Synchronising Physiological and Behavioural Sensors in a Driving Simulator

Ronnie Taib
National ICT Australia
13 Garden Street
Eveleigh NSW Australia
ronnie.taib@nicta.com.au

Benjamin Itzstein
National ICT Australia
13 Garden Street
Eveleigh NSW Australia
benjamin.itzstein@nicta.com.au

Kun Yu
National ICT Australia
13 Garden Street
Eveleigh NSW Australia
kun.yu@nicta.com.au

ABSTRACT

Accurate and noise robust multimodal activity and mental state monitoring can be achieved by combining physiological, behavioural and environmental signals. This is especially promising in assistive driving technologies, because vehicles now ship with sensors ranging from wheel and pedal activity, to voice and eye tracking. In practice, however, multimodal user studies are confronted with challenging data collection and synchronisation issues, due to the diversity of sensing, acquisition and storage systems. Referencing current research on cognitive load measurement in a driving simulator, this paper describes the steps we take to consistently collect and synchronise signals, using the Orbit Measurement Library (OML) framework, combined with a multimodal version of a cinema clapperboard. The resulting data is automatically stored in a networked database, in a structured format, including metadata about the data and experiment. Moreover, fine-grained synchronisation between all signals is provided without additional hardware, and clock drift can be corrected post-hoc.

Categories and Subject Descriptors

H.5.1 [**Information interfaces and presentation (e.g., HCI)**]: Multimedia Information Systems - *Artificial, augmented, and virtual realities; Evaluation/methodology.*

General Terms

Measurement, Reliability, Experimentation.

Keywords

Multimodal data synchronisation; data collection and storage; physiological and behavioural sensors; driving simulator; cognitive load.

1. INTRODUCTION

With the advent of cheaper and simpler sensors, the multimodal interaction research community and human-computer interaction practitioners at large have started to explore richer interfaces, by fusing a larger number of signals. Behavioural signals such as speech, gesture or facial expressions, which were until recently the basis of a majority of multimodal systems are now complemented by a mixture of behavioural and physiological sensors. They include gaze tracking and pupil dilation (e.g. using eye tracking glasses (ETG)), galvanic skin response (GSR), heart rate, blood volume pulse (BVP), muscular activity (EMG) and electroencephalography (EEG). Complex set-ups, such as functional magnetic resonance imaging (fMRI) scanners are also starting to be used in non-medical studies related to psychology and cognition. Environmental sensors, such as ambient temperature and noise, are also useful for ensuring experimental control conditions are maintained.

The benefits of fusing multiple sensor signals are to increase reliability, range, complementarity and sensitivity, as some sensors may not be accurate at all levels of activity, or may have a lower signal-to-noise ratio for the specific quantity to be studied. Cross-referencing signals improves pattern recognition, especially when external noise is present, or when discriminating similar conditions. The low cost and miniaturisation of many sensors also make them available not only to researchers, but to the consumer market, promising interesting deployments of multimodal systems.

However, the new sensors come with heavy requirements, especially in terms of annotation and analysis complexity. Physiological sensors are typically acquired in tens to thousands of Hertz, and often across multiple channels, such as in the case of EEG. Depending on the hardware or vendor, pre-processing may be available and offer high level features of interest to the researchers. But even so, the data rate and format are usually not compatible with other sensors. Practically speaking, fine-grained synchronisation of mixed signals is almost impossible with modern low-cost sensors, or comes at the expense of extremely long and tedious manual annotations.

This paper is based on several years of experimental work in our team, which have led to the creation of a usable, low-cost, synchronisation methodology for a wide range of sensors. It relies on the OML framework, and we describe its integration in our latest research on cognitive load measurement in a driving simulator. We discuss the major hurdles that researchers in our domain face, how we addressed them, and how our methodology can benefit other research community.

2. BACKGROUND

2.1 Physiological and Behavioural Sensors

As far as multimodal interaction is concerned, sensors are used to provide insights into human activity (behavioural sensor) or physiological reactions. In this paper, we distinguish the two in terms of conscious vs. unconscious process, whereby eye gaze is a behavioural process, whereas pupil dilation is a physiological signal, for example.

From the Polhemus used by Bolt et al. in early speech and gesture fusion, to the Kinect or Google Glass, it seems impossible to write a comprehensive list of behavioural sensors which have been used

ICMI'14, November 12–16, 2014, Istanbul, Turkey.

http://dx.doi.org/10.1145/2663204.2663264

to capture human activity, often to allow interaction with machines or other humans.

Even more variety can be found in physiological sensors, including much wider ranges of quality and cost. These are progressively being adopted by the human-computer interaction (HCI) community, for their ability to represent the user's psychological state, possibly in real-time. While this psychophysiological inference is complex and still open to research, multimodal fusion of physiological inputs and task space measures are promising to improve the robustness and complexity of the user representation [1], leading to finer models of human activity.

