A Web-Based Tool for Visualization and Collaborative Annotation of Physiological Databases

MB Oefinger, RG Mark

Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

In this paper we present a novel web-based service, available through Physionet (www.physionet.org), that utilizes a Java applet front-end for visualization and integrates seamlessly with WFDB-based back-end CGI scripts to facilitate simple point-and-click user interaction and data annotation. The service allows users to peruse visually all ECG databases available on Physionet through a web browser. The web tool displays waveforms and accompanying derived time series and histograms for information such as heart rate, QT-interval and ST-segment deviation. By virtue of its web interface and centralized storage of annotations, this service for database visualization and annotation promotes worldwide collaboration for the development of more robust databases in far less time than with conventional, manual practices.

1. Introduction

Physiological database annotation is a notoriously difficult task, both because it is manually tedious and because collaboration among disparate research groups imposes difficulties in data-sharing. The process of annotating databases has historically involved a panel of experts each reviewing a separate copy of a database, resorting to consensus opinion to comprise the final gold standard. Database annotation remains a mandatory part of creating gold standard databases such as those by which automated analysis algorithms are measured objectively for research purposes or FDA approval.

With the ubiquity of the Internet has come an obvious framework for centralized annotation storage and remote collaboration. However, only very recently have web-based software tools provided sufficient power and speed to support a fully interactive, web-based application-like software architecture such as that required by annotation committees.

Using Java and CGI, we developed a highly interactive web service that provides an easy-to-use yet very powerful service for researchers to visualize and annotate data. In this service model all records are pre-processed to provide annotators insight into probable areas of interest. For example, one panel of a record shows the time-series tachogram for the duration of ECG recording. By clicking on a point of interest in the tachogram plot, the underlying waveform corresponding to that point in time is presented in another panel. Another panel shows a histogram of instantaneous R-R intervals for all beats in the record. By selecting a given region of the histogram, the user may scan through each underlying ECG beat within the chosen histogram region. Similar derived parameters allow insight into ST-segment, QT-interval, arrhythmia, and other parameters pre-processed by the system.

Figure 1: screenshot of the web-based ECG visualization and annotation tool. The above sample shows data that has been pre-processed to create a histogram of R-R intervals (upper-right panel) and a tachogram (middle-left panel). Clicking on a point on the tachogram reveals the corresponding ECG waveform (lower panel) for the same point in time. A scope view (middle-right panel) with adjustable depth shows the previous n R-centered beats (where n is selectable) superimposed.

2. Methods

The system of interest in this paper, called Hermes (TM), was first designed as a web-based system for visualization of ECG data using SVG, an XML-based graphics language [1]. Limitations inherent in XML encoding and transmission of large datasets, along with the slow response time to point-and-click activity in SVG-rendered plots, motivated use of a more interactive platform to provide the strength of an application with the ease-of-use and ease-of-deployment of web content. For GUI design we utilized Java's Swing components, which provide an event-driven model permitting easy association of mouse and keyboard events with so-called callback routines. By utilizing Java's robust network socket modules and associating CGI (Common Gateway Interface) links with callback routines, we incorporated dynamic client/server rendering functionality and point-and-click functionality.

Dynamic rendering of data is the cornerstone of an optimal visualization tool for large data sets, as storing each segment of data as a static JPEG or GIF file would consume more disk space than the waveform it represents. As a solution one may, alternatively, consider utilization of a server-side rendering cache that creates a temporary JPEG or GIF image of the requested data for temporary access on the server, in turn providing the client with a link or embedded image to view the data. Such an approach is utilized in Physionet's chart-o-matic [2], for example. However, the absence of inherent mouse or keyboard tracking in returned plots means that one may not annotate resultant plots online.

To address the need for the server to render data dynamically *and* with the possibility of online annotation, we employ CGI to request the appropriate raw data from the server, which is in turn rendered on the client by the Java applet. The applet, designed with the ability to map pixel location to the corresponding absolute location in the data record, processes client-side requests for adding/deleting server-side annotations by sending a CGI request to the server to add or delete the appropriate annotation. Similar buttons with CGI links to the server permit the user to browse forward and backward through a record, or even view the record in an adjustable-speed streaming mode as though viewing an ECG stream at real-time speed (or in slow motion or high speed). Functionality for browsing is enabled by tying button clicks to CGI scripts that tell the server to provide the client with a new time-slice buffer for rendering. Data streaming is enabled by an adjustable timer interrupt (adjusted by the user with an intuitive GUI slider bar) in the GUI that, with each internal interrupt, retrieves data via CGI from the server and subsequently renders the next time-slice of data.

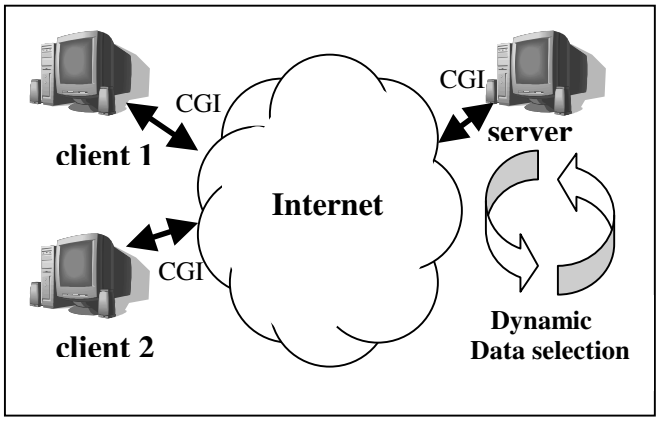

Figure 2: A diagram of dynamic rendering. Each client makes a request to the server via CGI. The server returns the appropriate section (time-slice) of raw data to the client, which renders the plot in its viewing area.

