Future User Interface

Dandelion

Adrian Hänni,
Padideh Pezeshki,

Introduction

An obvious property of multimodal systems, is that it involves simultaneous stream of various signal types continuously generated by users. The overlapping of these signals is a rare issue requiring special attention. However, it is a common experience that users usually tend to introduce small delays of 1-4 seconds between two modal signals, either from user or from the machine.

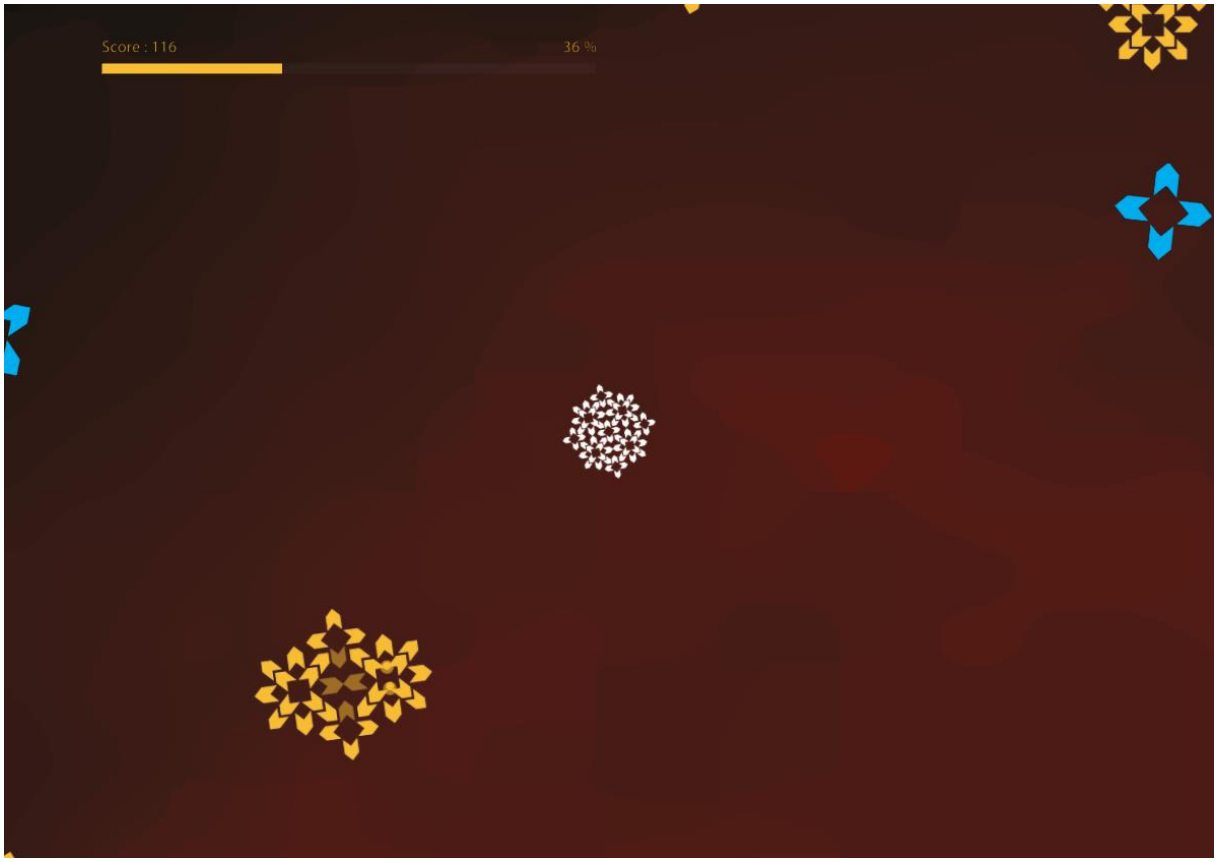
Multimodal systems rely on multimodal interfaces to collect and present information from and to the users. Such interfaces include speech, handwriting, touch, gesture, and other natural forms of communicative movements like shrugging of the one's shoulders. In any case, input and output modalities should be generated in coordination, when using a common multimodal interfacing backbone. This class of interfaces are built around the new concept of recognizing and synthesizing natural human language, and behaviour to create a high level of user friendly natural interaction between humans and machines [S.Oviatt et al..,2002].

