Exploratory Design of a Hands-free Video Game Controller for a Quadriplegic Individual

Atieh Taheri
atieh@ece.ucsb.edu
University of California, Santa Barbara
Santa Barbara, CA 93106, USA

Ziv Weissman
zivweissman2008@gmail.com
Palo Alto Senior High School
Palo Alto, CA 94301, USA

Misha Sra
sra@cs.ucsb.edu
University of California, Santa Barbara
Santa Barbara, CA 93106, USA



Figure 1: A screenshot from the Temple Looter adventure game. The facial action units corresponding to a smile (AU6 + AU12 or Cheek raiser + Lip corner puller) are recognized from realtime video input data to make the video game character walk.

ABSTRACT

From colored pixels to hyper-realistic 3D landscapes of virtual reality, video games have evolved immensely over the last few decades. However, video game input still requires two-handed dexterous finger manipulations for simultaneous joystick and trigger or mouse and keyboard presses. In this work, we explore the design of a hands-free game control method using realtime facial expression recognition for individuals with neurological and neuromuscular diseases who are unable to use traditional game controllers. Similar to other Assistive Technologies (AT), our facial input technique is also designed and tested in collaboration with a graduate student who has Spinal Muscular Atrophy. Our preliminary evaluation shows the potential of facial expression recognition for augmenting the lives of quadriplegic individuals by enabling them to accomplish things like walking, running, flying or other adventures that may not be so attainable otherwise.

This work is licensed under a Creative Commons Attribution-NoDerivs International 4.0 License.

AHs '21, February 22–24, 2021, Rovaniemi, Finland
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8428-5/21/02.
https://doi.org/10.1145/3458709.3458946

CCS CONCEPTS

• **Human-centered computing** → **Accessibility**; **Accessibility systems and tools**;

KEYWORDS

Accessibility, quadriplegia, facial expression recognition, video gaming, input methods, hands-free, facial expressions

ACM Reference Format:
Atieh Taheri, Ziv Weissman, and Misha Sra. 2021. Exploratory Design of a Hands-free Video Game Controller for a Quadriplegic Individual. In *Augmented Humans International Conference 2021 (AHs '21), February 22–24, 2021, Rovaniemi, Finland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3458709.3458946

1 INTRODUCTION

With COVID-19 pandemic induced stay-at-home lifestyles, video conferencing has become a prime way to connect, whether it is

for work, school, fitness or socializing. Interest in video games has also risen as sequestered people around the world take to games not only as an escape from the impositions of the pandemic but also for work related meetings [1], and as a new way to connect with family and friends [2]. For a lot of people, video games are about being able to do things that are often not possible in real life, about experiencing great adventures and visiting new places. Yet, as prolific as gaming is, it is inaccessible to a significant number of people with disabilities [53]. According to the CDC, 1 in 4 US adults have some type of disability[3]).

With growing interest in playing video games and with video games increasingly being used for purposes other than entertainment, such as education [12], rehabilitation [16, 24] or health [21, 47], game accessibility is increasingly critical, and even more so for players with disabilities who stand to benefit greatly from the opportunities video games offer. But, games are usually far more demanding than other entertainment media "in terms of motor and sensory skills needed for interaction control, due to special-purpose input devices, complicated interaction techniques, and the primary emphasis on visual control and attention" [15].

Individuals with neuromuscular diseases can have difficulties grasping, holding, moving, clicking, pushing or pulling due to lack of muscle control. Loss of muscle function could happen due to degenerative neurological diseases such as Muscular Dystrophy (MD), Spinal Muscular Atrophy (SMA), Multiple Sclerosis (MS), or Amyotrophic Lateral Sclerosis (ALS or Lou Gehrig's disease), or from brain and spinal cord injuries, for example, due to motor vehicle accidents and stroke. Standard game input devices such as a mouse or a keyboard are not well suited to the needs of users with severe motor disabilities [6]. A number of input devices have been developed to allow motor impaired players to interact with games such as mechanical switches [35], mouth and tongue controllers and joysticks [23, 34, 38], brain-computer interfaces [37], and eye-gaze controllers [13, 40]. The type of device that can work depends on an individual's needs and requirements that are determined by the targeted muscles and the degree of function of those muscles. However, most of these devices are typically constrained with regard to the input they can provide when compared with the types of input required in conventional games.

In this work, we present a hands-free human-computer interaction solution designed in collaboration with a quadriplegic user. The solution is based on realtime facial expression recognition (FER). By recognizing muscle movements in a webcam video stream, facial expressions (FEs) are identified and mapped to a PC keyboard to control actions in a video game. The system also includes speech recognition which serves as a secondary input modality. Unlike brain-computer interfaces or other input technologies designed for motor-impaired users, our system is inexpensive and does not encumber the user with sensors and devices.

The main contribution of this work are:

- A fully functional prototype of facial expression recognition based video game control for individuals with quadriplegia.
- Collaborative design approach for design and testing of facial expression recognition.
- Design of three different games that demonstrate the mapping of facial expressions to game actions with focus on user agency, user comfort, ease of use, memorability, and reliability of recognition along with some design reflection.

2 RELATED WORK

There has been a lot of research in the past twenty years aimed at developing assistive technology (AT) devices to increase independence and in individuals with motor impairments of various origins (e.g., locked-in-syndrome, amyotrophic lateral sclerosis, spinal muscular atrophy, quadriplegia, muscular dystrophy, cerebral palsy, etc.) [36]. We present a subset of that work here that is specifically related to interaction with video games, both for a broader audience of disabled users and for those with motor disabilities.

2.1 Accessibility in Consumer Video Games

More consumer games are starting to include accessibility options. For example, The Last of Us: Part II released in June 2020 offers around 60 accessibility options like directional subtitles and awareness indicators for deaf players or auto-target and auto-pickup for those with motor disabilities [4]. The accessibility options in Naughty Dog's 2016 release Uncharted 4: A Thief's End [5], support features like auto-locking the aiming reticle onto enemies, changing colors for colorblind users, or adding help to highlight enemies. Sony has included a number of accessibility functions in the PS4 system[6], including text-to-speech, button remapping, and larger font for players with visual and auditory impairments.

