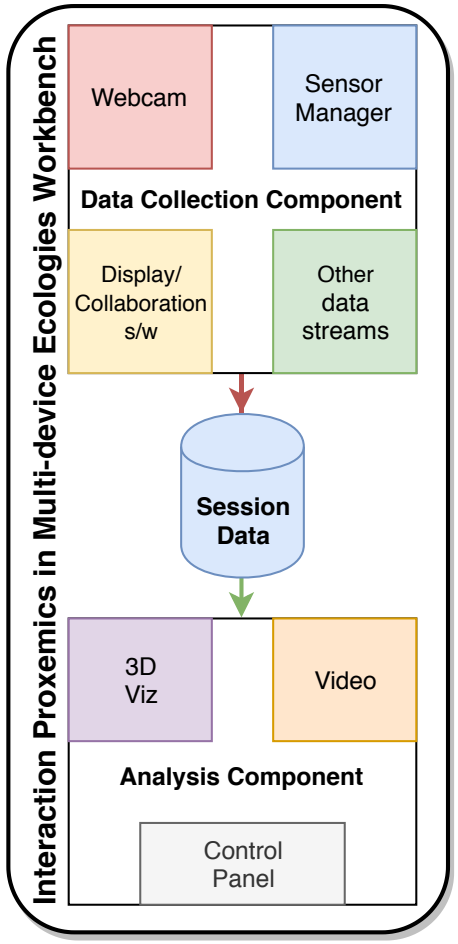

Figure 1: We present the IPME workbench, a data processing tool for mixed-methodology studies of group interactions.

IPME Workbench: A Data Processing Tool for Mixed-Methodology Studies of Group Interactions

Harish Naik
University of Illinois, Chicago
Chicago, IL, USA
hnaik2@uic.edu

Debaleena Chattopadhyay
University of Illinois, Chicago
Chicago, USA
debchatt@uic.edu

ABSTRACT

Today's small group interactions often occur in multi-device, multi-artifact ecosystems. CHI researchers studying these group interactions may adopt a socio-behavioral or sensing/data mining approach or both. A mixed-methodological approach for studying group interactions in collocated settings require collecting data from a range of sources, like audio, video, multiple sensor streams, and multiple software logs. Analyzing these disparate data sources systematically—with opportunities to rapidly form and correct research insights can help researchers who study group interactions. But engineering solutions assimilating multiple data sources to support different methodologies, ranging from grounded theory methodology to log analysis are rarely found. To address this frequent and tedious problem of data collection and processing in mixed-methodology studies of group interactions, we introduce a workbench tool: *Interaction Proxemics in Multi-Device Ecologies* (IPME). The IPME workbench synchronizes multiple data sources, provides data visualization, and opportunities for data correction and annotation.

INTRODUCTION

Small group interactions are increasingly happening in spaces filled with multiple computing devices—along with other artifacts like paper, pens, and pencils. These multi-device, multi-artifact ecosystems

CHI'19 Extended Abstracts, May 4–9, 2019, Glasgow, Scotland UK.

ACM ISBN 978-1-4503-5971-9/19/05.
https://doi.org/10.1145/3290607.3312805

KEYWORDS

Toolkit; Spatial; Interaction Proxemics; Sensors; Multi-Device Ecosystem; Data Formats

Table 1: A list of software components used in the IPME workbench.

Software	Version
GCC (C++17 support)	8.2
Qt	5.12
Boost	1.68.0
OpenCV	3.4.2
Google Protobuf	3.6.0.1

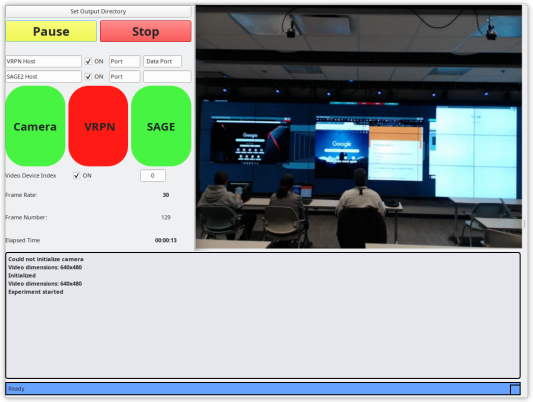

Figure 2: During data collection, the IPME workbench provides a snapshot of control buttons, connection inputs, status indicators, live webcam feed, and time string.

pose an array of interesting research questions for CHI researchers who want to study group interactions. For instance, if following a positivist approach, computing technologies are built functional—by design, but when intertwined with social processes, they come to create new challenges and opportunities in human-computer interaction [2, 3].

