MASTER EN SCIENCES INFORMATIQUES À FINALITÉ SPÉCIALISÉE EN SOFTWARE ENGINEERING

Gestural Interaction for an Artistic SoundControl System

Trinon, Christelle

Award date:
2020

Awarding institution:
Universite de Namur

UNIVERSITÉ DE NAMUR

FACULTÉ D'INFORMATIQUE

Gestural Interaction for an Artistic Sound Control System

Christelle Trinon

RUE GRANDGAGNAGE, 21 ● B-5000 NAMUR(BELGIUM)

Abstract

In recent years, the interest of numerous fields for gestural interaction and gesture recognition has risen. While some of them can afford the use of various devices like gloves and armbands, artists favor more discreet techniques to maximize the impact of the integration of gestural interaction on their audience. The goal of this master thesis was to study and assess the potential of such interactions for interactive artistic systems, only using computer vision. By means of an Intel RealSense D435 camera and the Cubemos Skeleton Tracking SDK, different movement features to sound features mappings have been tested in order to identify the most intuitive and guessable ones for the users. This work presents the followed approach and the results it provided.

Keywords: HCI, Gestural Interaction, Computer Vision, Sound Control, User Experience, Artistic Installation

Résumé

Ces dernières années, de nombreux domaines se sont intéressés à l'interaction gestuelle et à la reconnaissance de gestes. Certains domaines peuvent se permettre l'utilisation de divers équipements, comme des gants ou des bracelets, mais les artistes préfèrent utiliser des techniques plus discrètes pour maximiser l'impact de l'interaction gestuelle sur leur public. L'objectif de ce mémoire était d'étudier et d'évaluer le potentiel de ce type d'interaction pour des systèmes interactifs à but artistique, en utilisant uniquement la vision par ordinateur. A l'aide d'une caméra Intel RealSense D435 et du Skeleton Tracking SDK de Cubemos, différents mappings entre caractéristiques de mouvement et caractéristiques de son ont été testés dans le but d'identifier les plus intuitifs et devinables pour les utilisateurs. Ce travail présente l'approche suivie et les résultats qui en ont découlé.

Mots-Clés: IHM, Interaction Gestuelle, Vision par Ordinateur, Contrôle du Son, Expérience Utilisateur, Installation Artistique

Acknowledgements

I would like to start by thanking Pr. Bruno Dumas for his guidance and his reactivity all along this year. The subject of this thesis and the internship were a perfect fit for me and I would not have had the chance to have this experience without his interest in local companies such as Superbe.

I am thankful to Mr. Libertiaux and Mr. Bertrand, who welcomed me into their team and trusted me from the beginning. Their availability and their extensive advise during my internship with them have allowed me to develop skills that will help me in my professional career and for that I am beyond grateful.

Lastly, I want to thank my mum, my dad and my sister for their support throughout this semester, my academic journey (even when it was tough) and, perhaps most importantly, my whole life. Special thanks to my dad who read and corrected this work. I am also thankful to my friends who encouraged me during the writing of this thesis and put up with my absence these past few months.

Contents

Chapter 1

Introduction

The popularity of gestural interaction and of gesture recognition for human-computer interaction has drastically grown in the past few years [47] and will continue to increase in the future. The democratization and the increased quality of the devices needed to experiment with this topic has helped the scientific community and independent companies to launch numerous studies and research. Those are driven by different goals and a multitude of industries could benefit from new natural interfaces [37, 40]. While some aim to upgrade our healthcare system [51, 64, 57] or improve the lives of speech and hearing impaired people [13, 75, 74], to just name a couple of examples, a part of them is entertainment oriented. This entertainment sector is itself divided in diverse research areas, including, but not limited to, video game development [30, 58, 49] and artistic expositions and live performances [36, 28, 19].

This thesis will focus on the artistic part of the field, more specifically on gestural interaction in real-time for sound generation and sound control in artistic settings. Sound generation systems allow the user to fully create his own music while sound control systems only enables the control of existing sound.

The project was realized for the Namur-based artistic studio *Superbe*[1]. It develops technological installations with an artistic purpose and was cofounded in 2011 by Gaëtan Libertiaux, creative director, and Gaël Bertrand, electronics engineer. Mr. Bertrand is in charge of developing the hardware and software components of the installations. They are both artists and their work is profoundly influenced by their passion for music and art. Superbe's story started as they created an interactive beatbox installation, called *MusicOmaton*[2], which got a lot of success. It inspired them to develop more and more of these interactive installations and they now tour festivals and exhibits in different countries (Belgium, France, United States of America,...). The most notable projects they created are *PAF*[3], which was their first installation as Superbe, *Geometric Music*[4] and *SMing*[5], which is detailed later in this chapter as it was the starting point for this thesis. But they don't limit themselves to original musical

[1] http://superbe.be
[2] https://vimeo.com/11422539
[3] http://superbe.be/work/paf-photo-box
[4] http://superbe.be/work/geometric-music-app
[5] http://superbe.be/work/sming

installations, they have also created custom-made arcade games and other interactive systems for different companies and brands. Furthermore, they have collaborated with the University of Namur on a project named *The Big Bang Machine*[6]. It was created with Dr. André Füzfa and was meant to be interacted with on stage during conferences. The goal was to show how various parameters can influence the course of the evolution of our universe.

Figure 1.1: SMing in action. Source: Superbe (`http://superbe.be/work/sming`)

Superbe, in collaboration with *Dog Studio*[7], has also initiated the *KIKK Festival*, as Mr. Libertiaux is co-founder and art director of *KIKK*[8]. The festival has taken place every year since 2011 in Namur and has a different theme each year. The general topic however remains the same: digital and creative cultures.

The objective of the internship at Superbe was to provide an evolution to SMing, their latest project, an interactive choir installation. The name came from the combination of the words *sing* and *me*. It allows the users to first record their voice and face, then use a conductor's baton equipped with a gyroscope[9] to control a choir. The special feature of the choir is that it is solely composed of the user himself. A software modifies the user's voice to create all the voice types that compose an actual choir (from baritone to soprano). The user can control the rhythm and the intensity of the choir: the melody is extracted from an existing Musical Instrument Digital Interface[10] (MIDI) file and, when the gyroscope records an angular change, the baton sends an Open Sound Control

[6] `http://superbe.be/work/the-big-bang-machine`
[7] `https://dogstudio.co`
[8] `https://www.kikk.be`
[9] Device used to measure angular velocity
[10] `https://www.midi.org`

(OSC) message[11] to a Processing[12] sketch that triggers the launch of the next MIDI note. As a result, the rhythm of the music (i.e. the frequency at which the notes change) is based on the frequency at which the user changes the angle of his movement. The intensity of the movement (i.e. the speed of the baton) defines the playing volume of the note.

The desired evolution path for SMing was to remove the conductor's baton and allow the users to control the system in a similar fashion but with their body movements. Superbe advertise themselves as "Magic Makers": consequently, one important constraint for the new system was to only use computer vision technologies and absolutely no wearables, such as gloves or armbands for instance. Indeed, asking the users to wear some piece of equipment would "break the magic". It would also be too time-consuming.

Taking this into account, the research question to be addressed in this thesis can be defined as: *How to integrate computer vision-based gestural interaction in an artistic sound control system, while keeping the interaction attractive, playful, understandable and artistic?*

This research question is answered the following way: the first step was to identify the best technology to use. The most promising camera was the Intel RealSense D435[13] and the best pairing for it was with the Cubemos Skeleton Tracking SDK[14]. Once this duo was approved, a first prototype was developed. Mr. Libertiaux and Mr. Bertrand experimented with it and gave their feedback. Two more iterations of this process were necessary before they were satisfied with the system.

This thesis presents the three proofs of concept that were designed to answer the research question. Its chapters are structured as follows: the next chapter explains the difference between gestural interaction and gesture recognition. Then, a state of the art on these two areas is provided to allow the audience to become acquainted with the topic. It also introduces the concept of guessability. The subsequent chapter is dedicated to detailing the research and discussions that lead to and followed the different prototypes. It also offers implementation decisions for each prototype as well as an evaluation process. The final chapter concludes this works with insights into the possibilities for future academic work and projects for Superbe.

[11] http://opensoundcontrol.org/introduction-osc
[12] https://processing.org
[13] https://www.intelrealsense.com/depth-camera-d435/
[14] https://www.cubemos.com/skeleton-tracking-sdk/

Chapter 2

Gestural interaction and gesture recognition

The purpose of this chapter is to review what gestural interaction and gesture recognition are in the context of human-computer interaction and to expose the distinction between them.

2.1 Gestural interaction

Gestural interaction is simply a way of interacting with a computer or a system using gestures or body movements instead of a more classic modality, like a mouse, a keyboard or a touchscreen interface for instance (see Figure 2.1). Indeed, the definition of the word *interaction*, as given in the *Collins English Dictionary*[1], is "a mutual or reciprocal action or influence"[16]. This means that the computer receives an input from the human, processes it and reacts accordingly to the input to provide the appropriate output.

Performing gestural interaction requires the processing of raw data and of the features that can be extracted from them [18]. The raw data are directly extracted from the input modalities, usually RGB-Depth camera(s), glove(s), muscle sensor(s) or a combination of these, and the different data from similar modalities can be fused (data-level fusion) to create new and maybe more useful and meaningful raw data. The features, for their part, are derived from the raw data. In case of multimodal interaction, the features of the different modalities can be processed separately or they can be fused together (feature-level fusion) if the modalities are intimately tied together.

[1] https://www.collinsdictionary.com/

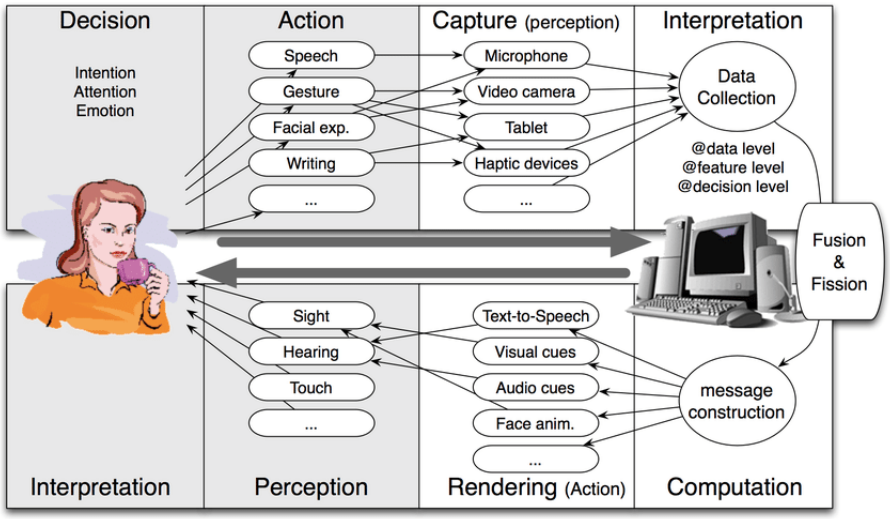


Figure 2.1: The human-computer interaction cycle, as depicted by Dumas et al. [18].

2.2 Gesture recognition

In contrast to gestural interaction, gesture recognition systems are able of genuinely recognizing gestures. This means the system is capable of capturing a gesture or movement, analyzing it and interpreting it to associate it to the appropriate semantic label. For instance, such a program could link the ring gesture to its most widespread meaning, "OK" (Figure 2.2).

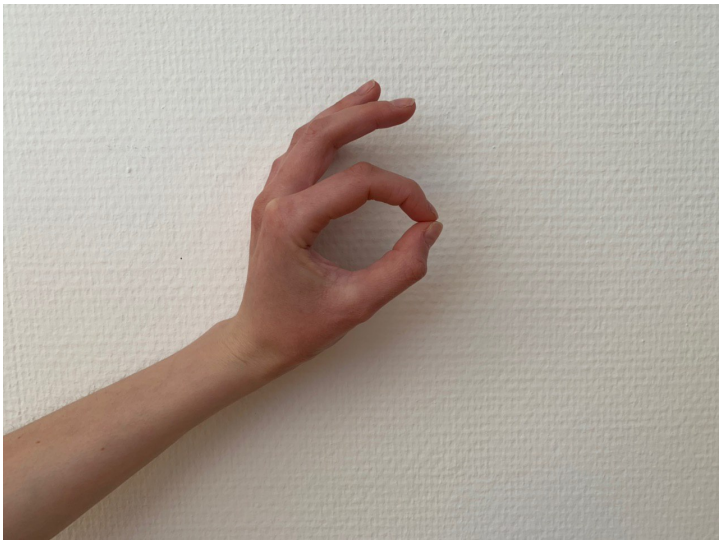


Figure 2.2: Ring gesture, commonly called OK gesture.

To achieve this, an additional level of processing is required: the data and the features are interpreted to allow decision-level treatment [18]. It is also possible

to proceed to decision-level fusion, which can be useful when working with less bound modalities. It is at this level that the movements are coupled to their corresponding semantic labels. Consequently, a gesture recognition algorithm has to learn what gestures it has to identify and must be provided with a mapping of the various gestures to their desired meaning.

This learning is accomplished through training and in nearly all cases involves artificial intelligence techniques [37], such as artificial neural networks [6, 10, 50] and other deep learning techniques [4], machine learning [43, 60, 5] or data-driven, stochastic techniques, like hidden Markov Models [47] and decision trees and forests [35, 29, 17]. Artificial intelligence being a entirely separate topic, only brief explanations about it, specifically in the context of gesture recognition, will be provided in this thesis. The usual general approach is to create one class/model per gesture and to integrate a classifier or classification module, which is trained (using training datasets) to classify the data properly. Then, to test the efficiency of the system, more datasets are presented to it. The more datasets it processes, the most powerful and accurate it should become.

Additionally, implementing and training efficient artificial intelligence algorithms or statistical models is a laborious task in itself and not what Superbe was seeking, at least not at the time the details of the solution were defined. Their intention was to experiment with gestures, to investigate what potential this type of solution was holding and evaluate whether or not it was a possible lead for future installations. Also, there is no computer scientist on Superbe's team. The solution could not integrate such algorithms, as it would be considerably complex for them to evolve and maintain it. Furthermore, they did not have a graphics processing unit (GPU) available for testing for the duration of the internship.

Thus, it was decided from the beginning that no artificial intelligence techniques would be applied to the solution. This thesis is therefore focused on gestural interaction through computer vision in artistic settings, with data- and feature-level processing.

The next chapter presents a state of the art on gestural interaction and gesture recognition, with an emphasis on research on gestural interaction linked to sound generation, sound control and artistic performances. It also reviews the notion of guessability for these kinds of systems.

