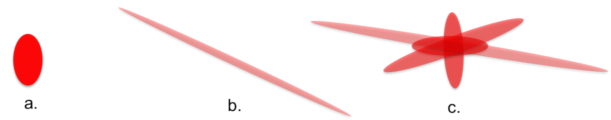

Fig. 2. A signal location is derived from the sensor's location, therefore, an ellipse is drawn to show the error cone (a). A benefit of the error ellipse is it can show us the relative position of the plane (same angle as the larger of the two radius). As the area of the ellipse increases, the opacity is reduced (b). Several ellipses can converge to better approximate a transmitter's location (c).

3 SYSTEM OVERVIEW

The overall workflow of our system is shown in Figure 1. Data is streamed in from the detection platform and accumulated for our system. Once the data is loaded and goes through the preprocessing step, the user can run one or more of the operations to explore and interact with the views. The system is built around exploration so where the user starts can depend on the task at hand. Depending on if the user is interested in repeaters, communication, digital signals, or a combination, he would run the corresponding algorithms before exploring the results; if the user is interested in geospatial or data specific (timeline) information, then he can start to filter and explore through the views before running any calculations.

Data Collection The data is collected from an airborne detection platform which continuously ingests detected signals' properties such as their strength, their power, their signal to noise ratio (SNR), and the direction they are coming from relative to the platform. As the collection platform detects these signals, the platform pre-processes them to determine geographic locations and to derive continuous signals from the same location. Since the platform detects transmitters as a directional ray, computing the geospatial information is dependent on both the plane's and transmitter's locations, as well as the orientation of the plane. But due to limits to the sensor's precision, there is an error cone around this ray, and the source of any terrestrial signal would lie somewhere in intersection of this cone on the surface of the earth. For simplicity, this intersection area is approximated by an ellipse, as shown in Figure 2(a). As the distance between the plane and the source increases, this intersection elongates as shown in Figure 2(b), and thus the uncertainty also increases, with an extreme limit when the ray hits the Earth's horizon. Interference, such as terrain or other noise can also affect the size/shape of this ellipse. However, after subsequent analytics show that multiple signals were detected in the same region, overplotting these ellipses can aid in identifying more precisely where the transmitter actually is, as shown in Figure 2(c).

Once the data is loaded, we also apply another preprocessing step, in order to help reduce noise in the data and pre-compute simple metrics. For instance, signals have a minimum frequency difference so they do not interfere with each other. When a signal is measured, it can have slight variation in frequency but will be within the legal band for that signal. So we use this step to bin frequency bands that are approximately identical. Additionally, the system sorts the data by frequency bands and time. We also compute additional metrics that are useful in later, more complicated analytics, such as the proximity of the plane to the signal. All these calculation and the data are also converted and stored for faster future loads.

Analytics Our system currently has five implemented analytic methods; four of them designed around specific tasks, plus one for more general exploration and discovery. The repeater algorithm finds signal repeaters by looking for collections of signals that have the same start and duration times. The communication detection algorithm is a rule based approach that tries to find a set of signals that make up a communication pattern based on time between signals and their locations. The windowed variance analytic algorithm finds series of signals that have high variance in temporal duration by applying a coefficient of variation metric. The digital pattern distinction algorithm also uses the coefficient of variation across a sequence of events, in order to separate series of signals that make up a digital patterns that exhibit consistently low variance from the remainder of analog signals. Lastly, the wavelet transformation is for more general analysis instead of a specific task; it samples the data from time windows defined by user parameters, then projects the higher dimensional results of the wavelets to a two dimensional representation using PCA. These analytics were developed to be modular, so that the user can dynamically link up the analytics as desired, or easily implement new analytics.

Visualizations When creating the visual system we had two goals in mind: keep the visualization intuitive for our expert users, and allow the users to gain insights for making crucial decisions. The visual representations must be kept simple because of the sheer volume of data and associated analysis tasks. Many classification tasks are difficult to compute automatically with certainty, such as determining whether signals are part of a communication, or if they are digital or analog transmissions. Rather than making these decisions completely computationally, we use analytics to compute probabilities that a series of events are one or more of these types of signals. We then plot these candidates and allow the user to inspect and group them as appropriate.

As Figure 3 shows, our system has three different views: the main, map and timeline. The main view shows the results of the algorithm methods and can toggle between different outputs and their views. The map view provides geospatial information and a reference point for the results. The timeline allows the user to filter the data based on attributes in the data and also serves as the overview for the data. The signal inset is triggered when an aggregate data point is clicked on and shows the underlying pattern. The workflow panel exposes the data workflow to the user, and enables the user to connect and combine the analytics as desired. If the user is interested in a particular analytic or combination of analytics, he would put the corresponding algorithms into the workflow and link the results to the visualizations.

4 ANALYTICS METHODS

In this section we describe algorithms we use. Most identify specific features of interest, such as repeaters or communications, while the wavelet algorithm is more for general exploration.

4.1 Windowed Variance

The Windowed Variance analytic was developed to measure how repetitive or consistent the detected signals are. We compute this works by first creating time windows based on user defined parameters: window size and step size, which allows for overlapping windows. We then create lists of events that fall within each window. As long as events either start or end inside the time window, they are included in the calculation. While this does duplicate overlapping events, trimming an event to fit inside the window would introduce variance into sequences which had little to no variance, such as digital signals. This would unfavorably bias the algorithm, so instead we always use the whole signal lengths.

Once we have the sequences of signals per time window, for each sequence we compute the Coefficient of Variation (CV), defined as the standard deviation divided by the mean, for both the durations of each signal and the gaps between signals. Time windows that have only one or two events are skipped because there is not enough data to calculate the CV for the gap and duration. The run time of this algorithm is $\Theta(2N)$: one pass to create the time windows and a second to the calculate the CV. As each band and window is independent, this process is parallelized to make it even more efficient by using threads.

In this manner, signal patterns of high variance (such as communications) can be separated from those of low variance (such as digital signals). This approach is general enough to handle multiple cases in between these extremes, and provides the spectrum of occurrences to the user in case there are interesting patterns in the middle somewhere. However, if the user is only interested in isolating just the digital signals (or filtering out the digital signals), the DPD algorithm is more focused to that specific task.

4.2 Digital Pattern Distinction (DPD)

Digital signals often exhibit an extremely regular pattern of consistent signal durations, whereas analog signals are more varying. That is, we define digital patterns as sets of signals where transmission durations and gaps between transmissions are very consistent. Being able to identify a digital or analog signal pattern can help to reduce the problem set; for instance, communication should generally only consist of analog signals when the conversation is among people.