In addition to adding accessibility options in commercial games, many special purpose games have also been developed for blind players [10, 32, 52]. Canetroller [55] is device that enables visually impaired individuals to navigate a virtual reality environment with haptic feedback through a programmable braking mechanism and vibrations supported by 3D auditory feedback. Virtual Showdown [48] is a virtual reality game designed for youth with visual impairments that teaches them to play the game using verbal and vibrotactile feedback. Players of a recently release game, Animal Crossing: New Horizons are using the game's customization options to make the game more accessible. For example, a blind player demonstrated how they modified the game in ways that do not rely on sight while another player low-vision player covered their island in grass and flowers to force fossils and rocks to spawn in specific spots [7]. Not all commercial games are customizable which leaves some players with disabilities are to rule out those games or seek help of a friend or assistant to "play" the game.

The leading example of an accessible game controller is Microsoft's Xbox Adaptive game controller that allows people with physical disabilities who retain hand/finger movement and control, to be able to interact and play games [30]. By connecting the adaptive controller to external buttons, joysticks, switches and mounts,

[1] https://www.nytimes.com/2020/07/31/business/video-game-meetings.html
[2] https://www.nytimes.com/2020/04/21/technology/personaltech/coronavirus-video-game-production.html
[3] https://www.cdc.gov/media/releases/2018/p0816-disability.html
[4] https://www.playstation.com/en-us/games/the-last-of-us-part-ii-ps4/accessibility/
[5] https://dagersystem.com/disability-review-uncharted-4/
[6] https://support.playstation.com/s/article/PS4-Accessibility-Settings?language=en_US
[7] https://kotaku.com/how-animal-crossing-new-horizons-players-use-the-game-1844843087

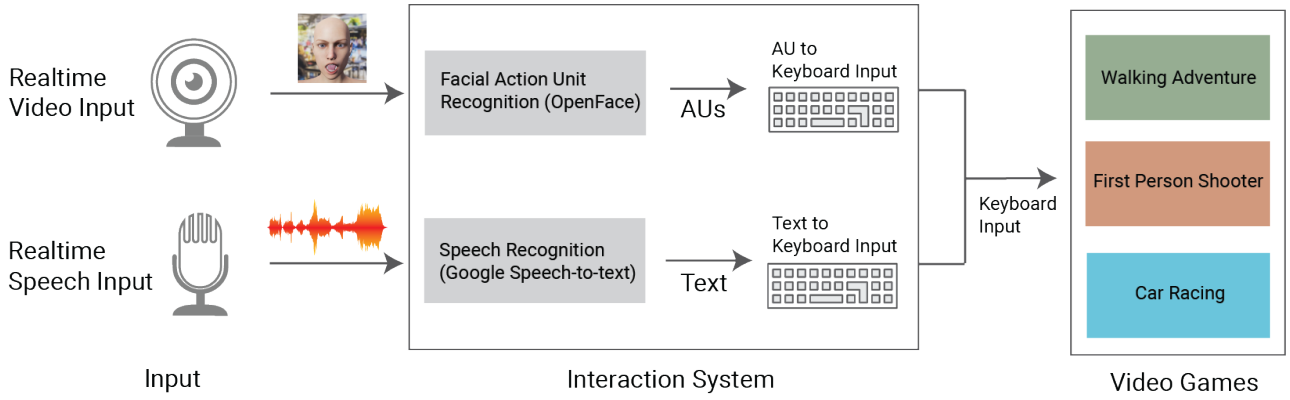


Figure 2: The system pipeline showing input of video and speech data that is processed and converted to keyboard bindings for controlling action in each game.

gamers with a broad range of disabilities can customize their setup. The device can be used to play Xbox One and Windows 10 PC games and supports Xbox Wireless Controller features such as button remapping [1].

The solutions presented here, while accessible, are not usable by those with severe motor disabilities as most of these solutions rely on hand-based control. Additionally, while both software and hardware solutions can make gaming accessible, we believe software solutions can provide a more economical and customizable solution. Thus, in this work we explore the design of a software based input system.

2.2 Game Input Methods for Quadriplegics

For individuals with quadriplegia or those with severe motor impairments, there are a few approaches that can enable playing video games without assistance. Most of these are based on acquiring signal from different parts of the body such as the tongue, brain, or muscles that the individual has voluntary control over. There are several Tongue Machine Interfaces (TMIs) such as tongue-operated switch arrays [42] or permanent magnet tongue piercings that are detected by magnetic field sensors [18, 23] to enable interaction with a computer. Lau et al. [25] created a radio frequency transmitting device shaped like an orthodontic retainer containing Braille keys that could be activated by raising the tongue tip to the mouth superior palate. Leung et al. [26] presented a theoretical framework for using a multi-camera system for facial gesture recognition for children with severe spastic quadriplegic cerebral palsy. Chen et. al. [7] mapped eye and lip movements to a computer mouse for a face-based input method.

Guangyu et. al. [4] showed a high-speed spelling system based on a brain-computer interface (BCI). BCIs can provide non-muscular communication and control to people who are severely motor impaired. Non-invasive BCIs using scalp-recorded electroencephalographic (EEG) brain activity along with adaptive algorithms have not become popular among users despite being researched since the early 1970s [5, 29, 46, 50] due to limitations like bandwidth and susceptibility to noise and interference [17].

Buttons or switches are a commonly used device to enable motor impaired players to interact with games as they can be operated with any part of the body that is able to produce voluntary movement, enabling actions like sip and puff, pull, push and squeeze [53]. Depending on the severity of the motor impairment one or more switches may be usable with binary input being the smallest amount of interaction provided with a switch.

There are a few consumer products that allow motor impaired users to play video games. One Switch[44] is a non-profit dedicated to arcade style games that can be played with one switch. The QuadStick [38] is a mouth operated device available in three versions that allow interacting with a computer or playing video games. Depending on the version, it includes sip/puff pressure sensors, a lip position sensor, and a joystick with customizable input and output mapping. Quadstick has enabled players with severe disabilities play videos games and engage in social interaction through streaming on Twitch [8]. Axis controllers [11] enable players who are able to move their wrists to play console games with an ergonomic layout featuring large buttons and joysticks. Game Box Controllers [8] are attachments for existing controllers to enable players who may simply need larger buttons or taller joysticks to play.

All these systems and devices have their unique affordances and limitations. For example, Quadstick is the most popular video game controller for quadriplegics, though it will not work with the upcoming PS5 [38]. There are several games where it is not possible to map a physical option on the Quadstick to a game action because of the large number of game actions possible. Our software based design allows creating macros, commonly used by video game players, where a sequence of game actions (e.g. jump + turn left) can be mapped to a single facial expression. BCI interfaces require the user to wear a headset which may be difficult to wear and use for extended periods of time for playing games [43]. Our solution does not require the user to wear any sensors, trackers or devices. To our knowledge, facial expression recognition has not yet been investigated in the context of game interaction for quadriplegic individuals.

