Multimodal Interaction Framework Based on Firebase Real-Time Database

Youssef Guedira, José Rouillard

Multimodal Interaction Framework based on Firebase Real-Time Database

Youssef Guedira and José Rouillard

Univ. Lille, CNRS, Centrale Lille, UMR 9189 – CRIStAL
Centre de Recherche en Informatique Signal et Automatique de Lille
F–59000, Lille France
youssef.guedira@univ-lille.fr, jose.rouillard@univ-lille.fr

Abstract. Relying on one technology with a single interaction modality may benefit some users but would certainly exclude a lot more if they have impedances to use that modality. The solution then becomes the inclusion of multiple modalities in the initial design of the interactive system making it more adaptable to the needs of many more users. Including many modalities can rapidly increase the number of interaction objects that need to receive the stream of user commands. This is especially true if the user needs to interact with multiple artifacts in a home automation environment. In this paper, we present the general architecture of an ongoing project for multimodal home automation system. This system relies on a web based database called Firebase for the exchange of user input and the issuing of commands to the multiple artifacts. The user input is acquired using a smartphone and a webcam equipped computer. They capture the user's tactile input, vocal phrases, eye gaze as well as head pose features like tilt and face direction. We were able to achieve a reliable data transfer between the database and the different input acquisition interface. As a first step in the prototyping of the system, we were able to control two separate game interfaces developed using Unity3D software.

Keywords: Quality of life technologies, Multimodal, Interaction design, disability, special needs

1 Introduction

With the advancements in the development of smart homes and connected environments, people nowadays are able to control their home environment through intelligent and adaptive interfaces. In fact, these technologies greatly benefit users with special needs especially. This is especially the case when moving around the house, reaching for artifacts and manipulating them becomes a heavy and painful task, sometimes impossible without external assistance. However, with the varying profiles of this category of users, inclusion matters come into play. A system relying solely on voice command would exclude any user with speech impediments. The solution is to adopt a multimodal interaction approach from the beginning stages of the system design.

In this paper, we present our ongoing work on the notion of multimodal interaction in the context of Human-Computer Interaction to improve quality of life. Since Richard Bolt and his famous “Put that there” [1], it is well known that multimodal interaction can provide more natural and easy to use interfaces. In our research, we focus more on the use of multimodal interfaces to interact with a connected environment especially for user with special needs. In fact, our discussions with patients and healthcare professionals in care-centers revealed the need for such environment control. On one hand, the connected environment allows the user to interact with multiple objects without needing to move around the space and with minimal to no need for physically reaching for the objects. This is especially beneficial when the person suffers from reduced mobility. On the other hand, the use of multiple modalities for interaction would allow users having various levels of ability with regards to sensory-motor functions to still benefit from the use of such technologies.

Elouali et al. [14] suggested benefiting from this mobile technology for multimodal interaction. More specifically, smartphones are nowadays equipped with multiple sensors (gyroscope, compass, microphone, camera…) which can be used to acquire a large spectrum of user input and utilizing a various interaction modalities. In fact, the work of Guedira et al. [12] introduces power wheelchair steering on a smartphone application. Combining these ideas can give rise to a holistic interactive system that utilizes the smartphone technology to both interact with the environment and drive the wheelchair.

In our work, we particularly focus on users suffering from neuromuscular diseases. For instance, in our team, we are working on Hybrid BCI (Brain-Computer Interfaces) for Duchenne Muscular Dystrophy (DMD). DMD is a severe pathology of skeletal musculature. This genetic disorder causes an absence of dystrophin, a protein that supports muscle strength and muscle fibers cohesion, which leads to progressive muscle degeneration and weakness [2]. Hybrid BCI means that various other ways to communicate are used beside EEG (Electroencephalogram). As we consider multiple user profiles and various interaction possibilities, one of the challenges to the system design is that users can change the way they interact when they become tired, for example, switching from a direct muscular interaction to another one (voice, gaze, EEG…). Interfaces that handle these changes may not be very easy to conceive, implement and manage. One of the challenging aspects is the exchange of data bits between the multiple input acquisition interfaces, the central interaction engine and the output devices. Jacket el al. [13] proposed an architecture for an ambient assisted living framework relying on a Zigbee protocol to relay information between multiple components of the interactive system. Nowadays, various tools allow us to store and retrieve data in the cloud instantly. In this paper, we explore the use of a lighter weight communication through a Firebase web database

The rest of the article is organized as follow: in section 2 we give the architecture of the proposed multimodal system. We detail the different modalities that are used to get user input. After that, we detail how the input is centralized using the web based database Firebase. We then give an overview from the literature on how the input coming from different modalities can be leveraged to then send it as a user command to the object of interaction. In section 3, we illustrate via two computer simulations

how this data exchange was achieved between two or more separate devices. Then, in section 4, we give a brief overview of the upcoming step in our design which is about an experiment using a Wizard of Oz technique. We conclude in section 5 with a summary of the work and a brief outline of this next experiment.

2 Architecture

For better efficiency, we have chosen a modular architecture. Multiple interfaces acquire user input, each interface capturing one or more interaction modality. This would allow each user to select the input interface(s) that best suit their needs without affecting the rest of the interaction chain. The different input signals are then centralized on a web based database and treated to synthesize one user command. The system then reacts accordingly. Throughout the execution of the intended task, the system can prompt messages to the user either asking for more clarification or informing him/her on the progression of the task. The prompted messages themselves can be conveyed through different channels in order to accommodate for the user's needs/preferences without being scarce or cumbersome. The schematic in Figure 1 gives a large scale overview of the architecture. To the latter is added a Wizard of Oz [9] (see section 4) which interacts, through the same database, with the user's home automation environment.