While the goal of the Windowed Variance is to determine how consistent or inconsistent patterns are within constant sized windows, the DPD algorithm was designed to detect and classify sequences of highly consistent sequences of arbitrary length, in order to extract the digital patterns specifically. To compute the DPD, for each frequency band, we spawn a thread that iterates over events, keeping track of a running coefficient of variation (CV) for the durations of both the signals and the gaps between them. To avoid having to completely recalculate the mean and standard deviation at each iteration, we use an incremental formulation to compute the CV:

$$\begin{aligned} E(x_{n+1}) &= \tfrac{n \cdot E(x_n) + x_{n+1}}{n+1} \quad & E(x_{n+1}^2) &= \tfrac{n \cdot E(x_n^2) + x_{n+1}^2}{n+1} \\ \sigma(x_n) &= \sqrt{E(x_n^2) - (E(x_n))^2} \quad & CV &= \tfrac{\sigma}{E(x_n)} \end{aligned} \tag{1}$$

where $E(x_0)$ and $E(x_0^2)$ are zero and $E(x)$ is the running average.

For each event, if both the gap CV and the signal CV are below a threshold then the event is appended to the current sequence, and the algorithm continues on to the next event in the frequency band. If adding the event to the list would exceed the CV's threshold, the algorithm will terminate the current sequence and save the statistical metrics. The algorithm will then continue with the event that it could not add and repeats the process. In this manner, repetitive digital signals will form long sequences of low variance, while analog signals will not. There is no consensus in the literature of a good CV value. We use 10 % because there is a slight variance in duration and gap that can be due to many factors e.g. noise or calibration. The CV can also be changed by the user to fit their needs.

One constraint of using the CV is that it does not work on interval scales, but since the durations of both the signals and gaps are positive ratio scales, we do not run into this problem. Another potential side effect of this approach is that as more events are added, each new event has less impact on the mean and standard deviation. Thus, a sequence of events could start very regular and gradually become more erratic but still be added to the sequence. As there is some data collection error though, this actually helps in creating longer sequences, even if there is some noise or dropped signals. Even in the case where an event is not captured correctly and a long sequence is split in half, both parts should still have the same CV in both gap and duration and so

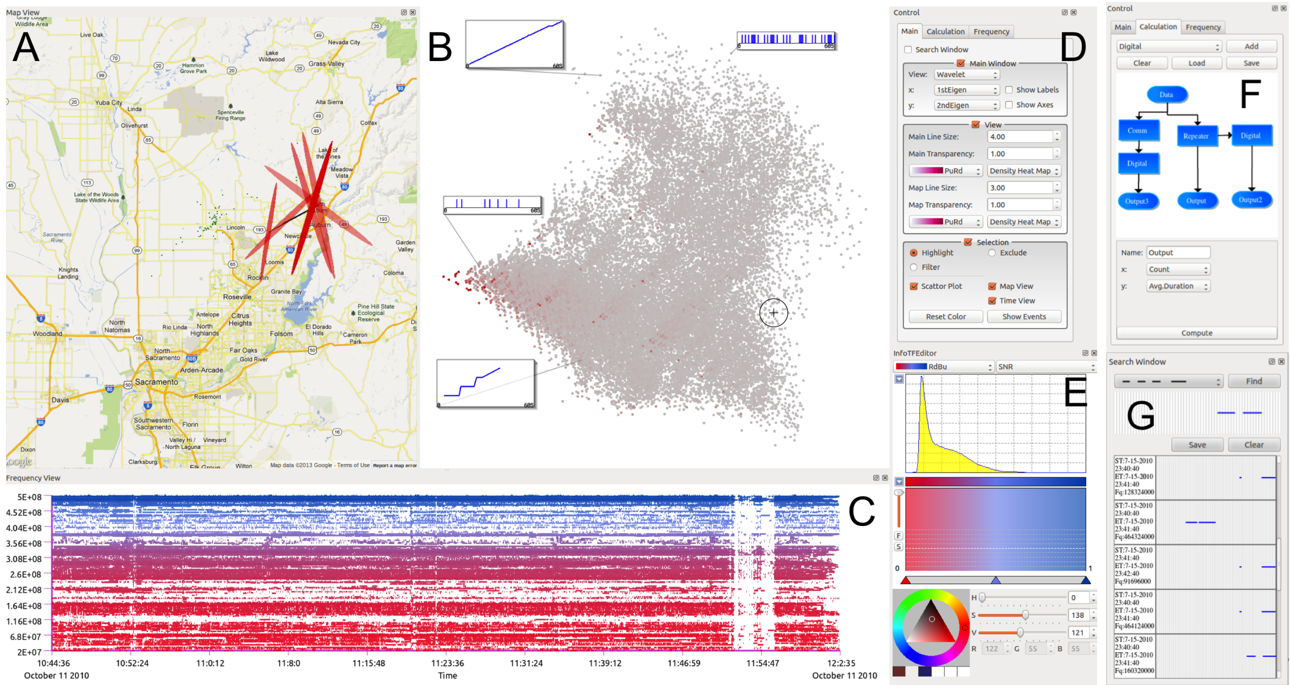

Fig. 3. The system has three primary views (A-C) and several interface tools (D-G). The map view (A) shows the locations of selected signals' sources geospatially. The main view (B) shows the results of the analytic algorithms, and allows the user to inspect or select them. The timeline view directly plots a parameter of the signals over time as lines for the signals' durations (frequency bands shown here), and allows the user to filter by that property or by time. The menu control (D) provides additional user controls for the three different views. The color editor (E) is used to create or modify a color gradient and provide histogram of each data property . The algorithm workflow (F) allows the user to visually select the algorithmic methods by directing the flow of data from source to view. The search window (G) gives the user the ability to draw a specific signal pattern and search for it in the results. Previously drawn patterns can also be loaded.

would still be plotted together. The run time of this algorithm is O(N), and like the Windowed Variance algorithm, it is heavily parallelizable due to the calculation being independent of the frequency band.

4.3 Repeater detection

A repeater is a device that receives one set of signals and rebroadcasts them - often to extend the range between low powered devices or to cross terrain such as hills that would block communication. In our signal data, a repeater shows up as a series of events that have the same start time and duration across at least two different frequencies (i.e. pairs of the initial transmissions and the rebroadcast transmissions).

To detect these, the repeater algorithm iterates through a sorted list of all events, and groups events that share the same start time and duration as candidate repeated signals (i.e. synchronous events from two or more frequency bands). Then we group candidates with identical frequency bands, as multiple repeated signals on the same frequencies are likely the same repeater. While it is possible for two different repeaters to share some subset of frequency bands, they would generally interfere with each other if they were operating on the same set of frequencies. This process is straightforward since the initial candidates have their events sorted by time. We do not incorporate the geographic information in this computation because of its low precision. Also, it is possible for a repeater to be mobile. And even pairs of signals in which both signals lack good geographic information can be relevant as they can provide insight into the use of the repeater. We group candidates together regardless of the elapsed time between them.