[8]https://www.washingtonpost.com/video-games/2019/10/14/its-my-escape-how-video-games-help-people-cope-with-disabilities/

Facial AUs Combinations or Facial Expression			Keyboard Key
AU6 > 2.0 ⊗ AU12 > 2.0	Cheek Raiser + Lip Corner Puller	Happiness	1
AU1 ⊗ AU4 ⊗ AU15	Inner Brow Raiser + Brow Lowerer and Lip Corner Depressor	Sadness	2
AU9 > 1.4 ⊗ AU10 > 2.0	Nose Wrinkler + Upper Lip Raiser	Disgust	3
AU2 > 0.5 ⊗ AU5 > 1.5	Outer Brow Raiser + Upper Lid Raiser	Wide Eyes	4
AU7 > 1.4 ⊗ AU23 > 1.0	Lid Tightener + Lip Tightener		5
AU4 ⊗ AU25 ⊗ AU26	Brow Lowerer + Lips Part + Jaw Drop		6

Table 1: Facial AU combinations with their descriptions and their approximate equivalent facial expressions mapped to keyboard keys. The numbers in the first column are the thresholds for AU intensity values (no number means AU presence/absence binary value was used.) If each AU intensity in an expression combination is greater than the specified threshold value then the corresponding key press in that row is triggered.

3 SYSTEM DESIGN

Interaction design often focuses on creating solutions that are generalizable to a large portion of the population. In contrast, Assistive Technologies (AT) are usually designed for the individual. Prior research shows the best effects of an AT are seen when it is developed with and tested by the potential end users [43]. In this work we take the AT approach to create software game input system and three test games in collaboration with Atieh, a quadriplegic engineering graduate student in our lab. We follow a similar design approach working with Atieh as used by Lin et. al. [27] who created a game controller and a mouse for a quadriplegic teen.

The fundamental idea behind our system design was to make the most out of the small muscle movements available to Atieh. In our first meeting, we learned that Atieh could voluntarily control only one finger so hand-based systems were not an option. Mouth-based systems like the Quadstick [38] were not an alternative due to Atieh's limited jaw range of motion. Lastly, Atieh's prior experience with gaze-based systems was frustrating and thus that input modality was also discarded as an input method. After some more brainstorming done in a Zoom session with Atieh where we discussed different input methods and reviewed prior research, we settled on camera-based facial expression recognition as the input method. Using a camera-based system over other methods like Interferi [20]] or Earfieldsensing [28] which require the user to either wear a facemask or an ear-plug, makes our system usable with devices that most users are likely to already own, e.g., smartphone or a desktop webcam or a laptop with a built-in webcam.The next step was testing which facial expressions Atieh could make, the robustness of detecting those expressions, and the positioning of the webcam to capture their facial input in relation to the computer and display. The design and ideation for the system was conducted with Atieh over Zoom as social distancing rules made meeting in-person impossible.

3.1 Pipeline

The system has four main parts: 1) facial expression recognition (FER) or facial action unit (AU) recognition, 2) speech recognition, 3) interaction design (AU and text to keyboard mapping), and 4) game design and gameplay. Figure 2 shows the system pipeline. Input from the webcam and the microphone is sent through the AU recognition system to the keyboard mapper, which converts them into keyboard inputs for each game. We used Unreal Engine (UE) version 4.25.1 for building the Walking Adventure (WA) and the First Person Shooter (FPS) games and UE version 4.23 for the Car Racing game. Game logic was created using UE's Blueprint system.

3.2 Facial Input

In order to standardize facial expression recognition, Ekman and Friesen [9] categorized facial muscle movements into *Facial Action Units (FAUs)* to form the *Facial Action Coding System (FACS)*. Over the years, two main types of approaches have been explored for FAU recognition - those that use texture information and those that use geometrical information [22]. In our system, we use the OpenFace 2.0 toolkit [3] based on the texture approach, hereon referred to as OpenFace. The toolkit is capable of facial landmark detection [54], head pose estimation, FAU recognition [2], and eye-gaze estimation [51]. Of these features, we integrated FAU recognition into our pipeline to extract Action Units (AUs) from the input video stream in realtime.

In our pipeline, AU detection takes two forms: 1) AU presence - a binary value depicting whether a particular AU is present in the captured frame or not, and 2) AU intensity - a real value between 0.0 and 5.0 depicting the intensity of the extracted AUs in a captured frame. OpenFace provides access to AUs 1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 15, 17, 20, 23, 25, 26, 28, and 45. After testing, we discarded AUs

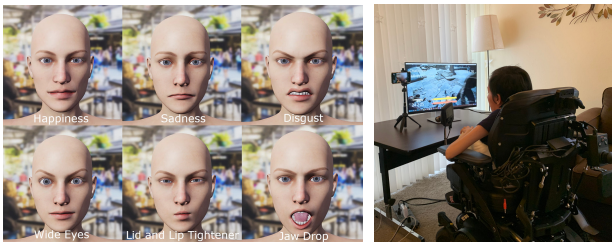


Figure 3: Left: Six facial expressions used for playing the games. Top row left to right: Happy face, Sad face, Disgust. Bottom row left to right: Wide open eyes, Pucker, Jaw drop. Right: Atieh playtesting the FPS game at home.

