OpenGesture: a Low-Cost Authoring Framework for Gesture and Speech Based Application Development and Learning Analytics

Marcelo Worsley
Stanford University School of Education
Stanford, CA 94305
mworsley@stanford.edu

Michael Johnston
AT&T Research
Florham Park, NJ 07932-1082
johnston@research.att.com

Paulo Blikstein
Stanford University School of Education
Stanford, CA 94305
paulob@stanford.edu

ABSTRACT
In this paper, we present an application framework for enabling education practitioners and researchers to develop interactive, multi-modal applications. These applications can be designed using typical HTML programming, and will enable a larger audience to make applications that incorporate speech recognition, gesture recognition and engagement detection. The application framework uses open-source software and inexpensive hardware that supports both multi-touch and multi-user capabilities.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces**]: User Interfaces – *graphical user interfaces, input devices and strategies, natural language, user-centered design, voice i/o.*

General Terms
Design, Human Factors.

Keywords
Educational technology, multi-modal interfaces, embodied interaction.

1. INTRODUCTION
With the recent release of the Xbox Kinect and Playstation Move, physically interactive gaming has become an increasingly prevalent. Furthermore, children of all ages tend to appreciate these new ways of interacting. As a testament to this, we are seeing a proliferation in the number of applications and research tools that are geared towards fostering increased student interaction through gestures [1], speech [2], haptics [3] and a combination of these techniques [4]. Unfortunately, the software for developing these embodied experiences remains inaccessible to most practitioners and researchers in the education. In order to make these applications, developers must be well versed in traditional programming languages (i.e. C, Java, and Python). The developer also faces a significant challenge in integrating a number of disparate systems in order to achieve multi-modal input. Beyond this, the hardware for these systems tends to be prohibitively expensive when considering anything beyond purchasing a single device. Because of these many constraints, we have endeavored to create an application framework that allows for anyone with basic skills with HTML to develop multi-modal applications that leverage speech recognition, gesture recognition and user engagement (as perceived via face detection). What's more, we have developed this application framework using open-source software and a collection of low-cost hardware that is already available to many children.

This application framework follows our previous work on the topic [5] in which we created a multi-modal application framework for enabling open-microphone interactions. Open-microphone presents a key advantage for full-body interaction since it affords the user to engage in more natural interactions with the system. This implementation makes key additions to our previous framework in that it supports a larger collection of acceptable gestures, does not use any proprietary software, and has been further tailored for ease of application development on the part of teachers, students and education researchers. In addition to this, the current software has been enhanced to allow for more systematic, detailed, "play by play" tracking of student actions as a way for providing rich learning analytics for education practitioners and researchers.

The following sections briefly describe that architecture of the system and give two sample applications as an example of what one can design using the application framework. Finally, we conclude with a brief discussion of the range of applications one could build using this framework.

2. HARDWARE
At the core of this application framework is the Nintendo Wiimote (a 1024 x 768 infrared camera), an infrared source, and a microphone. These devices, when combined with a computer, enable low-cost sensing of gestures and audio capture. In addition to these, however, there are a number of inexpensive additions that enable increased functionality of the application. The additional items include: a large screen display (presumably already available in most education settings), a high resolution web camera and an array microphone. An important feature is that these additions can be selectively incorporated to the system without changes in software.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

IDC 2011, June 20-23, 2011, Ann Arbor, MI, USA.
Copyright 2011 ACM 978-1-4503-0751-2…$10.00.

3. SOFTWARE

As previously noted, this application framework leverages open-source technology to create a cross-platform tool. The selection of freely accessible software libraries was an intentional design decision that would enable the tool to be more easily adopted by schools and teachers that are already facing shrinking budgets. Furthermore, there is nothing about the solution that would preclude a parent or student from setting this up at their residence for a more engaging way of interacting with educational content or media – many families might already own most of the hardware components.

The application framework is built in Python, and includes libraries for capturing and manipulating audio, video and Wiimote data, using GStreamer, Opencv and PyWii, respectively. For speech recognition, we are using PocketSphinx, a highly extensible open-source tool that permits users to supply their own language models and grammars [6]. Finally, the system is tied together using PyQt's Webkit. Webkit allows Python applications to display HTML content, and also features Javascript integration. Furthermore, Webkit allows a Python application to modify the content on a given web page with ease. Thus, by combining the capabilities of Webkit, with speech recognition and gesture recognition software, we are able to make a rich user interface that frees individuals to utilize more natural modalities for interaction.