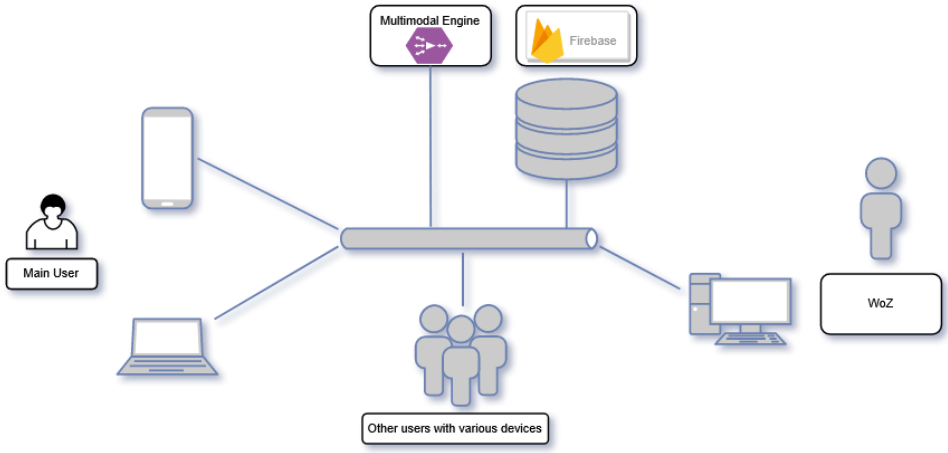

Fig. 1. Architecture of our multimodal interaction framework based on Firebase Realtime Database

In this section we give a detailed description of the system's architecture. For each module/sub-module, we will give a technological solution that can be used in the system. The specific solutions given in this section can be thought of as lightweight especially for prototyping but can also be used in real life deployment of the system. Other solutions can be used as long as they provide similar functionalities.

2.1 User input acquisition

When designing a system to improve the quality of life of users with special needs, one should keep in mind that different users may have varying levels of ability on each basic function. If we take neuromuscular diseases [3] as an example, the spectrum of manifestations can be very large: some patients may only have mobility problems while others may not be able to speak or even breathe naturally. In [4], authors discuss multiple manifestations of these diseases and how they can impact the design of a wheelchair steering system. The wide spectrum of deficiencies caused by such disease makes it hard to encompass a large spectrum of users with a single modality. In order to level up the plane field in terms of interactive system design, we need to think about including diverse input interfaces that can make use of each person's residual abilities to allow them to accomplish the needed tasks. For this reason, the input interface presented in this paper consists of several input acquisition modules, each one captures user actions through a particular channel and makes use of a specific modality.

Tactile Interaction. The user can issue commands by touching a tactile interface. This is performed via two different paradigms: continuous-input tactile pad and discrete tactile buttons.

These two paradigms themselves can either be physical (physical buttons) or virtual (tactile graphical interface). In our implementation, we have chosen to use a mobile application that contains both a virtual tactile pad and discreet buttons (Figure 2 top).

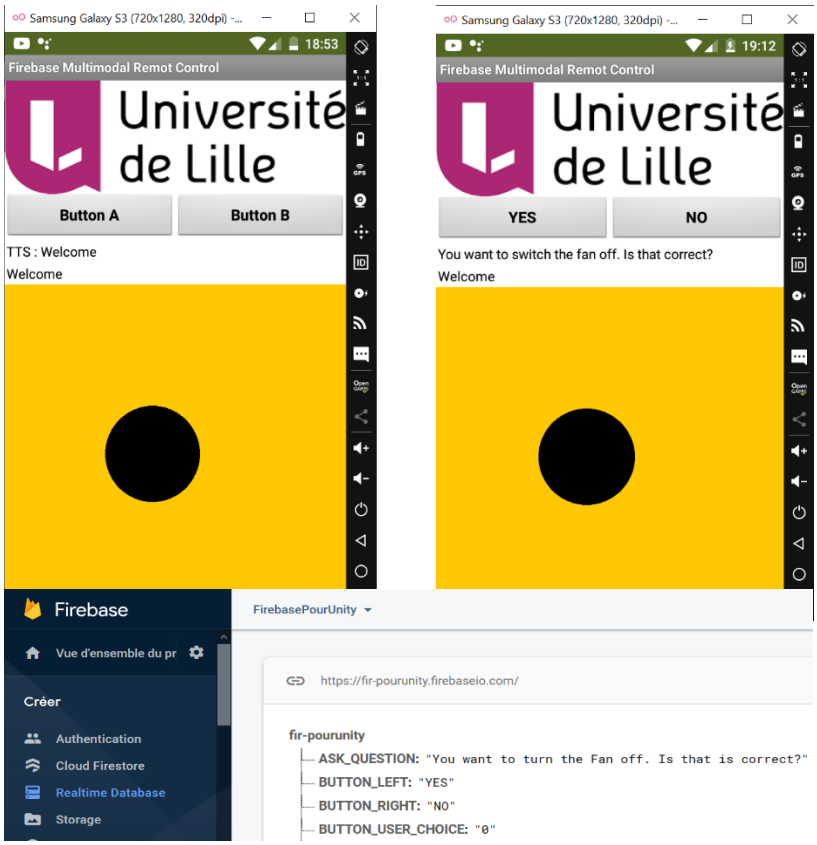

Fig. 2. A screenshot of the tactile control interface (top) and how it interacts with the Firebase web database. The tactile pad is represented by the orange rectangle. The user's touch is represented by the black circle. TTS stands for Text To Speech

The virtual pad (orange rectangle in Figure 2) registers the users' touch (represented using the large black circle in Figure 2) which allows them to perform free gestures. The discreet buttons allow for the selection between different choices (for example between tasks).