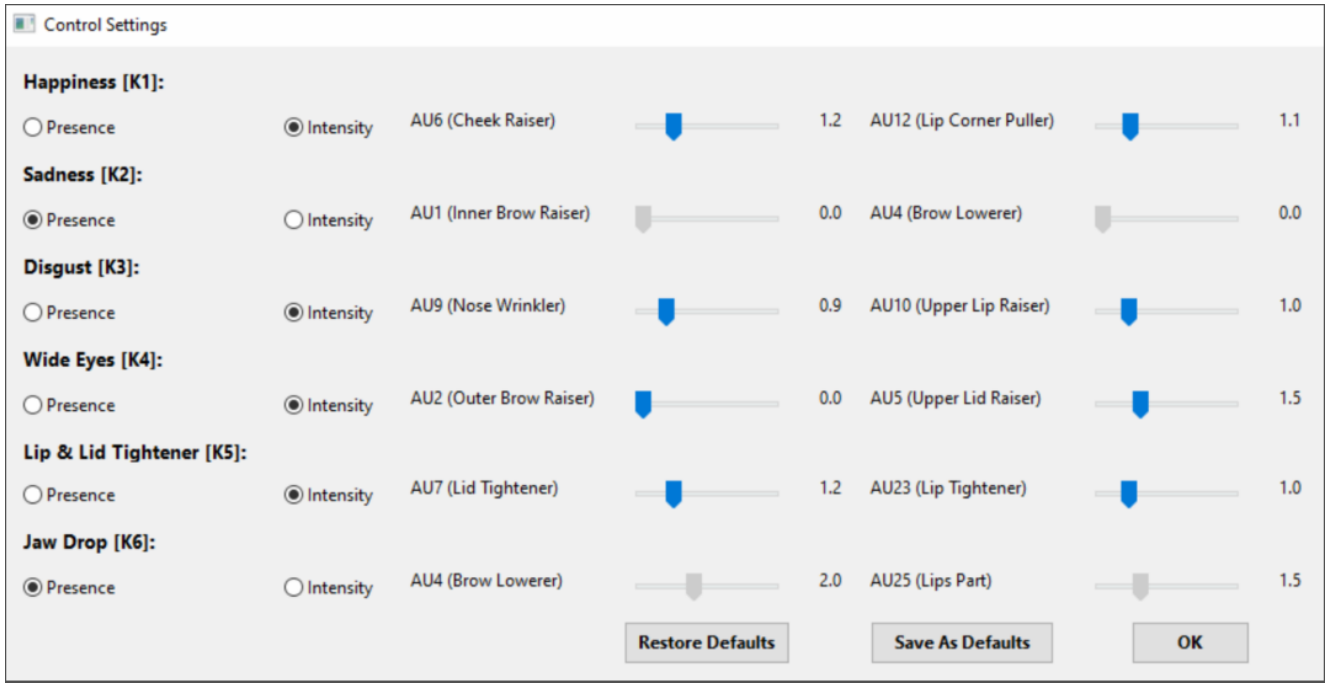


Figure 4: We built an interface to enable users to change AU detection thresholds and mappings for each game.

14, 17, 20 because they were similar to other AUs. AU45 detected blinking and was therefore unusable as input.

In each input frame, the FER system outputs AU presence and intensity values for each of the 18 AUs. These values are sent to the our keyboard mapper for converting them to game input. Figure 3: Left demonstrates the six FEs the player makes for taking actions in the games. Each one of these FEs is obtained by combining two or three facial AUs (Table 1). Figure 3: Right shows Atieh playtesting the FPS game at home.

The AU combinations used in each game are experimentally determined by Atieh since it was not possible for them to make every facial expression that an able-bodied person might easily make. Additionally, some AUs had higher reliable detection than others. The experimentation involved testing with AU presence values first and replacing with AU intensity values if AU presence alone failed to be accurately detected. The numbers in Table 1 show the AU intensity threshold values for the AU combinations that were used in the games. These values are also experimentally determined with help from Atieh. All AU combinations use intensity values except two FEs; sadness, a combination of AU1 + AU4 + AU15, and jaw drop, a combination of AU4 + AU25 + AU26. These two FEs were reliably detectable with AU presence values alone.

3.3 Speech Input

We used a Python speech recognition library [45] to communicate with Google Cloud Speech API [14] for converting spoken commands to text. The text data was scanned for specific keywords like "Walk" or "Yes" and converted into keyboard input using Pynput [33] and mapped to keys previously programmed in Unreal Engine for each action in each game. Speech interaction served as a backup modality to AU recognition, always available for use but not necessarily required.

3.4 Key Mapping

Facial expressions and text keywords are both mapped to keyboard input through the keyboard mapper. During testing, it became evident that AU recognition and mapping per input frame was frustrating to the user due to the system making multiple keyboard mappings per second. To resolve the issue, we set a threshold for the number of consecutive frames an AU combination needed to be visible in before getting mapped to the keyboard. This helped provide more control to the user and improved reliability. After testing all AU combinations were set to a five frame threshold. Having a high frame threshold helped prevent the system from responding to unplanned facial muscle movements.

3.5 Flexibility

While the mappings presented in this work are best suited for Atieh, we created an interface that allows the user to change the mappings as needed. For example, if a particular expression is not as robustly detected, or if the user finds the mapping of a smile to walking or running not intuitive for them, they can easily change which expression they would like to use through a front-end interface and map it to a game action of their choice (Figure 4).

4 GAME DESIGN AND GAMEPLAY

There are more than 91 different video game genres [49]. We chose three different ones to implement and test our facial AU recognition based input system in games that are vastly different from one



Figure 5: Left: The player bending over to pick up a flower in Nature Walk with the requisite facial expression shown as inset. Right: Two different facial expressions or speech input allow the player to make the decision to turn left or not in Nature Walk.

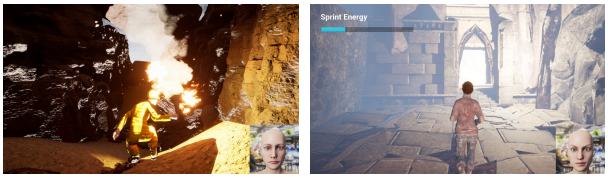


Figure 6: Left: Player pauses to look around for crystals in Cave Explorer using the facial expression shown in the inset. Right: Player sprinting in Temple Looter with facial expression input shown in the inset.

another. All games were collaboratively and iteratively designed with Atieh's feedback. For maintaining Atieh's privacy, we have removed their face from the gameplay in all the figures and replaced it with a virtual character wearing the same expressions. The mapping of AU combinations to keyboard keys is shown in Table 1.

4.1 Walking Adventure

Knowing that Atieh has not played any video games due to being severely affected by SMA for years, we wanted to introduce them to the game controls and genres step by step, going from fewer to more interactions and from slower to faster paced games. Walking Adventure has three levels: Nature Walk, Cave Explorer and Temple Looter. The keyboard to game action mapping is presented in Table 2: Left. Each level is visually different, designed to immerse the player in a different environment with its own set of tasks and goals. The game character is created in Adobe Fuse [31] and rigged and animated using Mixamo [41]. All three games use ambient and task related sounds. Each time a facial expression is correctly detected, the system provides audio feedback to the player. Thus the player always feels in control of the game character's actions with multimodal feedback from the visual and the audio channels.

