


Democratic and Popular Algerian Republic
Ministry of Higher Education and Scientific Research
Yahia Fares University of Médéa
Faculty of Sciences
Department of Mathematics and Computer Science

Final Master Project

Branch: Computer Science

Speciality: Systems Engineering and Web Technologies

Automated Arabic Algerian Sign Language Translation System Based on 3D Avatar Technology

---

Proposed and directed by:
Dr. TAHA ZERROUKI

Prepared and presented by:
AMINE MAMI
MOHAMED ELFARES SLIMANI

Year: 2023-2024

Acknowledgment

We would like to express our deepest gratitude to everyone who contributed to the successful completion of this project. First and foremost, we thank our advisor, Dr. Taha Zerrouki and the co-promoter Mr. Redha Mazari, for their continuous support, insightful guidance, and invaluable feedback throughout this research. Their expertise and encouragement have been instrumental in shaping this project.

We extend our heartfelt thanks to the laboratory responsible Mrs. Khadidja Louz for providing the resources and support necessary for this project. Your willingness to sacrifice time and effort to ensure we had everything we needed was invaluable. The stimulating environment you created allowed us to thrive and explore our ideas to their fullest.

Our sincere appreciation goes to the 3D modeling professional, Mr. Youcef Benyahia, whose expert advice and contributions significantly enhanced the quality of our 3D avatar and animations, thank you very much. We also acknowledge the support of the Unity and Blender communities on Discord for their valuable resources and tutorials, which helped us overcome numerous technical challenges.

We would like to express our deepest gratitude to Mrs. Dharbou Maroua, a sign language specialist whose expertise and dedication have been invaluable to this project. Her meticulous review of the signs and her tireless efforts to help improve more than 300 Algerian Sign Language (ASL) gestures have significantly enhanced the accuracy and quality of our work. Thank you for your guidance, support, and unwavering commitment to improving communication for the deaf community.

We are profoundly grateful to the teachers of the school for the deaf in Beni Slimane, Medea, for their participation in testing the prototype. Their feedback was crucial in refining the system and ensuring its relevance and effectiveness.

We would also like to thank our fellow students and friends for their constant encouragement and for providing a stimulating environment that motivated us to strive for excellence.

Last but not least, we are grateful to our families for their unwavering support and understanding throughout this project. Their patience and encouragement have been a source of strength for us. Thank you all for your invaluable contributions to this project.

Dedication

First and foremost, we thank God for all His blessings and for giving us the strength and energy to complete this work. His guidance and support have been our anchor throughout this challenging journey. Without Him, we have no power or knowledge to accomplish anything, and we are deeply grateful for the wisdom and perseverance He has bestowed upon us. We recognize that even the most basic blessings come from Him, and we are humbled by His continuous mercy and grace.

This work is dedicated to:

Our families, especially our parents, whose unwavering support, patience, and encouragement have been a constant source of strength throughout this journey. To our brothers and sisters, who have always been there to cheer us on, and to our extended family members for their understanding and belief in us. Your love and support have been the foundation upon which this project was built.

Our university teachers, whose dedication and passion for teaching have profoundly impacted our academic and personal development. Your unwavering support, insightful guidance, and constant encouragement have been a source of inspiration. Thank you for challenging us to push our boundaries and for believing in our potential.

Our friends and close fellow students, as we reach the end of our academic journey, I want to express how much I have enjoyed your company. I am deeply grateful to God for blessing me with such good comrades. I wish you all the best in your careers and hope that our paths cross once again in the future.

And to all the individuals who strive to break down barriers and foster inclusivity for the deaf and hard-of-hearing communities. This project is a small contribution to your tireless efforts.

Our brothers and sisters in Palestine and Gaza, who are facing unimaginable oppression and a relentless war against overwhelming odds. Thousands of lives are lost every day, countless untold stories perish with them, and the people in the north of the strip suffer from hunger while the world turns a blind eye. We are deeply sorry for your suffering and are certain that God will not let your lives, dreams, and hopes be in vain. All we can do is pray for you, and we do so with all our hearts.

Thank you for being our motivation and inspiration.

Abstract

Aiming to reduce communication barriers between deaf individuals using Arabic Algerian Sign Language (AASL) and hearing individuals unfamiliar with this language, an automated translation system has been developed. This system uses a 3D avatar to dynamically and accurately represent signs. The methodology includes the Notation System Method (NSM), which primarily relies on encoding sign language using the Hamburg Notation System (HamNoSys), and the Gesture Animation via Motion Capture (GAMC) method for recognizing and animating gestures using intelligent motion capture models. The study provides a comprehensive analysis of the linguistic complexities of AASL and the challenges in developing technological solutions to address these issues. To enhance translation accuracy, collaboration with experts in AASL translation was conducted at each development stage. The results indicate significant improvements in translation accuracy and user engagement, highlighting the system's potential to improve accessibility for the deaf community in various fields in Algeria, based on the differences revealed through the comparison of the two methods employed.

Keywords

Algerian Sign Language, 3D Avatar, Signwriting, Motion Capture, HamNoSys, Deaf Community, Translation System.

الملخص

يهدف التقليل من حواجز التواصل بين الأفراد الصم الذين يستخدمون لغة الإشارة الجزائرية والأشخاص السمعيين غير الملمين بهذه اللغة، تم تطوير نظام ترجمة آلي للغة الإشارة الجزائرية العربية باستخدام أفاتار ثلاثي الأبعاد لتمثيل الإشارات بشكل ديناميكي ودقيق. ولتحقيق ذلك تم اعتماد منهجية تشمل طريقة نظام التدوين (NSM) والتي تعتمد بشكل أساسي على ترميز لغة الإشارة بنظام هامبورغ للترميز (HamNoSys) وطريقة تحريك الإيماءات عبر التقاط الحركة (GAMC) للتعرف على الإيماءات وتحريكها بالاعتماد على نماذج ذكية في التقاط الحركة. تقدم الدراسة تحليلاً شاملاً للتعقيدات اللغوية للغة الإشارة الجزائرية والتحديات القائمة في تطوير حلول تقنية تعالج مشاكل هذه اللغة الإشارية. ولتحسين دقة الترجمة تم التعاون مع خبراء في ترجمة اللغة الإشارية الجزائرية في كل مرحلة من مراحل التطوير. تشير النتائج إلى تحسينات كبيرة في دقة الترجمة وتفاعل المستخدمين، واستنادا إلى الفروقات الناتجة عن المقارنة بين الطريقتين التي ينتهجهما النظام تبرز إمكانيات هذا النظام في تحسين الوصول لمجتمع الصم على مختلف الميادين في الجزائر.

الكلمات المفتاحية

لغة الإشارة الجزائرية، أفاتار ثلاثي الأبعاد، الكتابة الإشارية، التقاط الحركة، نظام هامبورغ للترميز، مجتمع الصم، نظام الترجمة.

Résumé

Dans le but de réduire les barrières de communication entre les personnes sourdes utilisant la langue des signes algérienne arabe (AASL) et les personnes entendantes qui ne connaissent pas cette langue, un système de traduction automatique a été développé. Ce système utilise un avatar 3D pour représenter les signes de manière dynamique et précise. La méthodologie comprend la méthode du système de notation (NSM), qui repose principalement sur l'encodage de la langue des signes à l'aide du système de notation de Hambourg (HamNoSys), et la méthode d'animation des gestes par capture de mouvement (GAMC) pour la reconnaissance et l'animation des gestes à l'aide de modèles intelligents de capture de mouvement. L'étude fournit une analyse approfondie des complexités linguistiques de l'AASL et des défis dans le développement de solutions technologiques pour traiter ces problèmes. Pour améliorer la précision de la traduction, une collaboration avec des experts en traduction de l'AASL a été menée à chaque étape du développement. Les résultats indiquent des améliorations significatives de la précision de la traduction et de l'engagement des utilisateurs, mettant en évidence le potentiel du système à améliorer l'accessibilité pour la communauté sourde dans divers domaines en Algérie, en se basant sur les différences révélées par la comparaison des deux méthodes employées.

Mots-clés

Langue des signes algérienne, Avatar 3D, Écriture des signes, capture de mouvement, HamNoSys, Communauté sourde, Système de traduction.

Contents

List of Figures

List of Tables

List of Abbreviations

AASL - Arabic Algerian Sign Language.
ALSL - Algerian Sign Language.
ArSL - Arabic Sign Language.
ASL - American Sign Language.
BSL - British Sign Language.
BVH - BioVision Hierarchy.
CGSLR - Continuous Gesture-based Sign Language Recognition.
CMSLR - Continuous Multimodal Sign Language Recognition.
COLLADA - Collaborative Design Activity.
CSLR - Continuous Sign Language Recognition.
CWASA - CWA Signing Avatars.
DC - Dependency Chain.
DGS - German Sign Language.
DSC - Dynamic Sentence Creation.
EBMT - Example-Based Machine Translation.
eSIGN - eSign Editor.
FBX - Filmbox.
GAMC - Gesture Animation via Motion Capture.
GIZA++ - A software tool for word alignment.
GSL - Greek Sign Language.
IDE - Integrated Development Environment.
ISL - Indian Sign Language.
ISLR - Isolated Sign Language Recognition.
JSON - JavaScript Object Notation.
JSL - Japanese Sign Language.
LSF - French Sign Language.
MB - Memory Bound.
MDD - Motion Designer's Data.
MT - Machine Translation.
NSLT - Neural Sign Language Translation.
NSM - Notation System Method.
RBMT - Rule-Based Machine Translation.

SiGML - Signing Gesture Markup Language.

SLR - Sign Language Recognition.

SLT - Sign Language Translation.

SMT - Statistical Machine Translation.

XML - Extensible Markup Language.

General Introduction

The advent of digital technology has ushered in an era where communication barriers can be significantly reduced, fostering inclusivity and enhancing accessibility. One critical area where technology can make a profound impact is in the communication between the deaf and hearing communities. Sign languages, which are visual languages using hand gestures, body movements, and facial expressions, play a pivotal role in the daily lives of millions of deaf individuals worldwide. However, the lack of universal understanding and the distinct variations of sign languages pose challenges for effective communication.

This thesis explores the development of an automated translation system for Arabic Algerian Sign Language (AASL) using 3D avatar technology. By leveraging advanced systems engineering and web technologies, this project aims to bridge the communication gap between deaf individuals who use AASL and those who do not understand it. The proposed system uses a 3D avatar to visually represent the signs, providing a dynamic and interactive translation platform that enhances the user experience and ensures accurate representation of the language.

The thesis is structured into several key chapters, each addressing critical aspects of the research and development process:

1. **Sign Language:** This chapter provides an overview of sign languages, including their history, structure, and the specific characteristics of Algerian Sign Language. It delves into the composition of signs and the development of Arab sign languages, setting the context for the necessity of this project.

2. **Literature Review:** A comprehensive review of existing research in sign language detection, recognition, translation, and the use of 3D avatar technology. This chapter highlights the technological advancements and methodologies that have been instrumental in enhancing sign language translation systems.

3. **Conception:** This chapter outlines the design and architecture of the proposed translation system. It covers the significance and impact of the system, detailing the system design,

functional and non-functional requirements, interaction design, and the technological choices made during development.

4. **Implementation:** The practical aspects of developing the system are detailed in this chapter. It discusses the development environment, hardware and software specifications, and the implementation details and results of the notation system method and gesture animation via motion capture.

Each chapter builds upon the previous one, culminating in a robust and comprehensive system designed to enhance communication for the deaf community in Algeria.