Nowadays, different biometrical sensors are often utilised together to gather appropriate data and investigate the behavioural or physiological patterns of drivers. The most popular devices utilised include electroencephalogram (EEG) sensors and the eye trackers or front camera, due to their high sensitivity and convenience for live monitoring [2], [3]. Heart rate and skin conductance level were considered reflective of cognitive load changes, and in many studies they were studied together [4], [5]. Many devices are capable of monitoring electrocardiogram (ECG) and measuring skin conductance (GSR) at the same time, such as the commonly used Vitaport Temec BV, MEDAC or Procomp Infiniti. Some other sensors used involve detection of the electrooculography (EoG) or electromyography (EMG) changes of the eye or facial muscles [6], [7], and gestures of the hand [8]. However, any single sensor used in current driver monitoring systems, is not capable of reliably detecting a driver's mental or physical status under all circumstances. For example, hand movement detectors are reported capable of detecting early stages of fatigue [8], however some false alerts may occur if the driver is driving along a straight lane. Eye tracking cameras determine a driver's fatigue state via the pupil and eyelid movements, which is sometimes sensitive to illumination and distance of the camera to the eye [9], [10].

Combined usage of the sensors aims to improve the sensitivity and accuracy of driver mental state monitoring, and an increasing number sensors have been integrated and deployed in studies, due to the fact that physiological sensors are becoming cheaper and transportable. They are even making their way into other field studies such as the evaluation of museum interfaces [11]. However, HCI researchers and practitioners are faced with practical issues:

- **Quality:** reliable (quantitative) measurements
- **Ease of use:** deployability, portability/mobility
- **Cost:** sensor, consumables, maintenance
- **Usability:** ease of use, subjective experience/preference

These aspects are often tightly coupled, so the selection of sensors generally comes down to a trade-off between purchase cost and signal quality. This can happen to the detriment of the ease of use and ancillary costs, in particular:

- **Storage costs** can be high when sensors provide large volumes of (uncompressed) data. Conversely, some sensors may only provide compressed data losing important information. An example are the raw eye video feeds provided by SMI's Eye Tracking Glasses, which only provide a 1Hz sampling rate;
- **Annotation requirements** may be high when the signal is analog and the sensor does not provide pre-processing for high-level features. For example, audio and video recordings are typically useful to capture the context of the experiment, but very time consuming to annotate for fine-grained annotation, such as for multi-participant multimodal input analysis [12]. Any manual annotation requirement precludes the use of the sensor in real-time systems;
- **Processing requirements** may be high when pre-processing is performed on an acquisition computer instead of within the sensor's hardware. This, combined with the number of computer ports required by the sensor, may preclude the use of many sensors on the same computer;
- **Deployment complexity** is increased because more time and procedures are required to deploy all the sensors. Interferences between different sensors may exist, and failure of one single sensor may result in the invalidity of all the data;
- **Synchronisation** between multiple signals is difficult to perform *post hoc*, because signals are often not timestamped consistently across vendors. Moreover, timestamps are sometimes too coarse, inexistent, or subject to clock differences and drifts between acquisition machines.

This latter aspect is the main topic addressed by this paper.

2.2 Data Logging and Synchronisation

The preferred output of a multimodal data logging system would keep all data universally timestamped, i.e. along the same timeline, as proposed in [13]. However, the limited processing power of acquisition machines, and to some extent the bandwidth and storage costs, impose that multiple computers be used for data collection. To facilitate data aggregation and to address the synchronisation requirement imposes that these machines be networked and that attention be paid to differences in their clocks. Hardware solutions such as multi-relay cooperation scheme [14] or GPS disciplined oscillators are effective but require additional, often complex and costly hardware, and may not function indoors. In this paper we focus on low-cost software solutions.

A framework for multimodal data collection and synchronisation was proposed by [15]. It relies on the Network Time Protocol (NTP), the standard clock synchronisation protocol used on the Internet. A limitation is that it uses proprietary timestamping of data, such as video frames. Also, the target signals (video, audio and electronic pens) could be synchronised with an accuracy of up to 120ms in only 80% of the cases. Furthermore, their pen data suffered large synchronisation offsets of up to 120 seconds, which could not be corrected *post hoc*. In other research, a distributed fusion architecture using a central track file was used to collate multiple sensor data, with variable sampling rates, in a vehicular environment [16]. However, they did not specifically address synchronisation and instead relied on fast network data transfer to a central location. While this may work satisfactorily in a car, with streaming sensors, a larger computer network may adversely affect synchronisation and the approach doesn't readily apply to buffered or processed sensor data.

Looking at standard ways to perform these functions, and with better synchronisation, we found two main technologies able to provide data transfer and centralised storage, in consistent format.

The MQ Telemetry Transport (MQTT) is an open source protocol defining broker-based message passing, and offers complex features such as message queuing and selectable levels of quality.

While it is designed to be open, simple, lightweight and easy to implement [17], [18], and offers application programming interfaces (API) for a multitude of languages, we found that the protocol comes with many business-oriented features which may not be relevant or easy to deploy for user study experimenters. Furthermore, its message passing technology has heavy dependencies, especially IBM WebSphere. Most importantly, while the quality of service can be tuned to regulate message ordering for example, the protocol does not seem to include any timestamping or specific support to synchronise messages at a fine level.