The two paradigms can be used either as alternatives or as complements to each other. When used as complements to each other, the user can make a gesture on the pad and either confirm or undo the gesture by tapping on one of two buttons. When used as alternatives to one another (in a two-choice selection for example) the user can make a gesture to the right on the pad (equivalent to tapping the right button) and make a gesture to the left (equivalent to tapping the left button) and vice versa.

Speech recognition. The user can utter a sentence or a phrase to issue a command to the system. In our implementation of the multimodal system, we have chosen to make use of the Google API for speech recognition[1].

The utterance can initiate an interaction, specify it or halt it. The significance of the utterance can be absolute or contextual. For example: if the user needs to close the windows for the bedroom and the guest room. The user can say “close the windows of the bedroom”. In this case, the utterance can be interpreted at face value and the system will close the windows of the bedroom. To close the windows of the guestroom, the user can either substitute “bedroom” by “guestroom” in the previous utterance or utter “and do the same thing for the guestroom”. The system should understand from the context that the overall interaction is not done yet and that “the same thing” is a contextual substitute for “close windows”.

Last but not least, the utterance can specify the interaction. We can take the previous example. The user can utter “close the windows”. The system recognizing multiple rooms, can ask for specification of the command. The user can then specify “bedroom” or “guestroom”.

Head movement command. By the means of a normal web camera, the system captures the head movement of the user. The system detects head orientation and tilt using the face detection feature of the Dlib and OpenCV libraries. For the first iteration, we decided to limit the user's actions to a left/center/right face orientation (Figure 3) and a left/straight/right head tilt.

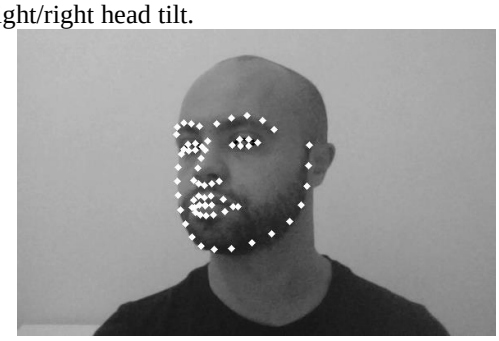

Fig. 3. Illustration of head movement (face looking to the left) detection using OpenCV and Dlib libraries

Eye gaze. The user can make eye movements in order to interact with the system. For prototyping purposes in our implementation of the multimodal system, the eye gaze is captured via video camera relying on OpenCV library for face feature detection (Figure 4). The reason behind it is that we did not opt for a precise target acquisition using this modality. It comes as a picking mechanism for a limited number of discrete choices presented on a screen in front of the user.

[1]More information on the API available on the official page: https://cloud.google.com/speech-to-text

The risk while using eye gaze to interaction is that this channel can be the main tool for exploring the environment. Thus, the user might be simply looking at an object in the environment while the system may register this action as an instruction bit. For this reason, the eye gaze detection interface requires a confirmation via a blink in order to register the gaze action. The user's gaze is detected as either left, or right as a mechanism of choosing between two different options each presented on the left or the right side of the screen. In order to send a gaze instruction the user needs to gaze in the intended direction then blink so that the gaze action is registered.

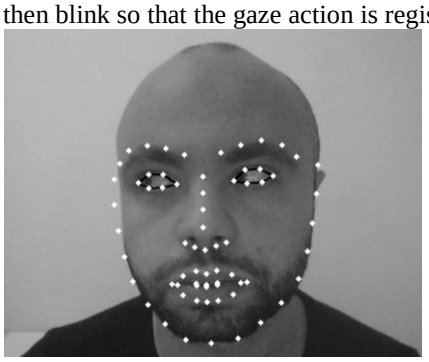

Fig. 4. Illustration of eye gaze detection (eyes looking to the right) using OpenCV and Dlib libraries.

In our current implementation, we have included all the mentioned modalities. As stated above, we mainly relied on existing libraries for input acquisition as our goal is not to re-implement them but rather give a proof of concept of their integration in a holistic system. Namely, we have used the Google API for speech recognition, and OpenCV was used to process camera input from the user. We are well aware that trying to explore each modality on its own and trying to optimize it thoroughly can improve the quality of input acquisition and may have a positive impact on the quality of interaction as a whole. However, following the main goal of this first implementation, the focus is more on the integration rather than the study of each modality. Now that these building blocks are fully integrated in the system, the modularity of the latter makes it feasible to isolate each modality and optimize it as much as possible without hindering the functioning of the system as a whole. Further developments should introduce the use of Electromyography (EMG) signals [15] that gives data about muscular activation that reflects muscular activity as well as a brain-computer interface through the use of Electroencephalography (EEG) signals [16]. The multimodal fusion engine should be well calibrated to adapt to a given user depending on the acquired signals and the input acquisition time window.

2.2 Centralizing user input

In order to benefit from the multiple modalities in the interaction, it is necessary to centralize all the actions coming from the user on a single platform. This platform needs to have constant updates of the input signals coming from all the acquisition devices then transmit them to the central decision making component that should make sense of all the input signals and interpret them according to the situation. For our implementation, we have decided to use a web-based database called “Firebase Database” to centralize the user input.

Communication with Firebase. Firebase[2] is a web NoSQL database provided by Google that supports storage as well as real-time data access. It is a lightweight solution for exchanging information between multiple clients that are subscribed to the database. It provides support for multiple development platforms such as Android, IOS. Support is also provided to connect the database to a C++, C# or a Python script.

In our implementation, each client is a component in the multimodal system architecture. Each component is connected to the internet and sends data to Firebase. In the real-time database, the state of each input acquisition interface is registered and changes upon user action. The data are updated in real time so the central decision making component can periodically and frequently check for new arriving commands. Figure 5 gives a brief screenshot of the database we use in our architecture.