Chapter 3

State of the art

The goal of this state of the art is to introduce the various concepts and technologies used in the field of gesture recognition and to review existing creative projects and studies. As the desired solution is an artistic sound control system, this literature review is mainly centered around this precise topic and includes a section on the concept of guessability, which can help to develop the intuitiveness component of a system.

The first section is dedicated to musical gestures, the gestures linked to music. Then, the devices typically used for gesture interaction and recognition are introduced. Next, the steps for gesture interaction and recognition are addressed alongside with studies mapping gestures and sounds. Afterwards, the guessability is described and analyzed in exhibit contexts. Finally, a few recent existing projects mapping gestures with sound or musical effects are examined.

3.1 Musical gestures

The term *gesture* can have a variety of meanings, depending on the context in which it is use. Cadoz and Wanderley [7] have covered a number of them in a survey on gestures and music. Musical gestures themselves can be defined manifold ways, including as the performer's physical actions and as the precise action of triggering musical sounds (interacting with a musical instrument for instance). The number of existing definitions hampers the writing of a universal definition, as each is valid in its own context. The common factor between most of them is the human's involvement.

The remainder of the survey focuses on what Delalande's gesture classification [15] refers to as "effective gestures": the gestures that effectively produce a sound, like blowing in a saxophone or plucking a guitar string. The other gesture categories are the "accompanist gestures", in essence the movements that the body makes when an effective gesture is performed, and "figurative gestures", the ones the spectators can detect but that do not clearly coincide with the physical movements producing the sounds, like head nods following the melody. Cadoz and Wanderley further refine the gestural vocabulary of music

with the introduction of instrumental gestures and analyze three case studies (cello, clarinet and bagpipes).

The percussionist Jean Geoffroy [20] associates what he dubbed *innate* gestures to sounds: our bodies tend to automatically materialize the sounds and their rhythms. Indeed, it is tricky to stand still whilst listening to some music or singing. These gestures however have a learnt dimension, as they often come from imitations of one's environment. He then describes the *instrumental* (or *expert*) gesture, that is to be learnt through teaching, like playing of an instrument. This gesture is more than a mere mechanical reproduction of a movement, the same expert gesture will not have the same feeling, the same personality from one performer to another, as they transmit part of their identity to the expert gestures. This is what is expressed when different artists interpret the same piece: not two interpretations are the same, as they are imbued with the personality of each performer. This way, both types of gestures are correlated, as the innate gesture is part of the expert as much as the expert gesture is part of the innate one. Artistic gestures require these two components: they demand work and accuracy but they also need a personality, a visceral element in order to truly come alive and deliver the artistic message.

Gritten and King (2006) [21] have published a first book of essays studying gestures, music and their relations from theoretical and practical points of view. They, like Cadoz and Wanderley did, identified variations in the definition of (musical) gestures, depending on the context. The trending common denominator between those definitions seems to be the initial hypothesis that a movement turns into a gesture when an interpreter (human or material) gives it matter, relevance. Musical gestures hence are dynamic, carry and transmit information; they can be categorized as manipulative (use of musical instruments) or bare-handed, intentional or not and more. The last four chapters of the book focus the accompanying gestures of the musical performance, analyzing the breathing of pianists and the body movements of clarinettists and other artists. These studies found that the additional movements have a communicative purpose and that they bring a personal dimension to the performance. This is in line with what Geoffroy explained about the expert gestures being more than simply mechanical. The book was followed by a second volume [22] on how the field of musical gestures evolved since the first publication. It introduces the concepts of musical body and musical cognition and presents a selection of essays on these subjects. The essays are more oriented towards psychobiology, psychology and cognition.

In 1974, Walter Thompson invented a gestural language for live artistic composition, called *Soundpainting* [61]. It is now composed of over 1500 body and hand gestures and allows the conductor (referred to as the Soundpainter) to lead a group of artists, composed of an assortment of performers, from musicians through actors to dancers. The peculiarity of this sign language is that it allows the Soundpainter to fully improvise. Each musical gesture is mapped to a specific meaning or action, that anyone who learned the language can recognize, making it universal. This enables real-time composition and production of novel artistic work. The instructive gestures follow a precise syntax, consisting of four types of statements: the first type indicates *Who* has to play, it can be the whole group, the musicians or a subsection of them for instance. The second kind is *What*, stating what is expected from the playing group, like long

or short tones, high or low. It can also stipulate which specific performer has to play, with the scan gesture for example. The Soundpainter scans the group with his arm and the performers who are in the continuity of his arm are the ones playing. Then comes the *How* statement, which can define the desired volume (volume fader) or tempo (tempo fader) and is optional. The final statement specifies *When* the execution has to happen (play now, stop playing, start/stop gradually). One complete gesture can be one statement, like the stop sign, or it can be composed of several statements, for example: the *Strings* are instructed play a *Long Low Tone* with a *Slow Tempo* and start *"Now"*. The statements can be categorized as functional or as sculpting gestures. The first category is for Who and When, the latter for What and How.

In this work, the chosen definition for musical gesture is any gesture that produces (musical) sounds, directly (human-world interaction, like the use of an instrument) or indirectly (human-human interaction, a conductor's gestures for example, and human-computer interaction, such as sound production using a computer). It emphasizes on effective gestures but accompanist gestures can also play a role in gestural interaction, as they can be picked up by cameras and disrupt the gestural data. Furthermore, the focus is put on gestures that are performed with bare hands, as opposed to instrumental gestures which involve a physical interaction with an object.

3.2 Devices

3.2.1 RGB, depth and RGB-D cameras

When it comes to gestural interaction through computer vision, cameras are a mandatory device. While standard webcams (RGB) can be used to capture the images, their main purpose is not gesture recognition and their images therefore can require particular processing before they can be used for it. The algorithms also has to take the camera's specifications into account. Sánchez-Nielsen et al. [53] have developed a hand-gesture recognition system that uses simple hardware such as webcams by processing and segmenting each image fast and by using the Hausdorff distance to compare the frames to a visual memory, which stores the postures' characteristics, for quick recognition. The cross-platform framework MediaPipe[1] [31] also provides, amongst other machine learning examples, a powerful hand detection and tracking solution[2] that can use only the webcam of a computer. Bazarevsky and Zhang [5] have used the framework to implement a hand gesture recognizer. They achieved this by creating a machine learning pipeline [5] composed of:

1. A model for palm detection, that isolates the hand from the original image.
2. Another model that provides the 3D keypoints of the palm detected by the first model.
3. A third model that recognizes the gestures by associating the detected gesture to the right set of gestures.

[1] https://github.com/google/mediapipe
[2] https://viz.mediapipe.dev/runner/demos/hand_tracking/hand_tracking.html

It is also possible to use classic RGB cameras in a stereo setting (passive sensing) in order to retrieve depth information [77]. A stereo configuration operates like the human eyes: using both eyes allows the retrieval of all depth information while closing one eye impairs our perception of depth.

There are however other cameras available on the market that are specifically designed for gestural interaction: RGB-Depth cameras and depth sensors (active sensing). The addition of a depth module (i.e. an infrared (IR) projector and an infrared camera) to the classic RGB module grants these devices the ability to retrieved depth information from the images and thereby provides depth maps. There are two ways of measuring the distance between objects and the camera and getting a depth map [77]:

- Structured light projection: the camera has an image of the projected pattern (projected on a plain, flat surface) stored in its memory and computes the depth according to how it is deformed by the captured scene by comparing the stored image and the captured one. The pattern can be unique or sequential.
- Time-of-flight calculation: this method computes how long it takes to the radiation to come back to the camera in order to assess the distance. The advantage of this technique is that it is not sensitive to lighting conditions.

Using the depth information also allows the recognition to be color insensitive.

These cameras have their limits, such as a limited range and a sensitivity to light (structured light cameras only) and some surface (translucent, reflective,...), as exposed by Zollhöfer [77] and Alhwarin et al. [2], but they are widely used, as they are mostly inexpensive. Alhwarin et al. provide an approach to address these problems by setting up two RGB-D cameras (Asus Xtion PRO, structured light) as a stereo system. However, using more than one RGB-D camera in such a configuration leads to an interference problem, due to the infrared projections crossing. They overcame this issue and the surface sensitivity one by calibrating the cameras, fusing the two infrared images to generate a new depth map through correspondence matching. Using a combination of this additional depth map and the original infrared images allows them to provide a precise, textured depth image. Their approach has proven efficient in improving the detection of the problematic surfaces and increasing the range of use.

The most well-know RGB-D cameras are the Microsoft Kinect[3]. They have released four of them over the years: Kinect for Xbox 360, Kinect for Windows (v1), Kinect for Xbox One, Kinect for Windows (v2). The two firsts are structured light sensors while the other two use time-of-flight calculation. The two versions for Windows were the personal computer adaptations of the ones for the gaming consoles and required the use of the Kinect for Windows SDK[4] (Software Development Kit), which provided assistance for speech and gesture recognition but is now deprecated. Plenty of researches [28, 19, 26, 9, 47, 72] have used these sensors over the years but Microsoft has discontinued the Kinect for Xbox One, the last one still produced at the time, in 2017 [1]. Although the camera is far from being the best one for tracking, its considerable benefits are

[3] https://developer.microsoft.com/fr-fr/windows/kinect/
[4] https://www.microsoft.com/en-us/download/details.aspx?id=44561

that it was not sensitive to light (Kinect v2, due to its time-of-flight calculation), extremely easy to set up, as no calibration is necessary, and cheap [68]. But Microsoft did not give up on RGB-D cameras: in May 2018, they announced the launch of the Kinect Azure[5], which was available in China and the United States of America in July 2019 [33] and in Germany, Japan and the United Kingdom in April 2020 [34]. Due to the recent release, few, if any, studies using it have been published but a fair amount of them should be expected in the near future and its popularity should grow as it is made more widely accessible.

Another RGB-D sensor is the Asus Xtion PRO[6], a structured light camera. Unlike the Kinect sensor, it was specifically designed for personal computers. Their SDK supports gesture detection, with a set of predefined gestures, and body tracking, with the possibility to track several users.

Intel has also entered the market and launched the Intel RealSense DSeries[7], a collection of three RGB-D cameras (structured light). The particularity of these is that they are stereo cameras: they are composed of the RGB module, the infrared projector and two infrared cameras. This makes them active and passive sensors at the same time and solves the interference issues pointed out by Alhwarin et al. [2], as the two lenses share a single infrared projection. These cameras are efficient in low-light settings as much as in brighter environment, where they switch to passive stereo [77]. Intel caters the Intel RealSense SDK and code samples. However, they only provide the hardware and the SDK. For skeletal tracking, gesture recognition and applications, they have teamed up with third party companies[8].

The Leap Motion Controller[9] is the most accurate sensor for hand tracking and hand gestures recognition. It is a depth camera: it is composed of two infrared cameras and an infrared projector. It is popular amongst virtual reality developers, as it can easily be mounted on headsets[10]. However, due to its focus on hand tracking, it has a restricted range and is not suitable for the recognition of other body movements.

3.2.2 Wearables

Implementing exclusively vision-based gesture recognition algorithms is possible and is the ideal solution, as other types of sensors, such as gloves and armbands, are wearable and therefore invasive. They can however be useful for researchers because they provide additional body data that can contribute to a better understanding of the operation of our body when it is interacting with gesture-based systems and consequently improve gesture recognition systems. Due to the nature of the wearables presented in this part, the accompanying studies are focused on hand tracking and hand gesture recognition.

Wang et al. [69] as well as Mazumdar et al. [32] have used colored gloves. The former use the unique pattern of colors on the gloves to deduce the posture of the

[5] https://azure.microsoft.com/en-us/services/kinect-dk/
[6] https://www.asus.com/fr/3D-Sensor/Xtion_PRO/
[7] https://www.intelrealsense.com/stereo-depth/
[8] https://www.intelrealsense.com/software-for-intel-realsense/
[9] https://www.ultraleap.com/product/leap-motion-controller/
[10] https://www.ultraleap.com/product/vr-developer-mount/

hands and the latter use monochrome gloves to ease background removal and the isolation of the hands from the rest of the body. A variation of these techniques is to place markers on the hands, such as finger caps for example. Markers can also be placed on hand-held devices, like Nymoen et al. [43] and Caramiaux et al. [11] did with either a pole or smaller devices. In these instances, the focus is more on full gesture recognition than on hand posture recognition.

CyberGlove Systems[11] manufacture a wide variety of data gloves that provide 18 to 22 joint-angle data from finger movements, depending on the design. These gloves have been used in several studies, a lot of them focused on sign language recognition [38, 46], as that was the first purpose of the CyberGlove, but others are centered around other fields, like medicine [76, 52] for instance. There are other data gloves on the market, like the HandTutor[12] or the 5DT Data Glove Ultra[13] for instance. The issue with all these is that even the basic models are expensive. It is also possible to build custom smart gloves and to equip them with any desired sensor(s), such as accelerometers, gyroscopes, inertial motion units (IMU), electrocardiograms (ECG) and more, like Perng et al. [48] did with two accelerometers and Fernández et al. [19] with two R-IoT sensors.

Another useful wearable device was the Myo armband[14], developed and commercialized by North[15]. It was equipped of electromyographs (EMG) and measured the muscular activity of the forearm through the electrical signals sent when muscles are used [42]. The muscle activity of the forearms allows to identify if the hands are in motion and what fingers are in moving. The armband also featured an inertial motion unit, used to perceive if the forearm is moving and to determine its orientation. Dongo et al. [17] as well as Tanaka et al. [60] have used it to analyse musical gestures. It was also used in a number of studies with various topics, from virtual map navigation [39] to finger placement prediction on musical instruments [14], and seemed to have a promising future. It was nonetheless discontinued in October 2018.

3.3 Skeleton detection and tracking

Skeleton or skeletal tracking is essentially applying a tracking algorithm over a human pose estimation algorithm. This latter's goal is to give an approximation of the skeleton and body joints of the user(s) using camera images.

Shotton et al. [56] have played a great role in modern skeletal tracking, as they have developed the human estimation pose core of the Kinect SDK's skeleton tracking algorithm. They were inspired by object detection systems and attempted to adjust the method to body parts detection. From a single depth image, their algorithm starts by removing the background by comparing the depths of the pixels. Then, it examines each remaining pixel and associates it to the most likely body joint it belongs to using a color code (one color per joint) using a randomized decision forests algorithm. They built huge and varied training sets for their randomized decision forests algorithm by using motion capture

[11] http://www.cyberglovesystems.com/
[12] http://meditouch.co.il/products/handtutor/
[13] https://5dt.com/5dt-data-glove-ultra/
[14] https://support.getmyo.com/hc/en-us
[15] https://www.bynorth.com/

images instead of real life images, which was very cost-effective. For each colored body part, the algorithm estimates the accurate joint position based on mean shift and a weighted Gaussian kernel.