Another option is the Orbit measurement library (OML) framework [19] which can be used as a reporting system for any sort of set-up requiring moving sampled data from decentralised sensing devices to a centralised storage database. It offers automatic timestamping facilities, and implementations in multiple programming languages allowing easy integration with various sensors, even if they come with different APIs.

The OML framework consists of a lightweight client library through which instrumented applications report individual sensor readings, and a data collection server which automatically stores them in a database for later analysis, using a consistent format. We found that the server installation using Linux packages was trivial and involved few dependencies. Further to its simplicity, OML is an open standard and seems to be adopted by an increasingly wider community.

3. USE-CASE: EXPERIMENT DESIGN

3.1 Experiment Overview

The user study used here as a test bed for the OML logging examines the performance of drivers under varying levels of cognitive load. The study is carried out in a simulator, as we are eliciting potentially dangerous levels of inattention. The main task is related to real-life driving on a 2x1 lane country road, with some surrounding traffic present. Secondary tasks involving mental recall and processing are used to control the level of cognitive load experienced by the participant, and are carefully crafted to:

- Be easily conducted while driving;
- Induce scalable levels of cognitive load;
- Resemble everyday mental processes while driving.

The experiment was conducted in five sub-sessions of about 5 minutes each. Since the focus of this initial experiment was to validate the data collection and synchronisation framework, we only recruited six volunteer participants. All had to own a driving license, because traffic rules applied during the experiment, and we didn’t want cognitive load generated by inexperienced drivers. Participants were also restricted to those not requiring glasses or contact lenses, due to a limitation in the eye tracking system.

The driving simulator itself is a low-fidelity simulator, composed of a real car driving seat, Logitech Driving Force GT gaming steering wheel and pedals, and three monitors (1920x1080 pixels each), as pictured in Figure 1. The Unity3D game engine was used to implement a customized virtual driving environment matching the task design requirements.

Figure 1: Driving simulator

Emphasis was placed on creating immersive realistic conditions, through the use of ordinary tourism cars, coaches and trucks, as well as items of scenery such as a farm building or cows. The car model, e.g. acceleration, grip and noise, was tuned to provide as realistic a handling as possible. Unity3D provides some fine rendering features, making the experience more immersive, including realistic sky, water bodies and reflections on the car hood. It also generates sound for passing cars and wind on the windshield, based on vehicle speeds.

3.2 Sensors

The sensors used in the experiment were relatively low cost, and easy to use. They all had different sampling rates and numbers of channels, and could not be run concurrently on the same computer because of their computing requirements (EEG and ETG in particular), so the data acquisition was spread across several networked machines. An example of an acquisition client is shown in Figure 2. A comprehensive list of sensors is provided in Table 1, along with the actual location of these sensors shown in Figure 3.

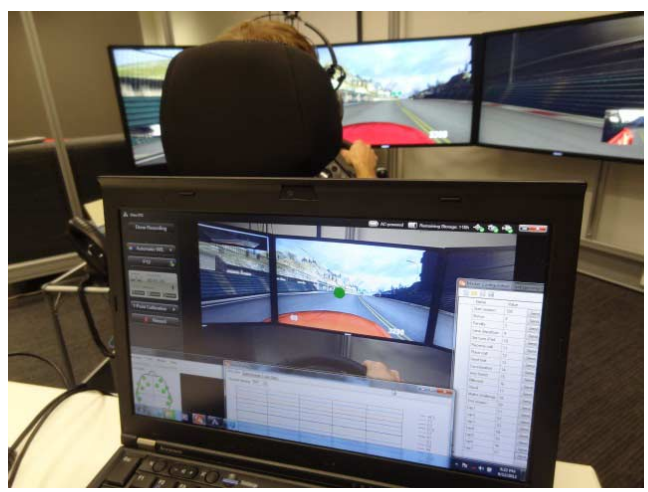

Figure 2: Sensor acquisition machine (OML client)

A microphone and camera pointing at the participant’s face were also recording the general interaction and facial expressions. Our goal is to provide mental state detection in real-time, without the need for manual annotation. Hence this microphone and camera are only used as a reference in our analysis.

Table 1: Sensors and characteristics.