Chapter 1

Sign Language

1.1 Introduction

Sign languages represent a visual and non-verbal mode of communication primarily utilized by individuals who are deaf, mute, or hard of hearing. A prevalent misconception suggests that sign languages possess universal comprehension, yet this notion is inaccurate. Similar to spoken languages, sign languages are diverse and distinctly tailored to specific regions. They exhibit substantial differences in lexicons and linguistic grammars, posing challenges for hearing individuals attempting to communicate without proper training. This discrepancy contributes to a notable communication gap between the deaf and hearing communities.

In contrast to spoken languages, sign languages are innate forms of communication relying on visual and non-verbal elements, such as hand gestures and body language. Despite their distinctions, both sign and spoken languages share a fundamental purpose: to facilitate communication.

Within this chapter, we present an introduction to the realm of sign languages, providing an overview of various universal sign languages and their distinct characteristics. Additionally, we delve into the specifics of Algerian Sign Language, exploring its origin and structural elements. Furthermore, we investigate the concept of automatic sign recognition and its associated advantages in enhancing communication within the deaf and hard of hearing communities.

1.2 Sign Language Overview

1.2.1 Background

Sign languages were erroneously perceived as universally comprehensible, yet contemporary understanding acknowledges their linguistic diversity. Contrary to the previous misconception of their universality, sign languages exhibit variation and cultural specificity [3]. Globally, deaf communities employ distinct sign languages, underscoring the diverse linguistic landscape within this community. Not all deaf individuals affiliate with a Deaf community; some develop individualized gesture systems, termed ”home sign” to communicate with hearing individuals [3]. This multiplicity challenges the concept of a universally understood sign language. Furthermore, hearing individuals interact with intricate sign systems distinct from those of deaf sign languages, known as ”secondary” sign languages [3].

Plains Indian Sign Language in North America serves as an illustrative example, acting as a lingua franca among tribes with varied spoken languages. It possessed distinct syntactic rules, differing lexically and grammatically from American Sign Language (ASL) used by deaf individuals. Some hearing American Indians, such as Mr. J. L. Clark, a deaf Blackfoot Indian, employed it [3].

Various communities, including religious orders and monasteries, adopt sign languages for social or speech-restricted situations. In isolated communities like Martha's Vineyard in the seventeenth century, high incidences of congenital deafness led to the development of sign languages used by both deaf and hearing individuals. Similar situations exist in Grand Cayman, Yucatan, Bali, and other regions worldwide [4].

While rare, communities with both hearing and deaf members utilizing sign languages exist. The next focal points are the number and distribution of these sign languages among global Deaf communities, addressing fundamental questions: "How many languages are there?" and "What is the size of the deaf community?" [4].

Sign languages, as a visually-based mode of communication, enjoy widespread popularity within the deaf community, comprising over 70 million individuals globally. This preference stems from the perception among many deaf individuals that their deafness constitutes a distinctive aspect of their identity, leading them to take pride in expressing themselves through sign languages [3]. Moreover, sign languages transcend the boundaries of the deaf community, finding utility among hearing individuals who encounter challenges in verbal communication or have deaf relatives. This inclusive communication modality facilitates connections and interactions among individuals, irrespective of potential hearing impairments [3].

Sign languages are often misunderstood and subject to biases, despite their extensive use. One common misconception is that sign languages are mere visual translations of spoken languages. However, this belief is unfounded, as sign languages differ significantly from spoken languages, particularly in terms of grammar and vocabulary. For instance, the English word "right" has multiple meanings, including direction and correctness, whereas no single sign in sign languages such as BSL or ASL can convey all of these meanings simultaneously [5]. This highlights that each sign in sign languages has a distinct meaning, unlike words in spoken languages.

The history of sign language is marked by pivotal events, including the establishment of systematic sign language in the 18th century by Abbé Charles Michel de l'Épée. In 1760, Abbé de l'Épée established the National Institute for the Deaf and developed a sign alphabet to facilitate communication among deaf individuals, with the language evolving through the active involvement of the deaf community [6]. In 1880, the 2nd International Congress on Education of the Deaf in Milan marked a crucial turning point when resolutions were passed to prohibit the use of sign language in educational programs. This decision marginalized sign language in deaf education, leading to lasting negative consequences for millions of deaf people globally [7].

The 1880 Milan Conference had a long-lasting impact, negatively affecting the education and lives of deaf individuals. In 2008, a campaign started to reject the 1880 resolutions and seek a

formal apology. This led to the ICED conference in Vancouver in 2010, where a declaration was made acknowledging the mistakes of the 1880 Congress and highlighting the harmful effects of its resolutions. The declaration emphasized linguistic human rights and urged nations to follow principles outlined in international conventions, including the Convention on the Rights of Persons with Disabilities[8]. The Deaf community's acknowledgment of historical mistakes and demand for a formal apology is a significant step towards addressing the impact of the 1880 resolutions. The World Federation of the Deaf (WFD) supported and endorsed the declaration, showing commitment to partnering with the Deaf community to secure global educational rights[8]. This historical moment reflects a shift towards recognizing and embracing sign language as an integral part of deaf culture and identity.

1.2.2 Composition of a Sign

The signs constituting sign languages have long been regarded as simple and unanalyzable gestures, lacking any internal organization. "Stokoe will be the first linguist in 1960 to demonstrate that, just like words in spoken languages, signs have an internal structure. In contrast to spoken languages, gestural languages have a structure that reflects the simultaneity of parameters composing their vocabulary" [9]. Stokoe (1965) identified three parameters for describing signs: configuration, movement, and location, to which orientation was later added. What does this entail? [10]

- The Configuration: corresponds to the shape of the hand or hands forming the sign. These configurations are noted using "brief evocations" (index, duck bill...), letters evoked by the shapes of the hands in fingerspelling (see further below), or numbers.

- The Movement: is a complex parameter of the sign. It can have a strong phonemic character, as a change in movement can change the meaning of a word. "An example of this can be cited in LSF with AIMER (to like) and NE PAS AIMER (to dislike): for these two terms, the location (the chest) and the configuration (flat hand) are the same. Only the movement changes, in the first case the movement is upwards and in the second, downwards" [9]. These signs AIMER and NE PAS AIMER are also found in ALSL.

- The Location: corresponds to the place where the signing speaker performs the sign. The signing space is limited according to the constraints imposed by the visuo-motor channel to ensure maximum management of perceived information.

- The Orientation: is determined in relation to the palm of the hand. It is a parameter that remains vague and debatable insofar as it is difficult to describe precisely the orientation of

the palm of the hand or hands forming the sign; especially when the hand is at an oblique angle, thus between two defined directions.

- Another parameter allowing the analysis or description of a sign is facial expression. This makes it possible to distinguish between two identical signs in realization. For instance, one can cite the example of ”AIMER” (to like) and ”NE PAS AIMER” (to dislike) in LSF (French Sign Language): for these two terms, the location (the chest) and the hand configuration (flat hand) are the same. Only the movement changes [11], in the first case the movement is upwards and in the second, downwards [9]. These signs, ”AIMER” and ”NE PAS AIMER,” are also found in ALSL (American Sign Language).

1.2.3 Categories of Gestures

In exploring the application of sign language, it becomes essential to comprehend the concept of gestures. Gestures, constituting a diverse array of movements, may include temporal components [12]. These movements are commonly sorted into categories based on the involvement of specific body parts. Typically, three fundamental types of gestures are recognized [10]: In the domain of body language research, the investigation into gestures encompassing the entire body holds paramount importance, particularly in endeavors such as enhancing athletic performance through meticulous gesture analysis. These gestures can be categorized broadly into two distinct types: dynamic and static gestures [10]. A static gesture, often termed as a posture, elucidates the arrangement of the body or its constituent parts at a precise moment in time. In contrast, dynamic gestures entail a continuous sequence of postures. This differentiation, rooted in temporal and dynamic attributes, substantially enriches the understanding of body language’s communicative efficacy [10].

Although head and facial gestures may lack explicit semantic definitions, they wield substantial influence in delineating the field of vision [10].

Hand and arm gestures emerge as a pivotal category of interactive gestures, particularly pivotal in the realm of sign language communication due to its reliance on non-vocal modalities [13]. In the context of sign language, the transmission of information hinges exclusively on the intricate movements of the hands and arms, underscoring the imperative nature of precise recognition and interpretation [13]. Moreover, nuanced variations within these gestures can exert profound influences on the conveyed message, accentuating the necessity for meticulous technique and extensive rehearsal. [13]

1.3 Development of Arab Sign Languages

Sign language in the Arab World has gained recognition and documentation in recent times. Efforts have been made to standardize and disseminate sign language in countries such as Jordan, Egypt, Libya, and the Gulf States, aimed at the Deaf community and stakeholders [14] . These efforts have resulted in the emergence of multiple sign languages, nearly matching the number of Arabic-speaking countries, albeit with similar sign alphabets [14].

ARSLs have primarily developed independently, although some have drawn from the experiences of others [15]. The origins of ARSLs can be traced to various sources, including borrowings from European and American sign languages, the creation of conceptual signs, mimicking actions and natural elements, and linguistic expansions such as compounding and blending [15]. Additionally, regional signs, inherited and used by local communities, have contributed to the lexicon of ARSLs [15].

the figure 1.1 illustrates the sign for ”dead,” incorporating nonmanual features. The sign is characterized by specific facial expressions: lips pressed together and eyes slightly closed, combined with a corresponding hand movement.

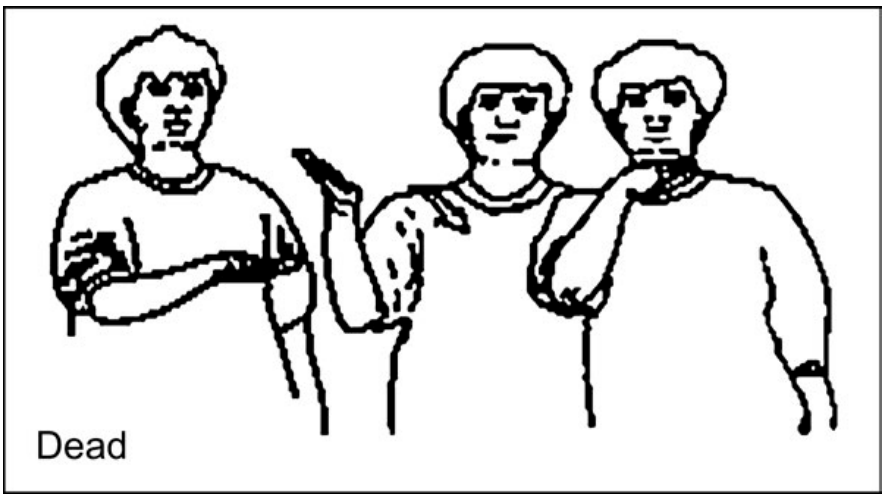

Figure 1.1: *Using non-manual features.[14]*

1.4 Algerian Sign Language

1.4.1 Definition

sign language has attained official recognition as the primary language for the deaf in Algeria, a status granted by the law enacted on May 8, 2002. This groundbreaking development positions Algeria as the pioneering and exclusive nation in the Arab world and Africa to extend such official acknowledgment, presenting a distinctive and inclusive approach tailored to the needs of the local deaf population[16].

The statement underscores the value of sign language while emphasizing the ongoing need for initiatives supporting the deaf community in Algeria. It stresses the importance of enhancing accessibility, fostering understanding, and promoting societal integration for individuals within this community [16].

The Algerian Sign Language (ALSL) is a vibrant communication method using a range of signs made through different body movements, including hands, face, shoulders, and body [12]. It shares characteristics with spoken languages, having organized syntax and unique vocabulary [12]. Understanding ALSL involves recognizing its alphabet formed by hand and body postures, with each sign conveying a specific meaning, highlighting the complexity and detail of ALSL's gestural system [12].