By allowing the system to group candidates independent of time, the algorithm can generate several potential scenarios. The best case is that many pairings belong to a single repeater, as having more data improves our chances of triangulating the correct location of the repeater. There is the possibility that two or more repeaters that share frequency bands could get paired together into one repeater. However, this can easy be seen in the map view, as there will be two or more geographic centers. Finally, one limitation to our approach is that it can also group less desired points such as white noise or points with no geographic information. If there are only a few signals grouped in such a way, then the user would likely ignore them. But if the signal count in such a grouping is high, then further investigation could reveal a repeater whose location was not identified with sufficient precision, as it is unlikely for so much noise to group together at random. Even in such cases, where there is not enough information to precisely triangulate the repeater, just knowing that a repeater exists is an important insight that could warrant another flyby to determine its location.

The most expensive computation here is sorting, which takes $O(N \cdot \log_2(N))$ where N is the number of elements. Luckily, this step can be avoided when the data is recorded chronologically (as in real-time analysis) or when the data is presorted beforehand. The repeater algorithm itself takes O(N) to find candidates and $O(M^2)$ to group the candidates, where M is number of candidates. In large data sets the grouping step can take longer to compute than finding candidates. A potential speed up is that once a candidate is found, we could sort the frequency bands that comprise the candidate, then use the sorted frequency bands as a key that maps to an array. This would take $O(M * \log(K))$, where K is the number of keys and K << M.

4.4 Communication detection

For communication, we created a rule based algorithm. We looked at several papers on the proper duration of gaps and pauses between communicating individuals. Most of the research, however, is done on conversation that is either over the phone or in person. Since our data is comprised of half-duplex(i.e only one transmit at a time) handheld transceivers, conventional conversation rules do not apply quite as rigorously. Thus we applied some more relaxed rules. First, the duration of each signal in the communication needs to be at least one second. We found this to be reasonable since any confirmation takes more than one second to transmit when following radio transmission etiquette (e.g. "Roger that. Over."). Similarly, we set the maximal gap between each signal to be no more than 30 seconds, as pauses longer than that could signal the end of a conversation. We also require that the initial signals are from different locations, and thus we ignore any signals that do not have positional information. However, due to the precision of the instruments, determining if two signals have the same location is not that straightforward. We consider points to be in the same location if their error ellipses overlap, and different locations otherwise. The communication algorithm runs in O(N). Since the frequency bands are independent, this algorithm can also be heavily parallelized. In our system, we use threading to speed up the process, where each thread is assigned to one frequency band.

4.5 Wavelets and Dimensionality Reduction

One way to look at the data is to analyze patterns of activity according to their similarity. We can define these patterns by treating each frequency band in the data as a time series. Then the sequence of captured signals expresses itself as a square wave in this time series and similar patterns can be detected through frequency analysis. We choose to use wavelet scalograms [20] both because they are naturally tuned to such square wave patterns (unlike Fourier analysis which works with sinusoids), and because wavelets are rather resistant to phase shifts and noise: similar patterns will have similar wavelet signatures even if the patterns are shifted slightly or parts of the pattern are missing. Conversely, different signals should produce different wavelets.

While wavelets are often useful in signal processing applications for general frequency analyses, in which the data is continuous, the captured signal data is stored as discrete data made up of events with start times and durations. To generate time series to use in the wavelet scalograms, we first sample the data according to sliding time windows, which are defined by user controlled parameters such as window size, sampling rate and overlap amount. Since the data is sorted temporally within in each frequency band, these windowed time series can be generated in a streaming manner. For each frequency band, we first initialize all sample point values in each window's time series to 0. Then we iterate over the events that intersect temporally with the time window, setting the values in the intersection to 1. Each window's time series now comprises the D_0 array used in the wavelet calculation. The scalogram $(\mu_0, \mu_1, \ldots)$ we calculate recursively as:

$$
\begin{aligned}
D_k &= (d_{k,1}, \ldots, d_{k,2^{n-k}}) = \left(\tfrac{d_{k-1,1}+d_{k-1,2}}{2}, \ldots, \tfrac{d_{k-1,2^{n-k}-1}+d_{k-1,2^{n-k}}}{2}\right) \\
S_k &= (s_{k,0}, \ldots, s_{k,2^{n-k}}) = \left(\tfrac{|d_{k-1,1}-d_{k-1,2}|}{2}, \ldots, \tfrac{|d_{k-1,2^{n-k}-1}-d_{k-1,2^{n-k}}|}{2}\right) \\
\mu_k &= \sum \tfrac{S_k}{2^{n-k}}
\end{aligned}
\tag{2}
$$

for $0 < k < n$ (where n is the smallest number large enough for D_0). At each recursion the μ values are the mean of the corresponding data series, which approximate the variance at each resolution, and hence at each frequency scale. More complicated wavelets can be calculated by changing the functions used to calculate D_k and S_k. We found that the basic D_k and S_k functions above provided sufficient results for our signal data. Though we are generating a significant amount of data from sampling and overlapping, the amount of stored data is only $\log_2(n)$ in size per wavelet.

Once we have computed the wavelets, we need a visual representation. Our goal is to place wavelets with similar signatures next to each other, so there are many possible techniques, such as clustering or dimensionality reduction. We chose to use the dimensionality reduction technique known as PCA [15], as it is simple, but good at extracting the most prevalent trends in the data. We found that it also arranges the points based on duration and consistency. While more complicated dimensionality reduction techniques exist, PCA produced reasonable results that were sufficient for our analysis.

5 Views

In order to interact with and understand the results of the analytic processes, we use a number of visual representations and interfaces.

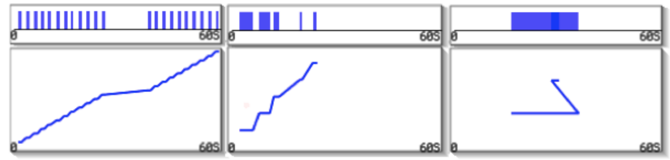

Fig. 4. A sequence of signals can be represented by plotting them in 1D over time (as in the timeline). The top set of images show this for a single sequence. We devised a line based method, where signals are plotted horizontally and gaps are plotted diagonally. Our method provides easier comparisons between signal event sequences. Small signal durations sequences can be detected (at left) as well as anomalies. On the right, it is hard to see the two overlapping signals in the top image while it is very apparent with our method(bottom image).