Sensor	Sample rate	Data streams	Device
Eye tracking glasses (ETG) 1	30 Hz	3 channels: gaze direction, pupil dilation, blinks	SMI ETG
EEG headset 2	128 Hz	14 channels EEG, 2 axis gyroscope	Emotiv Epoc
GSR (weak hand fingers) 3	512 Hz	2 channels: galvanic skin response, blood volume pulse	Procomp Infiniti
BVP (ear lobe) 4			
4 Pressure sensors on seat pan, 6 on backrest 5	500 Hz	18 channels: pressure and distance for posture, ambient temperature, vibration	3x Phidget 8/8/8
3 Distance sensors for head, 6 1 for chest, 7 2 for legs 8			
Temperature (beside seat)			
Vibration (see Section 4.3)			
Simulator events	50 Hz	10+ channels	Unity3D C# interface
Annotations (by experimenter)	125 Hz	1 channel: key logging	Keyboard

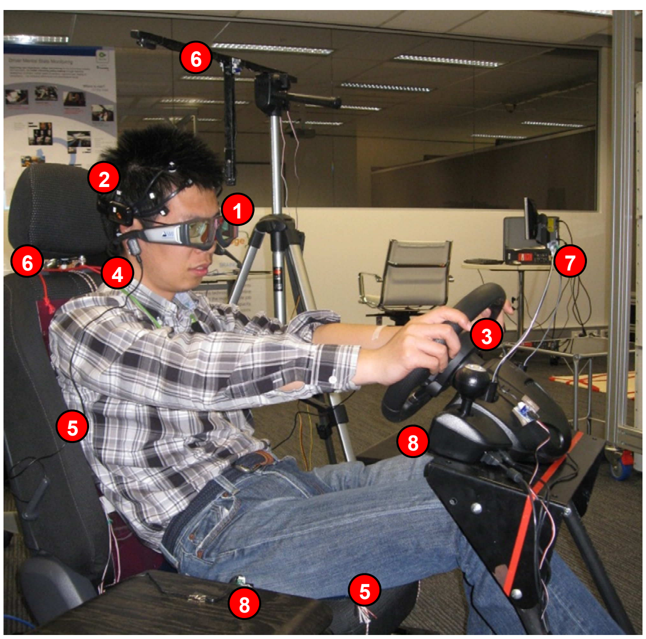

Figure 3: Sensor placements

The prominent practical issues we encountered and address in this paper include:

- Inconsistent acquisition methods: polling vs. event based, leading to a mix of regular streaming samples vs. sporadic samples, according to the sensor used;
- Diverse API programming and languages;
- Eclectic data formats reported by the devices, often geared towards manual review, not automated analysis;
- Most devices introduce internal sub-sampling, buffering and other mechanisms, that may obscure the true time of each data point. We assumed that these are negligible for our purpose, i.e. below 10ms;
- Some devices provide timestamping, based on the local computer clock, others on their internal clock, and others yet didn't provide any timestamping at all.

These issues were addressed through the development of custom parsers and the integration within the OML framework, as detailed in the following sections.

4. OML DEPLOYMENT

4.1 Integration of OML into User Study

OML clients were coded in C#, using the respective API of each sensor. Each client created a reporting thread to read data (pushed from some sensors, and polled from others) and used the OML client API to transfer the readings to the OML server for persistent storage. All sensors described in Table 1 were assigned an OML client, with the exception of the ETG, which did not provide a working API for real-time data acquisition. The eye tracking data had to be imported *post hoc* into the OML database, and the scene video and audio captured by the ETG were stored in files.

Due to the processing requirements of the various sensor drivers and APIs, the data collection had to be spread across a number of computers, as detailed in Table 2.

Table 2: Experiment network topology

Computer	Sensors
Windows PC 1	Simulator events, driver behaviour, live keyboard annotations
Windows PC 2	EEG, GSR & BVP, posture, temperature, vibration
Windows PC 3	Eye tracking glasses (eye tracking, scene video and microphone)
Linux server	[OML server]

Once pushed into an OML client, the data becomes a *measurement point*, which is automatically attributed a timestamp by the client, and gets linked to the current experiment, by way of a unique ID mechanism. This is convenient for our user study because metadata (experiment domain, application name and sender ID) gets automatically saved along with the raw sensor data, ensuring safe management of the data, and hence best practice in terms of data archival, without additional effort.

Every measurement point is transmitted as a single OML message over Transmission Control Protocol (TCP) and stored in the database on the server. Two standard timestamps are automatically appended to every measurement:

- oml_client_ts: the message send time, based on the client system clock;
- oml_server_ts: the message reception time, based on the server system clock.

For our data collection purposes, the OML server was configured with a PostgreSQL back-end, on a shared Linux server. A positive side-effect of storing the user study data in a centralised database is that we can leverage near real-time access to structured time series data, and so are able to realise a number of benefits:

- Visualisation of signals, for monitoring activity and to ensure sensor quality remains stable, throughout the experiment. This proved important for example to detect when the BVP sensor moved slightly out of place on a participant's earlobe. While not apparent to the experimenters, because the move was small, the impact on signal quality was significant;
- Feature extraction for immediate use in classification and prediction models;
- Data sets are automatically organised and stored in a consistent documented format, including metadata about the experiment.

Figure 4 shows a web-based session visualisation tool we developed, that can dynamically calculate and display any combination of time series features, for any session, by running SQL queries on raw data stored on the OML server, in real-time or *post hoc*.

