Distributed Logging and Synchronization of Physiological and Performance Measures to Support Adaptive Automation Strategies

Daniel Barber[1] and Irwin Hudson[2]

[1] University of Central Florida Institute for Simulation and Training Applied Cognition and Training in Immersive Virtual Environments Laboratory,
3100 Technology Parkway, Orlando, FL 32826
[2] U.S. Army Research Laboratory, SFC Paul Ray Smith Simulation & Training Technology Center (STTC), Orlando, FL
dbarber@ist.ucf.edu

Abstract. As advances in physiological sensors make them more minimally intrusive and easier to use, there is a clear desire by researchers in the fields of Augmented Cognition and Neuroergonomics to incorporate them as much as possible. To best support use of multiple measures, the data from each sensor must be accurately synchronized across all devices and tied to performance and environment events. However, each sensor provides different sampling frequencies, local timing information, and timing accuracy making data synchronization in logs or real time systems difficult. In this paper, a modular architecture is presented to address the issue of how to synchronize data to support analysis of physiological and performance measures. Specific design requirements are presented to ensure the ability to accurately measure raw sensor data and compute metrics in a distributed computing environment to support adaptive automation strategies in a research environment. Finally, an example system is described which combines multiple minimally invasive physiological sensors.

Keywords: Adaptive Automation, Closed-Loop Training System, Data Synchronization.

1 Introduction

Current advances and capabilities of physiological sensing technologies have made them minimally intrusive and portable enough for use in research efforts for subjective measurement of a user's state [1]. When developing a closed-loop adaptive system or one in which the human acts as a sensor, the ability to measure the workload and potential cognitive state accurately using multiple sensors and performance measures is critical [2]. However, one of the major issues with these devices is that each provides different sampling frequencies, timing resolution, and local time stamps making it difficult to synchronize data across disparate devices and other measurements (e.g. task performance). These problems become even more

D.D. Schmorrow and C.M. Fidopiastis (Eds.): FAC 2011, HCII 2011, LNAI 6780, pp. 559–566, 2011.
© Springer-Verlag Berlin Heidelberg 2011

pronounced when working within a distributed computing environment where multiple machines are used for different devices, tasks, and team members as shown in Fig. 1.

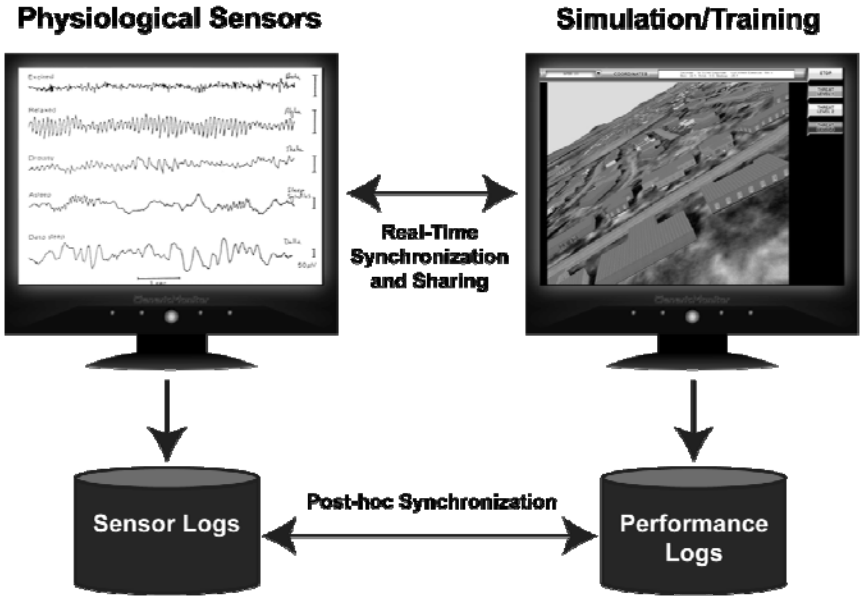

Fig. 1. Example distributed simulation and training environment using physiological sensors

As shown in the above figure, collected data can be shared and synchronized post-hoc, but depending on the timing information recorded this can be a tedious process, and usually performed manually by the researcher. If an appropriate minimum set of timing data, as described in Section 0, is collected it is possible to automate the data synchronization process and provide real time sharing of metrics to support closed-loop training and feedback as described in [1][2][3]. Moreover, the inclusion of unique identifiers for a participant and their group/team makes it possible to design larger systems that can adapt based on the physiological and thus cognitive state of a team. For example, a team of four tasked with supervising multiple unmanned ground systems in a reconnaissance and surveillance mission. During this mission each team member is required to respond to audio and text communications, monitor video feeds, and re-route vehicles based on changes in environment. Using physiological measures from an eye tracker (e.g. eye blink rate (EBR), pupil dilation (PD)) and electroencephalography (EEG), the system may identify that a member of the team is under a high workload condition and based on past measures is likely to enter a period of decreased performance. In this event, the system can decide to re-assign an unmanned system to another member of the team with the lowest workload level, ensuring overall team workload and performance stays within optimal levels.