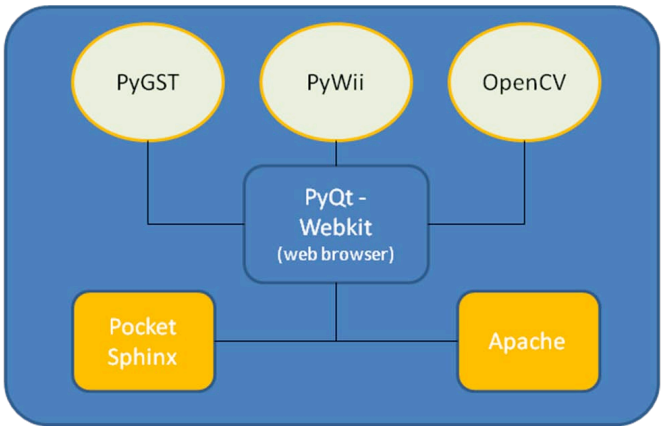

Figure 1. The software architecture consists of three primary Python libraries (PyGST, PyWii and OpenCV), combined with Pocket Sphinx for speech recognition, and Apache for locally hosted web pages.

3.1 How It Works

We have worked to make the application design process easy for user. Given an HTML page that a user *does not own*, the user simply needs to provide our application with the URL and HTML class/label for the entities that they wish to use gestures with. For example, consider the following basic HTML components that consist of a text box and an image hosting at http://justgesturesandspeech.com:

```
<input type="text" name="searchBox" class="input-text" />
<img src="physics.jpg" class="draggable_image"name-
"physics_img"/>
```

If the user wants to use speech to interact with the search box, for example, they would use the following command-line arguments when launching the application:

```
python multimodal_learning.py -url
http://justgesturesandspeech.com –speech_input_name
searchBox
```

If the user then wanted to also add in the ability to drag the image around the screen using gestures, they would launch the application as follows (this assumes that the HTML page already has drag-and-drop as a feature):

```
python multimodal_learning.py -url
http://justgesturesandspeech.com –draggable_class
draggable_image –speech_input_name searchBox
```

In addition to the aforementioned configurations, there are several other ways that the developer can specify how to use, speech, gesture and engagement. These include the ability to turn on and off different modalities, electing to capture and store audio, and the option to explicitly define which gesture to use for single and double mouse clicks (ie. One could associate a 3-second dwell with the mouse double-click action). Finally, the framework contains a number of optional configurations related to speech recognition. These configurations allow developers to specify the acoustic model, language model and dictionary that should be used with their application.

Developers that are designing their own HTML pages can elect to use our default values for the various clicking and selecting parameters, in which case the application incantation only requires the URL for the website which can either be hosted locally or externally:

```
python multimodal_learning.py -url
http://justgesturesandspeech.com
```

4. SAMPLE APPLICATIONS

In what follows, we will describe two sample applications built with this framework, and explain how teachers, practitioners and researchers can author multi-modal applications. In particular, we used open-source, Javascript-based, physics engines, to build these applications. This was done to demonstrate the low-threshold requirements of the framework. We believe that there is great potential in a tool that acts as 'glue' for the myriad online resources currently available for teachers, allowing for easy integration of gesture and speech.

4.1 G-Forces

G-Forces (gravitational and gesture forces) is a sample application for exploring Newtonian physics using gestures and speech. The application extends an existing web-based application that is based on a box2d.js Javascript physics engine. Our implementation extends the existing capabilities of the application to contain more explicit learning components. These learning components provide users with the opportunity to explore more structured learning tasks (eg. causing a certain ball to move in a specific trajectory, or generating a series of collisions in order to achieve a goal). We also add scaffolding that allows students to gleam more knowledge from the unstructured, exploratory learning scenario. For example, when trying to accomplish one of the tasks, students will be able to use speech to ask for hints or short tutorials on words and concepts in physics. These questions will also direct them to web-based resources that describe physics vocabulary and the underlying concepts that are relevant to understanding the theories of Newtonian physics.

On the analytics side, G-FORCES stores the activity of each user, through the application framework, as a way for recognizing the

contributions that each individual makes. This approach is closely tied to idea of Learning Analytics – which uses multi-modal data mining about learner behaviors, speech and sentiment to better understand learning processes [7][8]. Furthermore this close integration of analytics and practice helps practitioners and researchers recognize areas where students may have deep rooted misconceptions that need to be addressed.