There are other powerful and very accurate pose estimation and tracking solutions, like Openpose [16] and Densepose [17], for full body tracking, fingers included. Their algorithms are based on convolutional neural network [10, 50]. This is a complex deep learning technique and it will not be further discussed here.

Because of their flexibility and all the variations they can have, precise hand tracking is usually not supported by the basic SDK or libraries. Xi et al. [72] have developed a hand and finger tracker based on a recursive connected component analysis from a Kinect's depth frame. They start by assigning an identifier to the tracked user and process the camera's image only when the user is in the camera's field of view. They continue with occlusion recovery: it can happen that a needed joint is (partly) hidden behind another joint or an object so the system has to predict its position using invariable data like the length of the user's arm. They use the lengths of the forearm and of the hand to recover the hidden joint's position. Then they smooth the image with the help of a Kalman filter. After that, they isolate the hand from the background by removing all parts of the image that are past a specific threshold. The hand is then isolated from the rest of the body by using recursive connected component analysis (i.e. neighbour analysis). Next, they give an estimation of the hand skeleton, based on pixel-to-edge distances. Lastly, they determine the fingertips' positions by computing the geodesic distance, the shortest path between two points, on the hand skeleton pixel estimations from the previous step.

Most powerful hand tracking solutions use deep or machine learning techniques though. Mediapipe proposes a machine learning-based hand gesture recognition module [5], Simon et al. [59] have implemented a machine learning algorithm (convolutional pose machines) for hand joints detection and Sharp et al. [55] present a discriminative method and particle swarm optimization algorithm to optimize the number of possible matching models, to cite just a few examples.

The SDK corresponding to most RGB-D cameras usually support skeletal tracking and body joints detection or the camera manufacturers at least have a partnership with a company that produces such a framework, software or SDK.

3.4 Gesture recognition

There exists a large number of gesture recognition methods, most of them including artificial intelligence components. But it is not the case of all of them, like the $-family, a series of 2D gesture recognition algorithms. The $1 recognizer [71] is meant for unistroke gestures. The recognition is accomplished by template-matching, meaning that the to-be-recognized stroke, called candidate, is compared to template strokes and is recognized as the closest template stroke. To compare them, the candidate stroke has to be processed to fit the same characteristics as the template strokes. The algorithm is composed of four steps: first, the candidate has to be *resampled*. Depending on the speed of execution

[16] https://github.com/CMU-Perceptual-Computing-Lab/openpose
[17] https://github.com/facebookresearch/DensePose

and on the capturing hardware or software (touchless), a candidate stroke is not always composed of the same number of input points so it is resampled to have 64 equidistant points, like the template strokes. Then, the stroke is rotated so its *indicative angle*, which is the angle found between the gesture's centroid and its first point, is equals to 0°. This makes the algorithm rotation invariant, which means that it cannot differentiate a left-pointing arrow of an up-pointing arrow. Next, the rotated stroke is scaled up or down to a *reference square* for comparison purposes. This however makes the algorithm unable to discern small from big gestures. The stroke is also translated so its centroid matches with the origin (0,0). Finally, now that the candidate has been normalized, it is compared to the template strokes. The winning stroke is the one with the closest path, namely the one with the shortest Euclidian distance for the most points. In spite of limitations arising from algorithmic decisions, it has been compared to efficient 2D gesture recognition algorithms (Dynamic Time Warping, using template matching but using dynamic programming, and Rubine, using statistical matching) and has had better results than Rubine and similar results to Dynamic Time Warping. It can be extended to 3D gestures, as Pellegrini et al. [47] have shown.

Later came the $N recognizer [3], a multistroke evolution of the $1. There are also other algorithms, like the $P recognizer [66], which considers the gestures as point clouds (unordered) instead of strokes (ordered), or $Q [67], which is an optimized version of $P for end devices (phones, wearables,...).

Other types of distances have been investigated for template-based gesture recognition and compared to the traditional Euclidian distance. Vanderdonckt et al. [65] have compared it to the Mahalanobis distance for gesture matching and have also compared two types of edit distances, Levenshtein and Jaro-Winkler, for string matching: two gestures are converted into character strings of cardinal or compass directions and the edit distance between them is the number of operations required before one matches the other. Euclidian and Mahalanobis distances were compared using the $1 algorithm (the original one and a modified version for Mahalanobis distances) while Levenshtein and Jaro-Winkler distances were compared using the LVS recognizer (the original and an extended version for Jaro-Winkler distances). They experimented with several types of characters: lowercase letters, capital letters, shapes (flicks and marks) and arrows. The results were different from one type to another but they identified a trend: Mahalanobis distances performed a bit better then Euclidian distances overall, which themselves achieved significantly better results than Levenshtein distances. Jaro-Winkler distances were the less successful.

The general approach for gesture recognition through computer vision, as seen in [47, 29, 17, 28, 60] and more, includes the following elements:

- Gesture definition: the first step is to define what gestures the system has to recognize, what are their most representative characteristics and what state(s) or key pose(s) represent them the best, to extract the proper features and implement the algorithm so it responds adequately when a key pose is performed.
- Feature extraction: body data are extracted from the input modality (camera, armband,...). Usually, the body joints coordinates are extracted,

as most cameras/SDK provide them from the outset. They can then be processed to fit the needs of each solution. For instance, Pellegrini et al. [47] have normalized and translated them to correspond to a reference point while Jáuregui et al. [29] have transformed them (from Cartesian to spherical) to remove the impact one's morphology can have on their system's recognition.

- Gesture segmentation: the gestures are divided into smaller ones according to a time constraint in order to isolate the defined key pose(s) and recognize a gesture when it is performed.
- Training and classification: the algorithm is trained to recognize the demanded gestures. Very often for 3D gesture recognition, it is based on artificial intelligence and data science principles. It can be model-based [47], decision tree-based [29, 17], machine learning-based [60, 43],... The details of these techniques are however outside the scope of this thesis.
- Gesture to action mapping: the recognized gesture or pose is mapped to the desired action, allowing the system to trigger the action as a suitable gesture is performed. When the recognition system's purpose is to produce sound or music, the gestures are mapped to notes or sound features.

3.5 Gesture recognition and music

Nymoen et al. [43] have studied how sounds can influence a person's movements. Their goal was to discover whether sound features and movement features are correlated and if these possible correlations are personal or spread across the population. To do so, they used a Support Vector Machines classifier (machine learning) and evaluated if it was useful to highlight the correlations. They have hypothesized about several possible sound to movement relationships: the loudness of a sound influences the speed features of a movement, the pitch affects the Y coordinate (vertical displacement) and different sounds (with different features) result in different movements. They conducted an experiment in which the participants were asked to move a markers-equipped pole in the air according to the various sounds they heard. This process is called sound tracing. They have found a rather significant relationship between pitch and vertical movement: an increasing pitch results in an upward movement and vice versa. They have also identified cross-individual correlations, however slighter, between the rhythm of the sounds and the shake movements. Repetitions of the same sound sequences had a tendency to lead to repetitive movements. These results are experimental and a larger study should be conducted in order to confirm them or not.

Caramiaux et al. (2010) [11] have also conducted a sound tracing study and have analyzed the relationship between gesture features and sound features through canonical correlation analysis. This type of analysis allows to identify linear correlations between two datasets. They extracted a series of gestural features (position, velocity and acceleration coordinates, tangential acceleration, curvature, torsion and radius) but only the position, velocity and acceleration coordinates revealed relevant correlations with the selected sound features (sharpness and

loudness). They used two different sounds: an ocean waves sound and a note played on a single flute. For the first one, the loudness features was found to be correlated to the velocity and acceleration parameters of the gestures while the sharpness feature was more correlated to the position feature. The opposite was found for the latter, the position and loudness features being correlated and the other gesture features being correlated to the sharpness feature.

Later, Caramiaux et al. (2011) [12] have studied the influence the type of the listened sound can have on the features of the resulting gestures. They considered two categories of sounds: causal and non-causal. The first one is when the origin, the cause (action or object) of the sound is clearly identified, like a can opening sound or a bell sound. The latter is when the source of the sound cannot be properly recognized. They distinguished two types of gestures: symbolic and morphological. A symbolic gesture mimics the action that creates the sound whereas a morphological gesture follows the sound's temporal evolution and its acoustic features. Their hypothesis was that causal sounds would generate symbolic gestures whilst non-causal sounds would be associated to morphological gestures. The experiment they conducted verified this hypothesis, as participants imitated the action producing the causal sounds and thought of various metaphors for non-causal sounds, often referring to the sounds' temporal characteristics. However, they detected a wider variety of gestures for causal sound: a possible interpretation for this observation is that, even though the ideas behind the gestures were similar, each participant has his own visual representation of the action.

Tanaka et al. [60] proposed a machine learning-based sound tracing system that can be used to design gestures for sound synthesis. They use a Myo's IMU to extract gesture features from the user's gestures. Their system is meant for artists who are interested in integrating gestural interaction and machine learning in their shows or performances. It allows the user to select a synthetic sound and to chose which one of the four proposed machine learning algorithms (static regression, temporal modeling, whole regression or their proposed method, windowed regression) is going to be trained. Then it is time to design gestures that they think fit with the sound (sound tracing). Once the user likes the designed gesture, it is recorded and the training phase begins: the user records a training set and, once he deems the set sufficient, actually trains the chosen algorithm. This system allowed them to study the gestures designed by sound tracing and they seemed to confirm the theory of Caramiaux et al. [12], that the amplitude of the gestures tends to follow the amplitude and temporality of non-causal sound while causal sounds, like a bell sound for example, induce more powerful and sharp gestures, mimicking the cause of the sound.

When discussing gestures and music, the Soundpainting language immediately comes to mind. Indeed, it is composed of already meaningful gestures and has a formal syntax, making it a language of interest for studies combining gestures and music. Surprisingly considering the growing popularity of gesture recognition, only a handful of studies on Soundpainting gesture recognition have been conducted.

Pellegrini et al. [47] have worked on a proof of concept with that purpose, using a Kinect for gesture capture and hidden Markov Models for system training and gesture classification. They have recorded a professional Soundpainter executing

20 gestures and sequences five times with slight variations to create a database of Soundpainting gestures for gesture recognition algorithm training. Their recognition system starts by processing the captured color and depth images to recover the skeleton. They use hand blob tracking and the coordinates of the hands' barycenter (or center of mass) in order to optimize the tracking. Then, the 3D coordinates of the six relevant body joints (hand, wrist and elbow on each side) of each frame are sent to the gesture recognition module. To enhance the recognition, they use several time-sliding windows of various lengths, as gesture recognition needs a temporal dimension and not all gestures are executed with the same speed. For better performances, the skeletal data is then processed with a 3D extension to the $1 algorithm (gesture normalization, rotation and translation to fit a reference model, symmetry to deal with gestures executed with the left hand and with the right one) and forwarded to the classification module. They use one model per gesture and two to three states per model to recognize the gestures and the recognition is based on the Viterbi algorithm. In [24], they inform that they had to filter the outputs of the system, as the same gesture is identified multiple times as it is executed, due to the multiple states in the models. They consider some future possible applications for their algorithm, including using a computer for music production, its incorporation in Soundpainting performances, as an additional performer and Soundpainting teaching.

Guyot and Pellegrini [24] propose an application of the system detailed above for Soundpainting gestures analysis and annotation. This could simplify the dainty transcription task of Soundpainting scores, as it is not a regular practice amongst Soundpainters, favoring oral transmission and teaching even though Walter Thompson proposed notations. These scores would then help to analyze and document the performances. Their prototype however presents limitations: it does not detect finger movements, which are important in Soundpainting, and does not recognize gestures that require the relative position of two body limbs. For example, the system recognizes the volume and tempo faders but cannot detect the asked volume or tempo, as they are expressed by the position of the hand with reference to the forearm of the opposite side of the body Nevertheless, it can recognize the other learnt gestures and future work could solve the current issues.

Jáuregui et al. [29] have explored the real-time electronic music generation application of a Soundpainting gesture recognition system. It allows a Soundpainter to sign the six learnt gestures and produce a musical piece. The six gestures were selected because they could be detected by the camera they used, a Kinect. They simplified the recognition by basing it on a single key pose. For optimization purposes, only the upper body joints are kept by the algorithm, as the gestures they chose do not require the use of the Soundpainter's legs. The recognition process is in two phases: the first is the feature extraction step and consists of analyzing the joints' Cartesian coordinates rendered by the Kinect and extracting two features: the position of the wrists and the four arm angles (elbow angle and armpit angle, on each side). The first was chosen because wrists are a major part of Soundpainting gestures and the latter because these angles are not sensitive to scaling and therefore persistent throughout the population. Also, for this same reason, the Cartesian coordinates are translated into spherical coordinates. The second stage is the learning and classify one:

they use a Decision Tree algorithm as it can classify quickly new material. The model was trained to recognize and classify nine poses: one per selected gesture, the rest pose (arms doing nothing, lying along the Soundpainter's body) and two additional poses, used to prevent confusion. Then, the recognized gesture's code is sent over UDP to the sound generation module, *Pure Data*[18], and the appropriate sound is launched. They evaluated if their system could offer a recognition as good as humans but it cannot yet. Although, the system is expected to become more performing as it is used over time and as it witnesses the movements of various users: the system was trained to recognize the poses of one specific Soundpainter but there are different ways of signing Soundpainting gestures and each performer has his own way of performing, leading to slightly different gestures that the system could not properly identify at the time. The system offers visual feedback so the performer can see if the system recognized the gesture or not and adjust the executed gesture. They also evaluated the user experience of the system: the Soundpainter who tested it was satisfied and found the solution attractive, easy to grasp, motivating and efficient. Students also tested it in a learning context and had a similar feeling.

Recently, Dongo et al. [17] have tried to improve the aforementioned system by using a second input modality, two Myo armbands (one for each arm). By means of them, it is possible to tackle the lack of hands and fingers data, which are useful to recognize some Soundpainting gestures. Their goal was to assess:

- If the Kinect is more accurate than the Myo for upper body movements recognition.
- If the combination of sensors provides a higher accuracy than each individually.
- If combining the features from one sensor offers a better accuracy than using each on its own.
- If the data acquisition rate difference between the two types of sensors affect the accuracy.

To begin with, they created a Soundpainting gesture database, including 14 gestures with 50 repetitions each. Three of the gestures are variations of the same one. They made the database public, to encourage other researchers to replicate, test their recognition system and implement others.