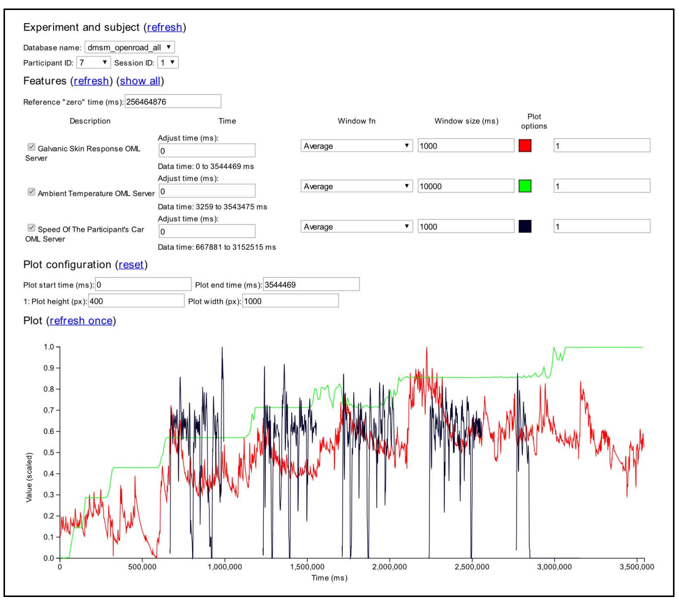

Figure 4: Visualisation tool, showing GSR, car speed and ambient temperature

4.2 Synchronisation Challenges

Generally speaking, synchronising clocks is a difficult problem, with a large body of research and development attached to it. Concrete solutions are provided in many applications, and generally require some trade-offs between accuracy and cost, depending on the needs of the application.

From the perspective of an HCI practitioner wanting to run user studies with multiple sensors, three primary synchronisation challenges exist:

- **Lack of sensor timing:** Some sensors do not provide accurate (if any) timestamping, nor any API allowing real-time data acquisition. This is the case for the version of the ETG glasses and software we were using. Moreover, these sensors may have uncertain or unreliable sample times, as their timestamps are relative to an unspecified start time and clock. For example, we found that the scene video, audio, and eye data, all being captured on the same ETG device for the same period, will vary in overall duration, sometimes in the order of several seconds. Data file creation and modification times are often the only time reference available for some sensors, and these do not provide much precision. In essence, we have no inherent indication of when the first sample was captured and of any systematic drift that occurred over the recordings;
- **Acquisition machine clock:** The network time protocol (NTP) is a well-established mechanism for fine-grained synchronisation of computers over networks. Unfortunately, while usually reliable on Linux machines, the Windows Time Service implementation of the protocol does not guarantee synchronisation with a time server with an accuracy any better than 5 minutes. In practice, the delay is usually much lower, but we will show in the coming sections how a computer clock may apply many corrections to synchronise to a server, introducing substantial jitter in the clock, hence dramatically affecting readings;
- **Lag in the sensor and driver:** Most sensors, especially cheaper ones, will have some lag between the time when an event occurs (and is sensed) and when it becomes available within the API. This is due to a combination of delay in the embedded electronics, software processing by the driver, and possibly multiple buffering stages. This information is usually not readily available, and extremely difficult to ascertain without using metrology equipment. In this paper, we assume that this delay is typically under 10 ms, hence below the target accuracy we set to achieve.

The next two sections detail how we addressed the first two issues, by respectively introducing a multimodal synchronisation event, and using OML to adjust timestamps *post hoc*.

4.3 Multimodal Synchronisation Event

In order to address absent system timings, and provide a control measure in this study, we designed a multimodal synchronisation event which appears on at least one OML-connected sensor for each system clock (on PC 1 and PC 2), as well as for the sensors which are not connected to OML (ETG, audio and video on PC 3). The event is also designed so that it generates clear signal patterns that would not be readily confused with ordinary behaviour in the experiment and could be automatically identified from the signal with some heuristics. For example, tapping on an accelerometer typically produces a sharp high spike in the signal.

In between each task of the experiment, (approximately every 7-10 minutes), the participants were instructed to perform a set of synchronisation actions *simultaneously*, as described in Table 3. This set was repeated 5 times in a row, at a 1 second interval roughly, to make it easily distinguishable in the signal.

Table 3: Synchronisation event actions

Simultaneous action	Targeted sensor
Tap the NumLock key with the provided vibration sensor in the right hand 	PC 1: keyboard annotations PC 2: Phidgets (vibration) Additionally: PC 2: Microphone PC 3: ETG (microphone)
Observe the NumLock light key with the ETG camera	PC 3: ETG (scene video)
Blink eyes deliberately	PC 3: ETG (eye tracking) PC 4: Face video
Tap the gyroscope on the EEG headset with the left hand 	PC 2: Emotiv EPOC (EEG/gyro)

Recording just one of these synchronisation sequences is sufficient to align all targeted data streams at a given point in time, simultaneously correcting any offsets due to differences in clocks. However, for redundancy and to identify and minimise the effect of any clock drift over the period of the experiment, we chose to periodically repeat the synchronisation sequence after each task. While we cannot rely on participants to perfectly align the simultaneous actions, we assume that they will all occur within 30ms, which is the threshold at which humans can typically perceive coincidence between events [20].

