Learning Symbolic Hand and Finger Motion Gestures

Tarif Haque and Chris Chockley

Advisor: Jeff Gray

Department of Computer Science – The University of Alabama

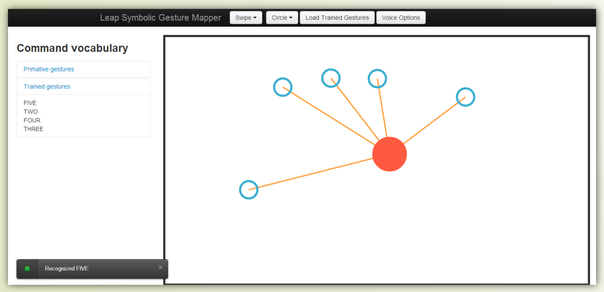
Abstract

Hand and finger motion gestures have not yet been widely integrated into human-computer interfaces. Detecting symbolic gestures, or gestures within a defined interval and mapped to a specific command, can be trivial to complex, depending on the nature of the gesture. Symbolic gesture recognition techniques were investigated with respect to the Leap Motion Controller, a hand and finger motion tracking sensor. To evaluate the performance of existing gesture recognition tools and explore the implications of symbolic gesture recognition, we developed a gesture mapping interface that is able to recognize gestures learned using exemplar-based machine learning approaches and compositions of primitive gestures that the Leap recognizes natively. To illustrate an application of symbolic gesture recognition, the interface provides a means to map learned symbolic gestures to speech; such a system could potentially be used to teach the vision impaired to sign.

Symbolic Gestures

In this project, we think about gestures as hand and finger motions with a defined start and end point. We call these gestures **symbolic gestures**. We distinguish between manipulations and gestures: a gesture does not seek to manipulate or move a virtual object in space; for our purposes, it has a single meaning and usually expresses a command. By nature, gestures have no inherent meaning; they have not been standardized or well-established across technologies.

Software Prototype



- Browser-based web application implemented using HTML5 and JavaScript Tools.
- Allows gestures trained via the *LeapTrainer* framework to be loaded and recognized by the interface.
- A set of primitive gestures which need not be trained are provided by the mapper.
- The system voices gestures upon recognition.

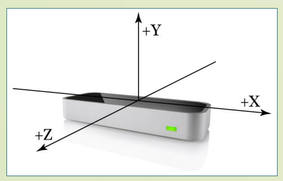


Figure 1. The Leap reports hand motion data in the context of a 3D coordinate system.

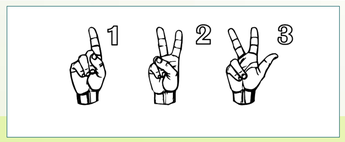
Primitive Gestures

If we wish to integrate gesture recognition into an interface, we must first determine which gestures our platform is able to natively recognize. We call these gestures **primitive gestures**. The LEAP recognizes these gestures out-of-the box:

1. Swipe – A linear movement of finger or hand
2. Circle – A single finger tracing a circle
3. Key Tap - A tapping movement by a finger as if tapping the keyboard.
4. Screen Tap - A tapping movement by a finger as if tapping a keyboard key.

Figure 2. The Leap natively recognizes the swipe gesture, a swiping motion of a finger or hand, which is provided by the LEAP Motion API.

Recognition Model for Trained Gestures



- If our hand stays still for some interval, **pose recognition** is triggered.
- **Gesture recognition** is triggered when the hand moves beyond a velocity threshold.

Thus, the interface will only attempt recognition anytime the user's (1) hand is still or (2) moves quickly. The interface then attempts to classify the gesture or pose as one of the learned gestures or poses.

Building a Gesture Vocabulary

Train Gestures Based on Examples
LeapTrainer Framework

Primitive Gesture Detection
Swipe / Circle
Key Tap / Screen Tap

Implementing Symbolic Gesture Recognition

- Mutual Exclusion - Only one gesture may be recognized at a time.
- Inject event listener for each trained gesture into the interface.
- Continuous gestures must be reduced to discrete gestures so they have a defined start and end point.

Symbolic Gesture Command Vocabulary

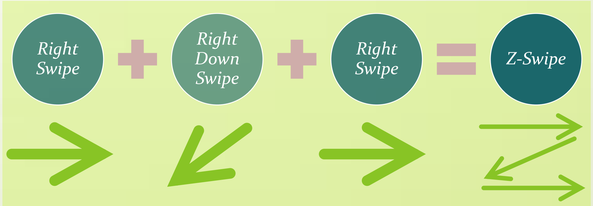
Exemplar-based gesture training

LeapTrainer is an open-source gesture and pose learning framework for the LEAP Motion distributed under the MIT License. It is intended to be used by developers who wish to integrate custom gestures into applications or explore gesture capture and recognition algorithms. We use this framework to train gestures based on example data and then integrate the learned gestures into the mapper.

Figure 3. Using *LeapTrainer* to train custom gestures.

Detecting Sequences of Gestures

We include a gesture vocabulary of eight directional swipes in the mapper. We allow the user to define more complex gestures by specifying a sequence of gestures. For example, a *right-swipe*, followed by a *right-down* and *right-swipe*, may be interpreted as a *Z-swipe*.



Sign to Speech

Vision impaired users may be challenged in reading material presented on a monitor. As input, they use standard keyboards or braille input devices. Speech output systems are commonly used to read screen text vision impaired users. Such systems motivated the implementation of a sign-to-speech system, which maps the interface's learned gestures to speech feedback. A sign-to-speech interface for vision impaired users could potentially be used to teach how to sign and afford a more intuitive way to interact with a computer.

Conclusion

- Existing tools for symbolic gesture recognition are limited in their gesture detection capability.
- Large gesture vocabularies can make gesture detection error-prone.
- Detecting complex gestures is far from trivial and generally requires training data.

Future Work

- Study symbolic gesture recognition systems using other motion tracking devices.
- Survey programmatic models used for gesture detection.
- Improve on exemplar-based methods for gesture detection.
- Design a system that composes complex gestures using low-level primitive gestures.
- Test system recognition performance formally using American Sign Language symbols.

Acknowledgements

Research supported by University of Alabama Department of Computer Science, The National Science Foundation, and UA Computer-Based Honors Research Program.

NSF

THE UNIVERSITY OF ALABAMA