The timeline, which can be seen in Figure 3.C, provides a series of simple plots to filter and examine the data. In all timeline plots, the x-axis corresponds to time, while the y-axis is either one of a number of derived values or an attribute from the dataset. For instance, plotting frequency band versus time (as in Figure 3.C) produces a Gantt chart that provides a simple and intuitive initial summary of the data set. In this example, the signals are colored via a user defined color map (defined via the color map editor in Figure 3.E).

One derived values that was found helpful is the proximity of the airplane to the signal sources, as closer signals are more detected more precisely. We use two proximity metrics: the first is the standard proximity which we calculate in a similar manner as in [6] and the second proximity metric divides the signals based on which side of the airplane a signal originated from. This allows us to see when the airplane makes a turn. It also separates entities that look close in proximity space but are on opposite sides of the airplane. While not that relevant to the results shown in this paper, our collaborators found this very helpful in analyzing the behavior of the data collection platform itself.

The map view provides geospatial reference to the user and is shown in Figure 3.A. We use the Google Maps API to generate the background map and plot the points using OpenGL. It was necessary to include the map not just to provide geospatial information but also to help interpret the results from the analytical methods. When a selection is made in the main view, the map view can provide additional functionality depending on which metric is being viewed. For repeaters, we draw lines between pairs of repeated signals. This helps identify which source is the repeater, as a single repeater would link to multiple transmitters. For communications, we draw lines between the initial transmitters and the first responders.

Sometimes the GPS locations are not accurate and thus showing error ellipses is useful. We map the size of the ellipse to transparency, where the bigger the ellipse the more transparent it becomes, illustrated in Figure 2. This makes accurate GPS information stand out while the less accurate fades away, and having several overlapping big ellipses accumulates to give a better approximate signal location.

The main view, pictured in Figure 3.B, holds the visual results computed by our algorithms. As the points in this plot are aggregations, and not individual signals, we can not employ the same per signal color mappings used in other views. Instead, color is mapped to another selected property, such as the density on screen or the variance of the aggregated data. The axes depend on the analytic. For the wavelet projection, the axes are the first two dimensions of the PCA projection. For Windowed Variance, the axes are the standard deviations of the duration of signals and of the gaps between signals for each time window, on a log scale. For the digital, communication, and repeater algorithms, we map number of associated events to the x-axis and average duration of the events to the y-axis.

We also added a search window, shown in Figure 3.G, which provides the capability to look for a particular signal pattern in conjunction with the wavelet view. The user either draws a pattern in the top segment of the panel or loads a previously saved pattern to commence a search. The system then calculates the wavelet scalogram of the search pattern, and uses the PCA's component matrix to project the search pattern into the wavelet visualization. We use a cross-hair representation as a glyph to help the user identify where in the PCA space the pattern is located, with a circle of a fixed radius around the target pattern to both make the target more easily visible and to indicate neighbors. Results inside the circle are displayed under the search button in the search window, ranked by proximity to the target signal pattern. Saving patterns from one dataset and loading them into another dataset allows the user to see how it is mapped in the different space, which is important since PCA is not consistent between datasets.

In the signal inset, there are two ways of showing the underlying signal pattern, which is shown in Figure 4. The standard way to visualize signal data is to draw lines that represent the start and end time of each signal on the x-axis. The y-axis splits the window up based on how many sets of signals are portrayed. Our method keeps the same x-axis setup but changes the y-axis so that each sequential event is above the previous one. Then we connect all the signals events forming something similar to a line plot. Steep slopes represent a quick succession of events, while gradual slopes show long pauses between events. One benefit of our method is it provides a visual metaphor for the signal patterns. For instance, digital patterns, on the left in Figure 4, show up as staircases. It can also show anomalies in the data. On the right, we can see that there is two signals overlapping each other, something not apparent in the standard view.

With a context menu, the user can pick which representation he wants to view. The same point can be viewed in both representations by opening two separate windows. Some patterns might look similar but have different temporal size. To differentiate them, we overlay a timeline and label the temporal spacing. For comparison, the user can either open different signal inset for each point or view multiple points in one signal inset. The viewer selects which method to use. Opening up different insets lets the user look at the general patterns while maintaining the same scale. Placing multiple points in one inset provides one-to-one comparison.

5.1 Opacity Tone Mapping

Our approach renders large amounts of data to the screen, often results in many points or lines overplotting. A common way to resolve this overplot is to make them semitransparent and use alpha blending to combine them. However, this very quickly runs into limitations as the number of elements increases. The standard 8-bit alpha buffer only allows for a maximum overplotting of 256. Furthermore, the opacity has to be set so low that outliers are nearly invisible. In order to keep both the opacity of outliers high and the combined opacity of dense overlap from overflowing the alpha buffer, we utilize opacity scaling techniques similar to [14]. In our implementation of this technique, we first render to a high precision density buffer D which keeps track of the total amount of overplot and to a high precision color buffer C which blends the input color information with opacity inversely proportional to the density information to result in an average color that is fully opaque. We then combine these buffers with a transfer function to render the final pixels P to the screen. In order to be able to handle many orders of magnitude variance in the data density, we then combine these buffers with a logarithmic transfer function to render the final pixels P to the screen, which is defined as :

$$P_{x,y} = C_{x,y} \times \left(o_{min} + (1 - o_{min}) \times \frac{log(D_{x,y})}{log(D_{max})} \right) \tag{3}$$

Where o_{min} is a user defined minimum opacity level and D_{max} is the maximum level of overplotting that occurred. By calculating the final opacity in this manner we guarantee that any outliers will have at least opacity o_{min}, that no overplotting exceeds the maximum opacity and that the system can handle orders of magnitude of overplotting.

In several views, we alternately use just the density buffer to generate a density heat map, discarding the initial color mapping. As before, we apply a logarithmic transfer function, but then we use the resulting value as a lookup into a 1D color map texture. In this heat map, the pixels are computed as:

$$P_{x,y} = ColorMap \left(\frac{log(D_{x,y})}{log(D_{max})} \right) \tag{4}$$

6 Exploration and Interaction

We aimed to keep our UI and interaction design as simple as possible. Within the main window, the user can mouse over or click on a point in the main view to create an signal inset that shows the underlying signal pattern(s) that the algorithm used to compute that point. Several of these windows can be opened with a click for comparison. The window can be resized or moved, and each window points back to the aggregated value. For a direct comparison, multiple points can be loaded into one window.

We use the rectangle and lasso selection in our system. The rectangle selection is used in the timeline while lasso is used in the other two views. In the timeline, the user is interested in a particular range for a given attribute, so it makes sense to use rectangle selection. Lasso selection is used when the selection process is difficult, as is the case in the other views where the user has to sort through multiple points to get the selection he wants.