4.4 Synchronisation without External Event

During the experiments, the clocks of PCs 1-4 were being actively synchronised to a time server by the Windows Time Service. At the beginning of the experiments all PC clocks agreed, as observed by the experimenters, however, plotting the signals corresponding to a multimodal synchronisation event unveiled a very important issue. Throughout the experiment, PC 2 significantly skewed its clock, in an effort to synchronise with the time server, making its clock tick at a different pace from the OML server and other PCs.

Figure 5 zooms on the second multimodal synchronisation event for one participant. The onset time of the initial spike was manually adjusted to align with the other signals. That lag was 104s for PC 2 and 0.7s for PC 3. Given that the ETG data on PC 3 was timestamped based on file modification time, such a static offset, of order 1 second, was expected.

After accounting for the clock offset PC 1 and PC 3 were relatively in sync, as can be seen by the aligned NumLock presses (E: large yellow spikes) and ETG blinks (D: red spikes). Since it couldn't be timestamped in real-time, ETG couldn't be synchronised by our system, and was only used it to assess delay/drift between machines during manual validation.

However, the EEG activity, calculated as the total energy on all the EEG channels (A: black), EEG gyro (B: blue) and vibration sensor (C: green), all captured on PC 2, are running quite ahead. In the short 4 second synchronisation event pictured here, the signals on PC 2 moved 1200ms ahead of the rest, as can be seen by their 5th peak being shifted to the right compared to the yellow spike.

Figure 5: Client times, with PC 2 signals running ahead

Figure 6 shows the same event, using the OML server time, i.e. the arrival times of reported measurements, with no adjustments. Clearly, they are well aligned, which confirms that the PC 2 clock is dramatically drifting. The slight lag observed between the yellow spikes and the other peaks can be explained by the fact that the NumLock press event (D: yellow) gets detected by the computer hardware when the key is being pressed half way down or so. The vibration peak (C: green) is observed when the key reaches its fully depressed position, hence the sensor gets stopped suddenly. The lagging EEG gyroscope (B: blue) and activity (A: black) suggests that this oml_client_ts (s) on the headset with a slight delay.

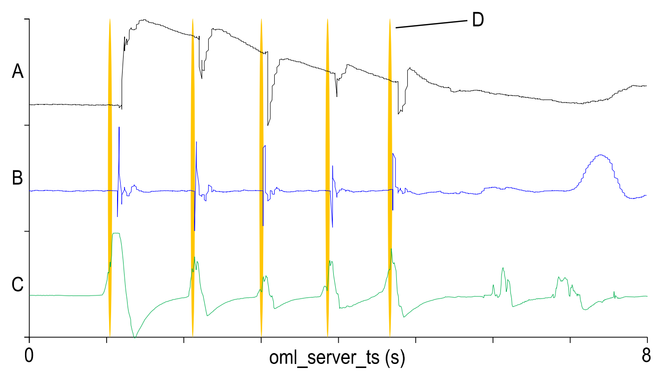

Figure 6: Server times, with PC 2 signals running ahead

Based on these results, it would be tempting to simply use the server timestamp as a reference for all the experimental data, and this would in fact provide an accuracy of the order of 100ms or better on a high quality local area network. However, we will find in the next two sections that other issues exist, but can be addressed by the framework.

5. SYNCHRONISATION CORRECTION

5.1 Timestamp Fusion

The server timestamp is dependent on the packet transmission time, hence the conditions of the network. When looking closely at the curves in Figure 5 and Figure 6, it can be noted that the former exhibits sharper features than the latter, for the same event. This can be explained by small fluctuations in the network, as well as potential buffering within the TCP stack.

The way to address this issue is to use the server timestamp to correct the synchronisation of the client timestamps, globally. In addition to the two standard OML timestamps introduced in Section 4.1 (oml_client_ts and oml_server_ts), we added a custom timestamp to our measurements:

- client_sample_ts: an educated guess of the time the data sample was captured, based on the system clock of the client, and the behaviour of the API and sensors, e.g. back-dating buffered samples via linear interpolation.

In a high performance real-time system, where samples are processed as soon as they are captured and the clocks are in sync, we would expect that client_sample_ts and oml_client_ts should be very similar to each other.

In an OML set-up including several clients, any drifting clock will become apparent when compared to the rest of the cohort. The timestamp for that clock can be adjusted according to the rest of the clocks. In the unlikely situation where all clocks would drift, the final timestamps would not represent absolute time correctly, but would at least all be aligned, which is paramount for any fine-grained HCI analysis.

5.2 Offset Correction

As noted before, the first event in Figure 5 and Figure 6 has been manually aligned to allow drift comparison. This relative time offset between computer clocks is easily measured by comparing the signals, using their original client timestamps (client_sample_ts). Figure 7 illustrates this, where the tap vibrations (green) are dramatically ahead of the NumLock press (black dots) and eye blinks (blue triangles).