The system-centric definition of multimodality by Nigay and Coutaz [1993] states that: "Multimodality is the capacity of the system to communicate with a user along different types of communication channels and to extract and convey meaning automatically." According to this definition, in a consistent multimodal system various types of data are combined/fused together in real-time, to generate a bi-directional communication route between a machine and its user. Such interface should apparently satisfy some predetermined timing constraints.

Contents

1 Dandelion

Dandelion is considered to be a prototype of an exergame aimed for people suffering Parkinson disease. The Dandelion game uses voice and gesture input for remote control. The goal of the game is to collect blue flowers in a two-dimensional space to increase the score while avoiding so-called thorn flowers shown in yellow color. A yellow progress bar on the top left indicates the remaining time.



1.1 Game Mechanics

The play time of one round is set to 60 seconds. Collecting of one flower adds 10 points to the players score and increases the play time by 2 seconds. An additional bonus score is increased each time by 1 point and is additionally added. The collision with a thorn flower subtracts 10 points from the score and decreases the play time by 5 seconds. Such a collision also resets the additional bonus score to 0.

2 Modalities

2.1 Speech

Although speech is considered the primary input/output modality in human-to-human communication, yet it may not be the default option in human-to-machine communication schemes. Speech is the modality with some specifications and constraints that should be taken into account

when used as an interface modality. Although the users can easily adapt, learn and enjoy voice controls, yet they should obey some guidelines for best performance. They should follow good spoken language using short sentences with a prosody clearly indicating end of words, to make it clear enough for machine to confidentially understand. Other technical limitations include ambient noise, disturbance, and others. Voice modality is usually helpful when the user's hands and/or eyes are busy with competing tasks, or when only a limited area is available on the keyboard and/or screen, for users with physical disability, and when the voice is the subject matter being used (e.g. reading, foreign language training, and more). Additionally, some users may prefer voice input for a variety of reasons such as its easy and straightforward interaction.

2.2 Gesture

Gesture is another naturally used modality. A basic version of gesture modality is used in computer mouse and trackpads. In fact, the more advanced gesture modalities are the natural expansion of such gestures. While pointing gestures are directed and reliable, sign gestures are synthetic, more complex and harder to be recognized by a machine. In this project, we employ MYO gesture armband to capture users' control gestures.

2.3 Game Controls

In our implementation, speech has a supportive role for those with excessive disability preventing from smooth exergaming. In such cases, some of the harder to perform gesture controls can be replaced by speech orders to enable them engage with the game. On the other hand, the use of gestures intends to animate patients suffering Parkinson disease to perform training movements.

2.3.1 Types of Interfaces

There are two types of multimodal interfaces implemented for the game. The first one will use MYO gesture to trigger the player's movement in the game and voice input to set the level of speed for the character, the Dandelion. It is referred as interface A. The second interface termed B will do the opposite using gesture to set the level of speed and voice for the movements. Both interfaces use voice and/or electromyographic gestures for the game's menu interaction.

2.3.2 Supported Commands

Type	Interface(s)	Application State(s)	Input	Consequence
Speech	A, B	Menu Screen	Start	Starts a new round of the game
Speech	A, B	Menu Screen	Help	Opens the Help Screen
Speech	A, B	Menu Screen	Score	Opens the Score Screen
Speech	A, B	Menu Screen	Exit	Terminates the application
Speech	A	Game Screen	Faster	Increases the speed of the player
Speech	A	Game Screen	Slower	Decreases the speed of the player
Speech	B	Game Screen	Up / Down / Left / Right	Sets the moving direction of the player