The control menu, pictured in Figure 3.D, has three tabs. The main tab controls which calculated view is shown, allows minor visualization adjustments, and changes global selection properties such as configuring which operations affect which views (e.g. the user might want to exclude all repeaters from the map but still keep them in the time view). There are three types of selections: filter, highlight, and exclude. The filter and exclude selections have to be handled carefully in algorithms where the data is aggregated. Removal of one signal from within an aggregated group of signals would invalidate or at least change the value of the group's derived metric. To avoid such errors, we simply remove the derived point when this happens. For highlighting, color interactions are applied only to their representation within the detailed signal insets instead of the aggregate point.

The user can adjust the colors by selecting a premade colormap or by manually changing the colormap, and can select which property to map to each color. The left color wheel provides color selection for the highlighting, and the color legend shows the current color map. The color legend is fully customizable with abilities to add new color or change existing ones. Any changes made to the color legend are saved, even when switching between colormaps. The calculation tab allows parameters to be changed and the algorithms to be queued up. For wavelet calculation, size of time window, sampling and overlap can be set. The last tab is the interface to the time view and controls what attribute is plotted.

6.1 Algorithm Workflow

We had two goals in mind when designing the algorithm workflow: easy to use and modular. We achieve these goals by providing the user with a visual metaphor while at the same time making it easy to add new operations to the system. For instance, for a programmer to add a new layout calculation, they would just need to inherit the operation class and fill in the virtual functions to create this new layout.

Figure 3.F shows the workflow view. The drop down menu adds operation panels. Selecting an operation displays its parameters at the bottom. Each operation has its own rules on how many and which operations it can be connected to. Clicking on another operations links them together. Operations can chain together, feeding the results of one operation as input into others. A link or operation can easily be removed with right click. The view has standard panning, zooming and moving of operations. Any changes are saved and loaded when the system is rebooted. The user can also choose to load a saved workflow or export the current workflow.

There are two unique operations, the data and output operation. The data operation is the starting point and does not take any input. The output operation grabs the preferred x and y axis of its input. When the compute button is pressed, all outputs are added to the drop-down menu for viewing. There, the user can select which results he wants to see and change x and y variables if he wants.

7 Case Studies

To evaluate our system, we provide case studies demonstrating each type of algorithm and their effectiveness. The particular data sets used

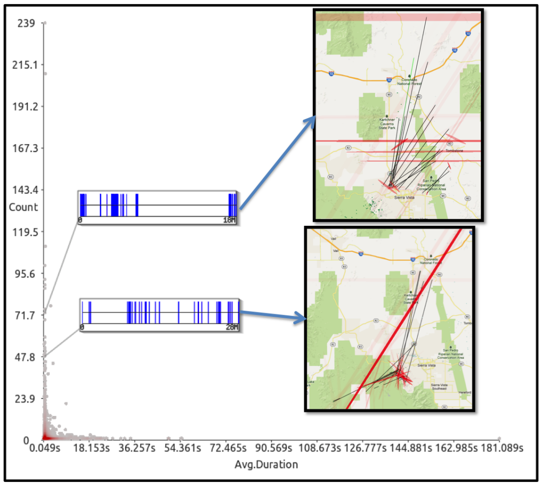

Fig. 5. The repeater candidates algorithm plots the number of signals versus their average duration for each sequence. The plot's density is mapped to color as in a heat map. For many likely candidates (those with high counts), selecting the sequence of signal pairs and plotting it in the map view shows that most of their pairs of matched signals approximately share one end point (within error ellipses) indicating likely locations of repeaters.

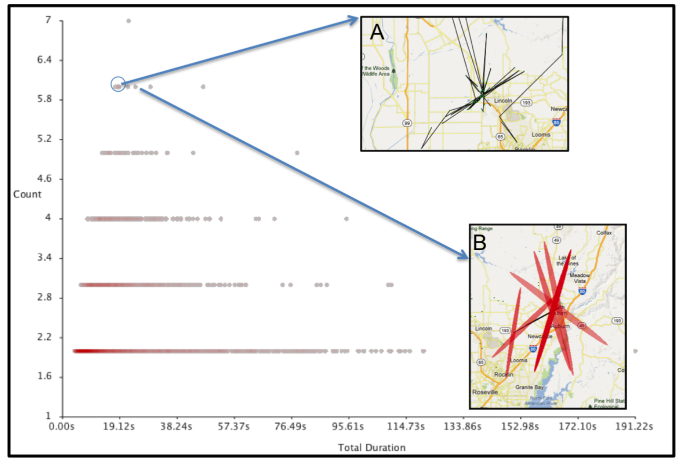

Fig. 6. The communication detection algorithm plots number of transmissions versus the overall duration of the conversation, and investigate two candidates. (a) The star burst pattern represents communications between a number of different places and one central hub, which is likely some sort of dispatch center. (b) The detected communication goes back and forth between two different locations. While the error ellipses are fairly large, their overlap refines the precision of the locations.

in this work were collected during test flights of a spectrum measurement detection platform in two geographical areas in the US. The data includes real world environmental noise and large volume of events from NSOIs that make the separation of real transmissions from noise induced transmissions difficult. Each flight generated a large log of transmission events, composed of geospatial information (including uncertainty ellipse), start/end times, signal quality, bandwidth, and Signal to Noise Ratio (SNR).

7.1 Repeaters

In this case, we use our system to explore one of the datasets collected from flight tests performed near the U.S.-Mexico border. We start by applying the repeater detection algorithm, and plotting it in the main view, as shown in Figure 5. Since we compute potential candidates, not every point is necessarily a repeater. For instance, some of the selected points do not have enough GPS data to determine whether they are repeaters or not, while others contain noise signals. For instance, there is a large concentration of noise points in the bottom left, as there is a high chance of finding incidental pairs with low count and duration. The points in the regions that extend almost asymptotically along both axes are more likely to be of interest. While this plot is rather skewed, it is these tails that are important, so it would be counter-productive to address this by applying a log-log scale.

To inspect these results in the map view, we select points from the upper left of the plot, as candidates that have more signal pairs are more likely to be repeaters. By mousing over each point, we can use the signal inset to inspect each pattern. Once a good candidate is found, it can be selected. Then, we draw lines on the map between each pair of events that have the same start time and duration. Ideally, we expect to see star patterns, where several client transmitters are utilizing one central repeater. In Figure 5, we show two different repeater candidates. The top candidate has most of the signal pairs pointing to a small region of space, while there are a few pairs that point a bit to right. When looking back at the signal inset we see a large amount of signal pairs and then a long break followed by a couple more signal pairs. This indicates that the first set of signals is the repeater on the left and the small set is either the same repeater moved or more likely a different repeater using the same set of frequency bands. In the bottom candidate, we can see constant stream of signal pairs which all points to a signal region, indicating a highly likely repeater location.