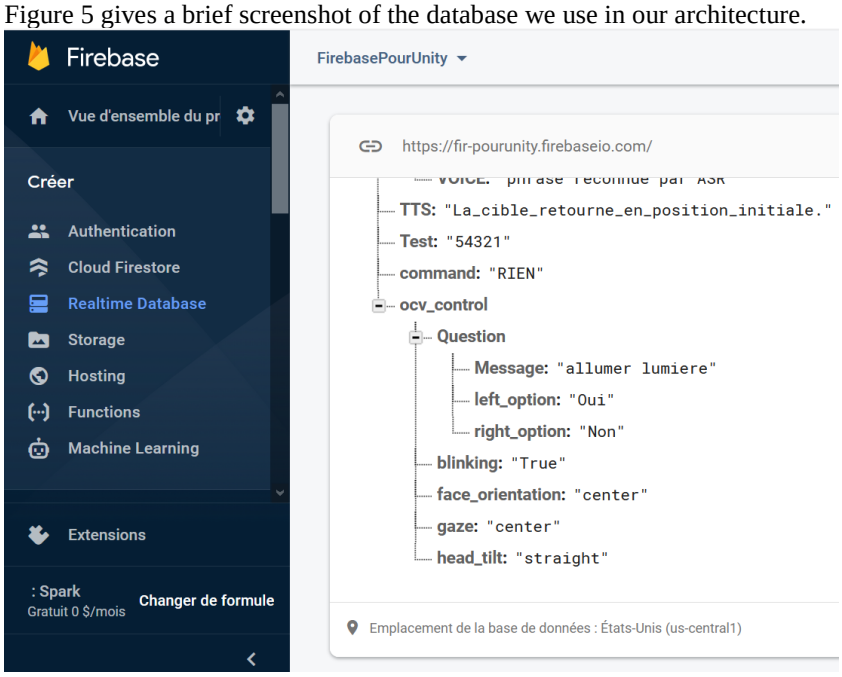

Fig. 5. Example of content available in a Firebase Realtime Database

[2] Details about the Unity Firebase integration available on: https://firebase.google.com/support/release-notes/unity

2.3 Synchronization of user input

As we have established, the system makes use of multiple modalities to interact with the user. Each input acquisition interface captures a certain modality and sends the registered input to the central decision making system. The system needs to make sense of the continuously changing state of the input interfaces in order to form a complete instruction. In addition, the instruction may be composed of multiple bits from different input interfaces. The question then becomes: when does the system detect the start/end of an instruction acquisition?

A time based acquisition of user input. The first paradigm in the acquisition of an instruction can be based on a time window. Within this time lapse, the user can issue multiple bits of the instruction through various channels. The system needs to wait for the duration of this time window after which it can start analyzing the user input. The time window can start from the moment the system asks a certain question like "What do you want to do?" If the user wants to dim the lights of the room, he/she can utter "dim" and click on the "lights" icon on the screen. The user is provided a time lapse of 10 seconds for example to complete all the needed bits for the full understanding of the command. After the 10 second time frame is over, the system can proceed to the processing of the instruction. In the case of an incomplete instruction, e.g. the user utters "dim" and does no further action before the 10 seconds time lapse is over, the system can use that bit of the instruction to ask for clarification: "What do you want to dim?" The user then is provided with an additional time lapse to complete the instruction. After several unsuccessful trials, the system can decide to halt the interaction.

It should be noted that the time lapse should be customized according to the capabilities of the user. If he/she suffers from a condition that renders his/her movements too slow, the time lapse should be large enough to accommodate for that. On the contrary, if the user is able to issue the command bits in a short period of time, the time lapse should be shortened.

On the plus side, this paradigm sets a relatively known duration of the input acquisition and can avoid getting stuck. On the minus side, it can be inconvenient to the user when he/she needs more time to think or just gets delayed by an external distraction. Hence, the user may not be able to complete the command bit in time which then lengthens the duration of the interaction and increases the needed exchanges between the user and the system.

An action based acquisition of user input. In this interaction paradigm, the system can issue an instruction request to the user. In order for the instruction sequence to start, a certain action is needed as a sort of "start recording" action. The changes to the state of each input interface is then registered. The user can then, through another key action signal the end of an instruction sequence. If the system was not able to make sense of a certain bit in the instruction sequence, it can notify the user asking for clarification. The same procedure is then required to issue the missing bit of information.

On the plus side, this paradigm offers the possibility for the user to take their time while issuing the different command bits. This can be beneficial especially for people suffering from a physical or mental condition that reduces considerably their action speed. On the minus side, it requires two additional actions (start and finish) which can add more physical and mental load to the interaction.

Choosing between these two input fusion paradigms may not be always clear. There can be situations where one is more adapted compared to the other. For this reason, we intend utilize both depending on the use case and the specific context of the interaction.

For the first implementation of our multimodal input fusion engine, we have utilized a time based approach. A lapse of time is given to the user to provide all the needed input bits for the task at hand. At the end of this time lapse, the system treats the multiple bits which are then converted into a complete instruction understandable by the system. This is better illustrated in a simple drawing application (Figure 6). The drawing application runs on the user's smartphone (smartphone 1). It provides a basic interface with "yes/no" buttons, a display for system message prompts, a large drawing area and a "clear" button. On a separate smartphone (smartphone 2) runs the multimodal fusion engine. This is a separate application that regulates the acquisition of user input and cycles every 10 seconds. As we established earlier, the information bit exchange between these two devices is mitigated through Firebase real-time web database to which both devices are subscribed. Within this 10 second time lapse, the user can touch on a location in the drawing area (on smartphone 1) and utters "draw house". At the end of this time lapse, the system gets the information that a house needs to be drawn at the same location indicated by the user's touch, then the application shows a small house icon.