Studying group interactions is thus interesting to researchers in computer sciences and social sciences alike. The modus operandi in such endeavors is to collect audio/visual data from sessions conducted in small room settings and use this data to study the phenomena of interest. Other data sources may include sensor streams and log data recorded by software. Researchers then commonly use commercial software tools to annotate this data via established coding schemes or discover new ones using grounded theory methodology (GTM) like practices.

Presence of sensor and log data add new dimensions to the existing audio-visual information. They gives insights into aspects of interaction and collaboration that are not perceivable from the video or audio. However, it is hard to attain a synchronized view of the activities. At best, based on device time stamps recorded during the activity, it is possible to establish a loose correlation between events from different data streams. The danger here is that this might cause us to miss minute changes in interaction aspects or lead us to mis-interpret constructs due to non-synchronicity of data from the different sources. One way to overcome such issues is to have a central process that collects and bundles data by the proper unit of analysis and records data in a streamlined fashion.

An example of frequently encountered problems, is that, in an IR based MoCAP system, the infra-red rigid body markers can move and become misaligned during the course of the session. It isn't always possible to identify such occurrences or the magnitude of discrepancy from the video data alone. It follows that, in order to ascertain the validity of such data, it is important to visualize pose information, as a first step. Collaboration suites like SAGE2 [8] software have the ability to disseminate geometric and other provenance information about currently active UI elements. We realized that visualizing this information in relation to the participants and devices pose data would give a better context and perspective of the spatial relation of all the elements in the scene. Given our experiences during experimentation, and the infrastructure available to us, our goal was to build a toolkit that could connect to multiple data streams, record, and visualize data. Moreover, we also needed a tool that could apply manual correction and the ability to perform basic annotations.

BACKGROUND

Table 2 shows a comparison of tools that provide features pertinent to our current discussion. We compare three proprietary tool-kits (VICON[14], OptiTrack[10], Kinect[15], three open source tool-kits (Creepy Tracker[13], SoD-Toolkit[12], Proximity Toolkit[7]) and two commercial software suites that are widely used for qualitative analyis (NVIVO[11] and ATLAS.ti[5]).

While all the other toolkits couple sensor specific handling, the IPME workbench takes a modular approach in which all the sensor registration and data multiplexing happens on the end of the

Table 2: A comparison of current tools with comparable features

Feature	NVIVO	Atlas.TI	VICON	OptiTrack	Kinect	Creepy-Tracker	SoD-T/k	Proximity T/k	IPME Wb
License	Prop.	Prop.	Prop.	Prop.	Prop.	OSS	OSS	OSS	OSS
Primary Sensor			VICON IR	OptiTrack IR	Kinect	Kinect	Kinect	VICON+Kinect	Multiplexed
OS Supported	Win./Mac	Win./Mac	Windows	Windows	Windows	Agnostic	Agnostic	Agnostic	Agnostic
Primary impl./API lang.			unknown	C++	C#	C#	Javascript	C#	C++17
Viz. / animation			✓(3D)	✓(3D)	✓	✓	✓(2D)	✓(3D)	✓(3D)
Multi-Sensor Type							✓	✓	✓
Ref. Video Support	✓	✓						✓	✓
Data Annotation	✓	✓							✓
Extensible Data Spec.									✓
Collab. sw / Disp. Intf.									✓

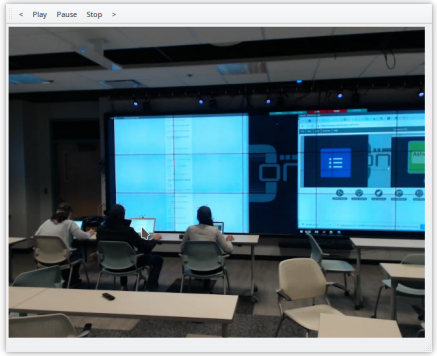

Figure 3: The video window with play/pause/stop controls. These controls also affect the animation in the visualization window.

VRPN service or any other service that it subscribes to. What matters is that the data received is convertible to the desired data format specification. The workbench handles this multiplexed data in an agnostic fashion using an extensible object oriented dispatcher framework. In our current implementation, the tool reads data from a VRPN service that disseminates rigid body motion capture data (VICON [14]/OptiTrack [10]), kinect data and touch data multiplexed all in a single stream. These data packets are appropriately tagged which allows us to optionally handle only the streams that we are interested in.