The first step of their approach was to sync the Kinect and the Myo armbands, as they originally do not have the same data acquisition rate. Then they segmented to gestures into poses: the initializing pose indicates the start of a new gesture and the following poses, each 840ms apart, are part of it. The gesture ends as the initializing pose is visited again. The extracted features were similar to the previous system [29] for the Kinect (hands positions, armpits and elbows angles and spherical coordinates) and were the mean absolute value of each armband, as each is composed of eight EMG sensors, and the waveform length for the Myo. They used a random forest algorithm for training and classifying. Before the classification step, they operated a reduction of the number of key poses, to only preserve five of them for each gesture, no matter how many

[18] https://puredata.info/

were captured. This way, the classifier is fed same-sized inputs and limiting the size of the inputs improves its performance. Then the classifier, based on the key poses, determines to what class the gesture can belong. In essence, a class corresponding to a gesture, the classifier suggests what gesture has been executed. Its output is the probability that the detected gesture belongs to each class. With their algorithm, any gesture, even if it is not part of the database, is classified, as the classifier outputs belonging probabilities.

They found out that the different data acquisition rates do not affect the accuracy of the system and that the Kinect is indeed overall better at body movement recognition than the Myo armbands, except for four of the gestures present in the database. This would imply that hand and finger movements are more critical for some gestures than for others. However, the combination of the sensors does not increase the accuracy of the system, besides for two gestures, and could even decrease the classifier's efficiency. A solution to this issue might be the use of other multimodal fusion methods. The combination of features from a same sensor does not improve the accuracy either. They plan on conducting correlation studies, to combine features that can work together to address this last statement.

3.6 Guessability

The guessability of an icon, symbol or command is the extent to which it is easy for the users to guess and understand what the purpose of the said icon, symbol or command is without previous knowledge of what it effectively achieves or means [70]. This definition can be scaled up to a whole system, where its guessability is the extent to which it is easy for the users to guess and understand how they have to interact with the system for it to perform properly. The guessabiliy measures the quality of any of the forementioned inputs and more. It is linked to well-known human-computer interaction concepts: usability and user experience.

Wobbrock et al. [70] have studied the guessability of symbolic inputs. An example of such an input is the use of a button on the screen to launch an application. If the users cannot deduce the purpose of the button, they will not use it. To tackle this issue, they asked the end-users to design symbols. The procedure they followed is the pattern for any classic guessability study: participants are asked to design their ideal input for the various tasks or system actions that are presented to them. At the end of the session(s), the inputs proposed by all the participants are gathered and compared. The similar inputs are clustered and analyzed. If similar inputs have been recommended for different actions, the most represented action of the cluster "wins" and is a candidate to winning the mapping to the action. This step is called "conflict resolution". The next step is the guessability computation of each proposed input and it outputs the proportion of participants who have recommended what input for what action. The input with the highest ratio is the selected one for the action. In other words, the input chosen by the most participants for an action wins. Then, measuring the agreement level can provide valuable information: this measure shows the proportion of participants that recommended the winning input for

each action. This approach has shown good results and can be used to evaluate existing sets of actions or symbols. It is also applicable to any type of human-computer interaction, including gestural interaction.

Cafaro et al. (2014) [9] proposed a variation of the classic guessability studies, using what they called the framed guessability. Their goals were:

- To discover whether or not the gestures proposed by the participants of classic guessability studies are correlated. In other words, to learn to which degree gestures designed for specific actions are similar from a person to another. This would imply that people imagine equivalent scenarios involving common concepts and knowledge (embodied schemata).
- To determine if the use of allegories increases the guessability rate and agreement percentage of the suite, as metaphors should guide the reasoning of the participants and therefore limit the total number of proposed gestures.
- To find out if a suite of control actions designed with the help of an allegory is composed of complementary gestures.

They have carried out a three stages experiment. The first one was conducted as a classic guessability study where participants were asked to propose a control action for each of the 12 animations that were presented to them. The second phase was the one of framed guessability and consisted of priming the participants then asking them to select, amongst the different gestures proposed during the first phase, the most appropriate gesture for each control action. The priming was realized on three levels with the allegory of the mirror, with a Kinect live streaming the participant's body shape on a screen to simulate a virtual mirror (visualisation level). The screen was referred to as a virtual mirror (instruction level) and the screen was framed like a mirror would be to reinforce the resemblance (physical level). The winning suite was different from the one of the first phase but some gesture-to-effect mappings won in both sets. The final stage's goal was to identify the preferred set: participants were presented the winning gesture suites of the previous phases and were asked to chose one or the other as the most fitting and intuitive one in their opinion. The favored suite was the one from the second phase, as they deemed it more intuitive and therefore more consistent, which confirms the third hypothesis.

The results of the experiment validated the other hypotheses too, as they showed that stage one participants produced related gestures by thinking of similar, known metaphors, even without priming. For instance, the popular control action to move items on the screen was moving the arm, like people use their arms and hands to move physical items. They also showed an increased agreement level from the first to the second stage: the use of a metaphor wired the minds of the participants and they were able to propose a less divided gesture set.

Recently, Cafaro et al. (2018) [8] have revised the original version of the framed guessability, as flaws in the approach were identified. First, the mirror allegory might not translate well in other contexts: a mirror cannot be easily disguised as a store or a gas station for example. Another weakness is that the framed guessability participants (second stage) were not actually asked to generate gestures, they only could chose from a selection of gestures. It makes it challenging

to tell if the increased agreement levels noted were due to the priming itself or to the fact the participants had limited options. Moreover, the restricted set presented might itself be compromised, as it was generated without priming. Hence, it might be composed of disconnected, unrelated control actions, which might undermine the study results.

They have conducted another study, where they have compared the discoverability of different gesture sets, generated through classic or framed guessability, in a real-life (in-situ) museum experiment. Their objective was to prove that the framed guessability suite would be more successful than the traditional guessability one. They created three groups, two were primed with two different contexts (group A with gym conditions and group B with a funhouse situation) and the third one was the unprimed control group (group C, classic guessability study). The first step was to do the priming with group A and B: participants were shown pictures related to their assigned situation (gym or funhouse), asked to list five things they would do in such a situation then to enact these five actions. Next was the (framed) guessability step for the three groups: all participants were presented six effects and were asked to suggest a control action for each. This generated one winning suite per group, as the control actions for groups A and B were influenced by the priming contexts, and concluded the in-lab part of the study. Afterwards, the control actions sets were evaluated in a museum exhibition, where visitors were asked to figure out how they could control the various effects of the systems. The effects were the same as the ones presented to the elicitation study participants and the visitors were not primed. They discovered more gestures of the A and B sets, compared to the C set, which confirms the hypothesis that framed gesture sets would be more discoverable than traditional sets. This study proves that gesture sets following a conductive thread are more user-friendly than disconnected sets.

3.7 Existing projects

3.7.1 Expressive Control of Indirect Augmented Reality During Live Music Performances

Hoste and Signer [28] have developed a Kinect-based gesture recognition system that allows artists and performers to control the visual effects of their performances[19]. This gives them more room for improvisation, as they can adapt their show to their mood or to their audience, instead of having to keep up with pre-scheduled effects. The gestures are pre-determined, as the performer has to know which gesture triggers which effect and the system has to know which effect to trigger when it recognizes a gesture, but it nonetheless give more flexibility to the artists. To perform well, such a system must reach high precision and recall levels: it must be precise enough to trigger the effects only when desired and it must always activate the appropriate effect when a key gesture is executed. This second statement is achieved through high recall.

The proposed approach was to take a single sample of the five full body gestures, each being divided in a sequence of key poses over time. Only the whole

[19] https://www.youtube.com/watch?v=nyVs_5TfN4c/

sequence timed properly triggers the visual effect, this provides a higher degree of liberty of movement while avoiding unintended triggers. The Kinect captures and sends using the OSC protocol a continuous live stream that is processed by a 3D gesture recognition extension of the Mudra framework [27], which is used to define declarative control points (i.e. rules) from 3D data. The use of a declarative, rule-based language provides a precise description for each gesture, that enables single sampling and facilitates the explanation of the various constraints the artist's gestures have to observe. The output of the system is the superposition of the live stream and fire visual effects on various body parts, depending on the gesture.

The proposed system performed perfectly and satisfied the artists as well as the audience in terms of responsiveness and accuracy. Even if it was only used to augment a single performance of the show, the performance brought excitement to the audience. Future work for this project involves expanding the system to more performances and the deployment of an integrated development environment (IDE) that would allow the community to experiment with their approach: it would help its users to create their own 3D gestures, generate the control points and translate them into comprehensive declarative rules.

Figure 3.1: Live performance of the fourth gesture, as shown by Hoste and Signer [28].

3.7.2 GeKiPe

Fernández et al. [19] present a gesture-based musical and visual live performance system called *GeKiPe*[20], which stands for "Geste, Kinect et Percussion" in French and translates to "Gesture, Kinect and Percussion" in English. It consists of a Kinect and two gloves equipped with R-IoT sensors. These sensors are composed of three gyroscopes, three accelerometers and three magnetometers each. They capture the movements of the performer and allow him to trigger and control the elements of the performance, namely the sounds and the visual effects and images. The dedicated zone for the performance is divided in 18 cubes: three blocks (low, mid, high) of six cubes each (front left, front center, front right, back left, back center, back right) are stacked and each of the three axes (x, y, z) can be mapped to a parameter for sound control.

[20] http://philippespiesser.com/projet/gekipe-geste-kinect-percussion/

The Kinect tracks the skeleton of the performer and the gloves sense the orientations, the accelerations, the inclinations of the hands and kick motions. They can also retrieve relative angles. All the data gathered are sent over the OSC protocol to the sound mapping module Antescofo[21], except for the R-IoT sensors' gestural data: they are analyzed within the sensor as it is faster to do so than to send them to another module over WiFi. Thus, the features (orientation, inclination, speed and angles) are instantly computed or retrieved and Antescofo directly receives the information it needs without having to process it itself. Antescofo dynamically maps, live, the gestural information it acquires from the sensors to sounds to provide continuous control to the performer. They implemented two types of mappings: one for sound generative performances (full improvisation) and one for existing sounds and effects triggering, still with room for improvisation. The mappings can change at any point during a performance: they can evolve over time, be affected by gestural data or musical sequences for example, leaving it to the performer to decide whether he wants to follow a composed performance or improvise. Improvisation will adapt the Antescofo score in real-time.

Visual mappings, for their part, are treated by an engine programmed with openFrameworks. The raw images can be treated directly, augmented or their vectorization can be further processed by the computer vision library OpenCV[22] for more visual effects (contouring, deformation,...). These mappings are also dynamic.

The system has been used in three different contexts: in the Sculpt performance[23], in workshops with educational purposes and for score transcription. The first is a live performance where the performer plays the invisible drums in mid-air. It is composed of two acts, a generative one and a scripted one. The second context intended to teach its participants to link body movements to audio and visual effects. The final one's goal was to write down scores for gestures in order to allow other musicians to play them.

GeKiPe has proven it performs as intended and the authors plan on providing a multi-users version as well as reinforcing the connections between sounds and visuals and working on image sonification.

[21] http://antescofo-doc.ircam.fr/
[22] https://opencv.org/
[23] https://ensembleflashback.com/spectacle-sculpt/

Figure 3.2: Excerpts from Sculpt (performance), as pictured by Fernández et al. [19].

3.7.3 Musical Brush

Valer et al. [63] have created an augmented reality-based mobile application for musical control[24]. It takes advantage of some of the sensors already integrated in smartphones: the touchscreen's pressure sensors, the accelerometer and the camera. The user has to press the screen then draw directly on it or move his smartphone in the air to generate and modulate one of the sounds available. The 3D position of the smartphone, its speed as well as the pressure applied to the screen are collected and used to control the selected sound. It is possible to record short performances and play up to four of them together to create a personal musical piece.

Their approach is the following: the first step was to define what actions would be used to control the system and to acquire the necessary smartphone data (3D position, movement and acceleration, pressure on the touchscreen). Then, they mapped these actions to the control of various sound features:

- Touching the screen launches a sound and removing the finger from the screen stops it.
- Moving the finger or the smartphone on the Y-axis causes the frequency (i.e. the pitch) of the sound to increase or decrease when moving up or down respectively.

[24]This paper is a preprint but has a Creative Commons licence.

- Applying more pressure on the screen increases the amplitude (i.e. the playing volume) of the sound and conversely when applying less pressure.
- The timbre of the generated sound can be chosen out of a list of four. It can also be referred to as the color or quality of a sound, of which it is one of the characteristics. It corresponds to the waveform of the sound, which is specific to each instrument and each voice, as it partly depends on the physical characteristics of the instrument producing the sound, such as the shape and material of the instrument for instrumental sounds and the shapes, openings and contractions of the mouth and vocal cords for voices [45]. Dynamic characteristics, such as the envelope[25] and vibrato[26]/tremolo[27], as well as harmonic contents[28] also take part in the determination of the timbre [41]. In short, it is what allows one to identify the different sources of sounds with identical pitch and intensity.
- Shaking the smartphone or sliding the finger fast on the screen generates an echo.

Next, they implemented the sound generation process using Pure Data. It creates a sound according to the data-to-feature mapping it receives and a converter transforms the created sound into proper sound. Sound is not the only feedback the user gets: visual feedback is provided, as the drawing corresponding to the sound fades in and out as the sound passes by, and haptic feedback (vibrations) is produced for the echo effect.

They have realized an evaluation of the application with 17 participants. They provided a mainly positive feedback but would appreciate a wider range of sounds to chose from, such as instrument-like sounds. They also identified a sensitivity issue with the amplitude control and found the echo effect trigger a bit counter-intuitive.

The authors plan on upgrading the amplitude control as well as the echo effect for future releases. They are also working on integrating more sounds.

NOTE: The application has been tested in January 2020 and can now record up to six short performances. It is difficult to assess whether or not the amplitude and echo control have improved, since the original application was not tested. However, the four original sounds are still the only ones available.

3.7.4 SICMAP

Héon-Morissette [26] has imagined and conceived this interactive system, whose acronym stands for "Système Interactif de Captation du Mouvement en Art Performatif". The project stemmed out of a rather philosophical reflection around her artistic praxis and the concept of transdisciplinarity, which aims to understand our world by combining aspects of various disciplines to build hybrid solutions [25]. The purpose of the installation is to make sound tangible and create what the author refers to as the "gesture-sound-space". In this space, it

[25] "Attack, sustain, and decay of a sound" [44].
[26] "Periodic changes in the pitch of the tone (frequency modulation)" [41].
[27] "Periodic changes in the amplitude or loudness of the tone (amplitude modulation)" [41].
[28] "Number and relative intensity of the upper harmonics present in the sound" [41].

is possible to perceive the sound with the ears but also with the eyes, through the performer's movements, referred to as "gesture-sound".

Substantially, the system allows the performer to use their body and gestures as a musical instrument, to generate sounds with their movements. They can also control visual effects. This is achieved through computer vision with a Kinect. It captures the motion of the artist and the video stream is treated by an application called Kinect Kreative Interface, used to represent the tracked skeleton in a 3D performance area containing subspaces mapped to the performer's body parts. This way, the body is virtually divided and different body parts can be mapped to different sounds or visual effects. Several mapping have been investigated, it started with one-to-one mappings but developed, after a long period of time, into one-to-many mappings. Due to their lack of the power capacities necessary for real-time tracking, skeletal data processing, mapping and audio processing all together, the system only offers simple sound modulation (reverb and various filters) to guarantee a satisfactory execution.