Fig. 6. Left: Illustration of the functioning of the multimodal fusion engine running on smartphone 2. Right: Multimodal drawing application running on smartphone 1.

2.4 Handling multiple input signals

Within the context of a multimodal interaction, the system listens to the user input coming from multiple channels. The question then becomes: how does the system handle the various inputs coming roughly at the same time? More specifically which signal to consider? And what role is it supposed to play in the interaction?
In the literature, this question can be answered in multiple ways depending on the design of the system and the requirements of the specific interaction. Multiple papers have laid out the different concepts and paradigms for multimodality [5] and [6]. These concepts and definitions have been revised since like in [10] but have retained the overall paradigms. In this section, we give a reminder of the main paradigms from these articles of literature and reposition them in the context of interaction for people with special needs.
Complementarity. In this paradigm, the interactive system relies on the combination of multiple signals coming from different modalities to get the full input message. Each modality brings one or more bits to the input message and without the contribution "synergistically" of the other bits from the different modalities, the input message is incomplete. An example in our case is the scenario where the user taps the "light bulb" icon on a screen and utters "switch ON". Each bit of information is incomplete by its own until associated to the other bit to give the instruction "switch ON the light bulb".
In the context of people suffering from neuromuscular diseases for example, this paradigm can be useful if for a person who has just enough motor ability to move over a couple of buttons while still being able to talk. When the same operations can be performed on two different artifacts (ON/OFF can be applied to a light bulb and to a TV set), then the tactile interface can be made more compact and more reachable by the user without diminishing the usability of the whole system.

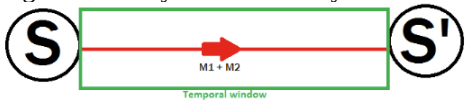

Fig. 7. Illustration of the combination of M1 (modality 1) and M2 (modality 2) brings the system from a state S to a state S' (taken from [7]).

Equivalence. In this paradigm, the multiple modalities allow for the same message being issued (Figure 7). In the case of switching on a light bulb, the user can tap a "light bulb switch" or utter "switch on the light bulb". Both modalities (touch and voice) allow the user to issue the same command. In the context of interaction for people suffering from neuromuscular diseases for example, the patient can have his/her motor abilities decline with various external factors like cold. When that happens and the user can hardly move his/her hand to the "light bulb switch", he/she can issue the command by voice.

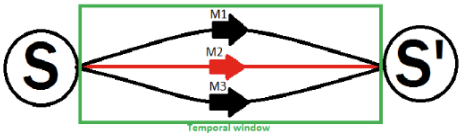

Fig. 8. Illustration of how M1 (modality 1), M2 (modality 2) and M3 (modality 3) can all bring the system from a state S to a state S'. The user chooses M2. Taken from [7]

When the user decides to issue the same command using more than one modality, then the paradigm is called a redundancy (Figure 8). This however puts the constraint on the system when fusing the data from the different modalities to handle any apparent conflict. If the user taps the "lights OFF" button and utters "dim the light", the system either needs to ask for clarification or ignore one of the channels.

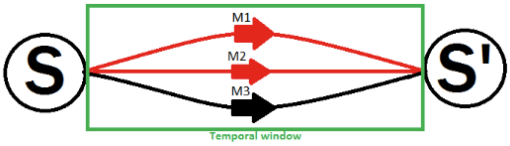

Fig. 9. Illustration of how M1 (modality 1), M2 (modality 2) and M3 (modality 3) can all bring the system from a state S to a state S'. The user chooses M1 and M2 redundantly. Taken from [7]

Assignment. In this paradigm, the user can issue the instruction only using one modality (Figure 9). When the combination modality-task is system imposed, we talk about a system assignment paradigm. Depending on the use case this paradigm may be obsolete when the user suffers from a disability that inhibits the utilization of that modality for any length of time. This would require a different design for each category of users (suffering from a given disease) or worse different designs for the same user when his/her condition fluctuates. If the switching on of the light bulb in the previous example were only possible by tapping the tactile switch, then the user would not have been able to perform the task in colder weather conditions.

Fig. 10. Illustration of how only M1 (modality 1) can bring the system from a state S to a state S'. Taken from [7]

A more adapted paradigm is the Agent Assignment (Figure 10). Here, the system design resembles that of the Equivalence case. However, the choice belongs to the user to always use one single modality. Taking the same light bulb switching example, if the user is completely paralyzed and can only interact with eye gaze, even if the interaction is achievable via multiple modalities, the user will always choose eye gaze interaction for that matter. The only caveat in this paradigm is that the system still listens for the other channels although they will not be used by the person. In this case, the system can dynamically prune one or more channels if, over time, they are not used.

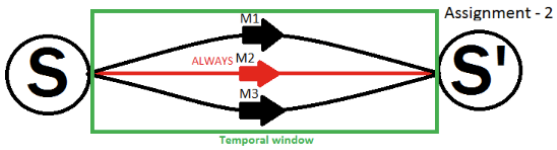

Fig. 11. Illustration of how M1 (modality 1), M2 (modality 2) and M3 (modality 3) can all bring the system from a state S to a state S'. The user always chooses M2. Taken from [7].

3 Simulation prototypes

As part of our prototyping process, we developed two applications on Unity3D game development software. The goal is to see how we can use multiple machines (clients) that are subscribe to Firebase. The user can provide input on one or more machines, the Firebase database collects this input and sends it to the machine that is required to perform the task. This section gives a brief description of the two applications.