1.4.2 Background

The origins of the Algerian Sign Language (ALSL) can be traced back to the French Sign Language (Langue des Signes Française, LSF). It is notable that ALSL is not a uniform language and has various regional variations, which is an interesting aspect of its history and evolution. This non-uniformity is something that sets it apart from other languages and is worth further examination to understand the regional differences and how they contribute to the overall language.

Understanding these variations is essential to the study of the language and its evolution, as it provides a unique insight into how the language has changed and adapted over time to cater to the needs of its diverse users. [17] These variations are not exhaustive and encompass, among other possibilities:

• Algerian Jewish Sign Language (AJSL), also known as Ghardaia Sign Language, originated and was primarily utilized in the village of Ghardaia by the Algerian Jewish community during that period [18] [19] [20] [21].

• Algerian Sign Language of Laghouat serves as a communication mode for numerous Deaf

individuals residing in Laghouat province and surrounding areas, including cities and villages [11].

• Algerian Sign Language of Oran is employed by the Deaf community in the northern regions of Algeria, particularly prevalent in the city of Oran [22].

• Algerian Sign Language of Adrar is utilized by the Deaf community in Adrar, situated in the southern part of Algeria [13].

The unique nature of the Algerian Sign Language (ALSL) is such that its variations can be region-specific, with distinct dialects developed by deaf communities across the various provinces and villages of Algeria. This phenomenon is likely due to the multiple deaf populations residing in urban and rural areas, which has resulted in similarities and differences among the various forms of ALSL. These variations are primarily used by the deaf community but are also understood by their acquaintances, relatives, and friends, as well as those who interact with them in a variety of ways. The existence of these unique dialects is a fascinating aspect of the language's evolution and diversity.

The study of these regional variations provides an intriguing insight into how this language has naturally adapted and evolved to cater to the needs of its diverse users across Algeria. It is essential to recognize the depth and complexity of these variations to appreciate the rich tapestry of the language and its users.

1.4.3 Algerian Sign Language structure

The movements of a hand can be analyzed based on five separate parameters, which are independent and can either be dynamic or consistent throughout the signing process [23]. These parameters have been defined as follows:

Handshape

The structuring of sign language configuration hinges upon the precise alignment of fingers and palm at a particular temporal juncture. Nonetheless, the morphology of hand shapes exhibits inherent interindividual variation attributable to anatomical distinctions, thereby engendering an element of irregularity. Consequently, the identical lexical units may present nuanced disparities in their gestural manifestations when articulated by diverse signers.

"Figure 1.2" illustrates an example of a configuration in Algerian Sign Language, depicting as a hand with three fingers spread out and the other two fingers, the thumb and index finger touching.

Figure 1.2: *Sign of the word « وجد »* [13]

Movement

Sign language involves hand movements, which can be in the form of lines, arcs, and circles, as well as head movements. There are three parameters for movement in sign language: trajectory, direction, and speed. These movements can be repetitive or periodic in some cases. As an example, in Figure 1.3, the sign " موعد " in Algerian Sign Language can be described by two parameters: configuration and movement. The movement parameter is represented by a hand moving in an arc trajectory.

Figure 1.3: *Sign of the word « موعد »* [13]

Location

Location refers to the position of the hand in space relative to the signer's body or a specific object, which can completely change the meaning of a gesture. For example, the hand can be placed in front of the signer, near the signer's head, on the mouth, eyes, arm, palm, etc. , the Algerian word " دواء " is represented in Algerian Sign Language by the sign shown in Figure 1.4, where the hand is placed in front of the face, near the mouth.

Figure 1.4: *Sign of the word « دواء »* [13]

Orientation

The different ways in which the hands and fingers are positioned can be categorized into specific orientations.[24], as illustrated in Figure 1.5. The orientation parameter describes the orientation of the hand with respect to the signer. Consequently, the direction of the hand is determined by the position of the palm relative to the signer's body [12].

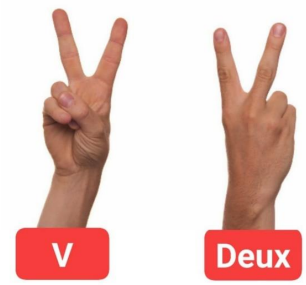

Figure 1.5: *Sign of the letter «V» and the word «two»* [13]

Facial Expressions

In reality, certain sign language expressions comprise hand gestures in conjunction with facial expressions, or facial expression mimics. The latter is crucial in ascribing meaning to a single sign and is fundamental to sentence construction. For instance, in Algerian Sign Language, a facial expression mimic is employed in the term " سعيد " as illustrated by Figure 1.6.

Figure 1.6: *Sign of the word « سعيد »* [13]

1.4.4 Algerian Sign Language Alphabet

Algerian Sign Language comprises a total of 42 alphabet signs, consisting of 35 static signs and 6 dynamic signs. Notably, these signs are executed with a single-handed gesture (see Figure 1.7). Additionally, each static sign is characterized by two unique attributes: configuration and orientation.

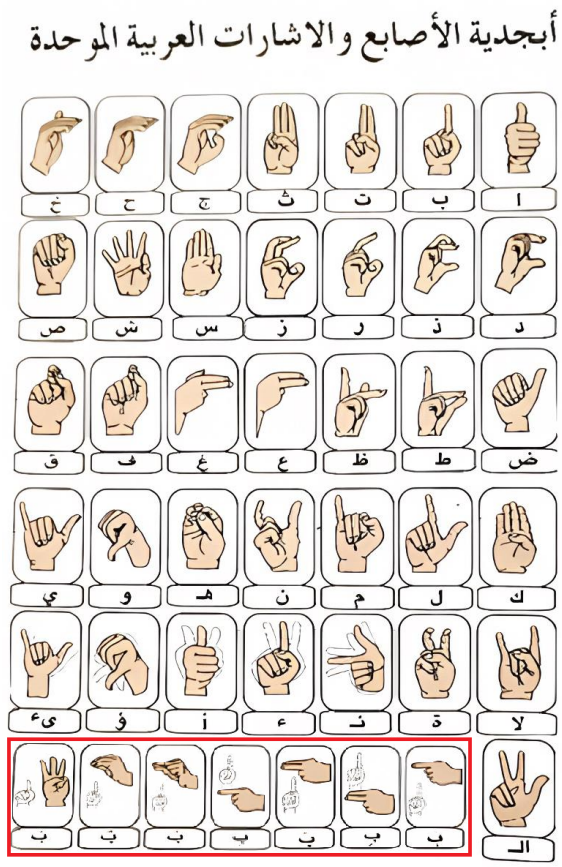

Figure 1.7: *Algerian Sign Language Alphabet* [12].

As an example, the sign for the letter " ب " in Algerian Sign Language is defined by two parameters [12]:

- Configuration: making a fist with the index finger extended.
- Orientation: the palm is facing upwards (or with the wrist in a downwards position) [13].

1.4.5 The Algerian Sign Language and Linguistic Standards

The Algerian Sign Language does not conform to the standards expected of a fully developed language with distinct linguistic features and grammar. Currently, our government lacks well-trained

educators proficient in sign language to effectively teach it. Although in 2017 , the first Algerian Sign Language Dictionary was finally published [25], in contrast to Europe and certain regions of the Arab world (such as Qatar, Egypt, and Saudi Arabia) where sign language is taught as a formal subject, complete with its own grammar and features comparable to any spoken language[25].

the Algerian Signs Dictionary, published in 2017 by the National Foundation for Media Contact of Algeria [25] dictionary utilizes two alphabets, Arabic and French. Its content primarily consists of images depicting sign language vocabulary, accompanied by captions in both Arabic and French. However, it lacks grammar rules and structured sentences, relying instead on Arabic language grammar and structure to convey meaning in sign language. The figure 1.8 makes clear of how Arabic and sign language are combined:

Figure 1.8: *Representations of how grammar get taught* [25]

1.5 Conclusion

This chapter offers a thorough examination of the landscape of sign languages, highlighting its various scholarly inquiries and available resources. With a focus on Algerian Sign Language, which forms the basis of our project, we explored its key attributes and unique characteristics. Our next step involves a deeper exploration of related projects in the following chapters.

Chapter 2

Literature Review

2.1 Introduction

In recent years, the field of sign language (SL) has seen significant advancements, particularly in the areas of detection, recognition, translation, and production. These advancements are crucial for improving communication between the Deaf and Hard of Hearing (DHH) community and hearing individuals. Chapter 3 aims to provide a comprehensive literature review on the various tasks associated with sign language, highlighting the technological developments and methodologies that have been instrumental in enhancing SL translation systems. By examining the historical context and the evolution of SL technologies, this chapter sets the stage for understanding the current state-of-the-art in SL recognition and translation, as well as the role of 3D avatar technology in these processes.

We begin by outlining the primary tasks involved in SL, including detection, identification, segmentation, recognition, translation, and production. Each task is explored in detail to understand its significance and the challenges it presents. The discussion then shifts to the technological advancements that have enabled these tasks, focusing on machine learning, deep learning, and 3d Avatar technologies. Additionally, the chapter delves into the historical development of SL translation systems, tracing their evolution from manual recognition techniques to the sophisticated deep learning models of today.

2.2 Sign Language various Tasks Overview

The realm of sign language (SL) encompasses various tasks, including detection, identification, segmentation, recognition, translation, and production [26]. Detection involves determining whether SL is being utilized, while identification entails discerning the specific SL being used, such as American Sign Language (ASL) or British Sign Language (BSL). Segmentation involves delineating temporal boundaries to isolate phrases or individual signs. Recognition, translation, and production are considered particularly challenging yet essential tasks for the Deaf and Hard of Hearing (DHH) community. These tasks are interlinked, with translation reliant on recognition. SL recognition (SLR) involves comprehending the meaning of signs by assigning labels to each sign, typically using glosses, though this method may not capture all nuances. Two variations of SLR are Continuous SLR (CSLR) and Isolated SLR (ISLR), the former dealing with streams of signs and the latter focusing on classifying cropped signs individually. SL translation aims for natural output, deviating from glosses to provide coherent interpretations of signs. Additionally, SL production involves generating signs or an SL representation from spoken language text. Tokenization of SLs,

akin to segmentation, is crucial for Neural Sign Language Translation (NSLT) systems, offering multiple approaches such as using glosses as tokens, extracting glosses from videos, or frame-level tokenization for embedding frames without discrete representation [26].

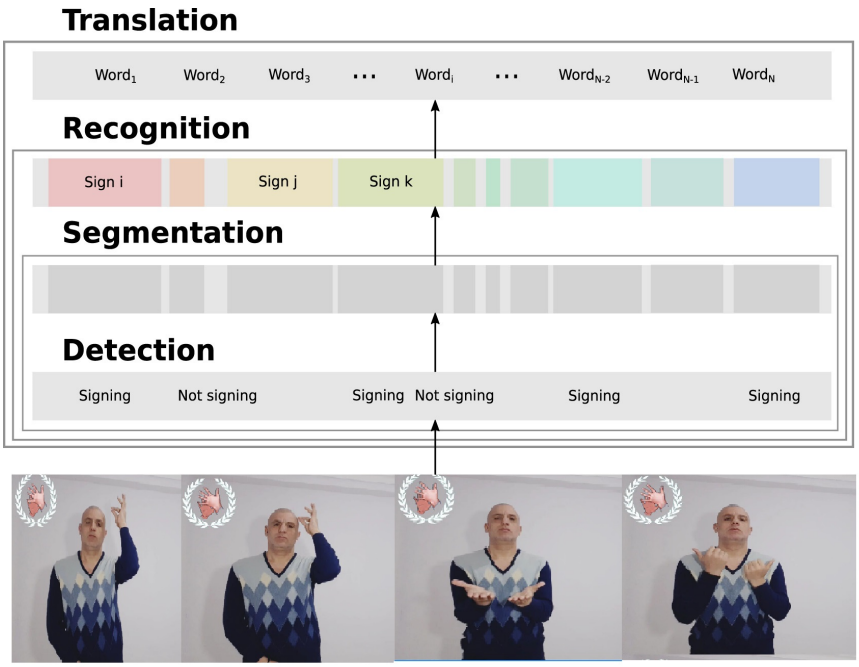

Figure 2.1: Sign Language various Tasks representation

2.2.1 Sign Language Detection

According to (Ilanchezhian, Singh, Balaji, Kumar, and Yaseen, 2023) [27] , Sign language detection employs machine learning techniques to interpret hand gestures used in sign language, thereby facilitating communication with individuals who are deaf or mute. This technology translates sign language by analyzing various hand parameters, including shapes, finger configurations, orientations, and relative positions. The process generally entails capturing images of hand signs or gestures via a webcam, annotating these images, training a model utilizing the TensorFlow object detection API, and subsequently detecting and displaying the meanings of the sign language in real-time through OpenCV-python.

