Rapid Development of Multimodal Interactive Systems: A Demonstration of Platform for Situated Intelligence

Dan Bohus
Microsoft Research
Redmond, WA, United States
dbohus@microsoft.com

Sean Andrist
Microsoft Research
Redmond, WA, United States
sandrist@microsoft.com

Mihai Jalobeanu
Microsoft Research
Redmond, WA, United States
mihaijal@microsoft.com

ABSTRACT

We demonstrate an open, extensible platform for developing and studying multimodal, integrative-AI systems. The platform provides a time-aware, stream-based programming model for parallel coordinated computation, a set of tools for data visualization, processing, and learning, and an ecosystem of pluggable AI components. The demonstration will showcase three applications built on this platform and highlight how the platform can significantly accelerate development and research in multimodal interactive systems.

CCS CONCEPTS

• **Human-centered computing → User interface toolkits**;

KEYWORDS

multimodal systems, situated interaction, integrative AI, streaming

ACM Reference Format:
Dan Bohus, Sean Andrist, and Mihai Jalobeanu. 2017. Rapid Development of Multimodal Interactive Systems: A Demonstration of Platform for Situated Intelligence. In *Proceedings of 19th ACM International Conference on Multimodal Interaction (ICMI'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3136755.3143021

1 INTRODUCTION

Recent years have been marked by fast-paced advances in several core perceptual technologies such as vision, speech, and language. However, despite these advances, developing end-to-end multimodal systems that bring together multiple such technologies to act autonomously or to interact with people in open-world settings remains a very challenging, error-prone, and time-consuming task.

Numerous engineering challenges arise from the complexity of the software involved and the lack of adequate infrastructure and development tools. For instance, reasoning about time, latency, and uncertainty is critical in these systems, yet there are no corresponding constructs in the fabrics of our programming languages. Tools for debugging and visualizing complex systems with multimodal streaming data are similarly lacking. Machine learning over these systems often involves training, evaluating, and fine-tuning multiple models over different components. However, while multiple toolkits exist for developing models, they mostly focus on going from a dataset to a model, and the rest of the machine learning loop from data collection to injecting a trained model into a running system typically requires additional effort, *e.g.*, coding for developing features, moving data and models across multiple tools and environments, and so on.

In this demonstration, we introduce and showcase the Platform for Situated Intelligence (or in short, \psi)—an open, extensible platform that enables fast development and study of multimodal, integrative-AI systems [1]. The platform has a similar motivation and thus bares comparison to other frameworks like ROS [3] for robotic systems, OpenSSI [5] for processing and fusing multimodal signals, and IrisTK [4] for multi-party interactions. \psi places deep emphasis on generality, extensibility, and performance, while leveraging the ease of programming provided by the .NET platform and embracing existing ecosystems like ROS.

ICMI'17, November 13–17, 2017, Glasgow, UK
© 2017 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
ACM ISBN 978-1-4503-5543-8/17/11…$15.00
https://doi.org/10.1145/3136755.3143021

2 PLATFORM FOR SITUATED INTELLIGENCE

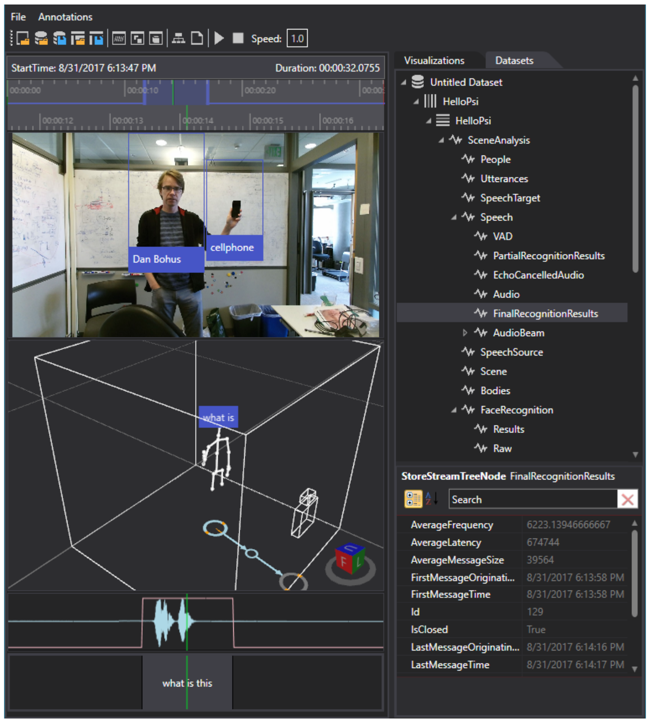
The platform provides (1) a *runtime* for time-aware, parallel coordinated computation, (2) a set of *tools* that enable visualization, data processing, and learning, and (3) an *ecosystem of components* that encapsulate various AI technologies.