Controlling SICMAP properly and generating artistic pieces with it requires accuracy and therefore a lot of time and practice, it is dedicated to professional and devoted performers. Future work involves experimenting with other sensors and other processing modules. The author plans on further refining the expressiveness of the system and on collaborating with other musicians.

The upcoming chapter deals with the implementation process for the solution, from the initial discussions with Mr. Bertrand and Mr. Libertiaux to the different iterations and associated technology trials, and with a possible evaluation method.

Chapter 4

Solution design and implementation

This chapter aims to explain in detail the thinking, discussions and technology assessments that lead to the implementation of the different proofs of concept. These will also be described here. It will be followed by a description of the evaluation process.

4.1 Requirements specification

The goal of the internship at Superbe was to provide a lead for a gesture-based evolution to Superbe's SMING installation. For reminder, the current installation allows the user to control MIDI notes using a conductor's baton equipped with a gyroscope.

Before anything else, the **scope of the project** had to be defined. Mr. Libertiaux and Mr. Bertrand were consulted in order to understand how they envisioned the system and what their requirements were. Their wish was to create a similar installation but incorporating gestural interaction. As the system's goal is to mimic a conductor, it was agreed upfront to limit the interaction to the upper body, specifically to the arms and hands. There were two possible evolution paths: completely removing the baton to use both hands for gestural interaction or keeping the baton and only using the free hand to allow the control of more sound features. It was decided to follow the first one, as it was deemed the most intuitive. Indeed, they were worried that some users could be puzzled in front of the other system, not knowing what is exactly expected of them or not using it to its full potential. Also, in case the two-handed interaction failed to meet their expectations, it would have been simple to join the second path, by downgrading the system for it to only focus on one hand. Another desire was to give the users more control over the system, by allowing them to control the pitch and the rhythm of the music.

As stated in the Introduction of this document, the prominent **constraint** for the system is that it must be exclusively computer vision-based. The integration

of wearables would make the technology behind the system highly apparent and the way the system works would therefore be too obvious. Moreover, the interactions with the system occurring in public gatherings, gearing up each user, then taking off the possibly multiple accessories would be extremely time-consuming and would impact the success of the installation. Another point that could have been a constraint was to code in Processing (Java-based), which is a language meant for beginner programmers and an IDE for coding visual arts applications. Mr. Bertrand indeed programs the installations Superbe create but he does not have a computer science background and Processing is the language he is used to. For exploration's sake, it was not made mandatory as it would have drastically limited the implementation options. A significant effort has been made to find and use a suitable library but in vain.

There is a single simple **use case** for this system: the users stands in front of the camera and moves his arms and hands and the system reacts to the movements.

This project is meant to explore the opportunities that gestural interaction offers in artistic installations. There is not one unique way of implementing such systems, each camera, each method has its advantages and limitations and there exists several possible movements to sounds mappings. The selection can only be done by trial and error. Consequently, there is no specific functional requirement regarding the system's reaction to the user's movements. The **functional requirements** are:

- The system should capture a video stream.
- The system should detect and track the skeleton and the hands of a single user. It should also render the coordinates of the upper body joints for each video frame.
- The system should compute features from the extracted data.
- The system should allow the control of sound features using movement features.
- The algorithm should communicate over OSC with Cycling '74's Max8[1], in charge of the sound treatment of the original SMing installation.

The user experience is important for any system but it is truly crucial for interactive artistic exhibits, as their main purpose is to generate interest and fun for the users. The usability of the system is paramount to reach satisfactory user experience levels. It includes various concepts, like the ease of use, the attractiveness, the effectiveness and the efficiency. In this context, there should be a balanced tradeoff between those last two concepts, as the system should allow the user to fulfill a task but it should also encourage him to explore the system to find out by himself what its purpose is. In other words, discovering the operation of the system is part of the fun so the system should not be overly straightforward but should not be excessively complicated either. This meets with the concept of guessability [70, 9, 8], also part of the usability. If the commands are exceedingly difficult to find out, the exhibit will be a failure, as most users will not do their utmost to understand what they are supposed to do: they

[1] https://cycling74.com/products/max

will get bored with the system and leave it there because their enjoyment and interest levels will significantly drop. Also, an interaction in a public exhibit imposes a timing constraint, one user cannot monopolize an installation indefinitely, there has to be a turnover to give a chance to every visitor to interact with the system. Considering all of this, the **non-functional requirements** of the solution are:

- The system should immediately react to the user's movements. Latency in artistic systems is an absolute deal breaker, as it deeply affects the interaction and, by extension, the user experience.
- The system should appeal to the user, have enough attractiveness to lure him in and make him want to interact with it.
- The system should be playful and fun to deliver a great user experience, otherwise it misses the point.
- The user should not be assisted to start using the system. Explanations would lengthen the interaction, reducing the turnover, and would also impair the fun component of the user experience, as the user would have been instructed what to do.
- The system should be easy to use at any age, as the public can comprise children and older adults. The technology should not be a barrier for less technologically experienced users.
- The system should be understandable and intuitive (guessable), as Superbe want to encourage the user to experiment and understand by himself the functioning of the installation. But the operation of the system must not be obvious to the point it would lower the level of interest of many users.
- The system should keep an artistic touch to give a satisfying result to the users, no matter their previous musical experience.

The scope, constraints and requirements were subject to change during the whole project, as the gradual introduction of different prototypes would refine ideas and lead to a better understanding and identification of Superbe's expectations.

4.2 General working methodology

The followed methodology was iterative and adaptive. Superbe does not have a team of programmers so the project did not involve team work. It was conducted by a single person, in charge of the analysis and of the implementation, who reported to the stakeholders, Messrs Libertiaux and Bertrand.

The first phase was dedicated to the evaluation of the available technology, namely the cameras and existing projects. The idea was to select the best available sensor and to find a fitting library or SDK to use for skeleton tracking, as creating one from scratch is a lengthy task and not a project of Superbe.

They already owned several RGB-Depth cameras and a depth camera and they were all tested. For each, a lot of libraries and a few SDK are available and a variety of them were put to the test as well. By the end of this phase, the various combinations alongside with their strengths and weaknesses were presented to Mr. Libertiaux and Mr. Bertrand so they could make an advised decision.

The second phase consisted of three programming iterations. During each of them, the basic features were developed first and were followed by more advanced ones. The first one resulted in the presentation of a first proof of concept, that was tested by Messrs Libertiaux and Bertrand. The project's requirements were adapted according to their opinions, based on the discussion that followed the test. The second iteration was similar to the first one except that the proof of concept could not be tested by either Mr. Libertiaux or Mr. Bertrand due to the COVID-19 pandemic and the ensuing lockdown. Instead, they were provided a demonstration video. This resulted in further changes in the solution's requirements. The proof of concept of the final iteration, also presented in a video, was judged satisfactory.

4.3 Technology assessment and selection

Superbe already own a handful of cameras. For economic reasons, the potential of these cameras was assessed before considering buying a new one. Two Microsoft Kinect (v2 then v1), a Leap Motion Controller and an Intel RealSense D435 have been put to the test. Figure 4.1 summarizes their specifications. They were tested on a total of three computers:

- First on a MacBook Air (2013) under Catalina, as Superbe originally did not provide a computer. The Kinect cameras being products of Microsoft, their SDK are therefore not compatible with macOS. The first step to use a Kinect on macOS is to install a driver, libfreenect for the Kinect v1[2] or libfreenect2 for the Kinect v2[3] [73]. Without it, the computer cannot use the data from the camera. The rest of the tests was a global failure. Dealing with skeletal data using a Kinect usually involves the help of OpenNI2, for which there was an installation issue. Moreover, there was no version of its Java wrapper for Processing, SimpleOpenNI[4], compatible with the Kinect v2 and a version of Processing that was still available to download. The Kinect v1 was tested in Processing with SimpleOpenNI and most of the samples were working. However, as the Kinect v1 is the oldest camera, it was decided to continue to experiment with other, more recent cameras and, if there were no other solution, come back to it later. Due to the recent release date of Catalina, most of the still active Kinect libraries are not compatible with it. An option was to run Windows in a virtual environment but Superbe offered to find a Windows computer instead. Indeed, they work on Windows so insisting on make it work on macOS would have been useless and a loss of time. The Leap Motion and the RealSense were also tested on this computer and performed terribly.

[2] https://github.com/OpenKinect/libfreenect
[3] https://github.com/OpenKinect/libfreenect2
[4] https://github.com/totovr/SimpleOpenNI

The issue with the first was a compatibility issue and was latency and low resolution with the RealSense SDK for the latter. There is however a library that brings the RealSense SDK to Processing[5], which significantly increased the performance of the sensor on this computer.

- Then came a light Asus laptop under Windows 10. It struggled a bit with heavy applications like Visual Studio but the sensors were working. The lender however needed it back after a short week.
- A Lenovo YOGA under Windows 10 was finally used. This computer was still basic but more powerful than the Asus and performed well overall. The detail of the tests is presented below.

	Microsoft Kinect v1	Microsoft Kinect v2	Leap Motion Controller	Intel RealSense D435
Type of camera	RGB-D	RGB-D	D	RGB-D
Type of sensing	Structured light	Time of flight	Stereo	Stereo structured light
Lighting sensitivity	Yes	No	Yes	No
Production (camera and SDK)	Discontinued	Discontinued	Ongoing	Ongoing
Range	≈ 0.4 to 3 m	≈ 0.4 to 4.5 m	≈ 0.25 to 60 cm	≈ 0.1 to 10 m
Availability of skeletal tracking	Yes but for how long?	Yes but for how long?	Yes	Yes

Figure 4.1: Kinect (v1 and v2), Leap Motion Controller and Intel RealSense D435 specifications.

4.3.1 Kinect

Kinect cameras have been use widely and a lot of libraries were created for them. They are however past their glory days and most of the libraries have now been forsaken, as the sensors are discontinued. This means they are not maintained nor evolved for recent operating systems. A lot of libraries do not offer compatibility to Windows 10.

Using the Kinect v2 in Processing on Windows would have given the same results as on macOS, as changing the operating system does not solve the SimpleOpenNI library version to IDE version incompatibility issue.

Mr. Bertrand's second choice for development environment was openFrameworks[6]. It is a series of tool for visual designs, much like Processing but in C. A lot of Kinect-related openFrameworks addons were available but the vast majority of them has not been update in over five years and is poorly documented.

Kinect-Finger-Tracking[7] proposes a finger tracking solution for Kinect but it was optimized for Visual Studio 2013 and this version could not properly be installed on Windows 10. It was tested in Visual Studio 2015 but there was a "type of solution mismatch" and no sample could run.

The framework Vitruvius[8], which could be called the commercial extension of Kinect-Finger-Tracking as it was created by the same person, provides code

[5] https://github.com/cansik/realsense-processing
[6] https://openframeworks.cc/
[7] https://github.com/LightBuzz/Kinect-Finger-Tracking
[8] https://vitruviuskinect.com/

samples and tools for easy development with Kinect. It is optimized for Windows 8 but they state that it should run on Windows 10. Its free version was first tested in Visual Studio 2017 and then is Visual Studio 2019 but there were again "type of solution mismatch" errors and the samples could not run. It was finally tested in Visual Studio 2015 where, after a few fixes, some samples could run. Vitruvius is also compatible with the RealSense but only the paying version.

As few to no support and new libraries for the Kinect v2 are expected in the future, Superbe chose not to further investigate it for the time, to focus on other, still available sensors instead and perhaps resume the research with the Kinect if no other camera is satisfactory. Moreover, creating a new system with a nearly obsolete sensor and an almost deprecated library could lead to performance and maintenance issues in the future.

4.3.2 Leap Motion Controller

A Leap Motion Controller has also been briefly tested. While the tracking of the hands was very pleasing, the major issue with the controller was its limited range.

4.3.3 Intel RealSense

Next up was an Intel RealSense D435. The SDK installation went smoothly and all samples were running properly from the beginning. The advantage of this camera is that it is partly advertise for low-light environments and this is the kind of situation needed for SMing, as the visual aspect of the installation involves projections.

Even though the samples were running (in Visual Studio 2017), the realsense-processing library was tested. It is efficient but does not support skeletal tracking. No library supporting it was found and Processing was then set aside. There is now an early version of a Processing library supporting skeleton tracking for the Intel RealSense[9] but the development of the solution had already started when it was posted. Also, as the library was brand new, it was decided to go on with the on-going implementation before starting over in Processing, to allow the library to mature a bit first.

Implementing a lightweight version of Openpose was considered but it would still have involved deep learning techniques. Also, finding a useful ready-made tracking solution was a better option than creating one, as it would have taken a lot of time before focusing on the artistic interaction.

In the end, there were two candidate SDK:

1. Nuitrack SDK[10]: this is a popular tracking solution and, from what the executable file showed, it performs well. However, there was an issue with the code samples: none of them could detect the Intel RealSense camera and it was not possible to run anything. The Nuitrack community's help has been solicited but no answer was provided.

[9] https://github.com/cansik/deep-vision-processing
[10] https://nuitrack.com/

2. Cubemos Skeleton Tracking SDK: this SDK is less known than Nuitrack and was found through Intel, which actually have a partnership with them. This warranties a good compatibility now and in the future. Another advantage was that they offer a month-long trial, which was ideal to test the SDK. The samples performed well and could be launched from Visual Studio 2017.

The Intel RealSense D435 paired with the Cubemos Skeleton Tracking SDK gave an acceptable tradeoff between satisfactory tracking quality, recent technology and pricing. This combination was therefore selected for further development.

4.3.4 Webcam

An option was to use a webcam in addition of the RGB-D sensor to accomplish precise hand tracking and hand gesture recognition and fuse the extracted features from the two sensors (multimodal fusion [18]). Mediapipe offers a potent hand tracker using a computer's webcam and Bazarevsky and Zhang have developed a hand gesture recognition system using it [5]. However, running it at the same time than the skeletal tracking SDK would add a significant load on the Lenovo computer and could result in latency. Also, according to Messrs Libertiaux and Bertrand's experience with SMing, the average user makes big arm movements and does not pay attention to details like hand gestures. Futhermore, as Dongo et al. have found out for Soundpainting gesture recognition [17], using hand data does not improve the recognition of arm movements. This lead was thereby abandoned, as it would have required a lot of time and additional work for limited added value.