An additional so-called "scope panel," providing overlaid, R-centered beats, permits the user to accelerate the process of searching for outliers (such as PVCs). Such a feature is present in many modern Holter software packages. Using a configurable depth for *n*, the user will see the previous R-centered *n* beats in the scope panel. Viewing the waveform in streaming mode, the user may quickly spot a PVC as a beat with morphology different from the otherwise very similar overlaid normal beats. By choosing *n* to be large and the streaming speed high, the user can search for outliers with tremendous speed and efficiency. A high value of *n* sets the scope panel with sufficient depth such that the PVC still shows in the scope panel even as the next *n-1* beats are quickly overlaid.

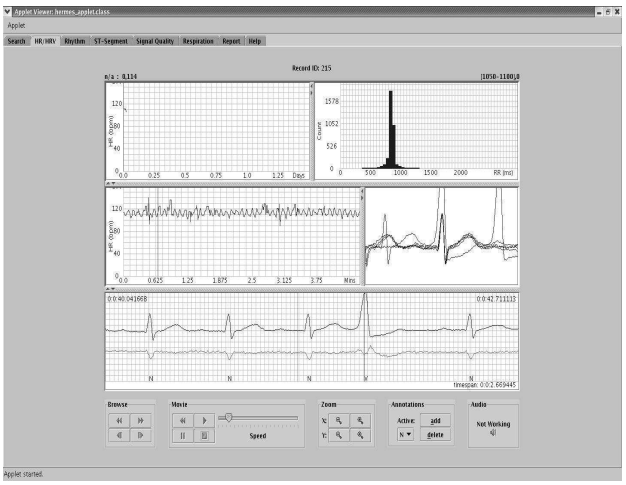

Figure 3: In the scope panel (middle-right), a PVC stands out from other previous normal overlaid beats (which are so similar that they appear as a single thick line).

3. Results

The utility of a web-based (or client/server) interactive data visualization and annotation tool can best be assessed by recognizing the limitations in pure client-side applications designed for the same purpose.

Installation of client-side applications requires a modicum of computer skills and, depending on the user, a fair amount of time. Upon installation the user needs to maintain his or her machine with appropriate upgrades or patches as they become available. A web-based service, in contrast, allows any user with a web browser to utilize the software immediately. The web service model permits changes (i.e. software upgrades or bug fixes) on the server to propagate immediately to all subscribers of the service with no subscriber intervention.

Another barrier imposed by a pure client-side software architecture is the difficulty of coherent data-sharing and collaboration. A web service model permits centralized storage of annotations so that users in disparate locations can simultaneously view the same data and form a consensus opinion. Centralized data storage also removes the need for each user to maintain his or her own backups; the server on which the server is run has inherent data redundancy through RAID and is backed up to mirror sites automatically for additional data redundancy.

Of additional merit in the tool's use in expediting data inspection are the pre-processed heart rate and other ECG-derived results providing time-series and histogram links to the underlying ECG waveform. With such a feature users may quickly identify areas of interest for further inspection and annotation in a manner similar to that permitted by modern Holter software. Additionally the user may then browse data in this region or view a stream of data, as though sitting in front of a real-time monitor, to gain further insight.

4. Discussion and conclusions

The tool described in this paper is continually under development to add further functionality of interest to users. Under current development is a feature to allow visualization of algorithm results as compared with any of the gold-standard annotated Physionet databases (eg MIT-BIH arrhythmia database). By uploading an annotation file to the server through a simple web interface, algorithm developers will be able to run a comparison between their annotations and the gold standard annotations. The server will provide sensitivity and positive predictivity performance measures along with a time-series plot of false-positives and false-negatives linked to the underlying waveform. Such a methodology will permit algorithm developers to identify quickly the areas in which algorithms are over- or under-sensitive.

Also under development is an audio feature that will complement the scope view to give users who are quickly scanning data a simple mechanism for detecting outliers. By emitting a tone with each beat that has a pitch proportional to the instantaneous R-R interval, the user will quickly hear a premature contraction or missed beat. The paradigm may be extended easily to other signals of interest. For example, a tone with pitch proportional to a derived ST-segment level would allow a user to scan through data at an extremely fast pace and hear the onset of ischemia or MI (provided that such events have corresponding ST-segment shifts in the record of interest).

Though the tool will continue to grow in its capabilities, the fundamental goal that drove its invention will remain in tact: that of providing the user with a simple means of visualizing and analyzing physiological data. The features that will be added should not come at the expense of perceived added complexity from the perspective of the end user. Rather, they should enhance the user's ability to understand or isolate interesting data with no additional effort.

The authors invite you to visit Physionet [2] and follow links for "Hermes" to utilize the tool described in this paper. We welcome your feedback on how the service works for you and how we might improve it in the future.

Acknowledgements

This research was supported by NIH grant PO1HL66105. The authors would additionally like to thank George Moody, author of WAVE, and Franc Jager, author of SEMIA [3]. Their tools provide a substantial foundation of visualization and annotation methodology that has been emulated in the web service discussed in this paper.

References

[1] Oefinger, M et al. *An Interactive Web-based Tool for Multi-Scale Physiological Data Visualization.* Computers in Cadiology 2004; 569-571

[2] http://www.physionet.org

[3] Dorn, R, Jager, F. *Semia: semi-automatic interactive graphic editing tool to annotate ambulatory ecg records*. Computer Methods and Programs in Biomedicine 2004; 235-249

Address for correspondence:

Matt Oefinger
Massachusetts Institute of Technology
77 Massachusetts Avenue, E25-505
Cambridge, MA 02139
oefinger@mit.edu