3.1 A Unity3D game on two distinct machines

The first application is a simple game developed on Unity 3D software running on a computer (Figure 11). At the start of each round, a green or red ball drops on the top level. On a smartphone, the user has a tactile interface with a number of button among which a "left" and "right". Using these two button, the user needs to move the ball down the different levels to the corresponding bucket.

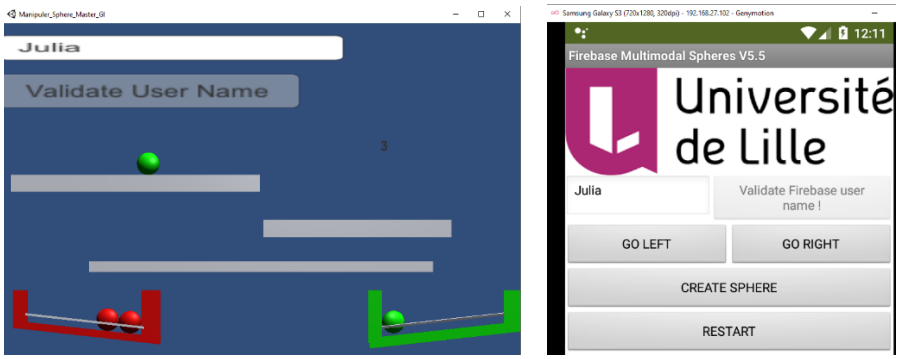

Fig. 12. Example of Unity standalone application (left) controlled by a smartphone Android application (right) across Firebase Realtime Database

The goal of this simple game is to show how, using Firebase real-time database, the user could provide input on one machine while the central system processes the input signal and executes it on another machine. Both are subscribed to the Firebase database. So, when the user presses a button, the smartphone application sends the data to the web database, the computer running the 3D game receives the update and then moves the ball.

3.2 Controlling a Robotic arm

The example presented in Figure 12 shows how a standalone application generated with Unity3D can be connected to an Android smartphone across Firebase. The user moves the target placement object (purple rectangle) thanks to the direction arrows of the mobile application. By clicking the button on the PC application, a trajectory plan is requested for the Nyrio robot arm, via a ROS (Robots Operating System)[3] instance, running in a Docker container[4].

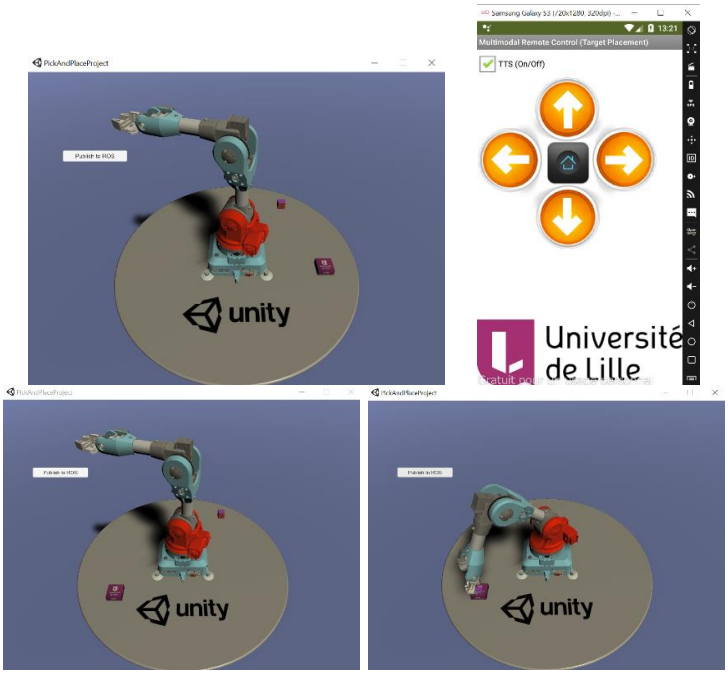

Fig. 13. Example of Unity standalone application controlled by a smartphone Android application across Firebase Realtime Database, using ROS on a Docker Container.

4 Examples of interaction scenarios: acting on the surrounding environment

In this section, we take three different scenarios of interactions that the user may be able to accomplish with the multimodal input of the system. In each scenario, we illustrate some of the paradigms in section 2.4 and detail how the system makes sense of different input signals in order to act on the user's environment. These scenarios constitute the basis for the first interactions developed for the user testing of our sys-

[3] More information on the Robots Operating System available on: https://www.ros.org/

[4] The implementation of the robotic control is inspired from the "pick and place" project provided by Unity3D: https://blogs.unity3d.com/2020/11/19/robotics-simulation-in-unity-is-as-easy-as-1-2-3/

tem. We are currently testing in laboratory the whole system usability, according to various parameters (user's disability level, kind of multimodality allowed, help from Wizard of OZ or not...).

In all these scenarios, the user is confined to the use of a power wheelchair. The latter is equipped with a computer (screen in front of the user for display), a tablet is attached to the armrest showing the interface Figure 12 capturing user taps as well as utterances to the voice command. A webcam is attached to the computer screen facing the user in order to capture eye gaze and head movement.

4.1 Switching on/off the lights

In this scenario, the goal for the user is to switch on/off the lights of the bedroom. In this scenario, the multimodality is used as an equivalence between vocal and touch interaction. The on/off switching can be performed either with vocal or touch interaction. We will illustrate one for ON and one for OFF.

The user is on his wheelchair about to enter the room and the lights are turned off. The user touches the speech button on the tablet and utters “switch on bedroom lights”. The computer plays the message “You asked to switch on bedroom lights, do you confirm?” The user then utters “Yes” and the lights are switched on.

As the user is leaving the room, we suppose that the user gets too fatigued by talking, so he needs to switch to another modality to perform the same task. Here, the chosen modality is touch.