In the light of the above decisions, Figure 4.2 pictures the global architecture of the solution. The processing and the sound generation are executed on the same computer but two computers might be used in real conditions. The captured video stream is processed by the system, then the relevant data are sent to Max8 using the Open Sound Control protocol and Max8 triggers the appropriate sound. Proper speakers would obviously be used for sound output in exhibit situation.

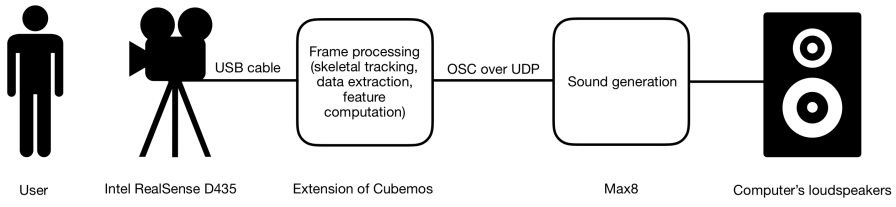


Figure 4.2: Architecture of the proposed system.

4.3.5 Extracted data and features

Working with a RGB-D sensor and a skeleton tracker for gestural interaction involves depth information and body joint coordinates retrieval. They can be

in 2D (usually Cartesian) or in 3D (usually Euclidian or spherical). From these coordinates, a series of body features can be computed: the length of body parts, the angles between them, their directions and their relative positions. Adding a temporal aspect allows to compute the speed of the movements, the velocity and the acceleration. Sequences of movements can also be identified.

In case of sound control or generation systems, it is important to specify the sound features that are to be interacted with. This allows the mapping of gestural data or features to the features they have to control. The characteristics of a sound are: its note (C, D, E, F, G, A, B), its frequency (pitch, i.e. how high the note is) and its amplitude (loudness, or volume), its duration and its timbre (or color, defined by the waveform and allowing to differentiate two voices or two instruments for example). In this project, there is no timbre control, as the sounds are derived from the user's voice. During the experimentation, synthetic voices have been used.

4.4 First iteration

It was decided to first start small and try with simple static mappings then, from there, increment as necessary. The hypothesis was that simple one-to-one mappings (one movement feature controlling one sound feature) would be the most guessable system for the users and therefore be the most satisfying and fun to use.

4.4.1 Implementation

A C# sample that was originally rendering the Cartesian coordinates of all body joints from a single frame was extended to deal with a continuous stream of frames. As the focus was on the upper body, the lower body joints were discarded to lighten the process. Some of the upper body joints were also removed as they were not used. The remaining joints were the left and right wrists and the head joint.

The algorithm was in charge of processing all the data then sending a codified OSC message to Max8 with sound control instructions. For this proof of concept, Max8 could receive messages composed of three arguments:

1. An integer between 1 and 4 indicating to which voice of the choir the rest of the message is addressed. When two voices has to play, two messages were sent, as they are voice-specific.

2. An integer between 0 and 7 stating which note is expected, as there were 8 notes, making up one octave (C, D, E, F, G, A, B, C).

3. An integer between 0 and 127 specifying the desired playing volume.

There were three possible cases, depending on the posture (see Figure 4.3):

1. Arms hanging along the body (rest pose): if the Y-coordinates of both wrists were below a defined threshold, no sound was being played. When

the algorithm identified this situation, it sent four OSC messages (one per voice) to Max8 to inform it to change the volume of each voice to 0.

2. One arm above the threshold (meaning that one arm is considered active): the field of vision was divided into a 4x8 grid (Figure 4.4), similarly to what Fernández et al. [19] did with GeKiPe but only in 2 dimensions. The horizontal axis was composed of four cells, each corresponding to one voice from the choir and therefore controlling the pitch. The deep, low voice was on the far left and the highest was on the far right. The position of the wrist on the Y-axis allowed the control of the note. In total, each of the four voices could play one octave, in its tessitura. The original idea was to map the movement on the vertical axis to the pitch, as shown by Nymoen et al. [43] and Valer et al. [63], but it made more sense for this installation to map the pitch to the X-coordinates as the different voices are emitted by different loudspeakers spread across the stage in front of the user: the deep voices come from the speakers on the left side and the higher voices from the speakers on the right side. Mapping the cells of the vertical axis to notes making up an octave still provided a sensation of highness or lowness control, as the notes of an octave gradually "go up" until the same note as the first one is reached again, only one octave higher. In this case, the volume of each note was the default volume of 65. If the arm was not moving, that is when the wrist coordinates were stable, the note was maintained.

3. Both arms above the threshold (meaning that both arms are considered active): this case was similar to the previous one, except that the horizontal axis was divided into two cells, allowing the four voices to sing at the same time. The two low voices were controlled by the left arm and both played the same note while the right one was controlling the two high voices, playing a common note too. If the two arm were moving together vertically (i.e. if the difference of the Y-coordinates of the wrists was small enough), the same note was played on both sides by all voices but it was also possible to play different notes on each side. Also, it allowed to control the volume of the notes. This was achieved by calculating the difference between the X-coordinates of the wrists: the biggest the difference, the louder the notes.

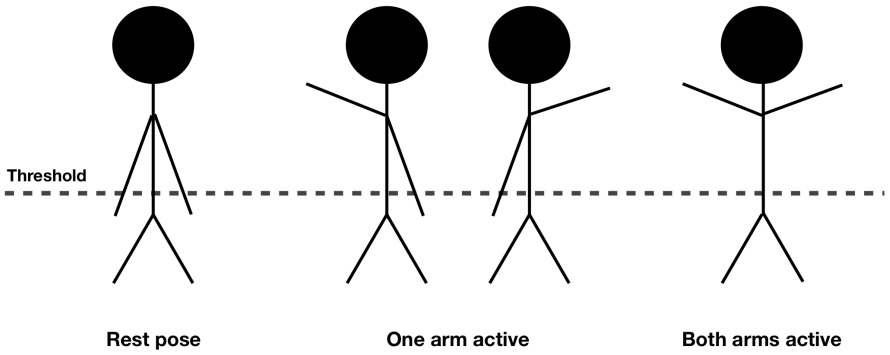


Figure 4.3: The body poses considered in this proof of concept and in the next ones.

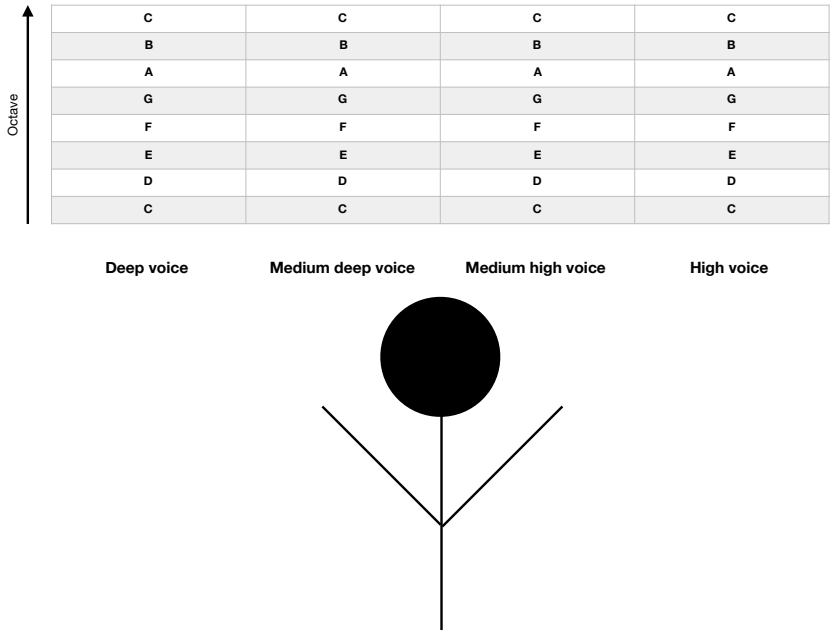


Figure 4.4: Grid system of the first proof of concept.

4.4.2 Limitations

This proof of concept had some limitations. Possible fixes for some of them were planned but the prototype was first off tested by the supervisors.

First, it did not provide control over the volume when playing with only one hand. This could have been fixed by computing the speed of the movements. The speed could then have been used to control the volume of the two hands case, allowing the use of the X-coordinates to control each voice individually.

Moreover, it was not possible to jump from a C to a G without playing all the notes in between. A feasible fix for this issue could have been to use the speed too, playing a note only if the wrist is moving slow enough in its cell. Another option would have been to process only every other frame or so but this could have resulted in latency.

Additionally, the freedom of composition was limited: even though there was a total of 32 notes available, only up to two different notes could be played at the same time. Inverting the mapping (notes on the horizontal axis and pitch on the vertical one) would have made even more notes available but would not have fixed the limited number of playing notes.

Also, the grid system required accuracy in the movements. This issue is difficult to bypass as the use of a grid automatically involves precision and limits the freedom of movement.

4.4.3 Supervisors' opinions

Mr. Libertiaux and Mr. Bertrand thought the system performed good but was too methodical and systematic. Indeed, the mappings being static, the same wrist position always produced the same note with the same pitch. Using a dynamic mappings could have eliminated the over-predictability of the system and add user engagement but could also have really thrown off some users. Moreover, the mapping on the horizontal axis was based on the stage layout, changing it could have been a bit incoherent. For example, using the right arm to control the voice emitted by the left hand side speakers could be confusing.

They were also worried about the usability and the user experience of the system for the general public: most people make big, unreflected gestures, trying to mimic what they think the gestures of a conductor are. Their first movements would render poorly with such an accurate system, impacting in the wrong way their interaction with the system.

They also did not find the artistic component they were expecting. Even though artistic practise call for accuracy and technique [20, 26], no advanced skills should be required to use this kind of system. According to Messrs Libertiaux and Bertrand, the simple static mappings and the above limitations made it too difficult to give the system a proper artistic feel. In addition to that, the freedom of note choice gave too much freedom to the users to guarantee artistic sound productions.

This discussion, based on their experience with interactive systems, mostly rejected the hypothesis but the system should nonetheless be tested by more users to effectively reject it: while the guessability of the system was judged acceptable, the usability was not, due to the accuracy the system was requiring. This also excluded the use of even simple gesture recognition algorithms. Indeed, an idea was to use the $1 algorithm [71] to recognize simple gestures but this would also have been systematic with one-to-one mappings and using many-to-one mappings would be peculiar for a gesture recognition system, especially since implementing a system in which any voluntary gesture or movement triggers some effect is possible without resorting to gesture recognition algorithms.

From there emerged some new functional requirements for the next prototype. They asked for a movement speed to volume mapping and a system globally closer to the original SMing installation: the system would be using the same MIDI file as SMing in order to limit the note choice and the arms would act like the baton does, triggering a new note as an angular change is detected.

4.5 Second iteration

The previous proof of concept showed that giving the user more freedom of choice over the notes actually decreases the usability of the system. The hypothesis for this iteration was that removing the control over the note and its pitch would add usability and provide a better artistic dimension to the system. The idea was to give the user an artistic creation feeling no matter the movements he was performing. Another hypothesis was that using the speed of the movement to control the playing volume would be more intuitive than the

difference between the X-coordinates of the wrists. One-to-one mappings were kept but were made less systematic.

4.5.1 Implementation

Another C# sample was used as the basis for this prototype. This sample was not selected for the first iteration because it has a full graphical user interface (GUI), which adds a significant load on the computer. It was however needed here as it renders 3D coordinates, which were to be used for angle calculations. Some parts of the code that were of no use were however removed in an attempt to lighten the algorithm. The unused coordinates were discarded, likewise the first proof of concept, but seven additional ones were needed: both shoulders and elbows, the neck joint in the middle of the shoulders and the top of the legs joints, between the legs and the hips.

The GUI highlighted a slight jiggle of the skeleton. Indeed, the coordinates are not perfectly stable and some body poses are not recognized as well as others. It is difficult the identify the origin of this issue. This might be due to the performance of the SDK, to the environment or to the camera itself, as its RGB module and the left infrared lens got lightly damaged. The camera should be tested in real conditions and the SDK and algorithm should be tested with a new RealSense camera to properly understand the cause of the jiggling.

Like for the previous proof of concept, all the data are processed by the algorithm and sent to Max8 over OSC. Here however, Max8 could receive two types of messages:

1. A note message with an integer (1 or 2) indicating to which voice the message is addressed and an integer for stopping (0), launching (1) or maintaining (2) a note. Launching a note in this context was meaning jumping to the next MIDI note and playing it. Maintaining the note allowed to stay on the same MIDI note and only changing its volume.

2. A volume message with an integer (1 or 2) indicating to which voice the message is addressed and with an integer between 0 and 66. The volume was limited to 66 as, above that, the speakers were saturated.

Here again, there were three different cases:

1. Rest pose: instead of using a pre-defined threshold, the top of the leg joints were used as reference points for the "not playing" state of the system. Indeed, when the arms are hanging alongside the body, the wrist joints are usually below the leg joints. To ensure that this relationship was standing no matter the body type, the coordinates of the leg joints have been revised to match with the top of the hips.

2. One active arm: similarly to the first prototype, for each arm, if the wrist was above its corresponding hip point, the arm was considered active and therefore playing. The left arm controlled the two low voices on the left while the right one controlled the two high voices on the right. There was a score for each group of voices, which meant that each arm could progress

through the MIDI file individually and was not impacted if the other arm was playing faster or not playing. Inspired by the features Jáuregui et al. extracted in their experiment [29], the launch of a new note was determined by the angle change of the elbow first and then of the armpit (the four arm angles). As not every arm movement induces a sufficient change of the elbow angle, the armpit angles was also assessed if the first condition failed. If none of these angles were different enough from the same angles in the previous frame, the note was simply maintained. The speed of the movement (i.e. the speed of the X-coordinate) determined the playing volume, a faster movement increasing it and vice versa. If the speed was below a decided threshold and the change in angles small enough, the note was maintained at the last recorded volume.

3. Both arms active: similarly to the one arm situation, this case launched a new note when the elbow angles, failing that, if the armpit angles were sufficiently different from the previous frame. The only differences were that all voices were playing, not just two and that, if the speeds of the two wrists were close enough, both sides were playing at the same volume to provide a sense of unity within the choir.

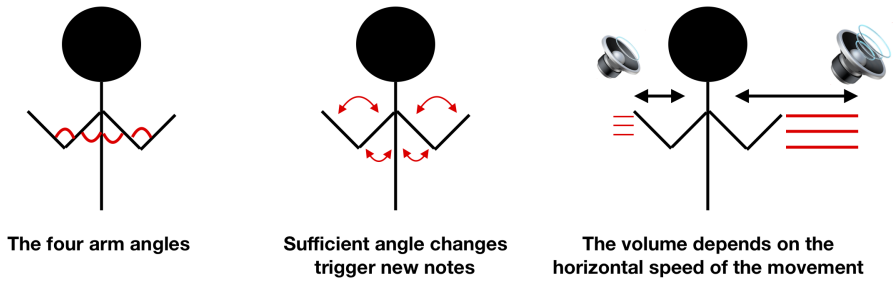


Figure 4.5: Operation of the second proof of concept.