7.2 Communications

This case searched for communication patterns using the communication detection algorithm to extract likely candidates, and plotting the number of transmissions versus the total length of the conversation for each candidate. We found that communication patterns occur less frequently than any of the other patterns. While this could be due to our algorithmic settings being too strict or not inclusive enough, it is also quite possible that there were simply very few conversations occurring when the data was collected. Further verification with ground truth knowledge would be needed to confirm either way.

As before, we inspect the candidates using the geospatial view. By plotting a line between communication transmitters in series on the map, one would expect the line to simply go back and forth between the communicators as they take turns talking. In Figure 6, we see two examples of this representation. By selecting group A, a star pattern is created by one transmitter that is mostly stationary and a number of surrounding transmitters. As there were multiple, temporally distinct conversations, and as one party was stationary, it is possible that this is a 'dispatch'-type communication. Selecting group B reveals a similar nearby pattern that at first would look like the communication comes from many locations, but displaying the error ellipses reveals that it is possible for the communication to simply be between two parties. The combination of the error ellipses better triangulate their locations than any single transmission by itself.

7.3 Wavelet Algorithm

In Figure 7, we show the results from the wavelet calculation mapping color to density heat map. We have opened up several signal insets for comparison. The wavelets separate windowed samples of the signal by pattern behavior, with longer duration signals tended towards the right, analog patterns toward the bottom, and digital patterns towards the top. This comes from the wavelet scalograms, in which digital patterns have a sharp spike in one or two dimensions in a scalogram while analog patterns are more even across dimension.

Wavelets are useful when the user is looking for a general pattern about the dataset. For instance, Figure 7 show wavelet structure of two different dataset. The image on the bottom has far fewer points in the top region, indicating that it has less long digital-like patterns. A benefit of the wavelets is that the user can filter map and timeline views based on durations or digital and analog behavior.

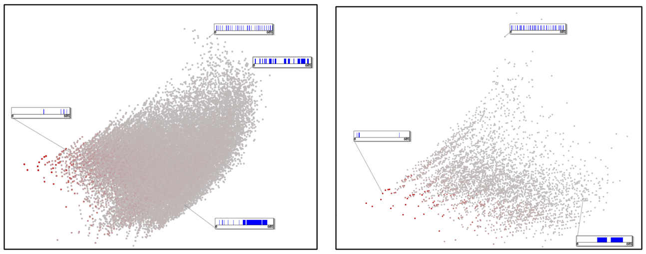

Fig. 7. The wavelet view plots sequence segments projected down from a high dimensional space. Inspection reveals that sparser patterns ended up towards the left, with more digital-like patterns at the upper right and more analog patterns in the lower right. Different datasets have different distribution of signal types. From the lack of points in mid upper region of the bottom image, it is clear that the top dataset has more digital signals than the bottom dataset.

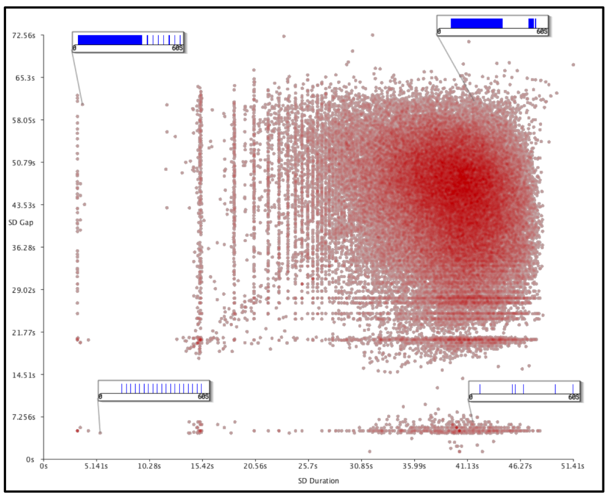

Fig. 8. The Windowed Variance distributes patterns based on their variance in gap and duration. The y-axis is the variance in the gap while the x-axis is the variance in duration. Analog patterns are found in the top right because they have large variance in both gap and duration.

7.4 Repetitive Patterns

We examine both the DPD algorithm, Windowed Variance and their views. We first start with the Windowed Variance. We set the window size to 60 seconds with no overlap between windows. Figure 8 shows the results of the Windowed Variance computation. The big cluster of patterns to the top right are analog signal patterns as they have large variation in both gap and duration. However, the striated groups to the left and bottom of the plot are generally digital signals, with low variation in signal length, gaps, or both.

The Windowed Variance view is useful when the user is looking for a pattern that is not entirely digital. For instance, there might be a series of events that have the same duration but different gap times. By plotting the SD of the gap and duration, the user can quickly decide how much variance to allow in either direction through selection. This gives Windowed Variance an advantage over DPD where it has to be recomputed. Another potential use for the Windowed Variance is to look at analog signal patterns. Just like in the digital selection, the user can decide how much variation to allow.

We computed the DPD algorithm with a 10% CV. The results are shown in Figure 9. Most digital signals would have high signal counts of low duration. So one unexpected feature of this dataset is that there are several peaks of high signal counts with moderate to high average duration. What is also interesting is that these high average high counts have low frequency bands, as normally digital signals are found in the higher frequency bands. We look at two low frequency bands points and notice that we can triangulate their position. With this piece of information, it would be possible to commence another fly-by to gather more information if warranted.

7.5 Combining Approaches

Our discussion so far has focused on each algorithm separately, and how each one can help the user find a particular pattern or generally explore a dataset. We also have shown how the time view and map view can be used in conjunction with the results to facilitate in the discovery of patterns. However, repeaters, communication and DPD/Windowed Variance can also be combined to find more complex patterns.

For instance, the user could combine the DPD and repeater algorithms. This would find digital patterns mirrored across multiple frequencies. While this could be a digital signal being retransmitted, it could also be a broadcast from a single source of a digital pattern across multiple frequencies, such as an alert or beacon. First, we would construct our workflow that would go from the data, into the digital algorithm and pipe those results into the repeater algorithm. We filter the map based on candidates we found, as shown in Figure 10. Mousing over the points, we notice candidates that have many error ellipses overlapping, indicating very likely candidates as all the signals are coming from the same location, as would indicate a beacon instead of a digital repeater. Zooming in on the candidates reveals that the signals originate from the McClellan airfield. Thus, it is reasonable to deduce that this is an air-control navigational beacon.

Each combination of algorithms would detect different patterns. A repeater communication would use a repeater to extend the range of a conversation. A common example situation is near mountains where the handheld devices would be communicating over a repeater at the top of the mountain. Digital communication could be two or more machines interfacing with each other. A combination of all three is possible as well: Two machines could be communicating over a repeater to extend the range or broadcast to other frequency bands.