TOOL DESCRIPTION

The current version of the tool has two distinct components: *the data collection component* and the *the visualization and annotation component*.

Data Collection Component

The data collection component is also called *Live Window* because it records the live activity part of the exercise. Figure 2 shows a snapshot of the data collection UI. It has UI elements for input to the VRPN service, camera index identifier, indicators showing which of the services are running and a live streaming video from the attached webcam device. It also shows the time since start of the experiment and number of frames recorded so far. There are also UI components to enable and disable services. At the bottom of the window is a text area showing messages logs from the tool.

Figure 7 shows the *data collection workflow*. The initialization phase makes all the necessary network and peripheral connections. It reports if any issues are found such that the user can take action before data collection. While there is no best way to perform the calibration step, one recommended way would be to ask the participants to look in a particular direction for a short period of time. The data is recorded

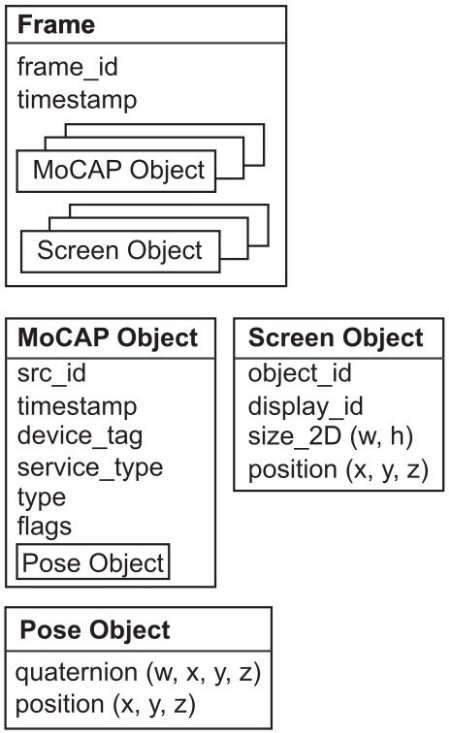

Figure 5: For each video frame, a data frame is constructed with all pertinent information, like both motion capture and screen information. A typical frame consists of *n* MoCAP objects and *m* Screen Objects. This particular data structure serves to mitigate different frame rates of data collection from different sources.

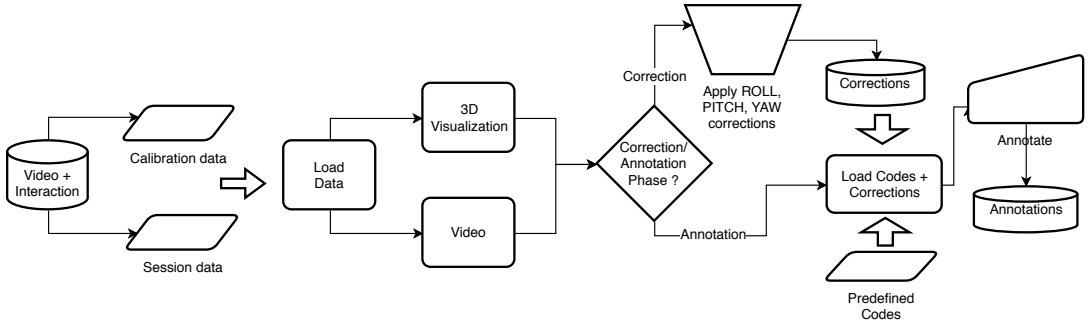

Figure 4: The analysis workflow using the IPME workbench.

and then loaded separately during the analysis phase to apply corrections. After calibration, recording of the real task can begin. This data is then saved to disk. Verification of the data currently involves a couple of steps. The user reloads the data and sanity checks the video and visualization data. As part of the suite, there is a command line utility to dump the spatial data to a JSON file. Then the user can manually sanity check this data looking for obvious clues like total absence of signal from a particular sensor.

Analysis Window

The analysis components consists of three main parts:

- **The Control Panel: Command/Control, Corrections and Annotations**: The main annotation interface is where most input controls of the visualization and annotations exist. Once the data has been annotated or coding has been performed on the data, it can be exported to CSV format.
- **Visualization**: shows two different renderings of the same frame. Users can opt to view the scene using a top-view or a front view. Apart from this, they can use mouse to control zoom, pan and rotate on these visualizations.
- **Video (replay)**: The video window as shown in figure 3 is meant to serve several purposes. The main purpose however is for use by annotators to observe the experiment and perform annotations. The control buttons on the video window, also control the 3D animation in the visualization window.