4.1.1 Nature Walk. Nature Walk is the first level in the Walking Adventure game. The goal is to enjoy walking and exploring the level and interacting with the environment. Nature Walk is a nature scene with realistic water, trees, and flowers that are organically scattered throughout the map built with assets from the Meadow Environment [9]. The level design consists of tree lined walking trails with branching paths. The level uses a modular spline path for the character to follow at a fixed walking speed. The player can stop to interact with the flowers that use an emissive texture to make them stand out from the rest of the vegetation. Interaction with the flower (Figure 5: Left) proceeds as follows:

(1) If the player is in the walking state, stop the player.
(2) Over a maximum of 2 seconds, turn the player to face the flower.
(3) Over another second move the player up to 90% of the distance from its current position towards the rose, such that when the character bends over to pick up the flower, their hand intersects with the flower's stem giving it a more natural appearance.
(4) Play the pickup animation and partway through the animation sequence, delete the flower on the ground, and spawn an identical flower attached to the player's hand.
(5) Wait for the animation to end, then let the player go idle with the flower in the character's hand for 3-4 seconds.
(6) When the idle time ends, play the put-down animation.
(7) When the player is bent half-way, destroy the rose that is in the character's hand to make it look like the rose was actually was put back down on the ground.
(8) Wait for the put-down animation sequence to end.
(9) Find a spot on the spline that is 100 steps ahead of where the player was before they chose to interact with the flower.
(10) Make the character face and walk towards that spot.
(11) Finally, make the character face forward and continue walking along the trail.

At path branches, we added a decision making option to enable the player to choose if they want to turn or continue walking along the main path (Figure 5: Right). In keeping with the lighthearted mood of the level, there were no timers or tasks.

4.1.2 Cave Explorer. For designing the Cave Explorer level, we downloaded free 3D assets from Quixel's Limestone Quarry [10] collection. The downloaded assets were assembled to create a complex cave environment as shown in Figure 6: Right The player explores a dimly lit cave bearing a torch in hand. Their task is to collect crystals and safely exit the cave. The crystal collection mechanic was added not only to mimic a task commonly found in commercial video games but also to make the level more interactive and goal oriented. The game character from Nature Walk is reused with different clothing and animations. To add an element of surprise, we implement a "jump scare" mechanic by hanging zombie skeletons from the cave roof that would abruptly drop down with appropriate animations and sound effects to scare the player at opportune moments.

4.1.3 Temple Looter. The last level in the Walking Adventure game is built by modifying a map in the free Infinity Blade: Fire Lands asset [11]. We created an ancient temple scene, similar to what one

[9] https://www.unrealengine.com/marketplace/en-US/product/meadow-environment-set

[10] https://quixel.com/megascans/collections?category=environment&category=natural&category=limestone-quarry

[11] https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-fire-lands?lang=en-US

Keyboard Key	Game Action	Input Type
1	Start Walking	AU
2	Stop Walking	AU
3	Pick up	AU
4	Sprint	AU
5	Turn Yes	AU or Speech
6	Turn No	AU or Speech

Keyboard Key	Game Action	Input Type
1	Start/Stop Walking Fwd	AU
2	Aim and Shoot	AU
3	Start/Stop Turning Left	AU
4	Start/Stop Turning Right	AU
5	Jump	AU
6	Pause	AU or Speech

Table 2: Left: Mappings of the keyboard keys to the actions defined in the Walking Adventure game. Right: Mappings of keyboard keys to the actions defined in the FPS game.

might see in an Indiana Jones movie. We again used the character from Nature Walk with an adventurer's clothing and different animations. The player is tasked with looting hidden treasure in the temple and escaping. We added a stamina bar that the player needed to fill up by not spending too much energy before sprinting out of the temple (Figure 6: Left). Running is a new action added to this level, something that Atieh expressed a strong desire to be able to do.

Atieh's response following gameplay was,

> "The three levels of Walking Adventure have nice environments. I loved taking actions in the game world like jumping, picking up, walking around on it. I'd like the game better if there was a story behind, but I realize it's only an exploratory design."

4.2 First Person Shooter

The second type of the game we created is a First Person Shooter (FPS). As opposed to Walking Adventure, which is a third-person game, the FPS allows Atieh to see through the eyes of the character. The FPS character was downloaded from Mixamo along with 40 animation sequences for covering typical movements in an FPS game. A blendspace was created to manage the animation logic playback with actions like walking, turning, jumping, aiming and shooting, reloading, and crouching. A single weapon option is added with sound effects and gunfire animation that showed as a flash at the tip of the gun (Figure 7: Left). The zombies from Cave Explorer are re-used with different animations to walk, attack, and die. Pathfinding logic is created for the zombies to move towards the player, if the player gets within a certain distance range. A horde system is implemented to spawn new zombies based on where the player is heading. We added trigger boxes on some paths in the environment to spawn a horde of zombies in a plausible location. This created the effect of there being more zombies than there actually were, which helped with performance and made the game feel higher action. The FPS map was created using elements from a free asset on the Unreal Engine Marketplace called Infinity Blade: Ice Lands[12]. Atieh's initial playtest revealed the game to be difficult for an unseasoned FPS player and too fast paced for using facial expressions or even speech as input. To help a first time gamer enjoy the FPS without frustration, the maximum number of zombies at any instant was limited to 15 and auto-aim was added. The auto-aim feature works as follows: when the player intends to shoot a zombie, the player character is turned by a calculated amount per frame and the scoped gun is pointed at the nearest zombie within a pre-specified range. Shooting takes place automatically after auto-aiming. The gun holds 25 bullets at a time and reloads automatically. Ammunition packs are strewn around the map. Like any FPS game, the player needs to manage the bullets to have enough in case a horde of zombies attack. Continuously holding down a key for character movement, as is common in traditional FPS games, was changed to using a key toggle. This meant the same facial expression could start and stop a character action like walking forward or turning left/right making it easier for the player to control the character. While speech input worked well for the slower paced Walking Adventure games, it was unusable for the FPS due to latency in cloud speech processing. Thus, speech was used only for pausing the game and not for the main actions. The facial expressions mapping to FPS actions is presented in Table 2: Left.

About the FPS game, Atieh said,

> "[I]t is smooth and it did not frustrate me playing whereas most of the time any assistive tool that comes out for people with disability, would somehow need the person an exhausting effort. If you notice, for example when the character shoots the zombies, it is only a matter of lowering your lips and your eyebrows which is very simple. For me, it is a fun experience...I just wish it was a multi-level game."

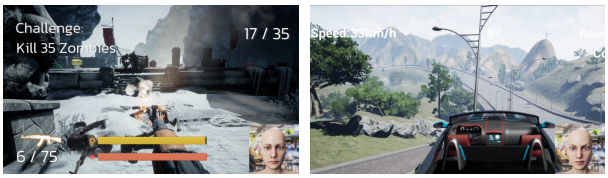


Figure 7: Left: Player shooting zombies in the FPS game using a facial expression shown as inset. Right: Player turning the car right using the facial expression shown as inset.