In the following sections, this paper describes an architecture framework for building systems capable of synchronizing physiological and other measures for post-hoc and real-time analysis of an individual or team state. The required minimum set of data for synchronization of disparate data is described in addition to the design of a system developed at the University of Central Florida's Institute for Simulation and Trainings ACTIVE Laboratory to support real-time adaptation in a closed-loop experimental environment.

2 Synchronization and Correlation of Data

As mentioned previously, sensors and other system modules provide time information for data in different formats, typically relative to when the device or data collection was activated. Using this information alone makes it impossible to match up the times of different data types on a single machine and across a network. Therefore, a time stamp that is from a global reference frame like Coordinated Universal Time (UTC) is required. UTC is a time standard based on International Atomic Time and includes the following information: Day, Hour, Minute, Second, Milliseconds. UTC is also used as the primary method for system time on most modern operating systems making it an ideal choice as a global time reference for tagging data. By recording the current UTC time stamp in a log or in software, when data collection begins for any sensor, it is possible to convert the local sensor time stamps to UTC time for data synchronization.

In addition to recording UTC time stamps for physiological sensors, simulation and operational environments must also be designed to use UTC timing in recorded data to correlate user state with environmental events. An example of a simulation environment that does this is the Mixed Initiative Experimental (MIX) Testbed [4]. The MIX Testbed is a simulation environment developed for research in supervisory control of unmanned systems. It provides log files necessary to support the correlation of physiological data to simulation events [3]. One such log file is called the "Simulation Time Log." This log provides UTC, Local Time (e.g. Eastern Time), and Simulation Time as events occur within the simulation (e.g. Simulation Start, Pause, Resume, Stop). Using this log file it is a simple to extract physiological sensor data from log files that correspond to when a user was performing a mission within the simulation environment post-hoc.

With the use of UTC times associated to data, the next step is to tag sensor and performance records with appropriate identifiers (e.g. User ID, Group ID). With unique identifiers an operational environment or log analyzer can pull sensor data for specific users and teams for analysis or triggering changes in the environment (e.g. re-assign tasking, prompt user). The minimum amount of data that must be associated with recorded data is presented in Table 1. Building off of this minimum set it is possible to create a datagram (i.e. byte array structure) format for transmission of physiological and environment data between computers in a distributed network. Also, with the inclusion of a time stamp representing when the data point was produced, any latency effects in sharing the data across a network will be minimal on filters which consume data to produce new metrics of user state.

Table 1. Record Data Requirements

Field	Name	Description/Interpretation
1	Time (UTC)	UTC Time when this data point was generated or captured by source.
2	Participant/User Number	Unique ID for the participant or user the data point is associated with
3	Group/Team Number	Unique ID for the group or team the participant or user the data point is associated with

3 Operational Psychophysiological Sensor Suite (OPSS)

The design of metrics to determine the real-time changes in cognitive state and workload of a human performing a task using psychophysiological measures is an ongoing field of research [1][5]. The Operational Psychophysiological Sensor Suite (OPSS) is a suite of state-of-the art wearable psychophysiological sensing devices and software tools assembled by the ACTIVE Laboratory at UCF-IST for measuring participants physiological response to cognitive state in real-time. This suite includes devices for eye tracking, Electro Dermal Response (EDR), Heart Rate Variability (HRV), and EEG measurement. To support the synchronization and correlation of data generated by these devices, a software architecture framework has been developed which expands upon the data requirements described in section 2 to support enhanced data logging, filtering, and real-time sharing over a network connection.

As shown in Fig. 2, this architecture design introduces the concept of a Sensor and Filter which both have similar capabilities that include data logging and network communication. A Sensor is an interface to an actual physical device attached to a computer (e.g. Eye Tracker). A Filter is a software module which can be "connected" to a single or multiple sensors and other Filters, and is designed to produce new metrics or cleanup of raw data. Connections between a Sensor and Filters can be made within the software application or through subscriptions over a network connection (e.g. UDP/IP, TCP/IP) producing a "Filter Graph." Although possible to transmit raw data from sensors, the Filter Graph concept provides the ability to reduce network bandwidth usage by calculating and transmitting only the metrics required by a 3rd party data subscriber. Data logging is an option that can be enabled or disabled for any Sensor/Filter and is used for record keeping or post-hoc analysis through batch processing.