4.2 Building Bridges

Building Bridges asks one to four students to explore concepts in static mechanics by simulating the process of performing mechanical load testing on different types of bridges. Similar to G-FORCES, students will have the opportunity to use gestures for dragging and dropping, rotating, scrolling and selecting. As a part of this application, students observe how different configurations of bridge components and different types of materials can impact the structural stability of the bridge. They can also be able to "shake" their bridges by gesturing (quickly moving their infrared device back and forth).

The application involves students being presented with a series of bridges that have different configurations. The student will put each bridge through load testing, but will be required to predict which bridge will perform the best, with the option of updating their selection throughout the activity.

As groups of students discuss and interact with the application, user activity is monitored, so that this tool can provide in-depth feedback on each student's level of engagement.

5. DISCUSSION

This framework is fundamentally designed to offer a low threshold to entry, and a high ceiling for what developers can do through it. At the most basic level, the framework can be used for capturing students' speech or engagement as they interact with an existing web page. Such an implementation would require no programming on the part of the practitioner or researcher beyond knowing the URL for the webpage and calling the application framework with the default incantation. On the other hand, this framework could be used to develop a tool similar to the Mathematical Imagery Trainer ("M.I.T") [1]. M.I.T. is a learning tool that enables students to learn math concepts, namely proportions, through a combination of spatial-temporal actions and visual feedback. Employing this technique of embodied interaction can assist in helping students to construct meaning for otherwise hard to grasp concepts [9]. Our framework looks to give a larger population of individuals the tools to design embodied-interaction applications, while also allowing for some of the affordances obtained through the use of multi-modal interfaces [10].

6. CONCLUSION

This paper describes a newly developed application framework to more easily create rich gesture and speech based interfaces that promote multi-modal, embodied interactions. As such, this framework looks to support the proliferation of interactive tools that are inexpensive, accessible - for people of all levels of programming ability - and readily modifiable, i.e. open-source, as such tools can help foster the expansion of high quality education.

7. REFERENCES

[1] Howison, M., Trninic, D., Reinholz, D., & Abrahamson, D. 2011. The Mathematical Imagery Trainer: from embodied interaction to conceptual learning. In G. Fitzpatrick & C. Gutwin (Eds.), *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*.

[2] Darves, C. & Oviatt, S. 2004. Talking to digital fish: Designing effective conversational interfaces for educational software, in From Brows to Trust: Evaluating Embodied Conversational Agents, Kluwer: Dordrecht, 2004, 271-292.

[3] McKnight, L. and Fitton, D. 2010. Touch-screen technology for children: giving the right instructions and getting the right responses. In: *Proceedings of ACM IDC10 Interaction Design and Children 2010.* pp. 238-241

[4] Rosenbaum, E. and Silver, J. 2010. Singing Fingers: fingerpainting with sound. In: *Proceedings of ACM IDC10 Interaction Design and Children 2010*. pp. 308-310.

[5] Worsley, M. and Johnston, M. 2010. Multimodal Interactive Spaces: MagicTV and MagicMAP. In: *Proceedings for the 2010 IEEE Workshop on Spoken Language Technology* (SLT), December 2010. pp. 149-150

[6] Huggins-Daines, D, Kumar, M., Chan, A., Black, A.W., Ravishankar, M. and Rudnicky, A.I. 2006. PocketSphinx: A free, real-time continuous speech recognition system for hand-held devices. In: *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing* (ICASSP), May 2006, pp. 185–188.

[7] Worsley, M. and Blikstein, P. 2011. Towards the Development of Learning Analytics: Student Speech as an Automatic and Natural Form of Assessment. Paper Presented At the 2011 *Annual Meeting of the American Education Research Association* (AERA), April, 2011.

[8] Blikstein, P and Worsley, M. 2011. Detecting Learning Analytics: Assessing Constructionist Learning Using Machine Learning. Paper Presented At the 2011 *Annual Meeting of the American Education Research Association* (AERA), April, 2011.

[9] Abrahamson, D. 2009. Embodied design: Constructing means for constructing meaning. Educational Studies in Mathematics 70, 1, 27-47.

[10] Oviatt, S. 2008. Multimodal interfaces, Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, 2nd ed, (ed. by A. Sears & J. Jacko), LEA: Mahwah, N. J., 2008, 413-432.