Type	Interface(s)	Application State(s)	Input	Consequence
Speech	A	Game Screen	Speed <Level>	Enables to set a speed level directly
Speech	A	Game Screen	<Speed> Fast	Sets the speed level to level Fast
Speech	A	Game Screen	<Speed> Medium	Sets the speed level to level Medium
Speech	A	Game Screen	<Speed> Slow	Sets the speed level to level Slow
Speech	A, B	Startup Screens, Score Screen, Help Screen, Game Over Screen	Next	Forwards to the next screen
Gesture	A, B	All screens	Double Tap <Pose>	Unlocks the MYO for further EMG gestures
Gesture	A, B	Menu Screen	<Double Tap> Wave In	Moves selection to the left
Gesture	A, B	Menu Screen	<Double Tap> Wave Out	Moves selection to the right
Gesture	A, B	Menu Screen	<Double Tap> Fist	Selects the menu element
Gesture	A, B	Game Screen	<Double Tap> Finger Spread	Sets the current orientation as rest orientation
Gesture	A, B	Startup Screens, Score Screen, Help Screen, Game Over Screen	<Double Tap> Fist	Forwards to the next screen
Gesture	A	Game Screen	Move arm to right / left / up / down	Sets the moving direction of the player
Gesture	B	Game Screen	Move arm up / down	Increases / Decreases speed level

2.4 CASE and CARE

In this project we didn't use fusion of modalities. The main reason behind this is that we didn't see a benefit in terms of usability for users. As this prototype game is aimed to be used by patients suffering Parkinson disease, a requirement to use speech as well as gesture to trigger one action may decrease user experience depending on the health condition. Because Parkinson is a degenerative disorder of the central nervous system mainly affecting the motor system some patients may not be able to perform certain gestures. The focus on providing alternatives, respectively equivalent inputs of different modalities, as provided by the interfaces can be considered to be beneficial to such users. In respect to the CASE and CARE models, the modalities can be classified as follows:

2.4.1 CASE

2.4.1.1 Exclusive

All commands triggered by modalities are exclusive and happen one after another in the order they were performed.

2.4.2 CARE

2.4.2.1 Assignment

Only FINGER SPREAD pose can set the MYO rest position that is needed to calculate the relative orientation drift from it in order to determine where the users arm is pointing to. Speed and direction for the players' character control can be performed only by one modality depending on the interface currently used.

2.4.2.2 Equivalence

The user has the choice to use either speech and/or gesture in the Menu screens.

3 Architecture

Several hardware and software components have to fit together that are introduced in upcoming sections.

3.1 Hardware

3.1.1 MYO Gesture Recognition Device

The MYO gesture armband developed by Thalmic Labs introduces a new way of interacting with our everyday computing environment by fitting around the upper part of forearm and detecting slight muscle movements through capturing electromyographic (EMG) data, acceleration as well as rotations. MYO armband consists of eight EMG-sensing modules, and is strapped onto the widest part of forearm. It senses the electrical impulses as one moves his/her hand, and translates the captured EMG data to a predefined set of hand gestures, to enable users to interact with a wide range of electronic devices or software applications in real time. The supported gestures are shown below:

MYO provides two different types of data streams. Spatial data indicates how the armband moves in 3D in terms of roll, pitch, and yaw. Gestural data is the second type of information generated by this armband and tells us what the user is doing with his/her hand. Spatial data encompasses angular velocity from gyroscope and acceleration data from accelerometer. Gestural data on the other hand, is provided by the proprietary EMG muscle activity sensors. Currently, the MYO armband is designed to detect 5 unique poses, but its future versions as promised may detect extra hand poses as well.

Further, it provides accelerometer data in vectors of units of g, gyroscope data in vectors of degrees per sec, as well as orientation data of the current roll, pitch and yaw values provided as quaternion. Quaternions are members of a non-commutative division algebra that can represent orientation or rotation.

In order to communicate with the MYO device, the MYO Connect application must be installed on the operating system. This application acts as link between the MYO device connected by a proprietary Bluetooth 4.0 Low Energy adapter and the applications that use it.

3.1.2 Microphone

For speech recognition a simple microphone that supports PCM_SIGNED 16000.0 Hz, 16 bit, mono, single channel input is used.

3.2 Software

3.2.1 Sphinx4

Sphinx4 is a speech recognition library using Hidden Markov Model (HMM) written in JAVA. It provides an API to convert the speech recordings into text using acoustic models. Beside speech recognition Sphinx4 is able to identify speakers, adapt models, perform transcription and more.