4.5.2 Limitations

The challenge in this algorithm was to find the right tradeoff between reactivity and appropriate note trigger. The system has to launch a new note when it detects voluntary movements (i.e. sufficient change in angles) but has to distinguish those movements from involuntary ones and coordinates jiggle movements. The jiggling makes finding the correct angle and speed windows a complicated task. It is however a paramount assignment to ensure a good user experience. All the combinations have to be implemented then tested to find the best working one and this is very time-consuming. The proposed combination was not optimal, as the notes changed too quickly, especially when the elbows were bent, but it was nonetheless good enough to provide a constructive preview of the prototype. This issue was partly fixed before the next iteration, following Mr. Bertrand's suggestion of using the angles formed between the neck-to-shoulder and the shoulder-to-wrist vectors, as it is more representative of the full movements.

Also, a new note being triggered by an arm angle change, several notes were launched during the execution of the movements, resulting in a series of uncontrolled notes no matter what gesture was performed, from simple line to zigzags.

The system was expected to perform like SMing, where direction changes involving angular velocity changes trigger the new notes. This way, drawing a square with the baton results in four notes and drawing a circle results in a continuous note, as there are no angles in circles. A similar behaviour was awaited for this proof of concept. However, when monitoring the elbow and armpit angles to determine the launch of the new notes, drawing a circle involves angle changes and therefore the launch of several notes. This limit is addressed in the next iteration.

4.5.3 Supervisors' opinions

Messrs Libertiaux and Bertrand did not have the opportunity to experiment with this proof of concept in person. The following comments were based on a video demonstration.

It is a complex task to evaluate the usability of a system without properly testing it but they were overall much more pleased with this prototype than with the previous one. The guessability of the system could however not be duly evaluated during this iteration, as a descriptive email was provided before the demo.

They appreciated the control of the volume using the speed of the movement, finding it intuitive. Although they thought that the movements triggered the new notes too promptly, depriving the user from a control sensation over the sound, they were satisfied with the direction taken. Indeed, it was a major improvement from the former prototype but they still wanted to witness a more accurate note trigger method in the next proof of concept.

From an artistic point of view, this system renders nicer with the MIDI notes than with the freedom of choice for the notes. This system provided a better artistic feeling.

This seemed to confirm the hypotheses of this iteration but the system should in spite be tested in person by several users to obtain a firm confirmation or rejection.

Two functional requirements for the next proof of concept were nevertheless identified. Mr. Libertiaux and Mr. Bertrand requested that the note stops when the active arms' movements pause. They also asked to use the change in direction of the movement to trigger a new note, like SMing. This way, drawing a line or a zigzag would not result in the same sound outputs.

4.6 Third iteration

As the previous proof of concept showed encouraging results, it was taken as a starting point for this iteration. One hypothesis here was that stopping the note as the movement stops would be more intuitive and render better. Another

hypothesis was that using direction changes to launch new notes would perform better than using the angle changes of the arms. An additional objective was to stabilize the note launching method in order to increase the sensation of control over the system.

4.6.1 Implementation

Max8 could receive the same two messages as previously but the note message could include an additional argument, the volume. This was realized in order to send less OSC messages. It was not an issue due to the current reasonable size of the system but it could have become one in the future, as the system could potentially grow into a complete, more complex installation.

Once again, there were three states: rest pose, one active arm, both arms active. The first one remained the same and the two others were improved in four connected ways:

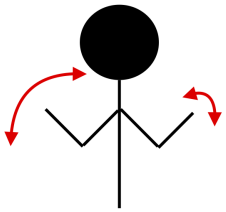
1. If the speed of a joint on each axis was smaller than a determined threshold and the traveled distance in each direction was smaller than a threshold as well, the joint was considered stable. The purpose was to counter the coordinates jiggle and therefore to improve the performance of the system.

2. This lead to the enhancement of the detection of paused movements, which were used to stop the notes.

3. The original idea for this proof of concept was to use the change in direction of the movements to trigger new notes. This was not possible as it would not have allowed the tracing of a circle to maintain the same note, as there are several directional changes in the movement. Instead, the change of note was based on the three-axis speed and travelled distance thresholds defined above. This method allowed to mimic SMing better, changing the note when the speeds and distances were small enough to indicate a change in direction. Indeed, as a movement changes of direction and creates and sharp enough angle, there is a point in time where the speed of the movement is equals to zero. This is fairly similar to calculating the angular velocity to decide if there should be a note change or not. By doing so, the system could produce a single note as a circle was drawn and launch four when it was a square. The second hypothesis of this iteration has hence been updated to fit with this implementation.

4. As the speeds on all axes were computed, each one could be used to determine the playing volume. If the travelled distance on one axis was sufficient, the speed on that axis determined the volume but if the traveled distances were big enough on more than one axis, they were evaluated individually and the first distance to be large enough determined the volume. According to Messrs Libertiaux and Bertrand, the users usually perform more sideways gestures so the distance on the horizontal axis was first assessed, followed by the distance on the vertical axis and then one the depth axis.

An overall improvement was to limit the number of potentially detected skeletons to one. This way, even if the SDK mistakenly detects another skeleton in the background or in the crease of a piece of clothing, the rest of the algorithm was not impacted. In the other prototypes, the case where a second skeleton popped up was not supported and caused the algorithm to fail on the related frame. A failsafe was however present to manage such an issue without impacting the output of the system. This security is still active to ensure a continuous tracking even in case of occasional joint loss.



Instead of angles, the overall movement (displacement and speed) determines the launch of new notes

Figure 4.6: Summary of the third proof of concept.

4.6.2 Limitations

In spite of the programming improvements, some issues remained.

Here again, a tradeoff between reactivity and adequate note launch has to be found. It has to be reached through the proper balance of the speed and travelled distance thresholds. In the current state of the system, changing the direction of the movement very fast results in the maintaining of the same note, as the system does not have the time to capture the series of frames it needs to assess the change of direction. For a similar reason, really slow movements trigger several notes, as the minimum speed and/or travelled distance required to maintain a note are not reached. In order to find this tradeoff, the system has to be tested in real lighting conditions, as slight tracking differences were noticed under different lighting conditions.

4.6.3 Supervisors' opinions

Again, Mr. Bertrand and Mr. Libertiaux were sent a video demonstration of the prototype. It was accompanied by a descriptive note. As the movements and features were essentially the same than during the previous iteration, the guessability could not be evaluated.

They were glad with this implementation. It met their expectations at the time and without testing it, they did not exactly know what improvements could be useful next. This demonstration was particularly frustrating for them: they truly wanted to test the prototype themselves, as it "now becomes interesting for them". Indeed, now that they seem to appreciate the system more and see what can be done with it, this iteration offers a stronger base for them to work with. This could imply that the system provides a good artistic dimension, is attractive and performs well.

The silence when there was no movement was considered more logic and the note launching using the speed and travelled distance windows appeared to provide a better sense of control. This seems to confirm the hypotheses. However, as previously, they cannot be duly confirmed nor rejected as long as the system has not been evaluated.

This iteration concluded the internship at Superbe but the final system has not yet been tested and evaluated by Messrs Libertiaux and Bertrand.

4.7 Evaluation

The goal of an evaluation for this system would be to assess its overall usability, its guessability and the user experience it delivers. Unfortunately, the confinement has made the organization of a study impossible. This section develops the evaluation processes that could have been followed under normal circumstances.

4.7.1 Superbe

Like Superbe do for their other installations, realizing an evaluation with the people working in the building, simply by placing the prototype in a common space would have been a useful step for the last proof of concept. It would have provided an interesting indicator of the usability of the system. Placing the prototype installation at Trakk[11] would also have brought interesting insights. In a normal situation, their feedback would have been used to fix the recurrent issues pointed out and improve the algorithm. Then, the improved system would have briefly been tested in the office again and, if sufficient, could have been tested during an event. Not all events are fit for the testing of experimental installations but, by experience, Superbe know in what types of events this is possible. Monitoring the participants' reactions would have enabled instant fixes and continuous improvement of the system. Superbe do not conduct formal in-lab evaluations, as simply presenting the installation to the general public instantly provides usability and user experience feedback. Modifying the system in-situ allows to test the improvements right away and assess the new usability.

4.7.2 Academic

From an academic perspective however, a formal study would have allowed to assess the differences in user experience, usability and guessability between the three proofs of concept and properly address the hypotheses.

A wide range of participants would have been recruited. As the system is to be placed in public artistic exhibits, all age categories and various professional horizons should be represented in the population sample. Also, the installation being musical and artistic, participants with more or less musical experience should be equally represented. Indeed, as the system makes the user the conductor of a choir, it should not strongly repel participants who know the proper gestures nor discourage individuals with basic to no musical knowledge.

[11] https://www.trakk.be/

The considered age categories for this study would have been determined by the average affinity with information technology. Generation Y (roughly born from early 1980s until mid to end 1990s) and Generation Z (roughly born from mid to end 1990s until early 2010s) could be part of the same category, as both generation have usually a good relationship with technology [23, 62]. Generation X (roughly born from early to mid 1960s until early 1980s) should be considered as a category by itself as there are high variations in technological affinity within its population: while most of them are used to technology [23], some tend to struggle with more recent technology. A last category would the Baby Boomers generation (roughly born from mid 1940s until early to mid 1960s), as they usually strongly dislike technology [23]. The previous generations are far less likely to volunteer in a study but, would it happen, they would be part of the latest category described.

Recruiting a diverse set of participants can be time-consuming. Depending on the time constraints, the study could have been conducted in two different ways:

1. Longer study: the participants would have been split into three equal groups (A, B, C), each one containing an homogeneous blend of the population sample. Each group would have tested one proof of concept (group A testing the first one, group B the second and group C the last). Each participant would have been immersed, individually but accompanied by a moderator, in conditions similar to the real ones and therefore been given audio and visual cues. Indeed, when no user is playing with SMing, a robotic arm uses the conductor's baton to control the system, inviting the user to do the same. As there is no baton anymore, playing a recorded performance of someone playing with the system while no one is standing in front of the camera could have had a similar effect. This statement is considered accepted for the study but this is also an hypothesis and it should be tested in another study, as designing the installation environment is beyond the scope of this thesis. The moderator would have invited the participant to play with the system, like Cafaro et al. (2018) did in their in-lab and in-situ study [8]. The guessability would have been assessed first, as the participant discovered the system. However, if some participants struggled to grasp the system for over five minutes, the moderator would have intervened, asking what is wrong, gathering the information given by the participant and vaguely explaining the operation of the system afterwards. Then, once they would have figured out how to use the system, they would have been allowed to play with the system for five minutes and the overall usability and user experience would have been evaluated using a French version of the User Experience Questionnaire [54] at the end of the interaction. Individual interviews could perhaps have been planned, to gather addition information.

2. Shorter study: the study would have been similar to the other one but only a third of the participants would have been recruited, meaning that the whole group would have tested each prototype. As the systems all have the same goal and therefore are similar, the test order would have been randomized from one participant to another. This would have prevented a familiarization effect to compromise the guessability results.

There would have been different control actions to discover in the different proofs of concept. The guessability of each one would have been measured by the average time it took the participants to figure it out. As this would not be a (framed) guessability study, no user agreement measurement would be needed. The usability and user experience would have been measured with the questionnaires.

The results of each proof of concept would have been compared to the others' results in order to determine which one was the most attractive, fun, understandable and artistic proof of concept.

The next and final chapter presents the conclusion of this project as well as future work possibilities.

Chapter 5

Conclusion and future work

Gestural interaction and gesture recognition are nowadays being integrated in a wide range of systems, with various purposes. Artists are no exception and are interested in the possibilities this technology offers as well. Superbe, a Belgian artistic studio, wanted to give it a try and to use it to evolve one of their existing installation: SMing, an interactive choir. This thesis explored the prospects of integrating computer vision-based gestural interaction in an artistic context and their implications for usability and user experience. There is however much more to do with body movements and music, only a small part of it is considered in this work.

5.1 Conclusion

After selecting the most appropriate RGB-D camera and skeleton tracking SDK, several proofs of concept were presented, each investigating different gesture features to sound features mappings. Unfortunately, two of the proofs of concept could not be properly tested due to the COVID-19 pandemic. This makes it difficult to draw strong conclusions on the usability and user experience of the proposed systems, as no evaluation study could be conducted. The last proof of concept however seems promising in terms of artistic feeling, control sensation and usability. It tracks the arm movements of the user and then allows the control of a melody and of its playing volume. If the arm movements are faster than a speed threshold and the travelled distance in each frame is over a distance threshold, the system produces the first note of a MIDI file. As the movements continue, the same note is either maintained, if there is no directional change, or the next note is triggered, if the system detects a change of direction. The speed of the movements controls the playing volume, faster movements producing louder notes, slower movements calmer notes and no movement resulting in silence. Each arm can control the system individually or they can be used together.

5.2 Future work

Future developments for Superbe will start with testing the last proof of concept in their office. From there, further development might be required before scaling the system up and preparing it for exhibits. If the system is satisfying, they will have to design the environment (or priming situation) in which the system will be presented to the public.

It might also be worth for them to test the new Processing library that supports skeleton tracking[1], as Mr. Bertrand has a preference for this IDE.

If they are not pleased with the system however, implementing a similar system with dynamic mappings or using machine or deep learning techniques might provide them with more development opportunities.

From an academic point of view, future work would be the conduct of a study to evaluate the usability, guessability and user experience offered by the proposed mappings in order to assess the emitted hypotheses.

[1] https://github.com/cansik/deep-vision-processing

Bibliography

[1] Kinect. https://en.wikipedia.org/wiki/Kinect. Date of access: 03.02.2020.

[2] Faraj Alhwarin, Alexander Ferrein, and Ingrid Scholl. Ir stereo kinect: Improving depth images by combining structuredlight with ir stereo. 12 2014.

[3] Lisa Anthony and Jacob O Wobbrock. A lightweight multistroke recognizer for user interface prototypes. In *Proceedings of Graphics Interface 2010*, GI '10, pages 245–252. Canadian Information Processing Society, Toronto, Ont., Canada, 2010.

[4] M. Asadi-Aghbolaghi, A. Clapés, M. Bellantonio, H. J. Escalante, V. Ponce-López, X. Baró, I. Guyon, S. Kasaei, and S. Escalera. A survey on deep learning based approaches for action and gesture recognition in image sequences. In *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, pages 476–483, 2017.

[5] Valentin Bazarevsky and Fan Zhang. On-device, real-time hand tracking with mediapipe. https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html. Publication date: 19.09.2019. Date of access: 24.02.2020.