[12]https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-ice-lands

Keyboard Key	Game Action	Input Type
1	Start/Stop Driving Forward	AU
2	Start/Stop Driving Backward	AU
3	Start/Stop Turning Left	AU
4	Start/Stop Turning Right	AU

Table 3: Mappings of the keyboard keys to the actions defined in the Car Racing game.

4.3 Car Racing

The game is based on the Car Game template from Unreal to which 3D assets consisting of hills and trees were added from CSDN[13], a community of game developers. The player competes against a clock on a closed loop track. To make the game more engaging, the player collects coins along the track, an idea inspired by the popular Nintendo car racing game, Mario Kart. The coins are created using simple cylinders with yellow material and spin animation attached. Other than coins, there are obstacles on the road that the player needs to maneuver around as bumping into them reduces the car's speed. During playtesting, Atieh found the default car speed to be too high to easily control with facial expressions. That made the car difficult to maneuver around obstacles which required many iterations of complex manipulations in a fast sequence. To help make the game more fun, the maximum car speed was lowered, the obstacle physics was simplified to enable turning the car easily with a facial expression toggle, similar to turning in the FPS game (Figure 7: Right). The mapping of facial expressions to car motions is presented in Table 3.

After playing the Car Racing game, Atieh said,

> "In the past, when I could still play with my hands and regular controllers, I loved playing racing games. However, losing my ability to use my hands, playing this genre of game became impossible. I could never have imagined that I would be able to play my favorite game genre again without putting in too much effort. When the idea of incorporating facial expressions in this genre first came up, I thought well, if we could make the game so customized, we could probably make it possible. Though surprisingly, after only the second attempt that lowered the vehicle speed, the game resembled what I used to play years ago and became very entertaining."

5 DESIGN REFLECTION

Designing our exploratory system was achieved iteratively through conducting a series of experiments with Atieh. Each stage of the design process gave us new insights into changing configuration of the expressions and game actions in ways that would best match the player's abilities. While our current design is focused on Atieh specifically, the system is easily modifiable. In fact, Atieh was able to create macros for testing the system with commercial games (Section 5.1), which is something we had not used with the games we created. Through our process of weekly interactions with Atieh over 6 months, we learned some valuable lessons that may help others design FER based input systems for quadriplegic users.

5.0.1 Design Input to Support Player Ability. While obvious, determining player ability is the first step to figuring out a potential input method that would work best for that individual. Our interaction system primarily focuses on using facial muscle movements because that's what we narrowed down options to during our first conversation with Atieh. Subsequently, we learned that the number of muscles employed in each facial expression also makes a big difference to the player's comfort level and to the system's ability to detect expressions reliably. Using too many muscles repetitively is tiring and difficult to control while using too few leads to false positives. Atieh tried several facial expressions to help us establish that expressions using 2-3 AUs worked best for them for two main reasons: 1) easy to make repeatedly, and 2) reliably detected without false positives.

5.0.2 Consider Frequency of Game Actions. There are some actions in every game that are more frequently taken than others. In our games they were: start/stop walking in the Walking Adventure, aiming and shooting in the FPS, and turning left or right in the Car Racing game. Keeping their frequent usage in mind, the facial expressions selected for these actions needed to be less demanding on the player. While the mappings used in the current games are comfortable for Atieh, they are also easily changeable if Atieh's abilities change over time (Figure 4). For Atieh, the four FEs (i.e. happiness, sadness, disgust, and wide eyes) were easier to make compared with the others and of these four, disgust and wide eyes were the simplest. Therefore, we mapped **disgust** and **wide eyes** to the most frequent game actions. We simultaneously attempted to map positive expressions with positive game actions, for example, smiling to move forward, to help the user remember the mapping.

5.0.3 Consider Multimodal Input and Feedback. Video games come in a large variety of genres. Some require fast responses while others are more forgiving. Relying on a single input system can limit the types of interactions and games that may work successfully. Including multiple input modalities, like FER and speech in our system, can help provide the user a backup input system for situations when the primary system is difficult to use. The inverse is also possible in that some games may not work well with one modality but work flawlessly with another. An example of this is our FPS game that would not work well with speech input due to latency induced by cloud-based recognition while the Walking Adventure game was perfectly suited for speech input. Similarly, including multimodal feedback through visuals, text and sound data can help the player stay in control, knowing that their expression was recognized.

5.0.4 Design Games for Personalization and Flexibility. Allowing the games to be personalized can help users enjoy playing them without getting frustrated. For example, mapping a specific expression to a game action, slowing down or speeding up a particular game action, reducing or increasing the frequency of enemies are elements that the user should be able to configure at the start. Commercial games provide this feature in the form of Easy, Normal or Difficult modes of gameplay and having this choice is helpful for players of all abilities. Unfortunately, a large percentage of ATs for

[13] https://www.csdn.net/

quadriplegic individuals are left unused for reasons from device performance to changes in user preference [19, 39]. Designing with a quadriplegic user helped us better understand the constraints and opportunities. Our system is flexible and configurable (Figure 4) such that the player can map any expression to any keyboard key and also create macros to allow for multiple key presses in sequence for more complex game input.

After months of playtesting and building, when Atieh played through the full games:

> "All these years that I have had lost my ability to use my hands to grab game controllers or even since when I realized that using a keyboard and a mouse have become impossible I desperately looking for finding a way to regain those things that one day were my only way of having some entertainment in my life and the more I pursued the more I became disappointed and eventually gave up. Even those accessible tools that came out like eye-gazing devices could not give me back the ability to play the video games, and it gradually became like a dream for me to play and I started only watching other people's gameplays. But trying out this tool gave me hope that I still can go into the game world once again."

5.1 Adoption in Other Video Games

We were curious to test our system with some commercial games to explore how it would perform outside our designed games. We were surprised to find that games that do not require a mouse or games where the mouse movements can be replaced with keyboard input, worked well with our system without requiring any modifications. We tested four atmospheric single player puzzles games on Steam like GRIS[14] and Inside[15], Limbo[16], and Little Nightmares[17]. For all these games, we created macros for game actions that required rapid key presses in a particular sequence to accomplish the game task. With that small change, the games were fully playable without requiring Atieh to make multiple facial expressions in quick succession. To switch to the macro mode and back, a FE was used. Based on this experiment, Atieh is excited about the potential of our system to work with other commercial games and they will continue testing. We believe that the set of games that work with our system will grow as more developers enable mapping game controls to a keyboard.