The user makes a circular gesture on the tablet pad and is then presented with a list of potential objects he can interact with displayed on the computer screen. One of the objects indicates the lights. Using the “left” and “right” buttons, he navigates to the lights icon then performs a check gesture on the tablet pad. The screen then shows the two options to interact with the lights “switch on” and “switch off”. As the user selects the right button corresponding to the right choice a message is shown on the screen to ask for confirmation. The user then performs a check gesture on the tablet pad to confirm the choice, and the lights are switched off in the bedroom.

These two interactions illustrate how the user can interact with an object from the surrounding (bedroom lights) using two different modalities interchangeably.

4.2 Switching an air conditioning fan and controlling the fan speed

In this scenario, we make the supposition that the user is on a wheelchair and suffers from a severe case of Duchenne Muscular Dystrophy. His ability to talk is reduced to single, simple words as talking tires him. He also has very reduced hand mobility but can still move a couple of fingers over his phone screen that is attached to the armrest of his wheelchair.

We make the assumption that the environment control system is context aware. In other words, as the user gets into a room, the list of objects that the user can interact with are displayed on the screen in front of him/her. A rectangle is drawn over the selected object. The user can still access objects that are in other rooms through a separate menu.

In this scenario the user is in the living room, on a hot day, and needs to switch on the air conditioning fan. The speed of the fan can be set to various levels of speed. A fan, a TV set, a light bulb and window stores are presented on the screen as the user enters the room.

The user's fingers are on the tactile pad drawn on the phone screen. With any movement on the pad, the selection rectangle is moved from one object to another. To select the intended object, the user blinks for 500 ms and the object is illuminated. To cancel the selection the user can blink again for 500 ms and the illumination disappears. When an object is illuminated (the air conditioning fan for example), a dialog window is displayed on the screen with two different options ON and OFF. The user tilts his head to the right to switch the fan on. The system keeps the dialog window for 5 seconds after the switching ON of the fan. If the user wants to cancel the action, he can switch the fan OFF by tilting the head to the left. After this 5 second time lapse, the interaction object menu reappears again to allow the user to interact with more objects in the room if needed.

As soon as the fan is switched on, the system registers that the fan is on and puts itself in the context. This is useful for the user in case he wants to change the fan speed. By simply uttering "faster" and confirming by a 500 ms blink the user can increment the fan speed. The user can--but is not required to--specify the intended interaction object since the fan is the only switched ON object in the context that has a "speed property".

As the user wants to leave the room, he would like to switch the air conditioning fan back OFF. He proceeds with the same steps as for switching on the fan.

4.3 Simulating the control of a waste separation robotic arm

In this scenario, the user is also on a wheelchair and is suffering from a mild case Duchenne Muscular Dystrophy [8]. She cannot walk but she can talk freely without getting tired and can even move her hands, and handle very light objects. To assist with heavier objects, she uses a robotic arm that can be positioned on her wheelchair or put independently somewhere else inside the house. Both her input acquisition interface (smartphone application shown in Figure 12) and the robotic arm's control interface are connected wirelessly to Firebase so she can, if needed, control the arm from while she is in another room. Her wheelchair is equipped with a screen where she can visualize a third person shot of the robotic arm so she can control it from a distance. It is worthy to note that she tries to be ecologically responsible so she makes sure to separate her waste bags into recyclable and non-recyclable. The bags however can be heavier than she can handle so she uses the robotic arm.

At the start of the interaction, she is in the living room on her wheelchair. The robotic arm is positioned on the kitchen counter, with the recycling bin on the right and the non-recyclable bin on the left and the trash bags to separate are in the middle. The trash bags and the bins are reachable by the robotic arm.

Using a tactile application similar to Figure 12, she is able to steer the robotic arm over the bag she wants to pick. Then she utters "pick". The robotic arm picks up the trash bag. We suppose that this bag needs to go to the recycling bin. When the robotic

arm picks up the bag, the user then utters “to recycle” and the robotic arm, knowing the position of the recycle bin through various sensors, moves the bag over the bin. We note that the robotic arm relies on movement planning through the Robotics Operating System (ROS) that runs via Transmission Control Protocol (TCP) connection on a server inside the house similar to the application shown in section 3.2.

However, we stipulate that due to occasional technical lags, the arm may position the bag slightly besides the intended bin. For this reason, a fail-safe was installed: if the arm is slightly off target, the user can correct its position using the tactile pad on her phone just like she did to pick up the bag, but this time, only a slight correction is needed. It is then that the user can utter “release” and the bag is released in the recycle bin.

5 Wizard of Oz

Wizard-of-Oz (WoZ) is a common technique enabling HCI researchers to explore aspects of interaction not yet implemented in a real interactive system [9]. In this architecture, the WoZ module can be seen as a client who subscribes to all messages passing through the network. Instead of automatically reacting as the other modules of the architecture, this one lets the evaluator interpret the user’s intentions and give an appropriate response.

The WoZ module is an important part of the system because it allows seeing, at a glance, all the necessary information needed for an evaluator to make a quick decision, letting the user think that this decision is taken by the “intelligent system”.

An evaluator (the WoZ) observes interactions between a user and the system (computers, robots, etc.) and decides to interpret or not various interaction elements (speech, gaze, gesture, EEG, etc.) performed by the user [11]. Hence, a supposed intelligent system could interpret the sentence “It’s too hot in this room”, by proposing to switch on a fan. If this part of the sytem (ASR, semantic interpretation…) if not yet really available, the WoZ can switch on a fan “manually” remotely, in order to let the user think that this reaction was made by the intelligent system.

6 Conclusion

Multimodal interaction is widely used and studied as a subpart of Human-Computer Interaction, and researchers of this domain are often trying to improve the users' quality of life.