[6] Amy Bearman and Catherine Dong. Human pose estimation and activity classification using convolutional neural networks. *CS231n Course Project Reports*, 2015.

[7] Claude Cadoz and Marcelo M. Wanderley. Gesture - Music. In Ircam-Centre Pompidou Marcelo Wanderley et Marc Battier, editor, *Trends in Gestural Control of Music*. March 2000. cote interne IRCAM: Cadoz00a.

[8] Francesco Cafaro, Leilah Lyons, and Alissa N. Antle. Framed guessability: Improving the discoverability of gestures and body movements for full-body interaction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, New York, NY, USA, 2018. Association for Computing Machinery.

[9] Francesco Cafaro, Leilah Lyons, Raymond Kang, Josh Radinsky, Jessica Roberts, and Kristen Vogt. Framed guessability: Using embodied allegories to increase user agreement on gesture sets. In *Proceedings of the*

8th International Conference on Tangible, Embedded and Embodied Interaction, TEI '14, page 197–204, New York, NY, USA, 2014. Association for Computing Machinery.

[10] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[11] Baptiste Caramiaux, Frédéric Bevilacqua, and Norbert Schnell. Towards a gesture-sound cross-modal analysis. In Stefan Kopp and Ipke Wachsmuth, editors, *Gesture in Embodied Communication and Human-Computer Interaction*, pages 158–170, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[12] Baptiste Caramiaux, Patrick Susini, Tommaso Bianco, Frédéric Bevilacqua, Olivier Houix, Norbert Schnell, and Nicolas Misdariis. Gestural embodiment of environmental sounds : an experimental study. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 144–148, Oslo, Norway, 2011.

[13] Ming Jin Cheok, Zaid Omar, and Mohamed Hisham Jaward. A review of hand gesture and sign language recognition techniques. *International Journal of Machine Learning and Cybernetics*, 10(1):131–153, 2019.

[14] David Dalmazzo and Rafael Ramirez. Air violin: a machine learning approach to fingering gesture recognition. In *Proceedings of the 1st ACM SIGCHI International Workshop on Multimodal Interaction for Education*, pages 63–66, 2017.

[15] François Delalande. La gestique de gould: éléments pour une sémiologie du geste musical. *Glenn Gould Pluriel*, pages 85–111, 1988.

[16] Collins English Dictionary. Definition of the word interaction. `https://www.collinsdictionary.com/`. Date of access: 24.04.2020.

[17] Irvin Dongo, David Antonio Gómez Jáuregui, and Nadine Couture. A study on the simultaneous consideration of two modalities for the recognition of SoundPainting gestures. In *Actes de la 31e conférence francophone sur l'Interaction Homme-Machine (IHM 2019)*, pages 13:1–12, Grenoble, France, December 2019. ACM.

[18] Bruno Dumas, Denis Lalanne, and Sharon Oviatt. Multimodal interfaces: A survey of principles, models and frameworks. In *Human machine interaction*, pages 3–26. Springer, 2009.

[19] José Miguel Fernandez, Thomas Köppel, Nina Verstraete, Grégoire Lorieux, Alexander Vert, and Philippe Spiesser. Gekipe, a gesture-based interface for audiovisual performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 450–455, Copenhagen, Denmark, 2017. Aalborg University Copenhagen.

[20] Jean Geoffroy. Le geste dans l'oeuvre musicale, la musique et le mouvement. *Le feedback dans la création musicale-Actes des rencontres musicales pluridisciplinaires*, 2006.

[21] Anthony Gritten and Elaine King. *Music and gesture*. Ashgate Publishing, Ltd., 2006.

[22] Anthony Gritten and Elaine King. *New Perspectives on Music and Gesture*. Sempre Studies in the Psychology of Music. Ashgate Publishing, Ltd., 2011.

[23] Dogan Gursoy, Thomas A Maier, and Christina G Chi. Generational differences: An examination of work values and generational gaps in the hospitality workforce. *International Journal of Hospitality Management*, 27(3):448–458, 2008.

[24] Patrice Guyot and Thomas Pellegrini. Vers la transcription automatique de gestes du soundpainting pour l'analyse de performances interactives. In *Journées d'Informatique Musicale (JIM 2016)*, pages pp. 118–123, Albi, France, March 2016.

[25] Barah Héon-Morissette. Transdisciplinarity, an artistic practice: Gesture-sound space and sicmap. 2014.

[26] Barah Héon-Morissette. Transdisciplinary methodology: from theory to the stage, creation for the SICmap. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, volume 16 of *2220-4806*, pages 253–258, Brisbane, Australia, 2016. Queensland Conservatorium Griffith University.

[27] Lode Hoste, Bruno Dumas, and Beat Signer. Mudra: a unified multimodal interaction framework. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 97–104, 2011.

[28] Lode Hoste and Beat Signer. Expressive control of indirect augmented reality during live music performances. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 13–18, Daejeon, Republic of Korea, May 2013. Graduate School of Culture Technology, KAIST.

[29] David Antonio Gómez Jáuregui, Irvin Dongo, and Nadine Couture. Automatic recognition of soundpainting for the generation of electronic music sounds. In Marcelo Queiroz and Anna Xambó Sedó, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 59–64, Porto Alegre, Brazil, June 2019. UFRGS.

[30] Hyun Kang, Chang Woo Lee, and Keechul Jung. Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, 25(15):1701–1714, 2004.

[31] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.

[32] Dharani Mazumdar, Anjan Kumar Talukdar, and Kandarpa Kumar Sarma. Gloved and free hand tracking based hand gesture recognition. In *2013 1st International Conference on Emerging Trends and Applications in Computer Science*, pages 197–202. IEEE, 2013.

[33] Microsoft. Azure kinect dk est désormais en disponibilité générale aux États-unis et en chine. https://azure.microsoft.com/fr-fr/updates/azure-kinect-dk-is-now-generally-available-in-the-united-states-and-china/. Updated: 15.07.2019. Date of access: 20.04.2020.

[34] Microsoft. Mise à disposition d'Azure Kinect DK au Japon, en Allemagne et au Royaume-Uni. https://azure.microsoft.com/fr-fr/updates/azure-kinect-dk-is-now-available-in-japan-germany-uk/. Updated: 02.04.2020. Date of access: 20.04.2020.

[35] L. Miranda, T. Vieira, D. Martinez, T. Lewiner, A. W. Vieira, and M. F. M. Campos. Real-time gesture recognition from depth data through key poses learning and decision forests. In *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, pages 268–275, 2012.

[36] S. Mirri, P. Salomoni, A. Pizzinelli, M. Roccetti, and C. E. Palazzi. "di piazza in piazza": Reimagining cultural specific interactions for people-centered exhibitions. In *2016 International Conference on Computing, Networking and Communications (ICNC)*, pages 1–5, 2016.

[37] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.

[38] Mohamed A Mohandes. Recognition of two-handed arabic signs using the cyberglove. *Arabian Journal for Science and Engineering*, 38(3):669–677, 2013.

[39] Tobias Mulling and Mithileysh Sathiyanarayanan. Characteristics of hand gesture navigation: a case study using a wearable device (myo). In *Proceedings of the 2015 British HCI Conference*, pages 283–284, 2015.

[40] GRS Murthy and RS Jadon. A review of vision based hand gestures recognition. *International Journal of Information Technology and Knowledge Management*, 2(2):405–410, 2009.

[41] R. Nave. Timbre. http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/timbre.html#c2. Date of access: 23.05.2020.

[42] North. How does the myo armband work? https://support.getmyo.com/hc/en-us/articles/202532376-How-does-the-Myo-armband-work. Date of access: 20.04.2020.

[43] Kristian Nymoen, Kyrre Glette, Ståle A. Skogstad, Jim Torresen, and Alexander Refsum Jensenius. Searching for cross-individual relationships between sound and movement features using an SVM classifier. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 259–262, Sydney, Australia, 2010.

[44] The Editors of Encyclopaedia Britannica. Envelope. https://www.britannica.com/science/envelope-sound. Date of access: 23.05.2020.

[45] The Editors of Encyclopaedia Britannica. Timbre. https://www.britannica.com/science/timbre. Date of access: 23.05.2020.

[46] Cemil Oz and Ming C Leu. American sign language word recognition with a sensory glove using artificial neural networks. *Engineering Applications of Artificial Intelligence*, 24(7):1204–1213, 2011.

[47] Thomas Pellegrini, Patrice Guyot, Baptiste Angles, Christophe Mollaret, and Christophe Mangou. Towards soundpainting gesture recognition. In *Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound*, AM '14, New York, NY, USA, 2014. Association for Computing Machinery.

[48] J. K. Perng, B. Fisher, S. Hollar, and K. S. J. Pister. Acceleration sensing glove (asg). In *Digest of Papers. Third International Symposium on Wearable Computers*, pages 178–180, 1999.

[49] S. S. Rautaray and A. Agrawal. Interaction with virtual game through hand gesture recognition. In *2011 International Conference on Multimedia, Signal Processing and Communication Technologies*, pages 244–247, 2011.

[50] Iasonas Kokkinos R{iza Alp Güler, Natalia Neverova. Densepose: Dense human pose estimation in the wild. 2018.

[51] Sriparna Saha, Monalisa Pal, Amit Konar, and Ramadoss Janarthanan. Neural network based gesture recognition for elderly health care using kinect sensor. In Bijaya Ketan Panigrahi, Ponnuthurai Nagaratnam Suganthan, Swagatam Das, and Shubhransu Sekhar Dash, editors, *Swarm, Evolutionary, and Memetic Computing*, pages 376–386, Cham, 2013. Springer International Publishing.

[52] Francisco M Sánchez-Margallo, Juan A Sánchez-Margallo, José B Pagador, José L Moyano, José Moreno, and Jesús Usón. Ergonomic assessment of hand movements in laparoscopic surgery using the cyberglove®. In *Computational biomechanics for medicine*, pages 121–128. Springer, 2010.

[53] Elena Sánchez-Nielsen, Luis Antón-Canalís, and Mario Hernández-Tejera. Hand gesture recognition for human-machine interaction. 2004.

[54] Martin Schrepp, Andreas Hinderks, and Jörg Thomaschewski. Applying the user experience questionnaire (ueq) in different evaluation scenarios. In Aaron Marcus, editor, *Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience*, pages 383–392, Cham, 2014. Springer International Publishing.

[55] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 3633–3642, New York, NY, USA, 2015. Association for Computing Machinery.

[56] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304, 2011.

[57] Francesco Luke Siena, Bill Byrom, Paul Watts, and Philip Breedon. Utilising the intel realsense camera for measuring health outcomes in clinical research. *Journal of medical systems*, 42(3):53, 2018.

[58] Chaklam Silpasuwanchai and Xiangshi Ren. Designing concurrent full-body gestures for intense gameplay. *International Journal of Human-Computer Studies*, 80:1–13, 2015.

[59] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[60] Atau Tanaka, Balandino Di Donato, Michael Zbyszynski, and Geert Roks. Designing gestures for continuous sonic interaction. In Marcelo Queiroz and Anna Xambó Sedó, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 180–185, Porto Alegre, Brazil, June 2019. UFRGS.

[61] Walter Thompson. Soundpainting. `http://www.soundpainting.com/soundpainting-2-fr/`. Update date: 2020. Date of access: 19.10.2019.

[62] Anthony Turner. Generation z: Technology and social interest. *The Journal of Individual Psychology*, 71(2):103–113, 2015.

[63] Rafael Valer, Rodrigo Schramm, Luciana P Nedel, Charles P Martin, and Jim Torresen. Musical brush: A creative ar app that connects music and drawing. 2019.

[64] K. M. Vamsikrishna, D. P. Dogra, and M. S. Desarkar. Computer-vision-assisted palm rehabilitation with supervised learning. *IEEE Transactions on Biomedical Engineering*, 63(5):991–1001, May 2016.

[65] Jean Vanderdonckt, Bruno Dumas, and Mauro Cherubini. Comparing some distances in template-based 2d gesture recognition. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI EA '18, New York, NY, USA, 2018. Association for Computing Machinery.

[66] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. Gestures as point clouds: A $p recognizer for user interface prototypes. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, page 273–280, New York, NY, USA, 2012. Association for Computing Machinery.

[67] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. $q: A super-quick, articulation-invariant stroke-gesture recognizer for low-resource devices. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '18, New York, NY, USA, 2018. Association for Computing Machinery.

[68] Gabriel Vigliensoni and Marcelo M. Wanderley. A quantitative comparison of position trackers for the development of a touch-less musical interface. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Ann Arbor, Michigan, 2012. University of Michigan.

[69] Robert Y Wang and Jovan Popović. Real-time hand-tracking with a color glove. *ACM transactions on graphics (TOG)*, 28(3):1–8, 2009.

[70] Jacob O. Wobbrock, Htet Htet Aung, Brandon Rothrock, and Brad A. Myers. Maximizing the guessability of symbolic input. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, page 1869–1872, New York, NY, USA, 2005. Association for Computing Machinery.

[71] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, page 159–168, New York, NY, USA, 2007. Association for Computing Machinery.

[72] Chenxuan Xi, Jianxin Chen, Chenxue Zhao, Qicheng Pei, and Lizheng Liu. Real-time hand tracking using kinect. In *Proceedings of the 2nd International Conference on Digital Signal Processing*, ICDSP 2018, page 37–42, New York, NY, USA, 2018. Association for Computing Machinery.

[73] Lingzhu Xiang, Florian Echtler, Christian Kerl, Thiemo Wiedemeyer, Lars, hanyazou, Ryan Gordon, Francisco Facioni, laborer2008, Rich Wareham, Matthias Goldhoorn, alberth, gaborpapp, Steffen Fuchs, jmtatsch, Joshua Blake, Federico, Henning Jungkurth, Yuan Mingze, vinouz, Dave Coleman, Brendan Burns, Rahul Rawat, Serguei Mokhov, Paul Reynolds, P.E. Viau, Matthieu Fraissinet-Tachet, Ludique, James Billingham, and Alistair. libfreenect2: Release 0.2, April 2016.

[74] Zahoor Zafrulla, Helene Brashear, Thad Starner, Harley Hamilton, and Peter Presti. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, page 279–286, New York, NY, USA, 2011. Association for Computing Machinery.

[75] Mahmoud M Zaki and Samir I Shaheen. Sign language recognition using a combination of new vision based features. *Pattern Recognition Letters*, 32(4):572–577, 2011.

[76] Jilin Zhou, François Malric, and Shervin Shirmohammadi. A new hand-measurement method to simplify calibration in cyberglove-based virtual rehabilitation. *IEEE Transactions on Instrumentation and Measurement*, 59(10):2496–2504, 2010.

[77] Michael Zollhöfer. *Commodity RGB-D Sensors: Data Acquisition*, pages 3–13. Springer International Publishing, Cham, 2019.