Detecting and translating sign language poses a number of significant challenges, One of the significant issues is the limited availability of large, diverse datasets containing continuous sign language videos paired with spoken language translations. Existing datasets typically feature individual signs or short sequences, which are insufficient for robust model training [28][29]. Additionally, sign language recognition is inherently complex due to the intricate hand shapes, finger configurations, hand orientations, and relative positions involved [27]. Furthermore, sign language incorporates facial expressions and body postures, adding another layer of complexity [28].

Recognizing continuous sequences of signs presents a greater challenge than identifying individual signs, as it requires segmenting the video into distinct signs and classifying each accurately [28][29]. Translating these signs into spoken language demands a deep understanding of the grammatical and linguistic structures of both sign and spoken languages, moving beyond mere sign recognition to generating coherent spoken language output.

For practical applications, achieving real-time performance is crucial. Models must recognize signs and translate them to speech with minimal latency [28][27]. Additionally, anonymizing sign language data poses difficulties since facial features and other physical attributes are vital for accurate sign interpretation [28].

Lastly, the lack of standardization across sign languages—varying by country and region—means models must be tailored to each specific language [29]. Overcoming these multifaceted challenges will require significant progress in creating comprehensive datasets, developing sophisticated model architectures, and refining translation techniques[29].

2.2.2 Sign Language Segmentation

Sign language segmentation stands as a foundational step within sign language recognition systems, pivotal for automatically discerning temporal boundaries between signs in continuous sign

language videos. The segmentation process typically unfolds across three primary stages [30][31] cite87: segmentation, modeling, and classification, with segmentation initiating the identification of sign start and end points [30][31]. Various image segmentation techniques, such as thresholding, have been employed in existing works to pinpoint hands and faces within sign language videos, with segmentation success contingent upon factors like hand size, contrast against the background, and noise levels[32]. Notably, Temporal Convolutional Networks (TCNs) have emerged as a promising avenue for sign language segmentation, leveraging 3D convolutional neural networks to capture both spatial and temporal cues indicative of sign boundaries[33][34]. Moreover, iterative refinement of temporal segments and pseudo-labeling techniques have been demonstrated to enhance performance by resolving boundary ambiguities and harnessing unlabeled data[35]. Evaluation on diverse datasets such as BSLCORPUS, PHOENIX14, and BSL-1K underscores the robust generalization of sign language segmentation models to new signers, languages, and domains [35][36]. The dissemination of open-source resources, including code and pretrained models for TCN-based sign language segmentation, further catalyzes ongoing advancements in this domain, culminating in heightened accuracy and generalization [36].

2.2.3 Sign Language Recognition

Sign language recognition (SLR) represents an evolving field of study aimed at deciphering the conveyed meaning of sign languages through the movements and gestures of signers. This burgeoning area of research entails the interpretation of sign languages performed by individuals utilizing data acquired from diverse sensors such as cameras and smartwatches. Depending on the objectives of recognition tasks, SLR can be broadly classified into two primary branches: isolated sign language recognition (ISLR) and continuous sign language recognition (CSLR). ISLR focuses on recognizing each individual sign language gloss separately, while CSLR involves interpreting entire sequences of glosses as cohesive units. In sign language, a gloss denotes a specific sign that conveys a meaning akin to a word or phrase in spoken languages, and it may encompass one or multiple subtle hand or body movements[13]. In the following subsections, we will conduct a comprehensive examination of both ISLR and CSLR.

Isolated Sign Language Recognition

Isolated Sign Language Recognition (ISLR) seeks to identify individual signs within sign languages and translate them into equivalent spoken language words or phrases. Nevertheless, most glosses in sign languages are intricate and necessitate video demonstrations rather than static images. ISLR research began with the recognition of simple finger-spellings in the 18th century [37], and since

then, numerous studies have explored the performance of various machine learning [38] and deep learning methods [39]. While many studies have demonstrated satisfactory results on certain ISLR (Isolated Sign Language Recognition) datasets, they often neglect a crucial and challenging preprocessing step: frame selection. Frame selection is essential in ISLR as it reduces data size, eliminates noisy data, and enhances both recognition speed and accuracy. However, most existing works utilize a sliding window with a fixed stride to select data frames from the raw data, without considering the nuances of frame selection. This approach is inadequate for ISLR for three main reasons [40]:

1. Certain data frames in ISLR are irrelevant as they do not correspond to any gloss (i.e., the gloss has either not yet begun or has already ended). To avoid selecting these frames, it is crucial to determine the start and end points of each gloss in every data sample. A sliding window with a fixed stride cannot fulfill this requirement effectively.

2. Sign language users perform gestures at varying speeds based on their personal preferences, leading to significant fluctuations in the time taken to complete the same gesture. It is essential to adjust the selection process according to the signing speed in each data sample to avoid selecting redundant or inadequate frames. The sliding window approach is insufficient for this purpose, as it fails to account for variations in signing speed.

3. Rapid movements by signers can cause motion blur and data loss, which significantly impacts the accuracy of subsequent recognition processes. Therefore, it is crucial to select frames with minimal motion blur and data loss during the frame selection process. A sliding window with a fixed stride cannot effectively address this issue, as it does not consider the quality of the data frames.

Continuous Sign Language Recognition

Continuous sign language recognition (CSLR) aims to interpret a continuous stream of sign language glosses into a coherent sentence of spoken words with the correct order. In contrast to ISLR, CSLR involves a larger number of glosses in each data sample. An instance of CSLR is illustrated in [41], where a signer conveys the meaning of "nice to meet you" or another example like the word "premier secours" that you can find in our dataset. In general, CSLR has a wider range of applications than ISLR because it aligns with the public's prefer-ence for communicating through sentences instead of isolated words or phrases [42]. Consequently, recent research studies in SLR have gradually shifted from ISLR to CSLR. Depending on the input sources, these studies can be divided into three major research directions: continuous vision-based sign language recog-nition (CVSLR) [43] [44], continuous gesture-based sign language recognition (CGSLR) [45] [46], and continuous

multimodal sign language recognition (CMSLR) [47][48]. Essentially, CVSLR and CGSLR only use RGB images and smartwatch data, respectively, as their input modalities, while CMSLR combines information from multiple modalities, including RGB images, hand images, depth images, joint coordi-nates, and smartwatch data.

2.2.4 Sign Language Translation

Sign Language Translation (SLT) represents the process of translating sign language into a spoken language, facilitating communication between deaf or hard-of-hearing individuals who use sign language and hearing individuals who do not. According to (Yin and Read, 2020) [49] SLT involves two primary steps: Sign Language Recognition (SLR), which extracts sign language glosses from videos, and a translation system that converts these glosses into spoken language. Recent advancements, such as the STMC-Transformer model, have demonstrated that glosses are inefficient representations of sign language, achieving state-of-the-art results in both gloss-to-text and video-to-text translations [49]. SLT faces challenges due to the multidimensional nature of sign language, which relies on both manual and non-manual cues, the absence of standardized written scripts for sign languages, and the need to translate complete sentences rather than individual words [49] [50]. The value proposition of SLT lies in its ability to provide accurate and efficient solutions for bridging the communication gap, thereby enhancing accessibility and inclusivity in various settings such as education, healthcare, and customer service.

2.3 Sign Language Translation Systems

2.3.1 Historical Development Overview

Early systems for SLT

Sign Language Translation (SLT) systems have experienced significant advancements, motivated by the need to bridge the communication gap between Deaf and hearing communities. This review encapsulates the historical progression of SLT systems, emphasizing key milestones and technological developments. Initially, from the 1960s to the 1980s, efforts focused on manual sign recognition and translation, utilizing basic image recognition techniques and heavily relying on human translators for recognizing and translating isolated signs [51] [52]. Before the advent of deep learning (DL) as the predominant methodology for machine translation (MT), alternative non-DL approaches were employed to tackle the task of SLT by decomposing it into discrete steps. Particularly critical was the recognition of continuous signs, necessitating its seamless integration into the workflow, especially

for video inputs. Typically, videos underwent manual annotation, with researchers concentrating on translating gloss into text. Statistical MT (SMT), rule-based MT (RBMT), and example-based MT (EBMT) emerged as the principal methodologies utilized for translation [53]. SMT, prevailing in contemporary literature, relies on statistical models derived from parallel corpora. RBMT relies on rules devised by domain experts for translation, whereas EBMT operates on a data-driven basis [53], necessitating the utilization of parallel corpora to maintain a translation memory. Additionally, word alignment models such as GIZA++ [53] were employed to facilitate the alignment of text and gloss/sign correspondences. Despite the ascent of DL and neural MT (NMT), conventional SLT methodologies continued to be proposed. It is noteworthy that, prior to the establishment of standardized benchmarks like the RWTH-Phoenix-Weather-2014 dataset [53], comparing various SLT approaches proved challenging due to the scarcity of datasets and the lack of definitive conclusions. Furthermore, the introduction of custom datasets, often proprietary, in research publications posed obstacles to conducting equitable comparisons between methodologies. Early research in the field of SLT witnessed the emergence of diverse methodologies and techniques. Kamata et al. (1989) [54] introduced one of the pioneering approaches by translating spoken language text into signs through the extraction of quantifiers, numerals, and word units, aided by contextual analysis for sign selection. Veale and Conway (1994) [55] presented the Zardoz system, marking one of the initial interlingua systems for SLT. This cross-modal MT system facilitated the translation of both speech and text into sign languages (ISL, ASL, and JSL) through the utilization of morphological rules, idiomatic reduction, unification grammar parsing, interlingua representation, schematization, anaphoric resolution, spatial dependency graphs (SD-graphs), and concept-to-sign mapping [53]. The transfer MT approach also found prominence, exemplified by the work of Lee and Kunii (1992) [56], who conducted text-to-SL translation by generating a dependency tree from morphological analysis, which was then transformed into an SL dependency tree. Smart glove technology gained traction due to limitations in gesture recognition from videos. Ohki et al. (1994) [57] developed an SL-to-text translation system using hand shape and position data from gloves, employing feature extraction and pattern matching. Similarly, Sagawa et al. (1996) [58] utilized hand shapes and positions from smart gloves for sign recognition through dynamic programming matching, enabling translation between video and text. Tokuda and Okumura (1998) [59] utilized a direct MT system, creating a word-sign corpus and implementing the SYUWAN MT system. In instances of missing entries, various techniques were proposed for translation, leading to the introduction of the Sign Language Description Method (SLDM). Bauer et al. (1999) [60] employed Hidden Markov Models (HMMs) for sign recognition from video, followed by translation into text using a translation model and a language model. Grieve-Smith (1999) [61] advocated for a literal orthography for

SL representation, employing syntactic structure transfer for translation into spoken language text. These varied methodologies underscore the evolution and exploration of approaches in SLT, setting the stage for subsequent advancements in the field [55];[57];[58];[60];[61].