8 Expert User Collaboration

This project was aimed at the development of a visual tool for expert analysts to use to derive actionable insights from large data sets of simple raw measurements, with the goals of providing them with highly efficient, interactive analytic methods and an interface that would be intuitive enough for them to learn. As such, throughout the development we worked very closely with expert users in the signal intelligence field, who have extensive experience in working with the airborne signal collection platform and interpreting its results. This interaction directly guided the development of the analytics which directly addressed their questions. And our joint exploration of the data revealed unanticipated applications to detection of hardware issues such as installation or measurement errors, which they used to further the development of the collection platform.

Due to the specific expertise of the target user base, it was unfeasible to find a sufficient number of users for a formal user study. Also, many potential applications for our system involve sensitive or classified data, so critical evaluation of the analysts' work flows or tool usage is also not viable. However, through regular user tests and feedback from the few expert users we were working with, we were able to informally evaluate the utility and intuitiveness of the tool. Here, we describe examples of their usage of most major components in the system, many of which led to concrete insights.

The analysts found the tools easy to use. In general, the frequency view was a clear base for the analyst to work from, as the users found it easy to understand. In particular, the time vs frequency view was often the initial view of choice of the signal environment. Previously, the analysts interacted with the data one frequency at a time. The timeline representation made understanding patterns out of a dense environment much easier while retaining the users' mental picture of the data space. The color editor allowed subtle shading within a data type and helped to highlight or exclude outliers visually. Features of the color wheel such as saving of the color wheel settings were found helpful for repeated analysis over several sessions or for collaboration.

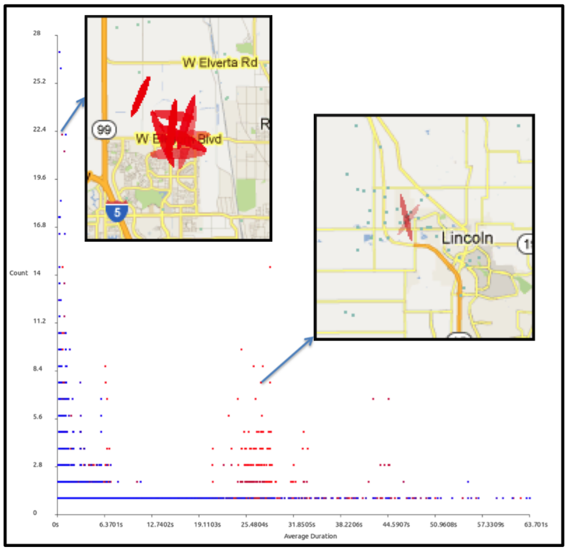

Fig. 9. The DPD view shows the count in the y-axis and average duration in the x-axis. Normally large counts of digital patterns have short average duration and are found in the higher frequency (blue). In this dataset, there are several high count patterns that have both high average duration and are found lower frequency bands (red).

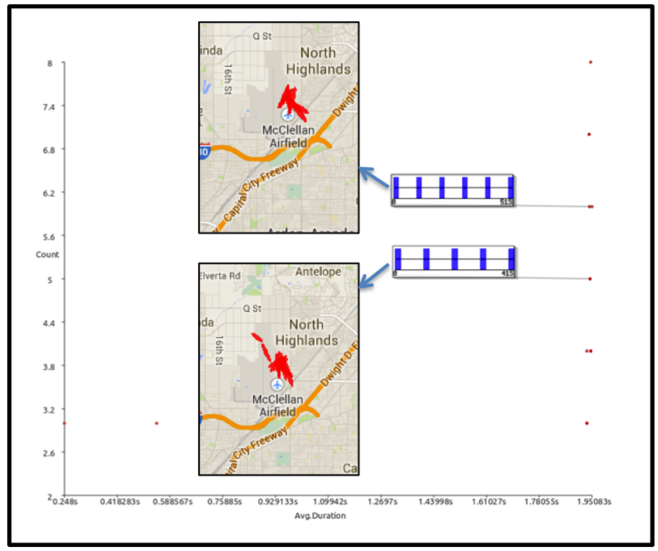

Fig. 10. An example of combining algorithmic methods. In this case, we ask the system to find us digital repeaters, a digital signal that spans multiple frequencies. The system presents several candidates. Examining the results, we find two promising clusters. Most of the signals originate from the McClellan airfield.

The addition of highlighting and filtering greatly improved the expert users exploration of the data set over their previous methods. Rapid selection of different data types and zooming into detail features for more in-depth analysis allowed the analysts to rapidly explore the features of the environment from the macro viewpoint down to detailed perspectives of related events.

Some of the data analytics (such as communication) often only identified a few events. The ability to see those events within the context of the entire frequency set allowed the users insight into whether the algorithm had selected a true communication channel or not. Because the number of such events may only be a few dozen out of the 500K to 1M events in the data, finding the related events needed to be efficient. Highlighting was sufficient for analytics with about 100 events. Filtering was important with fewer events. A common process was that the fast filtering would be used to remove all nonselected data and then zooming would be used to get closer to the selected data. Restoring all the events was a single button press which would put all the data back in the display. This sequence was intuitive to the users and helped in the rapid validation of low event analytic output.

The repeater analytic produced interesting results that still needs more analysis. The algorithm required absolute time synchronicity of the events that were put forward as candidates. On zooming into the details of the events, the expert users rapidly hypothesized new analytic algorithms for refined or different results.

Novel axis definitions in the digital, communication and repeater analysis helped translate the algorithms results into operator-centric understandings. The analytics produced many candidates and the novel axis definitions helped the experts in navigating those candidates, since they related to the experts' understood signal concepts. For example, one axis of the digital detection metric is the signals' mean duration, which lets users utilize their domain knowledge to separate digital signals into different duration subclasses. For the communication analytic the number of events in the communication was used as an axis, which allowed the users to easily distinguish by numbers of interchanges. These flexibilities made exploration of highly likely signals easier for the expert users to select, highlight, filter and map.

While not one of the original goals, the expert users also found the tool helpful for identification of subtle installation and measurement errors at a macro level. Our tool revealed indicators at the overview level which guided the users to drill down into specific times and measurements from a multi-variable perspective using colorization by data type and filtering and selection across the views. Although the data analysis was done on datasets after the flight tests were over, this analysis was able to point to subtle installation issues that were specific to particular frequencies and aircraft flight dynamics. These phenomena were not visible with previous data centric analysis methods as these subtle errors had not been previous hypothesized or analyzed.