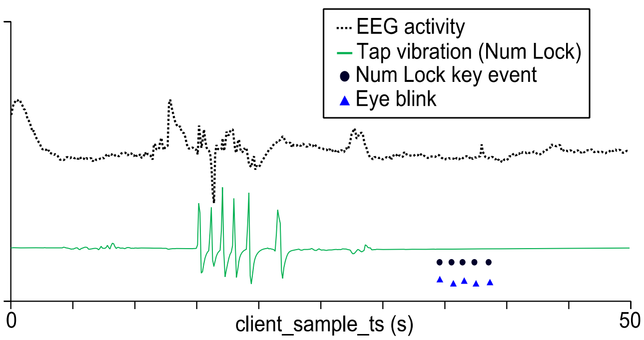

Figure 7: Offset between client clocks

More generally, by comparing oml_client_ts time to oml_server_ts for a measurement, any difference greater than 100ms (network transmission time) is likely due to an offset in the client's clock.

Detecting the spikes in the multimodal synchronisation event will also provide concrete confirmation of that offset if needed, as well as determining offset of non-OML connected data streams.

5.3 Drift Correction

A benefit of collecting multiple timestamps is that they can be used to quantify the amount of lag and drift on each individual clock in the system. In the case studies in section 4.4 where the clock of PC 2 appeared to be fast, we can actually get a much more precise insight into the situation by examining the difference between the client and server clock. The blue line in Figure 8 represents that difference over the one hour duration of the experiment. The line essentially reflects the correction pattern that the Windows Time Service was applying to the PC 2 local clock, throughout the experiment. It is made of series of linear corrections, getting the clock to alternatively tick too fast and too slow until convergence can be achieved. Clearly, the signals captured on PC 2 would continuously move ahead and behind the rest of the signals for no apparent reason should that pattern not be unveiled.

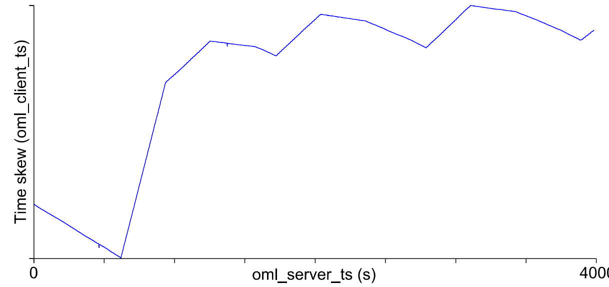

Figure 8: Clock drift pattern over the full experiment session

Once unveiled, though, it can be used to apply linear correction for each of the sections, hence completely repair the signal. At this stage, the process is manual, but it could be automated in the future, since the blue line can be obtained automatically.

6. DISCUSSION

Based on a combination of standard tool (OML) and empirical method (multimodal synchronisation event) we managed to provide synchronisation between a large number of heterogeneous sensor signals. This was achieved at very low cost, and improved synchronisation accuracy over traditional methods by an order of magnitude or more. We successfully implemented simple heuristic filters to detect tap vibrations and deliberate blinks from the data stored in the OML database. However, most of our participants tapped on the EEG gyroscope lightly, so the signal-to-noise ratio was quite low. We will explore ways to increase that ratio in the future, simply by instructing them to tap stronger, or providing a mechanical aid for tapping on the sensor.

Interestingly, we found that the EEG activity seemed to be a more robust indicator of the tapping activity. We are planning to explore whether this is due to the participant focusing on this specific task, hence producing specific brain activity, or whether it is simply due to the mechanical sensitivity of the EEG pads when the headset gets tapped on (even lightly).

We unveiled the dangers of relying on automatic clock synchronization, through Windows Time Service. Using Meinberg NTP software for Windows could reduce the observed client clock offset, but the protocol introduces drift by nature. We proposed a way to repair such errors *post hoc*.

In this paper, we assumed that the latency between raw sample capture and that data becoming available via the API is stable throughout the experiment. We will be on the look out for more research results in this field. We are planning to explore the user-to-user variation during the multimodal synchronisation event, to quantify the accuracy and consistency of this method.

7. CONCLUSION

This paper identified a set of key requirements for data collection in experiments where multiple physiological and behavioural sensors are being used. These requirements are low storage cost, minimal manual annotation, limited processing power, data synchronisation, and easy deployment. These typically impose that multiple networked machines must be used to pre-process and store the signals. Past research addressed the issue by using the network time protocol (NTP) and manual annotations, but these are limited by varying unreliability in some operating systems such as Windows, and also produce synchronisation accuracies of the order of 1 second at best.

We tested a set-up centred around the Orbit measurement library (OML) framework, in the context of a driving simulator, with multiple physiological and behavioural sensors in use. The framework provided simple-to-deploy data transfer, centralised storage and timestamping features. We showed that a severely drifting clock on an acquisition computer could be corrected effectively by cross referencing the timestamps of the server and several clients. Further levels of correction could be applied, leading to well under 100ms precision.

However, cheaper sensors may not come with real-time API so we also designed a multimodal synchronisation event, inspired from cinema clapperboards, whereby multiple signals get physically aligned. A benefit of that event over traditional manual synchronisation (e.g. using video recordings) is that it generates signal patterns that can be identified automatically using simple heuristic-based data filters.