2.3.2 Evolution of technology used in SLT systems

The evolution in the 1990s and 2000s, driven by computer vision and machine learning, introduced more sophisticated image processing methods, such as convolutional neural networks (CNNs), improving sign language recognition (SLR) systems. However, these systems were still limited to static hand signs [62][51]. The 2010s marked a surge in SLT research due to advancements in deep learning and the availability of extensive datasets. The introduction of recurrent neural networks (RNNs) and long short-term memory (LSTM) networks enabled the recognition and translation of continuous sign language sequences [62][51]. Recent years have seen transformative breakthroughs with transformer-based architectures and attention mechanisms, leading to more accurate and efficient SLT systems capable of handling complex sign language sequences [63][62]. Despite these advancements, challenges such as the need for diverse large-scale datasets, robust recognition algorithms, and practical applications persist. The advent of deep learning (DL) technology has spurred researchers to apply it to the field of machine translation (MT), yielding promising outcomes. Neural networks have played a pivotal role in eliminating the necessity for manual word alignments and language-specific rules, thereby facilitating the development of multilingual models [53]. However, this advancement has been accompanied by a substantial increase in data requirements, rendering small datasets ineffective in achieving significant progress. The availability of larger datasets, exemplified by RWTH-Phoenix-Weather-2014 [53], has been crucial in overcoming this obstacle, yet there persists a demand for even more extensive and diverse datasets, particularly when juxtaposed with domains like image classification, as demonstrated by the case of Imagenet [64]. The introduction of transformers [65] has heralded new opportunities in the realm of machine translation, including the application of transfer learning techniques to non-spoken language translation (NSLT) tasks [66]. Notably, methodologies such as data augmentation, initially pioneered in the field of natural language processing (NLP), are currently being investigated for their efficacy in NSLT applications. Prior to the ascendancy of transformers, RNNs constituted the cornerstone of NSLT architectures. The encoder-decoder framework, in particular, gained prominence, as exemplified by systems like DeepASL. DeepASL [67] employed a hierarchical bidirectional deep recurrent neural network (HB-RNN) coupled with a Connectionist Temporal Classification (CTC) loss function to facilitate word- and sentence-level translation of sign language. While this approach aimed to translate sign language into speech through wearable devices such as smart glasses, concerns were

raised regarding the unequal treatment between signers and non-signers. Specifically, the reliance on wearables placed an additional burden solely on signers, and the translation method favored written text over sign language for individuals who are deaf or hard of hearing. Future research aims to explore multimodal approaches, integrate SLT with other AI technologies, and develop more accessible interfaces.

2.3.3 Previous works in sign language translation

Previous research in sign language translation has predominantly focused on developing image-based models, highlighting the need for future studies to investigate the use of videos. This transition is essential as sign language is a multidimensional form of communication, incorporating both manual and non-manual cues that are challenging to capture with static images alone [68][62]. Current literature identifies several limitations, including biased assumptions and a lack of robust methodologies, resulting in low reliability [68]. To address these issues, researchers have explored various approaches, including machine learning and computer vision [68]. For instance, one study achieved 93.7% accuracy in recognizing 500 gestures in Chinese Sign Language using machine learning, while another employed deep learning to enhance translation accuracy [68]. Recent advancements, such as the STMC-Transformer model, have improved video-to-text translation, outperforming traditional gloss-based methods [62]. These projects demonstrate significant advancements in sign language translation technology, with a focus on improving communication between Deaf and hearing communities, enhancing accessibility, and addressing specific needs such as autism support. The most promising sign language translation projects currently underway include [69][68]:

- **SignON Project:** This European-funded Horizon 2020 project aims to bridge the communication gap between Deaf, hard of hearing, and hearing individuals by developing a sign language translation mobile application and open communications framework. The project focuses on facilitating information exchange among users of spoken languages and Deaf sign language users, particularly in the Irish, British, Dutch, Flemish, and Spanish sign languages, as well as English, Irish, Dutch, and Spanish spoken languages [70].

- **SimulSLT: End-to-end Simultaneous Sign Language Translation:** This project, presented at the 29th ACM International Conference on Multimedia, focuses on developing a real-time sign language translation system that can translate sign language videos into spoken language in real-time. The system uses a combination of computer vision and machine learning techniques to recognize and translate sign language [69].

- **LLMs are Good Sign Language Translators:** This research explores the potential of leveraging large language models (LLMs) for sign language translation. The study demonstrates the effectiveness of LLMs in translating sign videos by converting them into text and then translating the text into spoken language. The results show significant improvements in translation quality when using LLMs compared to traditional methods [71].

- **Real-Time Sign Language Translation Systems:** This project, presented at the 2022 11th

International Conference on Modern Circuits and Systems Technologies (MOCAST), reviews recent advancements in real-time sign language translation systems. The study highlights the importance of developing efficient and reliable systems that can effectively translate sign language into spoken language [69].

- **Development of a New Sign Language Translation System for People with Autism Spectrum Disorder:** This project aims to develop a sign language translation system specifically designed for individuals with autism spectrum disorder. The system focuses on improving communication between individuals with autism and their caregivers or therapists, enhancing the quality of life for those with autism [69].

2.4 3D Avatar Technology

Recent advancements in 3D technology have led to the creation of 3D body avatars, widely used in video games, virtual worlds, and various industries such as fashion, film, military, medical, and fitness. These avatars provide accurate sizing, virtual actors, and customization of uniforms and equipment, enhancing disease diagnosis [72].

In 1992, Neal Stephenson coined the term ”avatar” to represent a human in cyberspace [73]. Schlemmer and Trein (2008) noted that the avatar concept originates from Indian culture, symbolizing the manifestation of an immortal entity. The term ”avatar” comes from the Sanskrit word Avatãra, meaning descent or incarnation [73]. Indent computing, avatars depict users' personas in online forums and virtual communities, often as 2D icons or 3D models [74].

Technologically, avatars range from realistic human representations to imaginative constructs, serving as visual depictions within simulated settings [73]. Berdic, Mihic, and Dragan (2016) describe 3D avatars as dynamic graphical representations of individuals in virtual environments, created through various means such as 3D scanning [72].

Dear and Brian (2017) traced the term ”avatar” in gaming to the 1979 PLATO role-playing game and Norman Spinrad's 1980 novel, Songs from the Stars [75]. Neal Stephenson's 1992 novel Snow Crash further popularized ”avatar” in a virtual-reality context [76]. Diamandis (2020) highlighted the evolution of avatars from Ultima IV: Quest of the Avatar in 1985 to sophisticated 3D renderings in games like Second Life and World of Warcraft [77].

Peter (2020) discussed the transition from usernames to visual representations like AOL's ”Buddies” in 1996 and Yahoo!'s ”Yahoo! Avatars” in 2004, leading to personalized profile pictures on Myspace and Facebook [77]. In 2005, IKEA introduced anthropomorphic AI systems with Anna, paving the way for modern virtual assistants like Siri and Alexa [77].

2.4.1 3D Human Avatars

Structure of a 3D Human Avatar

The conventional definition of a human 3D avatar entails a representation composed of a 3D geometric mesh depicting a specific neutral pose, accompanied by a texture and a skinning mechanism that facilitates the transformation of mesh vertices in response to pose alterations. The addition of clothing and hair introduces additional complexity to the model, as accurately modeling these elements necessitates access to precise 3D data. However, such components are integral to rendering a complete avatar.

Prior to exploring the diverse methodologies for creating human 3D avatars, it is pertinent to provide a concise overview of the constituent elements of an avatar. Although avatars essentially comprise 3D models from varied sources, the environments in which they are utilized typically share common attributes. Consequently, ensuring that the avatar satisfies minimum requirements for successful rendering in its designated environment is paramount. This structural framework provides designers with guidelines to effectively accomplish their objectives.

Materials and textures are pivotal components in enhancing the realism of the model. Texture is contingent upon the underlying material and significantly influences the visual appearance of the model's geometry. The material properties of the model encompass various channels such as diffuse, ambient, and emissive attributes.

MPEG-V Standard

Creating avatars is a time-intensive endeavor, especially as virtual reality gains increasing popularity and diverse virtual environments are developed. In light of this, establishing a standard for defining the structure of virtual objects becomes imperative to facilitate interoperability. To address this need, the MPEG committee convened to develop a standard aimed at harmonizing disparities among existing and emerging metaverses [78]. This standard, known as MPEG-V Media Context and Control (ISO/IEC 23005), is divided into four parts, each addressing specific areas of standardization [78].

Part four of the MPEG-V standard is particularly relevant to this discourse as it focuses on providing data representation formats for describing avatars intended for exchange between different virtual worlds [78][79]. Notably, this part solely pertains to the metadata associated with avatars and does not encompass the representation of their geometry, sound, scent, animation, or texture[79].

Interoperability between virtual worlds is achieved in the MPEG-V standard through the implementation of a common schema [79]. This schema facilitates the mapping and transfer of various

attributes between virtual worlds, including the appearance and animation of characters [78][79]. The standardized schema encompasses 150 distinct parameters from various virtual worlds, organized into 14 attribute groups[79].

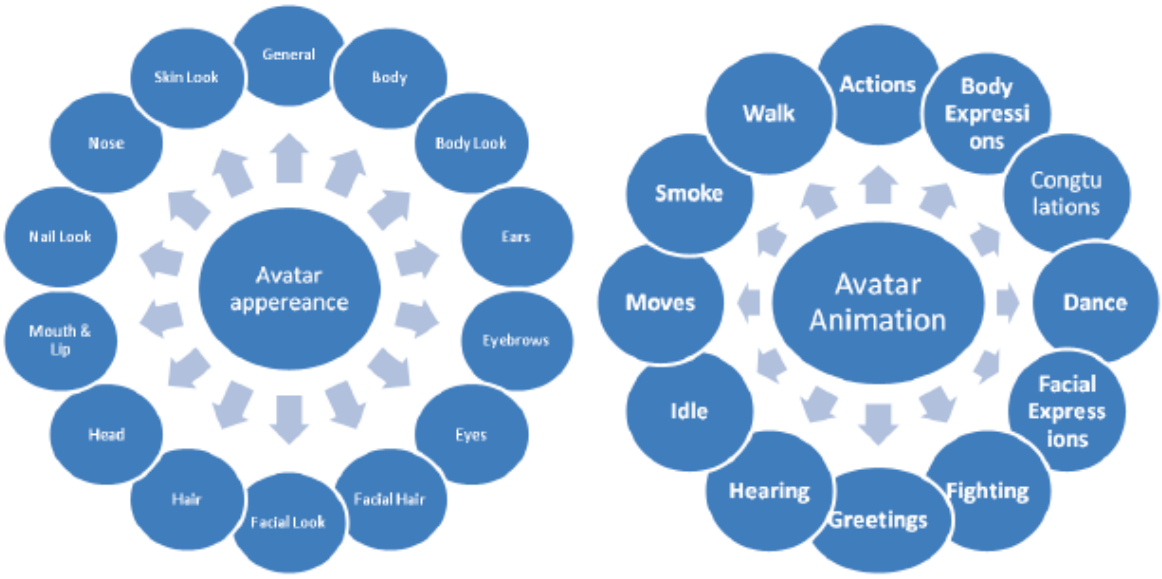

Figure 2.2: Avatar appearance & Avatar animation [79]

As an illustration, suppose an avatar from a particular virtual environment possesses the following properties [79]: a body height of 165 units, a body fat percentage of 15, an egg-shaped head, and is attired in a brown jacket. If this information is stored in a file named ”my_mesh.mp4,” the corresponding XML element generated using the standardized schema would resemble the following:

```
<Appearance>
        <Body>
                <BodyHeight value=165/>
                <BodyFat value=15/>
        </Body>
        <Head>
                <EggHead value="true"/>
        </Head>
        <Clothes ID=1 Name="jacket_brown"/>
        <AppearanceResources>
                <AvatarURL value="my_mesh.mp4"/>
        </AppearanceResources>
</Appearance>
```

Figure 2.3: the generated xml element by using the standardized schema [79].