9 Conclusion

In this paper we have presented a visual analysis system for exploring and analyzing radio signal meta-data. We showed how the visual representations help analysts identify, understand, and compare different signal patterns. In designing our system, we have developed novel algorithms for quantitative detection of patterns such as digital transmission or communications. We have shown how combinations of such algorithms along with visual interactions can lead to further insights. We have also demonstrated the usefulness of our system, particularly the algorithms and visualizations, with experimental results derived from data collected from actual airborne sensor platform tests. Such a system could only be more powerful when put into the hands of analysts who need to understand intercepted radio transmission patterns in the field, either forensically or in real-time.

Though the work in this paper has been focusing on exploring offline data, in the future, our system will move towards integration with real-time data collection. Our current analytics aim to be efficient enough for real-time usage, but future situations or additional analytics might necessitate further scaling and acceleration techniques, such as GPU techniques or out-of-core processing capabilities. Incorporating relevant real-time properties such as the location/orientation of the plane could also be relevant in real-time situations. Lastly, the contents of recorded signal data is generally sensitive information, so we only had access to the signal metadata. However, when the signal contents are available, their analysis would be a beneficial subsequent step; our approach can be used to identify sequences of signals that could form a communication, at which point the analyst would apply other communication analysis approaches based on content.

10 Acknowledgments

This research has been sponsored in part by the Northrop Grumman Corporation, National Science Foundation through grants IIS-1320229 and CCF-1025269, and Department of Energy through grants DE-FC02-06ER25777 and DE-FC02-12ER26072.

References

[1] N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[2] N. V. Andrienko and G. L. Andrienko. Visual analytics of movement: An overview of methods, tools and procedures. *Information Visualization*, 12(1):3–24, 2013.

[3] P. Barford, J. Kline, D. Plonka, and A. Ron. A signal analysis of network traffic anomalies. In *Proceedings of ACM SIGCOMM Workshop*, IMW '02, pages 71–82. ACM, 2002.

[4] T. Bergstrom and K. Karahalios. Conversation clock: Visualizing audio patterns in co-located groups. In *Proceedings of HICSS 2007*, page 78, 2007.

[5] T. Bergstrom and K. Karahalios. Conversation clusters: grouping conversation topics through human-computer dialog. In *Proceedings of CHI '09*, the SIGCHI Conference on Human Factors in Computing Systems, pages 2349–2352, New York, NY, USA, 2009. ACM.

[6] T. Crnovrsanin, C. Muelder, C. D. Correa, and K.-L. Ma. Proximity-based visualization of movement trace data. In *IEEE VAST*, pages 11–18, 2009.

[7] J. Donath, K. Karahalios, and F. Vigas. Visualizing conversation. *Journal of Computer-Mediated Communication*, 4(4):0–0, 1999.

[8] J. Edlund, M. Heldner, and J. Hirschberg. *Pause and gap length in face-to-face interaction*, pages 2779–2782. Speech Communication Association, 2009.

[9] J. Faith and S. Rajbhandari. The use of linear projections in the visual analysis of signals in an indoor optical wireless link. In *Proceedings of CSNDSP*, pages 576–581. IEEE, 2010.

[10] A. Graps. An introduction to wavelets. *IEEE Computational Science and Engineering*, 2:50–61, 1995.

[11] Y. Han, E. P. Stuntebeck, J. T. Stasko, and G. D. Abowd. A visual analytics system for radio frequency fingerprinting-based localization. In *IEEE VAST*, pages 35–42. IEEE, 2009.

[12] M. Heldner. Detection thresholds for gaps, overlaps, and no-gap-no-overlaps. *Journal of Acoustical Society of America*, 130(1):508–513, 2011.

[13] M. Heldner and J. Edlund. Pauses, gaps and overlaps in conversations. *Journal of Phonetics*, 38(4):555–568, 2010.

[14] J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *Proceedings of InfoVis '05*, pages 125–132. IEEE Computer Society, 2005.

[15] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, Oct. 2002.

[16] A. G. Landge, J. A. Levine, A. Bhatele, K. E. Isaacs, T. Gamblin, M. Schulz, S. H. Langer, P.-T. Bremer, and V. Pascucci. Visualizing network traffic to understand the performance of massively parallel simulations. *IEEE TVCG*, 18:2467–2476, 2012.

[17] A. M. MacEachren and M.-J. Kraak. Exploratory cartographic visualization: Advancing the agenda. *Comput. Geosci.*, 23(4):335–343, May 1997.

[18] A. M. MacEachren, A. Robinson, S. Hopper, S. Gardner, R. Murray, M. Gahegan, and E. Hetzler. Visualizing geospatial information uncertainty: What we know and what we need to know. *Cartography and Geographic Information Science*, 32(3):139–160, July 2005.

[19] N. Miller, P. C. Wong, M. Brewster, and H. Foote. Topic islandstm-a wavelet-based text visualization system. In *Proceedings of Vis '98*, pages 189–196, 1998.

[20] C. W. Muelder, K.-L. Ma, and T. Bartoletti. A visualization methodology for characterization of network scans. In *Proceedings of VizSEC '05*, pages 29–38, October 2005.

[21] W. E. Nagel, A. Arnold, M. Weber, H.-C. Hoppe, and K. Solchenbach. Vampir: Visualization and analysis of MPI resources. *Supercomputer*, 12:69–80, 1996.

[22] O. Rioul and M. Vetterli. Wavelets and signal processing. *IEEE Signal Processing Magazine*, 8(4):14–38, 1991.

[23] S. S. Shende and A. D. Malony. The Tau parallel performance system. *IJHPCA*, 20:287–311, May 2006.

[24] W. Spear, A. D. Malony, C. W. Lee, S. Biersdorff, and S. Shende. An approach to creating performance visualizations in a parallel profile analysis tool. In *Proceedings of Euro-Par '11*, the 2011 international conference on Parallel Processing - Volume 2, pages 156–165, 2012.

[25] J. Thomson, E. Hetzler, A. MacEachren, M. Gahegan, and M. Pavel. A typology for visualizing uncertainty. volume 5669, pages 146–157, 2005.

[26] J. M. Wilson. Gantt charts: A centenary appreciation. *European Journal of Operational Research*, 149(2):430–437, 2003.

[27] J. Wood, J. Dykes, and A. Slingsby. Visualisation of origins, destinations and flows with od maps. *Cartographic Journal, The*, 47(2):117 – 129, 2010.

[28] X. Xu, L. Zhang, and Q. Wan. A variation coefficient similarity measure and its application in emergency group decision-making. *Systems Engineering Procedia*, 5(0):119 – 124, 2012.

[29] O. Zaki, E. Lusk, W. Gropp, and D. Swider. Toward scalable performance visualization with Jumpshot. *IJHPCA*, 13:277–288, Aug. 1999.