In conclusion, the OML framework and multimodal synchronisation event can help HCI practitioners make the most of multiple low cost sensors. They can provide synchronisation support at least an order of magnitude more accurate than traditional systems, and can be deployed easily on operating systems with varying levels of clock reliability. Most importantly, they can dramatically decrease the need for manual annotations.

8. ACKNOWLEDGMENTS

The authors would like to thank the user study participants for their time and patience. NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

9. REFERENCES

[1] S. H. Fairclough, "Fundamentals of physiological computing," *Interacting with Computers*, vol. 21, no. 1–2, pp. 133–145, Jan. 2009.

[2] M. Bartels and S. P. Marshall, "Measuring cognitive workload across different eye tracking hardware platforms," presented at the Proceedings of the symposium on eye tracking research and applications, 2012, pp. 161–164.

[3] D. De Waard and V. Studiecentrum, *The measurement of drivers' mental workload*. Groningen University, Traffic Research Center, 1996.

[4] J. Engström, E. Johansson, and J. Östlund, "Effects of visual and cognitive load in real and simulated motorway driving," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 8, no. 2, pp. 97–120, 2005.

[5] J. Son, Y. Lee, and M.-H. Kim, "Impact of traffic environment and cognitive workload on older drivers' behavior in simulated driving," *International Journal of Precision Engineering and Manufacturing*, vol. 12, no. 1, pp. 135–141, 2011.

[6] M. Akin, M. B. Kurt, N. Sezgin, and M. Bayram, "Estimating vigilance level by using EEG and EMG signals," *Neural Computing and Applications*, vol. 17, no. 3, pp. 227–236, 2008.

[7] R. N. Khushaba, S. Kodagoda, S. Lal, and G. Dissanayake, "Driver drowsiness classification using fuzzy wavelet-packet-based feature-extraction algorithm," *Biomedical Engineering, IEEE Transactions on*, vol. 58, no. 1, pp. 121–131, 2011.

[8] T. C. Chieh, M. M. Mustafa, A. Hussain, E. Zahedi, and B. Majlis, "Driver fatigue detection using steering grip force," presented at the Research and Development, 2003. SCORED 2003. Proceedings. Student Conference on, 2003, pp. 45–48.

[9] S. Hu and G. Zheng, "Driver drowsiness detection with eyelid related parameters by Support Vector Machine," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7651–7658, 2009.

[10] R. F. Knipling, W. W. Wierwille, and I. America, *Vehicle-based drowsy driver detection: Current status and future prospects*. National Highway Traffic Safety Administration, Office of Crash Avoidance Research, 1994.

[11] A. J. Karran, S. H. Fairclough, and K. Gilleade, "Towards an adaptive cultural heritage experience using physiological computing," in *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, Paris, France, 2013, pp. 1683–1688.

[12] S. Oviatt and A. Cohen, "Written and multimodal representations as predictors of expertise and problem-solving success in mathematics," in *Proceedings of the 15th ACM on International conference on multimodal interaction*, Sydney, Australia, 2013, pp. 599–606.

[13] A. L. Thompson and D. Bohus, "A framework for multimodal data collection, visualization, annotation and learning," in *Proceedings of the 15th ACM on International conference on multimodal interaction*, Sydney, Australia, 2013, pp. 67–68.

[14] J. Zhang, J. Jia, Q. Zhang, and E. M. K. Lo, "Implementation and Evaluation of Cooperative Communication Schemes in Software-Defined Radio Testbed," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1–9.

[15] A. M. Arthur, R. Lunsford, M. Wesson, and S. Oviatt, "Prototyping novel collaborative multimodal systems: simulation, data collection and analysis tools for the next decade," in *Proceedings of the 8th international conference on Multimodal interfaces*, Banff, Alberta, Canada, 2006, pp. 209–216.

[16] A. Amditis, A. Polychronopoulos, I. Karaseitanidis, G. Katsoulis, and E. Bekiaris, "Multiple sensor collision avoidance system for automotive applications using an IMM approach for obstacle tracking," in *Proceedings of the Fifth International Conference on Information Fusion, 2002*, 2002, vol. 2, pp. 812–817 vol.2.

[17] IBM and Eurotech, "MQTT v3.1 Protocol Specification." IBM, 19-Aug-2010.

[18] V. Lampkin, W. T. Leong, L. Olivera, S. Rawat, N. Subrahmanyam, and R. Xiang, *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry*. 2012.

[19] O. Mehani, G. Jourjon, T. Rakotoarivelo, and M. Ott, "An Instrumentation Framework for the Critical Task of Measurement Collection in the Future {I}nternet," *Computer Networks*, vol. 63, pp. 68–83, Apr. 2014.

[20] B. Dainton, "Temporal Consciousness," in *The Stanford Encyclopedia of Philosophy*, Spring 2014., E. N. Zalta, Ed. 2014.