An example of a Filter Graph based on the OPSS System Architecture for an Eye Tracker is described in Fig 3. In this example, there are two filters in the graph: Eye Tracker Filter and Areas of Interest (AOI) Filter. The Eye Tracker Filter takes the raw eye gaze data it receives the Eye Tracker Sensors and produces classifications such as fixations and workload (e.g. Nearest-Neighbor Index) [6]. The downstream AOI Filter uses this data to identify what areas of the screen a user is focusing on for a given

point in time. Based on the metrics of the AOI Filter, a program can trigger system events to notify the user that they need to pay more attention to a specific area of the screen.

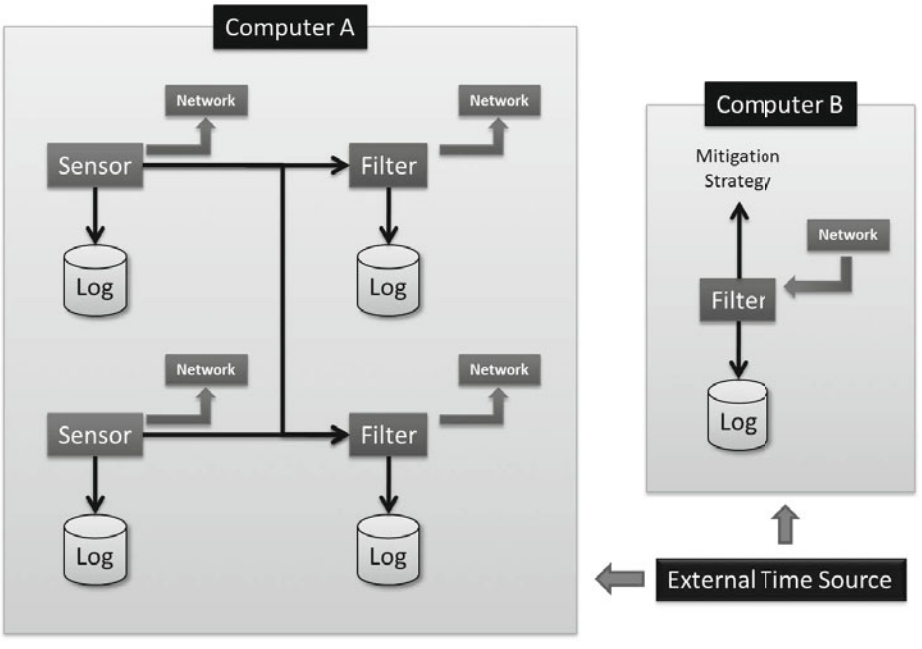

Fig. 2. System Architecture

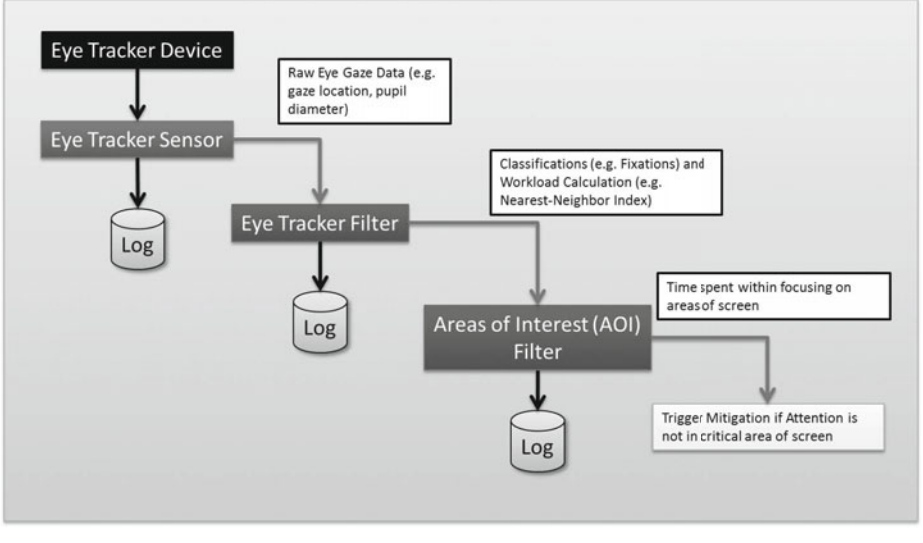

Fig. 3. Example Filter Graph for Eye Tracking Data

Another aspect of physiological sensors is that many provide large amounts of data at high sampling frequencies with timing resolution of 1ms. However, many of the devices provided run only on Windows operating systems, which provide UTC time resolution around 10ms [7]. To work around this issue, an external time source can be integrated into the system as shown in Fig. 2. The OPSS provides software tools to create external time sources which can be connected over a network connection to provide 1ms timing resolution in data collected.