The parameters described above serve as essential data for importing the character from one virtual environment into another that comprehends the same structure and format . This approach

facilitates the reuse of avatars across diverse virtual worlds and enables the connection of disparate environments. By adhering to a standardized structure and format, avatars can seamlessly transition between platforms, fostering interoperability and expanding the interconnectedness of virtual spaces [79].

Figure 2.4: 3D avatar in different virtual environments [80]

Motion Capturing

Motion capture (mocap) technology, utilized for recording and analyzing the movement of objects, people, or animals, has seen substantial advancements since its initial use in the 1960s [81]. It gained significant traction in the 1980s within the film industry, exemplified by its application in the 1989 film ”The Abyss” [81][82]. Today, mocap technology spans multiple domains, including video games, sports, medical applications, and sign language tasks. Various techniques such as optical, acoustical, mechanical, electromagnetic, inertial, and markerless systems are employed, each leveraging specific sensors and cameras to convert physical movements into precise digital data for animation and detailed analysis [81][83][82].

The integration of artificial intelligence (AI) has markedly improved the accuracy, efficiency, and versatility of mocap systems. AI advancements facilitate markerless motion capture, allowing for unrestricted and natural movement tracking using computer vision and machine learning algorithms [84]. AI-driven pose estimation and real-time motion tracking are critical for applications demanding high precision and immediacy, such as virtual reality gaming, sports analysis, and live entertainment [84]. These technologies provide low-latency data capture and processing, ensuring seamless execution of complex choreography and enhancing virtual production environments [84].

Furthermore, AI significantly enhances data processing and analysis in mocap. Advanced algorithms automatically detect patterns, anomalies, and specific gestures, contributing to fields such as biomechanics, sports performance analysis, and character animation. AI-powered facial motion capture and gesture recognition translate human expressions and movements into highly realistic animations, improving interactions within virtual environments [83][82][84].

The integration of mocap technology in sign language tasks involves capturing and analyzing the movements of signers to enhance the understanding and translation of sign language. Mocap technology captures the 3D movements of signers performing gestures and sentences, processes the data to identify markers, interpolates missing frames, and segments signs for analysis and classification [85][86]. This process aims to acquire quantitative knowledge about sign language movements, facilitating better understanding and synthesis of expressions [87][88]. Machine learning models are employed to modify and connect captured movements, enabling the creation of data-driven avatars for natural and realistic sign language synthesis[86][87]. Wearable sensors combined with natural language processing are used in sign language translation tasks, focusing on gesture recognition using electromyographic data and pattern recognition methods to translate gestures into text. Developing motion capture corpora, such as the LSF-SHELVES corpus [85][88], provides valuable resources for future studies on iconicity and spatial referencing, supporting linguistic analysis and machine learning applications in sign language research[85][86][87][88].

For effective utilization of mocap data formats like BVH (Biovision Hierarchy) and FBX (Filmbox) [89], 3D avatars must decode and apply hierarchical joint structures and animation keyframes [89]. AI algorithms enhance this process by accurately interpreting motion capture data, resulting in more lifelike avatar animations [90] [91]. Tools such as Plask Motion, Rokoko Vision, and DeepMotion Animate 3D exemplify how AI has democratized mocap technology, making it accessible and cost-effective by automating data processing and minimizing the need for specialized hardware [92].

In the programming realm, libraries and frameworks like OpenPose, MediaPipe, and OpenCV, alongside deep learning frameworks such as TensorFlow and PyTorch, are pivotal for mocap development and deployment. These tools support real-time multi-person 2D and 3D pose estimation, hand and face tracking, and integration with other machine learning models[93][94][95]. Companies like Dollarsmocap and Rokoko provide comprehensive mocap solutions, including full performance capture systems and AI-based video-to-animation tools, addressing the increasing demand for high-quality, affordable motion capture technologies in character animation and beyond [96][97].

3D File Formats for Avatars

A multitude of 3D file formats are employed in the representation of avatars, reflecting the expanding diversity in response to the evolving landscape of 3D commerce. Among the prevailing formats, FBX stands out as a proprietary format owned by Autodesk since 2006, facilitating the comprehensive exchange of both 3D geometry and animation data [79]. FBX files encompass crucial components essential for complete animation, encompassing bones, meshes, lighting, camera attributes, and geometric properties [79]. Moreover, the hierarchical organization of scene nodes within FBX files enables precise positioning, rotation, and scaling relative to their parent coordinate system [79]. Another prominent format, OBJ, developed by Wavefront Technologies, primarily emphasizes the representation of 3D geometry [79]. This open format intricately stores data pertaining to vertex positions, two-dimensional texture coordinates (UV), vertex normals, polygonal faces, and texture vertices, with the added flexibility of supporting unlimited colors and the definition of multiple objects within a single file [79]. Notably, OBJ's vector-based nature facilitates scalable object rendering without compromising resolution [79]. COLLADA, managed by the Khronos Group, adopts an open XML schema standard with a .dae file extension, defining an exchange format conducive to interoperability [79]. Furthermore, glTF and its updated iteration, glTF 2.0, maintained by the Khronos Group, are hailed for their versatility and efficiency, often likened to the "JPEG of 3D." These formats support a wide range of functionalities, including static models, animation, and dynamic scenes, with glTF 2.0 introducing advancements such as Physically Based Rendering (PBR) support and performance enhancements [79]. GLB, the binary version of glTF, offers expedited loading times and compact file sizes while encompassing critical supporting data such as textures, shaders, and geometry/animation [79]. Additionally, USDZ, developed collaboratively by Apple and Pixar, represents a closed, proprietary format tailored explicitly for Augmented Reality (AR) applications [79]. While these formats constitute the forefront of avatar representation, other formats like 3DS and SKP further enrich the landscape, demonstrating the diverse array of options available for avatar portrayal [79].

Creation of 3D Avatar

3D human representation, 3D avatar, is described with data acquired using 3D scanning. When the required data is gathered, usage of different software allows manipulation of the avatar. In this section we will discuss and provide a brief description of different techniques for 3D data and software which are most useful for their usage [98].

The exploration of 3D scanning methodologies reveals a diverse array of techniques and instruments employed in various fields [72].

- 3D Laser Scanning utilizes laser triangulation, time of flight, and phase shift methods either independently or in combination, enabling versatile scanning capabilities [72].
- White Light Scanning employs white light patterns projected onto objects, captured by sensors to extract three-dimensional data [98][72].
- Photogrammetry, rooted in conventional photography, involves capturing multiple images of objects for manual or automated identification of shared points, facilitating 3D measurements [99][72].
- Machine Vision, commonly used for 2D data detection like barcodes, employs stereo vision to create three-dimensional representations by integrating images and establishing corresponding points [72].
- Destructive Slicing entails capturing sequential images of objects while systematically removing thin sections, resulting in comprehensive 3D representations[72].
- MRI scans stack CT or MRI scans to generate 3D models, originally developed for medical use but expanding into manufacturing and industrial applications[100][101][72].
- Theodolites, integral to surveying, determine angular orientations and object distances through angular data acquisition, often featuring automated functionalities[72].
- Trackers monitor the position of measuring devices using various techniques, recording positions during measurements[72][98].
- Specialized software tools process data from different scanning methods to construct three-dimensional models, with popular software including Blender, SketchUp, and SolidWorks[102]. The profession of 3D avatar development requires a blend of artistic and technical skills, encompassing tasks such as modeling, texturing, lighting, animating, and compositing [72]. Additionally, 3D avatar software generators allow customization of avatar attributes such as body shape, clothing, and facial features[98][72].

2.4.2 Usage of 3D Avatars

Present Uses

3D technology is constantly advancing and its implementation is growing rapidly, making it difficult to keep track of developments. While 3D avatars are commonly associated with games and entertainment, they are also utilized in various industries for different purposes.

The medical field has seen advancements with the introduction of the VECTRA Whole Body 360 system, utilizing 3D avatars for expedited disease diagnosis and treatment, particularly in combating skin cancer [103]. This system involves positioning patients within a scaffold equipped with 46 high-resolution cameras, allowing for comprehensive scans of dermoscopic lesions to monitor changes over time and prompt further treatment if necessary [103].

In the fitness sector, 3D body scanning technology is emerging for assessing and monitoring physical fitness progress. However, consumer-oriented offerings are limited, with companies like Naked developing mirrors equipped with 3D depth sensors and rotating platforms for scanning users [72]. This technology enables individuals to track changes in their body shape and weight over time, with data transferred to their smartphones for analysis and monitoring [72].

Education has witnessed a transformation with the integration of technology, offering immersive learning experiences. Companies like Vcom3D [1] provide game-based simulations, such as STAT, to enhance critical thinking skills among medical practitioners [72] . Additionally, Vcom3D has developed Sign4me, a mobile application for learning sign language using 3D avatars, and SignSmith, an authoring tool for creating sign language animations [72].

In the fashion industry, 3D scanning and avatars are revolutionizing online clothing purchases by enabling virtual try-on fit personalization. Companies like Fitnect and Bodi.me utilize motion-capture devices to create visual avatars and recommend garment sizing [72], reducing returns and improving customer satisfaction.

Entertainment industries, including video games and films, are leveraging 3D scanning technology for character creation and immersive experiences. EA Sports employs 3D scanning in its FIFA franchise to capture stadium structures and player facial features, enhancing realism [72]. Networked virtual environments offer users real-time interactions and immersive experiences [72], indicating a growing trend towards the integration of 3D avatars in entertainment media.

Future Uses

the use of avatars is increasingly prevalent, particularly in gaming and cinema, thanks to the widespread adoption of 3D technology. Looking ahead, we anticipate a landscape filled with innovative uses for avatars, although their exact trajectory is uncertain. Currently, 3D avatars are being utilized in the medical field, albeit limited by technological constraints. There is potential for further exploration in creating detailed representations of the human anatomy, which could aid in diagnosing health issues with greater precision. According to Berdic, Mihic and D. Dragan (2016) [72], Physiotherapists could use 3D scans to develop tailored treatment plans, but progress is hindered by the

[1] http://www.vcom3d.com/company

lack of advanced 3D scanning technology. In sports, there is potential for integrating 3D avatars to enhance live broadcasts by providing immersive insights into gameplay. Additionally, there are efforts to use 3D avatars to improve the apparel shopping experience by allowing customers to visualize themselves wearing clothes in-store, potentially eliminating the need for physical try-ons [72]. While these developments are in their early stages, they offer promising prospects for the future, including personalized clothing design and augmented reality-assisted furniture browsing. These advancements have the potential to not only simplify daily tasks but also enhance life-saving capabilities.

2.4.3 3D Avatars Applications & Sign Language

The increased interest in 3D virtual human avatars has resulted in their widespread use, particularly in fulfilling the educational needs of deaf and hearing-impaired students [104]. These students face considerable difficulties in achieving proficiency in spoken language literacy. Sign languages serve as the primary mode of communication for millions of individuals with varying degrees of hearing impairment, both within their community and in interactions with those who can hear [105]. Hence, it is imperative to develop systems that assist them in overcoming social integration barriers. Sign language interpretation systems have been developed extensively to translate narratives, poetry, scientific terminology, and other textual content into signed languages (SLs) [104]. These systems prioritize accurate capture and organization of sign language into coherent linguistic structures, along with its presentation through avatars or synthesized videos that faithfully replicate the gestures conveying meaning in sign language [105].