5.2 Limitations and Future Work

Our work contributes to a body of research and design of hands-free gaming input for users with severe motor disabilities who need innovative solutions that can enable them to play independently. Our work identifies potential for future research in the field of hands-free input based on facial expression recognition. Because the current system was designed in collaboration with one quadriplegic individual, we did not build an interface to enable mapping expressions to keys. A future version will be helped with such an interface allowing players to customize the system according to their needs. An important future direction is improving the user interface, allowing for personalization and flexibility. Additionally, automated adaptation to disease progression would enable the system to continue being meaningful to the user for a long time. Once campus COVID-19 restrictions[18] are lifted, conducting a user study with a group of quadriplegic individuals would help us understand and modify the system design to improve flexibility and usability for individuals who are different from Atieh. The study would include both the quantitative methods, as well as the qualitative methods, to understand challenges in adopting a new AT and the value it brings.

[14]https://store.steampowered.com/app/683320/GRIS/
[15]https://store.steampowered.com/app/304430/INSIDE/
[16]https://store.steampowered.com/app/48000/LIMBO/
[17]https://store.steampowered.com/app/424840/Little_Nightmares/

6 CONCLUSION

In this paper, we explored the design of a hands-free interaction system for playing video games in collaboration with a quadriplegic player. We demonstrated the system in use with three different types of games. We found that the system enable Atieh to play and enjoy the experience without any frustration. This was the first time Atieh was able to play because they are unable to use input devices like the Quadstick [38]. Since every motor impaired individual has unique requirements, we believe our software solution can be easily customized for their abilities and purposes, with the assumption that the individual has voluntary control over their facial muscles. With a growing number of game developers and companies including accessibility options into the games, we are hopeful that facial expression recognition based input would also become an option soon, opening up a new world of gaming for Atieh and perhaps many others.

ACKNOWLEDGMENTS

We would like to thank Shiran Wang for work on the Car Racing game. We would like to acknowledge members of the Perceptual Engineering Lab in the Computer Science department at UCSB for their helpful comments on our work.

REFERENCES

[1] Deborah Bach. 2018. *Xbox Adaptive Controller — The next level in accessible gaming.* Retrieved August 1, 2020 from https://news.microsoft.com/stories/xbox-adaptive-controller/
[2] Tadas Baltrušaitis, Marwa Mahmoud, and Peter Robinson. 2015. Cross-dataset learning and person-specific normalisation for automatic action unit detection. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, Vol. 6. IEEE, 1–6.
[3] Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. 2018. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, 59–66.
[4] Guangyu Bin, Xiaorong Gao, Yijun Wang, Yun Li, Bo Hong, and Shangkai Gao. 2011. A high-speed BCI based on code modulation VEP. *Journal of neural engineering* 8, 2 (2011), 025015.
[5] Niels Birbaumer, Nimr Ghanayim, Thilo Hinterberger, Iver Iversen, Boris Kotchoubey, Andrea Kübler, Juri Perelmouter, Edward Taub, and Herta Flor. 1999. A spelling device for the paralysed. *Nature* 398, 6725 (1999), 297–298.
[6] José Cecílio, João Andrade, Pedro Martins, Miguel Castelo-Branco, and Pedro Furtado. 2016. BCI framework based on games to teach people with cognitive and motor limitations. *Procedia Computer Science* 83 (2016), 74–81.
[7] Cheng-Yao Chen and Jyh-Horng Chen. 2003. A computer interface for the disabled by using real-time face recognition. In *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE Cat. No. 03CH37439)*, Vol. 2. IEEE, 1644–1646.

[18]https://www.cdc.gov/coronavirus/2019-ncov/community/colleges-universities/index.html

[8] RJ Cooper and Associates Inc. 2020. Game Box Controllers. http://rjcooper.com/game-controller/, Last accessed on 2020-08-05.

[9] Paul Ekman and Wallace V Friesen. 1971. Constants across cultures in the face and emotion. *Journal of personality and social psychology* 17, 2 (1971), 124.

[10] Johnny Friberg and Dan Gärdenfors. 2004. Audio games: new perspectives on game audio. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*. 148–154.

[11] Bluetip Gaming. 2020. Axis 2 Pro. https://bluetipgaming.com/product/axis-2-pro/, Last accessed on 2020-08-05.

[12] James Paul Gee. 2003. What video games have to teach us about learning and literacy. *Computers in Entertainment (CIE)* 1, 1 (2003), 20–20.

[13] James Gips and Peter Olivieri. 1996. EagleEyes: An eye control system for persons with disabilities. In *The eleventh international conference on technology and persons with disabilities*, Vol. 1.

[14] Google. 2020. *Speech-to-Text: Automatic Speech Recognition | Google Cloud.* Retrieved August 10, 2020 from https://cloud.google.com/speech-to-text/

[15] Dimitris Grammenos, Anthony Savidis, and Constantine Stephanidis. 2009. Designing universally accessible games. *Computers in Entertainment (CIE)* 7, 1 (2009), 1–29.

[16] Jennifer Howcroft, Sue Klejman, Darcy Fehlings, Virginia Wright, Karl Zabjek, Jan Andrysek, and Elaine Biddiss. 2012. Active video game play in children with cerebral palsy: potential for physical activity promotion and rehabilitation therapies. *Archives of physical medicine and rehabilitation* 93, 8 (2012), 1448–1456.

[17] Xueliang Huo and Maysam Ghovanloo. 2010. Evaluation of a wireless wearable tongue–computer interface by individuals with high-level spinal cord injuries. *Journal of neural engineering* 7, 2 (2010), 026008.

[18] Xueliang Huo, Jia Wang, and Maysam Ghovanloo. 2008. A magneto-inductive sensor based wireless tongue-computer interface. *IEEE transactions on neural systems and rehabilitation engineering* 16, 5 (2008), 497–504.

[19] Amy Hurst and Shaun Kane. 2013. Making" making" accessible. In *Proceedings of the 12th international conference on interaction design and children*. 635–638.

[20] Yasha Iravantchi, Yang Zhang, Evi Bernitsas, Mayank Goel, and Chris Harrison. 2019. Interferi: Gesture sensing using on-body acoustic interferometry. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

[21] Pamela M Kato. 2010. Video games in health care: Closing the gap. *Review of general psychology* 14, 2 (2010), 113–121.

[22] Irene Kotsia, Stefanos Zafeiriou, and Ioannis Pitas. 2008. Texture and shape information fusion for facial expression and facial action unit recognition. *Pattern Recognition* 41, 3 (2008), 833–851.