2.1 Runtime

The *runtime* combines the dataflow and actor models to simplify the authoring of concurrent and distributed applications that interact with the real world under time constraints. \psi programs construct pipelines out of independent components communicating via streams of messages. Message contracts are strongly typed and enforced statically, at build time. The runtime brokers all message-passing, isolating components from each other through automatic cloning of messages. This enables it to freely schedule the execution of components based on the optimal level of concurrency given available resources, while allowing components to be single-threaded. Finally, the runtime provides time-based coordination primitives that allow components that operate on multiple streams to obtain consistent views over the input data with respect to the real world, even when their inputs are generated asynchronously with arbitrary relative lag. The end-result is that \psi programs are simple yet type-safe, thread-safe and time-aware while making optimal use of available resources (cores and machines).

2.2 Tools

A central tool in the \psi framework enables data visualization (see Figure 1). The tool allows users to navigate the set of streams



StoreStreamTreeNode FinalRecognitionResults

AverageFrequency	6223.13946666667
AverageLatency	674744
AverageMessageSize	39564
FirstMessageOriginati...	8/31/2017 6:13:58 PM
FirstMessageTime	8/31/2017 6:13:58 PM
Id	129
IsClosed	True
LastMessageOriginatin...	8/31/2017 6:14:16 PM
LastMessageTime	8/31/2017 6:14:17 PM

Figure 1: Visualization tool.

persisted or generated live and to construct complex visualization layouts that include timelines as well as instant 2D and 3D visualizers. It enables easy temporal navigation, data annotations, customization of visualization properties, persisting layouts, etc.

The framework provides end-to-end support for the machine learning loop, from data collection, feature development and model training, to evaluation and model deployment. It includes APIs and tools for grouping data stores generated from multiple runs of an application into larger datasets and performing data processing operations, such as computing derived streams or rerunning persisted sensor data through a reconfigured pipeline. Coupled with visualization, these tools enable experimentation and application tuning and debugging.

2.3 Components

The framework provides a wide array of AI components for sensors (*e.g.*, camera, microphone, Kinect, etc.), audio and visual processing components (*e.g.*, acoustic feature extraction, voice activity detection, speech recognition, face tracking, etc.), output components (*e.g.*, speech synthesis, avatar rendering, etc.), and more. In addition, bridges that allow for leveraging components from other ecosystems like ROS [3] or OpenCV [2] are available.

Finally, a toolkit targeting rapid development of multimodal systems that interact with people in multi-party, physically situated settings is currently under development. The toolkit provides perceptual components that integrate many technologies and allow authors to program against higher level constructs such as tracked people and their location, identity, attention, engagement, spoken utterances, etc.

3 DEMONSTRATION

The demonstration will focus on the platform and illustrate its various aspects via a set of different applications build on it. Participants will learn about the runtime, the set of tools and components available, and generally how to write applications with *\psi*.

The first application will demonstrate how to write a simple speech recognition pipeline, leveraging audio capture, voice activity detection, and speech recognition components, and will then show how to fuse facial tracking information from Kinect (mouth landmarks) to improve robustness in the voice activity detection. The application will illustrate core constructs, temporal synchronization, stream persistence, and data visualization.

The second application is a table-top mobile robot, Squeaky, constructed around a differential drive chassis and a Surface computer and designed to interact with children. It can drive on a table without running into obstacles or falling off. It can engage with users through face recognition and speech in the context of the objects present on the table, and can answer trivia and math questions. The application illustrates the integration of a wider array of components, including motor control, obstacle avoidance, face tracking and identification, speech recognition, language understanding and object recognition.

The third application highlights the Situated Interaction toolkit under development in *\psi*. The application displays an avatar that can interact with one or more people. The avatar attempts to detect which object a person is holding in their hand. The application couples conversational scene analysis and output components provided by the interaction toolkit with cloud services for object recognition and a simple, custom-written interaction manager.

These three applications illustrate some of the core concepts in Platform for Situated Intelligence. We hope the demonstration will inspire participants to use the framework in their research, both to develop future systems and to contribute novel components to the *\psi* ecosystem. We believe *\psi* will make it easier for the wider community to share advances and benefit from each other's work.

ACKNOWLEDGMENTS

The authors would like to thank Eric Horvitz and the entire Platform for Situated Intelligence development team, including John Elliott, Ashley Feniello, Don Gillett, Anne-Loomis Thompson, Chit Saw, Patrick Sweeney for their work and contributions to the platform.

REFERENCES

[1] 2017. Platform for Situated Intelligence. http://aka.ms/psi. (2017).
[2] Itseez. 2015. Open Source Computer Vision Library. https://github.com/itseez/opencv. (2015).
[3] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. 2009. ROS: an open-source Robot Operating System. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*. Kobe, Japan.
[4] Gabriel Skantze and Samer Al Moubayed. 2012. IrisTK: a statechart-based toolkit for multi-party face-to-face interaction. In *Proceedings of the 14th ACM international conference on Multimodal interaction (ICMI'12)*. ACM, 69–76.
[5] Johannes Wagner, Florian Lingenfelser, Tobias Baur, Ionut Damian, Felix Kistler, and Elisabeth André. 2013. The social signal interpretation (SSI) framework: Multimodal signal processing and recognition in real-time. In *Proceedings of the 21st ACM international conference on Multimedia (MM'13)*. ACM, 831–834.