3D avatars have several applications in the field of sign language, significantly enhancing accessibility, education, and research. One primary use is in sign language synthesis and translation, where 3D avatars can automatically translate written or spoken language into sign language by animating the avatar to perform the corresponding signs. This technology enables the creation of content in sign language from text or speech, thereby making information more accessible to the deaf community. Furthermore, in the realm of sign language learning and education, interactive 3D interfaces featuring signing avatars can facilitate learning by displaying and teaching sign language vocabulary and grammar[105].

This allows users to practice and master sign language at their own pace. Additionally, 3D signing avatars can be integrated into various digital platforms and applications, providing sign language interpretation and making content and services more accessible for deaf users [105]. In research and analysis, 3D avatar systems enable the study of sign language structure, linguistics, and usage by capturing and representing sign language digitally, allowing for in-depth examination and

manipulation [105]. Lastly, these systems contribute to sign language documentation and preservation by creating digital archives and libraries of sign language content, ensuring the longevity and continued use of sign languages for future generations [105].

2.5 Sign Language Representation Using 3D Avatars

Davidson (2006)[106] introduced PAULA, a signing avatar utilized for educational purposes, with a focus on fostering the development of sign language recognition skills through demonstrations and quizzes. Yorganci, Kindiroglu & Kose (2016) [107] emphasized the benefits of avatar-based tutoring over video content for evaluating sign language proficiency, citing challenges associated with the creation, editing, storage, and transfer of educational materials using videos. Projects such as ViSiCAST and eSign have standardized sign language specifications, employing HamNoSys and SIGML notations for synthetic signing rather than costly motion capture technology. Various initiatives, including the 'Research Toolkit Project' and the automated conversion system proposed by Kaur & Kumar (2016) [108], showcase the effectiveness of synthetic signing methods.

Papadogiorgaki et al. (2005) [109] introduced a VRML animation generation system based on SignWriting Markup Language, while Ferreira (2017) [110] developed the 3DSL system for translating Portuguese into Portuguese Sign Language.
Bouzid & Jemni (2014) [111] investigated alternative scripting approaches for sign language animation, and Murtagh (2019) [112] proposed a lexicon definition for Irish Sign Language animation grounded in Role and Reference Grammar.

Animation synthesis for sign language, as reviewed by Huenerfauth (2014) [113], Maarif et al. (2018) [114], and others, entails seamlessly blending individual words into cohesive animations. Challenges include capturing finger movements, simulating robotic motion, and ensuring fluid transitions. Initiatives like Content4All strive to develop lifelike sign language interpreters employing photorealistic 3D avatars [105].

Research endeavors are directed towards enhancing avatar intelligibility, involving deaf individuals in avatar assessment, and underscoring the significance of movement and facial expressions in sign language comprehension . Initiatives are underway to curate sign language corpora for avatar playback, leveraging motion capture technology and crowdsourcing techniques for data acquisition and annotation [105].

2.5.1 Hamburg Notation System

HamNoSys, developed by the Institute for German Sign Language (IDGS) at the University of Hamburg, is a well-established pictographic notation for deaf signing. It supports the definition of individual signing gestures and can represent the signs of various national sign languages through an alphabet of approximately 200 pictographic characters [115]. It does not rely on the sign language conventions differing from country to country and thus can be used internationally [116].

General structure of HamNoSys

HamNoSys for a given sign consists of transcription of the non-manual features, describing hand-shape, hand-orientation, location and movement of hand [108].

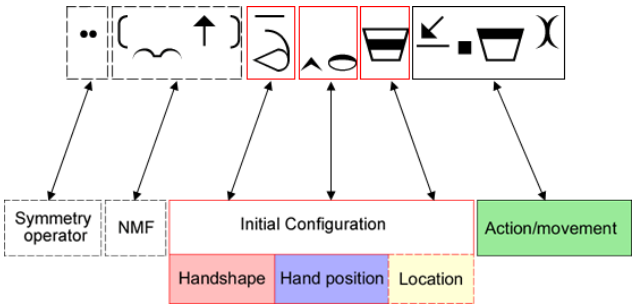

Figure 2.5: HamNoSys components [117].

- **Hand shapes:** The Hand Shapes are mainly grouped as Fist, Flat-Hand, Separated Fingers and Thumb combinations [108]. These four basic forms along with thumb variations (thumb extended or across the hand) and bending of fingers allows the user to write HamNoSys for any given Hand Shape [116]. Some of the fundamental Hand Shapes with flat and rounded hands are shown below in 3.3.

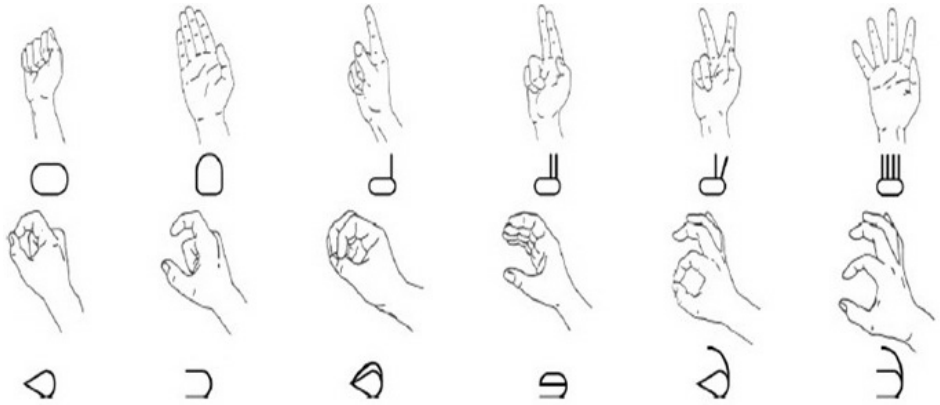

Figure 2.6: Basic Hand Shapes [117]

- **Hand orientation:** HamNoSys describes the orientation of the hand for a given sign by combining two components: extended finger direction and palm orientation [108]. There are three perspectives (signer's view, birds' view, and view from the right) which are used to show the direction of extended finger with respect to the signer's body. The palm orientation is also described with the same model for a given extended finger direction [116].

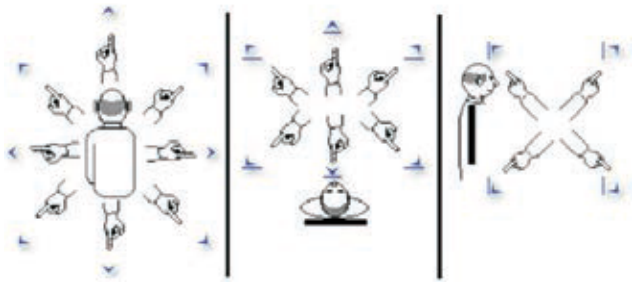

Figure 2.7: Extended Finger Direction [117]

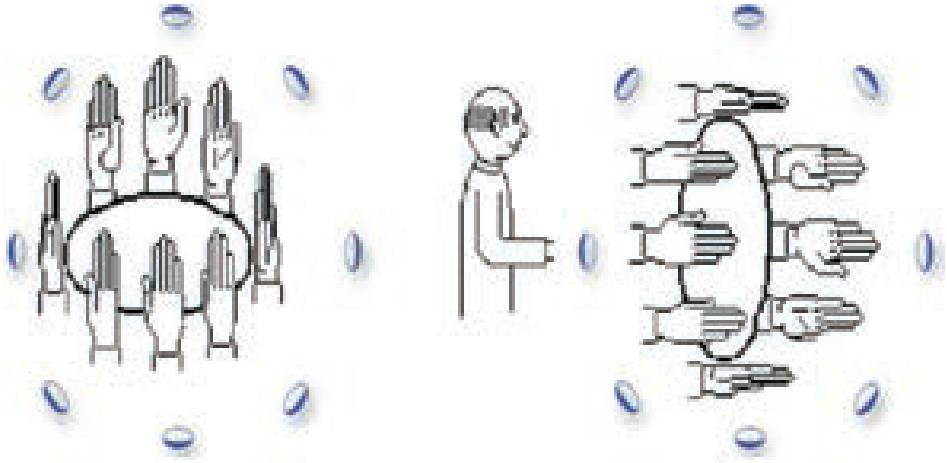

Figure 2.8: Palm Orientation [117]

- **Hand location:** The location specifications are used to tell the location of the hands of the signer, these are split into two parts [116]: The first part determines the location of the hand with respect to the body parts, where as the second part determines the distance of the hand with respect to the selected body part.

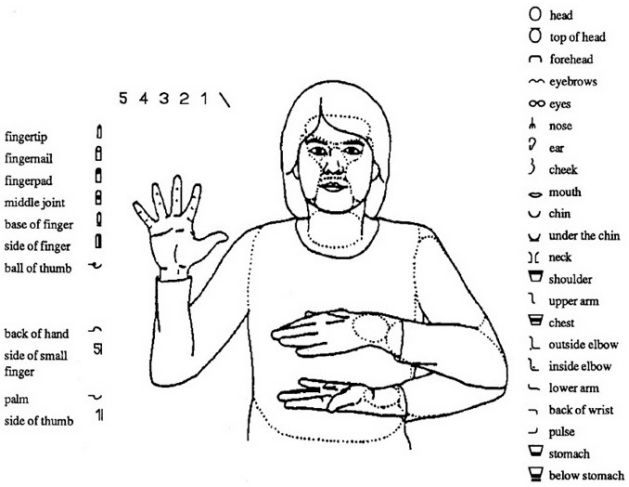

Figure 2.9: Hand Location with Respect to Body Parts [117]

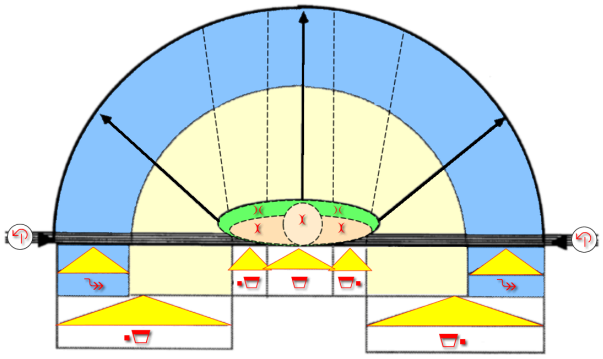

Figure 2.10: Downward view of person and signing space [117]

- **Hand movement:** The movement types are distinguished as straight, curved,wavy, zigzag, circular and spiral movements. The straight movements are either parallel to the body or with referent to the body of the signer [116][108]. The circular movements of the hand can either be clockwise or counter clockwise [116]. The manner of the movements can be any of the three degrees of size, i.e., large (expansive movement), normal size movement and small movement[116].

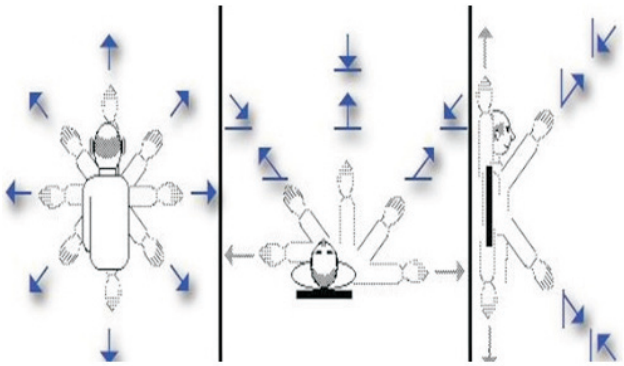

Figure 2.11: Straight Hand Movements [117]

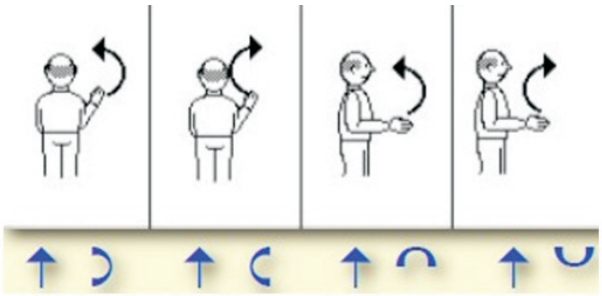

Figure 2.12: Curved Hand Movements [117]

- **Non-manual components:** The components which make the use of hands for signing are manual whereas those which include descriptions for shoulder shrugging, head movements, facial expressions or mouth movements are the Non-Manual ones [116]. There are various non-manual coding schemes which are defined for facial expressions (eye brows, eye gaze, eye lids and nose), limbs (head, shoulders and body postures) and gestures of mouth, but in ISL mainly the signer uses manual components [108].