[23] Gautham Krishnamurthy and Maysam Ghovanloo. 2006. Tongue drive: A tongue operated magnetic sensor based wireless assistive technology for people with severe disabilities. In *2006 IEEE international symposium on circuits and systems*. IEEE, 4–pp.

[24] Belinda Lange, Sheryl Flynn, and Albert Rizzo. 2009. Initial usability assessment of off-the-shelf video game consoles for clinical game-based motor rehabilitation. *Physical Therapy Reviews* 14, 5 (2009), 355–363.

[25] Cynthia Lau and Stephanie O'Leary. 1993. Comparison of computer interface devices for persons with severe physical disabilities. *American Journal of Occupational Therapy* 47, 11 (1993), 1022–1030.

[26] Brian Leung and Tom Chau. 2008. A Multiple Camera Approach to Facial Gesture Recognition for Children with Severe Spastic Quadriplegia. *CMBES Proceedings* 31 (2008).

[27] Henry WJ Lin, Leila Aflatoony, and Ron Wakkary. 2014. Design for one: a game controller for a quadriplegic gamer. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. 1243–1248.

[28] Denys JC Matthies, Bernhard A Strecker, and Bodo Urban. 2017. Earfieldsensing: A novel in-ear electric field sensing to enrich wearable gesture input through facial expressions. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1911–1922.

[29] Dennis J McFarland, Dean J Krusienski, William A Sarnacki, and Jonathan R Wolpaw. 2008. Emulation of computer mouse control with a noninvasive brain–computer interface. *Journal of neural engineering* 5, 2 (2008), 101.

[30] Microsoft. 2020. Xbox Adaptive Controller. https://www.xbox.com/en-US/accessories/controllers/xbox-adaptive-controller, Last accessed on 2020-08-05.

[31] Mixamo. 2020. *Adobe Fuse CC.* Retrieved August 10, 2020 from https://www.adobe.com/products/fuse.html

[32] Tony Morelli, John Foley, Luis Columna, Lauren Lieberman, and Eelke Folmer. 2010. VI-Tennis: a vibrotactile/audio exergame for players who are visually impaired. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*. 147–154.

[33] Moses Palmér. 2020. *pynput.* Retrieved August 10, 2020 from https://pypi.org/project/pynput/

[34] Qiyu Peng and Thomas F Budinger. 2007. ZigBee-based wireless intra-oral control system for quadriplegic patients. In *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 1647–1650.

[35] WJ Perkins and BF Stenning. 1986. Control units for operation of computers by. *Journal of medical engineering & technology* 10, 1 (1986), 21–23.

[36] Carlos G Pinheiro, Eduardo LM Naves, Pierre Pino, Etienne Losson, Adriano O Andrade, and Guy Bourhis. 2011. Alternative communication systems for people with severe motor disabilities: a survey. *Biomedical engineering online* 10, 1 (2011), 1–28.

[37] Gabriel Pires, Urbano Nunes, and Miguel Castelo-Branco. 2012. Evaluation of brain-computer interfaces in accessing computer and other devices by people with severe motor impairments. *Procedia Computer Science* 14 (2012), 283–292.

[38] Quadstick. 2020. Quadstick: A game controller for quadriplegics. https://www.quadstick.com/shop/quadstick-fps-game-controller, Last accessed on 2020-08-05.

[39] Gerwin Schalk, Kai J Miller, Nicholas R Anderson, J Adam Wilson, Matthew D Smyth, Jeffrey G Ojemann, Daniel W Moran, Jonathan R Wolpaw, and Eric C Leuthardt. 2008. Two-dimensional movement control using electrocorticographic signals in humans. *Journal of neural engineering* 5, 1 (2008), 75.

[40] J David Smith and TC Nicholas Graham. 2006. Use of eye movements for video game control. In *Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*. 20–es.

[41] Nazim Kareemi Stefano Corazza. 2020. *Mixamo - Get animated.* Retrieved August 10, 2020 from https://www.mixamo.com/#/

[42] Lotte NS Andreasen Struijk. 2006. An inductive tongue computer interface for control of computers and assistive devices. *IEEE Transactions on biomedical Engineering* 53, 12 (2006), 2594–2597.

[43] Boštjan Šumak, Matic Špindler, Mojca Debeljak, Marjan Heričko, and Maja Pušnik. 2019. An empirical evaluation of a hands-free computer interaction for users with motor disabilities. *Journal of biomedical informatics* 96 (2019), 103249.

[44] One Switch. 2020. OneSwitch.Org. http://www.oneswitch.org.uk/, Last accessed on 2020-08-05.

[45] Anthony Zhang (Uberi). 2018. *SpeechRecognition Python API.* Retrieved August 10, 2020 from https://pypi.org/project/SpeechRecognition/

[46] Jacques J Vidal. 1973. Toward direct brain-computer communication. *Annual review of Biophysics and Bioengineering* 2, 1 (1973), 157–180.

[47] Darren ER Warburton, Shannon SD Bredin, Leslie TL Horita, Dominik Zbogar, Jessica M Scott, Ben TA Esch, and Ryan E Rhodes. 2007. The health benefits of interactive video game exercise. *Applied Physiology, Nutrition, and Metabolism* 32, 4 (2007), 655–663.

[48] Ryan Wedoff, Lindsay Ball, Amelia Wang, Yi Xuan Khoo, Lauren Lieberman, and Kyle Rector. 2019. Virtual showdown: An accessible virtual reality game with scaffolds for youth with visual impairments. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–15.

[49] Wikipedia. 2020. Video Game Genres. https://en.wikipedia.org/wiki/Category:Video_game_genres, Last accessed on 2020-08-05.

[50] Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan. 2002. Brain–computer interfaces for communication and control. *Clinical neurophysiology* 113, 6 (2002), 767–791.

[51] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. 2015. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3756–3764.

[52] Bei Yuan and Eelke Folmer. 2008. Blind hero: enabling guitar hero for the visually impaired. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*. 169–176.

[53] Bei Yuan, Eelke Folmer, and Frederick C Harris. 2011. Game accessibility: a survey. *Universal Access in the information Society* 10, 1 (2011), 81–100.

[54] Amir Zadeh, Yao Chong Lim, Tadas Baltrusaitis, and Louis-Philippe Morency. 2017. Convolutional experts constrained local model for 3d facial landmark detection. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2519–2528.

[55] Yuhang Zhao, Cynthia L Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, and Mike Sinclair. 2018. Enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation. In *Proceedings of the 2018 CHI conference on human factors in computing systems*. 1–14.