We have presented a general architecture of a multimodal interaction framework based on Firebase Realtime Database. Obviously, when designing and developing multimodal user interfaces, it is useful to have a powerful and fast way to send and receive information through various communication devices.

Thanks to Unity Firebase SDK, it is now possible to create 2D and 3D standalone applications for Windows, Mac and Linux, connected to Firebase Realtime Database. Mobile applications can be also connected to Firebase to push and pull such kind of data. We used App Inventor[5] and an experimental component named "Firebase DB" in order to create quickly and easily some mobile applications to deploy and test our prototypes.

Within this framework, our preliminary tests have shown that it is possible to handle multimodal interactions from the user, thanks to a Firebase Realtime Database, used as a quick and robust bus communication. We successfully tested various kinds of interaction (touch on a smartphone screen, speech recognition, eye gaze, tilt head…) in order to allow equivalent multimodality.

As our proposition is based on a shared real-time database, we will also be able to carry out evaluations of tasks for which the protagonists could interact remotely with the multimodal systems (voice, teleoperation gesture, visual and haptic remote perception, etc.).

Future work will consist of evaluating the system for free (painting) or imposed (home automation) tasks. We also plan to test multiple fusion paradigms (such as complementarity) to perform these same tasks. We will determine if the system is usable according to the handicap declared by the user or detected by the system during interactions. We will be able to use the WoZ technique in our framework, in order to allow experimenters to simulate the behavior of machines. Finally, we will increase the number of sensors by adding endogenous (EEG, EMG ...) and exogenous (camera, temperature sensor, etc.) data.

[5] Web page available on : http://ai2.appinventor.mit.edu/

References

1. Bolt, R. “Put-that-there”: Voice and gesture at the graphics interface, ACM SIGGRAPH Computer Graphics,July 1980 https://doi.org/10.1145/965105.807503
2. Rouillard, J., Duprès, A., Cabestaing, F., Bekaert, M-H., Lecocq, C., et al.. Relevant HCI for Hybrid BCI and Severely Impaired Patients. HCII 2015, Aug 2015, Los Angeles, United States. pp 313-323, ⟨10.1007/978-3-319-20816-9_30⟩. ⟨hal-01361922⟩
3. Jasvinder, C. Stepwise approach to myopathy in systemic disease. In Frontiers in Neurology, 2. Article 49. (2011).
4. Guedira, Y., Brohm, P-E., Dervin, D., Farcy, R. and Bellik, Y. Conception et Evaluation d'une Interface Tactile pour le Pilotage de Fauteuils Roulants Electriques pour des Personnes Atteintes de Maladie Neuromusculaires. Journal d'Interaction Personne-Système (8), 1, 2019.
5. Caelen, J and Coutaz, J. "Interaction Homme-Machine Multimodale : Problèmes Généraux". IHM'91 (December 1991), Dourdan.
6. Y. Bellik, D. Teil, "Les types de multimodalités", dans les actes des 4ièmes Journées sur l'ingénierie des interfaces Homme-Machine, IHM'92, Telecom Paris Publ., Paris, (30 Nov. - 2 Déc., 1992), pp. 22-28
7. Pirau,J. Modeling and resolving conflicts and apprehensions in multimodal models, Student thesis: Master Thesis in Computer science, (2016).
8. Duprès A., Cabestaing, F., Rouillard, J., Tiffreau, V. and Pradeau, C. Toward a hybrid brain-machine interface for palliating motor handicap with Duchenne muscular dystrophy: A case report. Annals of Physical and Rehabilitation Medicine, Elsevier Masson, (2019), ⟨10.1016/j.rehab.2019.07.005⟩
9. Hoffman, G. OpenWoZ: A Runtime-Configurable Wizard-of-Oz Framework for Human-Robot Interaction, Conference: AAAI Spring Symposium on Enabling Computing Research in Socially Intelligent Human-Robot InteractionAt: Palo Alto, CA, (2016).
10. Martin, J.C.: Six primitive types of cooperation for observing, evaluating and specifying cooperations. Proceedings of AAAI (1999)
11. Salber, D. and Coutaz, J. Applying the Wizard of Oz Technique to the Study of Multimodal Systems, Third International Conference, EWHCI '93, Moscow, Russia, (1993), DOI: 10.1007/3-540-57433-6_51.
12. Guedira, Y., Brohm, P-E., Dervin, D., Farcy, R. and Bellik, Y. A Tactile Interface to Steer Power Wheelchairs for People Suffering from Neuromuscular Diseases. In HCI in Mobility, Transport, and Automotive Systems. Driving Behavior, Urban and Smart Mobility, Second International Conference, MobiTAS 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, July 19–24, 2020, Proceedings, Part II
13. Jacquet, C., Mohamed, C., Mateos, M., Jean-Bart, B., Bretault, P., Schnepp, I. and Bellik, Y. An Ambient Assisted Living Framework with Automatic Self-Diagnosis, International Journal On Advances in Life Sciences, vol. 5, n°1-2, 10p, 2013.
14. Nadia Elouali, Xavier Le Pallec, José Rouillard, Jean-Claude Tarby. MIMIC: Leveraging Sensor-based Interactions in Multimodal Mobile Applications. International Conference on Human Factors in Computing Systems, CHI 2014, 2014, Toronto, Canada
15. Sun, Y., Xu, C., Li, G., Xu, W., Kong, J., Jiang, D., Tao, B. and Chen,D. Intelligent human computer interaction based on non redundant EMG signal, Alexandria Engineering Journal, Volume 59, Issue 3, 2020, Pages 1149-1157.
16. Botte-Lecocq, C. and Cabestaing, F.. Les interfaces cerveau-machine pour la palliation du handicap moteur sévère. Handicap’2008, Jun 2008, Paris, France. pp.180-189.