3.2.1.1 Dictionary

Sphinx requires in its configuration a dictionary containing grapheme to phoneme (G2P) conversions of words that should be recognized. G2P uses rules to generate a pronunciation for a word in a textual description. The resulting dictionary can therefore be considered as a pronunciation dictionary. Consequently the resulting dictionary consists of entries of the form shown below:

help EH L P

help(2) HH EH L P

3.2.1.2 Acoustic Model

Standard acoustic models are provided by the Sphinx data package but can also be manually created. An acoustic model helps to adapt to a particular recording environment, audio transmission channel and accent of users. Dandelion currently uses “en-us” acoustic model, as it provides a good recognition accuracy.

3.2.1.3 Grammar

To describe the language that should be recognized, a grammar has to be provided. In Sphinx, grammars are created with JSpeech Grammar Format (JSGF) format. While the dictionary provides the phoneme descriptions, the grammar contains the set of words that actually can be recognized.

3.2.2 Java Library for MYO

This application uses the Java language bindings library for the MYO Developer Kit originally created by Nicolas Stuart. The Java Native Interface (JNI) libraries are provided for OSX as well as Windows for 32 and 64-Bit systems.

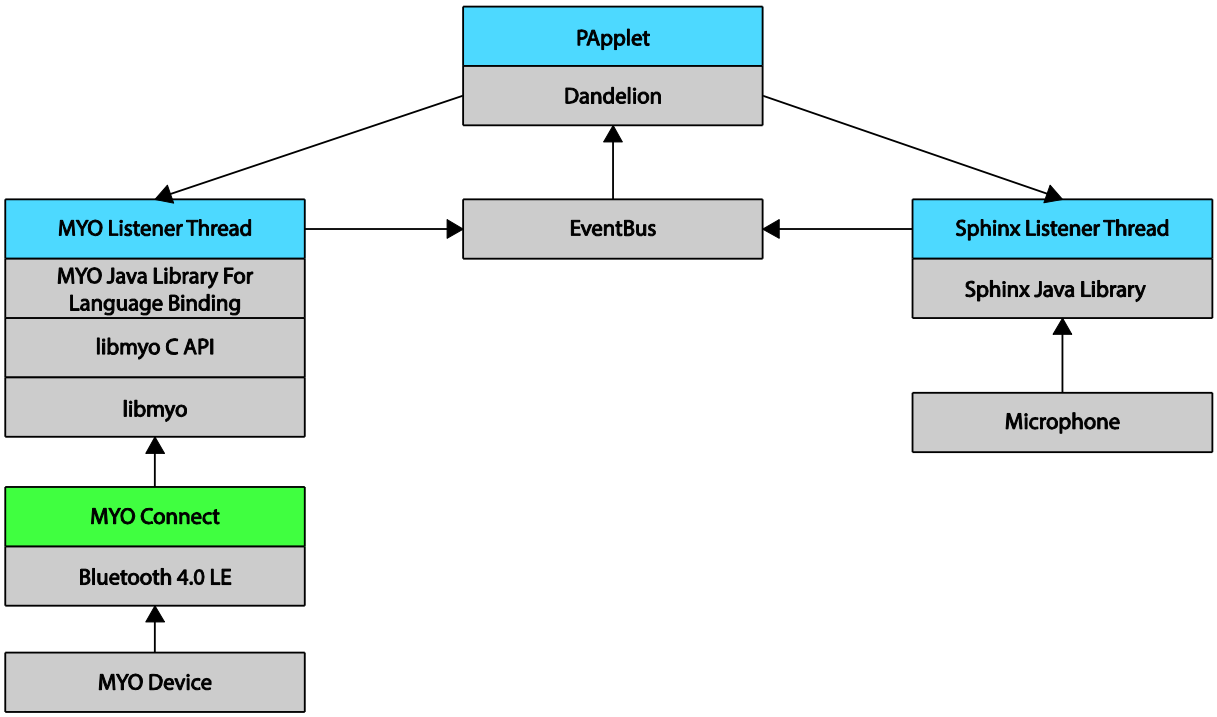
3.2.3 SimpleEventBus

Java SimpleEventBus is a library that provides subscribe and publish type architecture. The event bus helps to simplify event communication between multiple software components such as threads and promotes a more stable and simplistic decoupled interface between them. In general, an event bus is and software architecture, where objects can subscribe to receive certain events from the bus. When an event is published to the event bus, it will be propagated to any subscriber of this event bus. In

consequence, each component is able to couple solely to the event bus itself and not directly with each other.