The *analysis workflow* is shown in figure 4. The calibration data if collected separately, is first loaded. The annotator may then compare the head angles of the participants, with reference to the object in a known direction, in the video and the visualization window. They may then apply roll, pitch and yaw correction using the UI elements to the objects on the scene. These corrections can then be saved to disk to be later loaded during annotation phase. If the calibration was done in the same session as task, then these corrections can be retained in memory and applied during annotation. A JSON file containing predefined codes can be then loaded and applied during the annotation process.

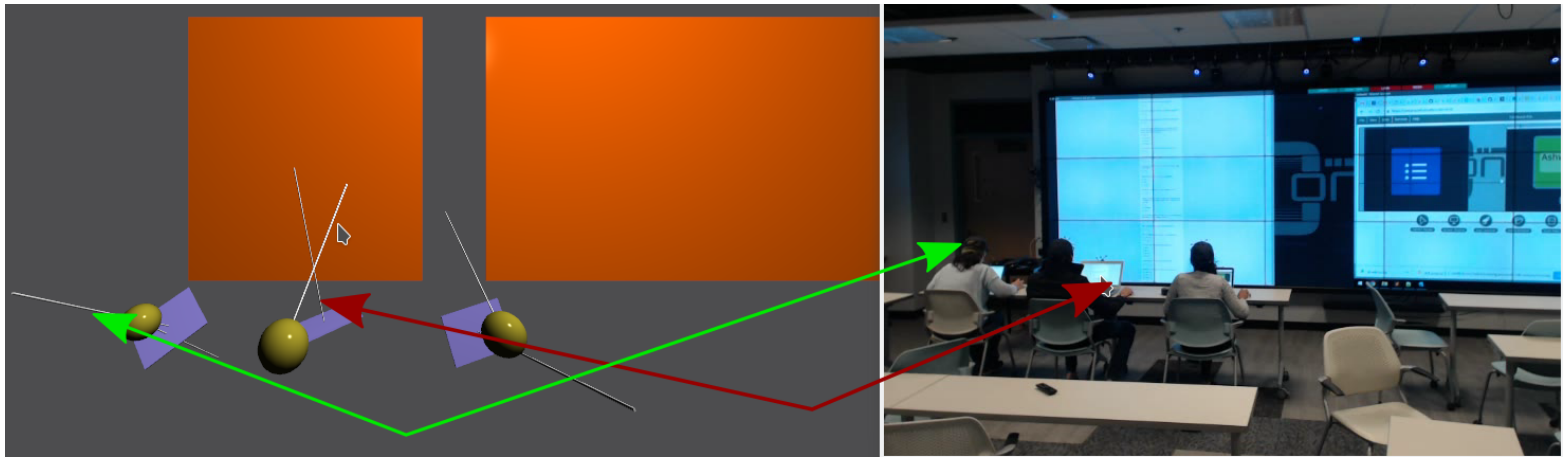

Figure 6: Shows data errors. The green arrows show the incorrect head direction as recorded in the data. The corresponding video snapshot shows that the user was not in fact looking to her right. The red arrows depict a discrepancy in the direction of the device.

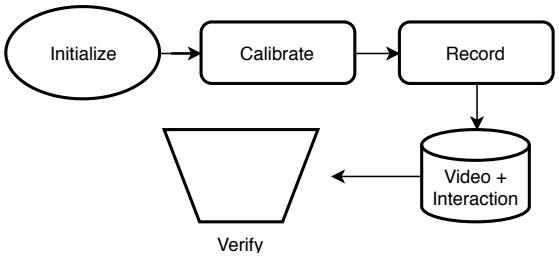

Figure 7: The data collection workflow using the IPME workbench.

We have also been using this feature for other purposes like sanity checking of the data. If we are to use the pose information of the objects (people and artifacts) in the study as a basis for inference, we need to ascertain the validity of the data. Since the video plays synchronously with the 3-D visualization, it is easy to tell if the data is corrupt or missing and identify data discrepancies as shown in figure6. In such cases, the tool has provisions to apply pitch, roll and yaw corrections.

Capabilities

The tool is currently able to support group interactions that involve people, personal devices and multiple large displays. We are able to capture the pose information of the participants' head and personal devices. Using a collaboration suite like SAGE2 [8] we are able to capture the positions of UI elements on large displays as well.

Data Format

Session Data. The data is stored in a directory containing the video file (.avi) and a single *Google protobuf* [6] file. Google protobuf message format is an industry wide standard used for serialization of data. A schematic representation of the data format is shown in figure 5.