Finally, for network sharing of data, the dataset from Table 1. is expanded upon (Table 2) to provide additional information necessary for distinguishing between types of data being transmitted so that subscribers can select only the data they require for processing. Within the OPSS, any record that a Sensor or Filter produces that can be logged can be modified to support conversion to the datagram format in Table 2 for transmission to subscribers.

Table 2. Message format for network streaming of data

Field	Name	Description/Interpretation
1	Message Start	4 Byte value containing unique identifiers for parsing message data and bit flags for priority and transmission confirmation
2	Time (UTC)	4 Byte UTC Time when this data point was generated or captured by source.
3	Participant/User Number	1 Byte Unique ID for the participant or user the data point is associated with
4	Group/Team Number	1 Byte Unique ID for the group or team the participant or user the data point is associated with
5	Sequence Number	2 Byte values representing the sequence number for a multi-packet sequence of data. First value always starts at 0 and increases for each packet in multi-packet stream. For standalone packets value is always 1.
6	Message Type	2 Byte Unique ID for the type of message data included in payload
7	Message Payload Size	2 Byte integer representing size of data in bytes

Table 2. (*continued*)

Message Start Format		
Bits	**Name**	**Description**
0-7	Start Byte	Byte representing the start of a message. Value is always equal to 0x02
8-15	Source Node ID	Unique Identifier representing the node source of the message. A node is defined as a PC or device with physical interface. Valid values range from [0,255]
16-23	Source Component ID	Unique Identifier representing the source application on the node. This is used to distinguish between multiple applications on the same physical source. Valid values range from [0,255]
24	Multi-Sequence Message packet	If value is 1, than this packet is part of a multi-packet sequence which is needed for transmitting messages greater than maximum bytes per packet of transport medium being used. Use the Sequence Number field for ordering of data packets. If 0, then the pack is stand alone and contains the full message.
25-32	Reserved	Reserved bits for future expansion

4 Conclusion

The synchronization process described in this paper provides a solution to the issues related to using multiple disparate physiological sensors. Specifically, how timing information is represented using these devices and methods for synchronizing the information so it can be correlated to external system events was discussed. An architecture framework for defining how sensors can be paired together and combined to produce new metrics for use in measurement of cognitive state and operator workload was described. The architecture implemented by the OPSS defines modules (e.g. Sensor, Filter) that make up a Filter Graph. If system developers choose to follow this framework in describing their own designs, facilitation of software and hardware re-use is achievable between researchers. Finally, a message structure format is proposed for the transmission of physiological data over network connections. It is the authors belief that the development of standards for sharing of

physiological data between different hardware/software implementations, similar to the Joint Architecture for Unmanned Systems (JAUS) [8], would be advantageous to the larger research community. Through the use of standards, and a desire by customers of sensors for their use, the cost of adding new sensors to adaptive systems will decrease and research results more easily shared.

Acknowledgements. This work was supported by the US Army RDECOM (W91CRB08D0015). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of RDECOM or the US Government. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Vartak, A.: Cognative State Estimation for Adaptive Learning Systems Using Wearable Physiological Sensors. In: 1st International Conference on Biomedical Electronics and Devices (2008)
2. Sciarini, L.W.: Assessing Cognitive State with Multiple Phyiological Measures: A Modular Approach. In: Human-Computer Interaction International, pp. 533–542 (2009)
3. Fidopiastis, C.M.: Impact of Automation and Task Load on Unmanned System Operator's Eye Movement Patterns. In: Human-Computer Interaction Interntional, pp. 229–238 (2009)
4. Barber, D.: The Mixed Initiative Experimental (MIX) Testbed for Human Robot Interactions with Varied Levels of Automation. In: 26th Army Science Conference, Olrando (2008)
5. Nicholson, D.: An Adaptive System for Improving and Augmenting Human Performance. In: Foundations of Augmented Cognition, pp. 215–222. Strategic Analysis, Inc., Arlington (2006)
6. Camilli, M.: ASTEF: A simple tool for examining fixations. Behavior Research Methods, 373–382 (2008)
7. Nilsson, J.: Implementing a Continuously Updating High-Resolution Time Provider for Windows. In: Microsoft, `http://msdn.microsoft.com/en-us/magazine/cc163996.aspx` (accessed March 2004)
8. Committee, A.-4.: JAUS Core Service Set. In: Society for Automotive Enginers. HYPERLINK, `http://www.sae.org/technical/standards/AS5710` (accessed December 2008)