3.3 Application Stack

Dandelion makes use of multiple threads. The main threat, executing the game logic in a PApplet, a Java Applet providing the functionalities of Processing programming language, the MYO Listener thread and Sphinx Listener thread are marked in blue color in the next figure. MYO Connect is a stand-alone application, which must be installed on the operating system that executes Dandelion, is marked in green color:



Once the MYO Listener thread is started, it will include the JNI library for the MYO device. This library is a programming framework that enables Java code running in a Java Virtual Machine (JVM) to call and be called by the MYO Connect application. The libmyo library is the core of the MYO Software Development Kit (SDK). The functionality of libmyo is exposed trough a plain C API. Applications normally do not interact directly with the C API but use language binding techniques corresponding to the programming language used by the application. This enables a wide variety of software written in different programming languages to interact with the MYO device.

The Sphinx Listener thread is responsible to gather voice input and to provide a hypothesis of recognized words.

Both, the MYO and Sphinx Listener threads, constantly sense information provided by MYO device or, respectively, microphone. If Sphinx recognizes words contained in the grammar it will trigger an interface event over the event bus. If a defined gesture is provided by the MYO device, it will also trigger an interface event. On the other hand, the main thread is a subscriber of the event bus and will therefore be notified about the event. Depending on the application state it will translate the event into an action in the game.

4 Evaluation

In order to evaluate the two different interfaces, we let the users play two rounds on each multimodal interface. To compare, we will take the average of the game score to figure out which interface performs better in terms of usability. Indeed there is a certain variation as the game's collectibles and thorn flowers are randomly generated in the playing area.

The game was evaluated with users that do not suffer Parkinson disease.

4.1 Quantitative Evaluation

	Interface A: Average Score	Interface B: Average Score
User 1	115	78.5
User 2	404.5	88
User 3	51.5	227
User 4	174	153
User 5	431.5	178.5
User 6 *)	-	-

*): User 6 had a thick forearm and was not able to perform the calibration gesture, the finger spreading after the unlock gesture, for the MYO device. After several attempts we decided to abort.

From the average score results we can see that with Interface A more points were collected. But we have to pay attention to the game mechanics. As the objects on the playing area are randomly set, some users found an area with a high amount of collectibles that give points, which they could harvest, and some did not. In consequence, luck is also a playing factor. The higher overall score using Interface A, can be traced back to the more responsive recognition when using MYO for the direction in contrast to the voice.

4.2 Qualitative Evaluation

Most users are first uncomfortable with the MYO device. Main reasons are that it sits tight on the forearm and they are unfamiliar with it. We observed in general that users could control the direction of the player character better with the MYO, but they tend to move the body too instead of only the arm. For all users the MYO hand gestures were difficult to perform and they tend to exaggerate the hand gesture using too much force. From our own experience, we witnessed the same at the beginning. The usage of hand gestures requires training. One has to condition himself to relax the muscles when wearing the MYO and only use a certain amount of force when performing a hand gesture.

After the test runs, we asked the users questions about how well they liked the interaction with the different modalities.

In general, they preferred the voice for the menu screen interaction but the MYO for controlling the direction of the player character. Voice has a too large delay to quickly react on the game screen, forcing the user to give the command before he actually wants to turn. For some users voice recognition performed poorly, so that they had to give the same voice command several times.

5 Conclusion

Our goal when starting the project was to create an exergame that could bring some benefit for Parkinson patients. But the evaluation with healthy users showed us that such an application need more improvements. The usage of hand gestures requires training and is maybe too sensitive for Patients that suffer disabilities on their motoric system. The MYO itself is currently in BETA state and the developer plans to constantly improve it. Currently it is not very user friendly. The calibration profile is saved on the MYO device itself and almost every time when it is worn again the user must set up a new profile, if it is not placed at the exact same position, to achieve a good recognition. Further, an additional warm-up phase when placing the MYO on the forearm prevents that the user can directly start with the interaction.
Voice recognition performed well overall but requires large CPU and RAM usage causing a noticeable delay on the game screen. For future work, a voice command that can trigger the MYO calibration would be beneficial, and a modified acoustic model could help to improve the recognition success rate.