Annotated Data. Currently only export to CSV is available. This is because, a common use of such exported data is to perform some kind of statistical analysis or inference and most tool-kits readily ingest CSV format. Export to other formats is planned in the future.

LIMITATIONS

Currently the tool can handle only three data streams namely, the Omicron multiplexed mocap service, SAGE2 environment and webcam stream. In order to make this widely applicable, we will need to extend it in order to support multiple sensor streams, interfacing with other desktop environments and audio support. Also, the way annotations are performed right now, each frame can only be assigned a single code or label. This can be easily overcome with minimal code changes. The tool currently has been developed and tested on a GNU/Linux desktop. However, since C++ compilers are available for all major platforms, it can be ported to those platforms with minimal effort.

TOOL AVAILABILITY

The tool is distributed under the GPL V3[4] license and can be freely downloaded from IPME source repository[9]. The workbench tool is written in C++ using opensource libraries like Qt, OpenCV[1] etc.

ACKNOWLEDGEMENTS

We thank the UIC Electronic Visualization Laboratory for the apparatus, software and technical support during the development of the software suite and experiments.

CONCLUSION AND FUTURE WORK

Data collection and annotation are vital phases in empirical group-interaction studies. They are mostly done using disparate hardware and software suites. We developed a tool based on our experiences,and the infrastructure that was available to us. Our current implementation of the tool aptly addresses our initial design goals in that, it functions as a central data collection tool multiplexing data from multiple sensor streams. It also satisfies the need for tool for carrying out basic annotations. However, a larger vision, is to make this a viable extensible tool for easing data collection and annotation. Given the previously mentioned avenues for future work, we would like to invite community participation and contribution in capacities ranging from bug reports and feature requests to code pull requests and design proposals.

REFERENCES

[1] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).
[2] Debaleena Chattopadhyay, Kenton O'Hara, Sean Rintel, and Roman Rädle. 2016. Office Social: Presentation interactivity for nearby devices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2487–2491.
[3] Debaleena Chattopadhyay, Francesca Salvadori, Kenton OâĂŹhara, and Sean Rintel. 2018. Beyond Presentation: Shared Slideware Control as a Resource for Collocated Collaboration. *Human–Computer Interaction* 33, 5-6 (2018), 455–498.
[4] Free Software Foundation. 2007. GNU General Public License. https://www.gnu.org/licenses/gpl.txt. [Online; accessed 04-January-2019].
[5] Scientific Software Development GmbH. 2002-2019. ATLAS.ti. https://atlasti.com/.
[6] Google. 2008. Protocol Buffers - Google's data interchange format. https://github.com/protocolbuffers/protobuf.
[7] Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 315–326. https://doi.org/10.1145/2047196.2047238
[8] T. Marrinan, J. Aurisano, A. Nishimoto, K. Bharadwaj, V. Mateevitsi, L. Renambot, L. Long, A. Johnson, and J. Leigh. 2014. SAGE2: A new approach for data intensive collaboration using Scalable Resolution Shared Displays. In *10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*. 177–186. https://doi.org/10.4108/icst.collaboratecom.2014.257337
[9] Harish Naik and Debaleena Chattopadhyay. 2017. Interaction Proxemics in Multi-Device Ecologies. https://github.com/hcilab-uic/ipme.
[10] Inc. DBA OptiTrack NaturalPoint. 2019. OptiTrack Motion Capture System. https://optitrack.com/.
[11] Lyn Richards. 1999. *Using NVivo in qualitative research*. Sage.
[12] Teddy Seyed, Alaa Azazi, Edwin Chan, Yuxi Wang, and Frank Maurer. 2015. SoD-Toolkit: A Toolkit for Interactively Prototyping and Developing Multi-Sensor, Multi-Device Environments. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces (ITS '15)*. ACM, New York, NY, USA, 171–180. https://doi.org/10.1145/2817721.2817750
[13] Maurício Sousa, Daniel Mendes, Rafael Kuffner Dos Anjos, Daniel Medeiros, Alfredo Ferreira, Alberto Raposo, João Madeiras Pereira, and Joaquim Jorge. 2017. Creepy Tracker Toolkit for Context-aware Interfaces. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. ACM, New York, NY, USA, 191–200. https://doi.org/10.1145/3132272.3134113
[14] Vicon Motion Systems. 2019. VICON Motion Capture Systems. https://www.vicon.com/.
[15] Zhengyou Zhang. 2012. Microsoft Kinect Sensor and Its Effect. *IEEE MultiMedia* 19, 2 (April 2012), 4–10. https://doi.org/10.1109/MMUL.2012.24