2.5.2 SIGML language

The Signing Gesture Markup Language (SIGML) is designed as a comprehensive framework for representing and transmitting information about signing sequences, facilitating the animation of avatars [108]. SIGML is intended to integrate within the XML standards framework being developed by the W3C and is based on the existing HamNoSys notation [115]. SIGML aims to describe signing sequences across several linguistic levels, including glossing (using English words or phrases), phonology, phonetics, and physical articulation (e.g., motion capture data) [115]. This notation also captures information about signing speed and other temporal constraints. Different

linguistic levels offer alternative methods for driving the signing avatar; for instance, complete articulation data may obviate the need for phonetic or gestural information [115]. Initially, SIGML will operate at a basic glossing level, linking each ”gloss” element to a lexicon of motion-captured signs to drive avatars. As the project progresses, SIGML aims to facilitate avatar animation using lower-level linguistic data, offering flexibility to present arbitrary signs defined at the gestural level [115].

SIGML transplants the HamNoSys gestural model into an XML framework, allowing the definition of gestures by specifying initial hand configurations, actions, and hand positions within the signing space. Primitive actions, such as hand movements and changes in hand configuration, can be combined to form complex gestures. These features can be specified for one or both hands, as required [115]. Additionally, SIGML will enhance the rudimentary non-manual signals (e.g., facial gestures) currently in HamNoSys [115].

XML provides a structured, textual format for representing documents, which can be stored and transmitted over networks like the internet [115]. An XML document’s structure is defined by nested elements, each delimited by start and end tags. The structure can be specified using a Document Type Definition (DTD), which outlines how the document is to be interpreted [115]. SIGML’s structure and valid sign forms are defined by its DTD, ensuring clarity and consistency in its implementation [115]. SIGML for word “DEAF” is shown in Fig 3.13.

```
<sigml>
    <hns_sign gloss="Deaf">
        <hamnosys_nonmanual>
        </hamnosys_nonmanual>
        <hamnosys_manual>
            <hamfinger23/>
            <hamextfingerui/>
            <hampalmu/>
            <hamear/>
            <hamtouch/>
            <hammoveu/>
        </hamnosys_manual>
    </hns_sign>
</sigml>
```

Figure 2.13: SIGML for Word “DEAF” [117]

2.5.3 ViSiCAST Project

ViSiCAST a project funded under the European Union's Framework V programme, endeavors to facilitate enhanced accessibility to information and services for deaf individuals through their preferred mode of communication, sign language[118]. Building upon the insights gleaned from preceding UK endeavors, which are elaborated upon in a complementary exposition, ViSiCAST engages key stakeholders including UEA Norwich, TeleVirtual Norwich, the Independent Television Commission, and the Post Office [118][119]. These prior projects focused on pioneering the development of virtual human-based signing systems, exemplified by avatars Simon the Signer and TESSA [118][119]. At its core, ViSiCAST encompasses 'enabling technology' work packages dedicated to advancing language and avatar technologies established in earlier initiatives [118]. These efforts support work packages spanning three distinct application domains: Multimedia and World Wide Web, Face-to-Face Transactions, and Broadcasting [118]. Subsequent sections offer succinct outlines of these applications, followed by an examination of the enabling technologies themselves [118].

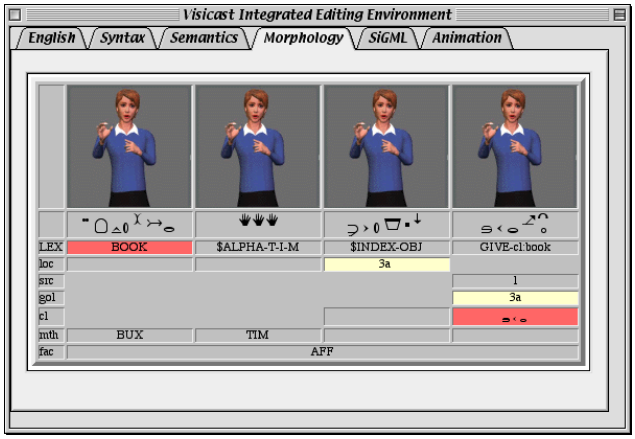

Figure 2.14: Visicast Integrated Editing Environment Interface

Overview of The Visia Avatar's Bones within MaskVR

The avatar's mesh is wrapped over a 'skeleton' which is composed of 78 'bones'. These bones act as control values for the mesh deformation algorithm [120]. They do not necessarily represent anatomical bones (although in the case of the main body and hands they do) [120]. Each bone is represented as 13 floating point values, and can be represented in either two forms [120]: Global or Local. A globalised bone has its values defined in terms of the global co-ordinate system, whilst a localised bone is defined in the co-ordinate system of a parent. Bones are always aligned along their x-axis. The supplied avatar provided by the ViSiCAST Control v2.0 is *Visia*, shown below.

Figure 2.15: Visia 3d Avatar [120]

There are three so-called, slap bones in the body: left_slap right_slap and neck_slap [120]. These are used in areas were x-axis rotations are common and can cause significant folding effect on the mesh [120]. Each slap bone is oriented in the same x-axis of a counter-part bone (i.e. right_forearm, left_forearm or neck), but with an approximately 90 degrees rotation about the x-axis [120]. The mesh attachment in these areas are specially set up in order to minimise folding during mesh deformation [120].

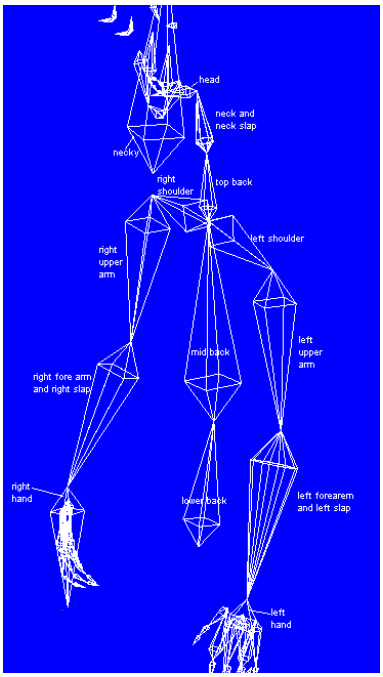

Figure 2.16: The bones of the body (torso, arms and neck) [120]

The following diagram shows the named bones of the left hand. Each bone is represented as a double pyramid [120]. As always the bone lies along its x-axis [120]. This axis, together with a triangle representing the XY plane has been highlighted in white [120]. There are two special bones in the hand called 'Thumb Magic' and 'Pinky Magic' [120]. 'Thumb Magic' is responsible for allowing the rotation of the thumb about the axis formed by the index knuckle and the meta-carpal joint [120]. 'Pinky Magic' allows for the curving of the palm caused by the Pinky Meta Carpal Joint bone, about the axis formed by the third knuckle and the base of the Pinky MCJ [120].

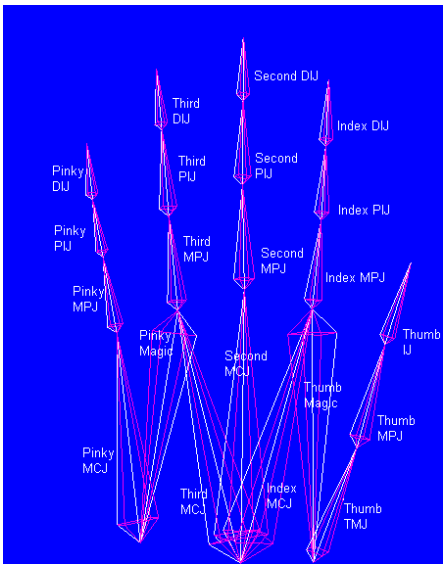

Figure 2.17: Bones of the Hands [120]

Finally, the following diagram shows the named bones of the Face.

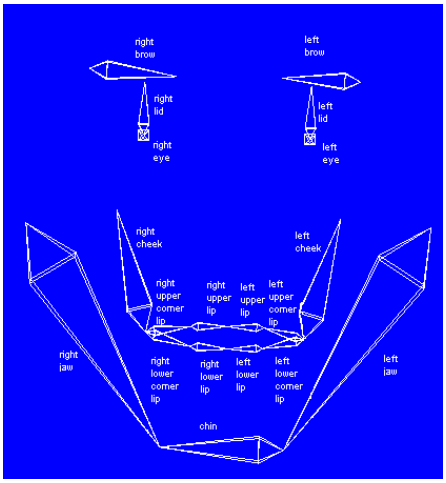

Figure 2.18: Bones of the Face [120]

JASigning

JASigning represents a pioneering virtual signing system designed to emulate natural sign language performances through virtual human characters. This innovative platform builds upon the foundation laid by its predecessor, the SIGMLSigning system, which was initially developed during the ViSiCAST and eSIGN projects [121]. Subsequent advancements were achieved through our contributions to the Dicta-Sign project.

Initially, the efforts predominantly leveraged Java JNLP applications for standalone and web-based usage [121]. However, due to the diminishing support for Java in contemporary browsers, this system has been deprecated [122]. Instead, recent strides have been made in the development of the CWA Signing Avatars, the present iteration of the system, which is built on HTML5 utilizing JavaScript and WebGL technologies [122].

CWA Signing Avatars

CWA Signing Avatars, abbreviated as CWASA, stands as cutting-edge virtual signing system engineered to replicate natural sign language expressions through virtual human characters [123]. This innovative platform represents an evolution beyond its predecessors, namely JASigning and SIGMLSigning systems, which were initially developed during the ViSiCAST and eSIGN projects [123].

Subsequent enhancements and refinements to the system were realized through our ongoing contributions to the Dicta-Sign project [123]. CWASA harnesses the capabilities of HTML5, employing JavaScript and WebGL technologies to facilitate its operations and user interactions [123]. This modern framework ensures optimal performance and compatibility across diverse web environments.

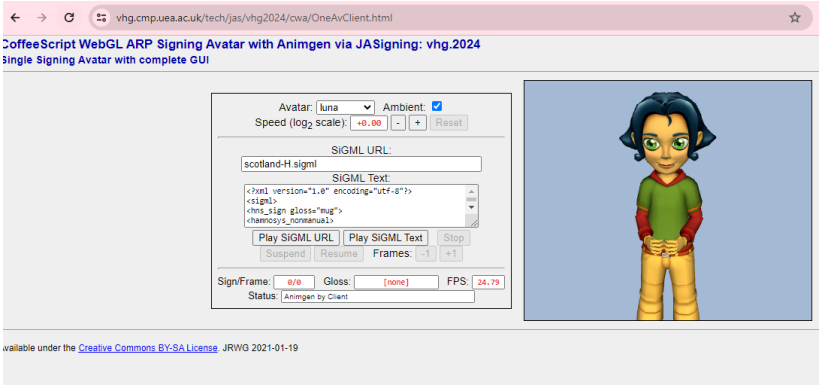

Figure 2.19: HTML5 Test Pages using JavaScript and WebGL from CWASA [124]

eSIGN Editor

The eSIGN Editor software allows the user to compose signed text to be performed by the eSIGN Avatar [125]. The editor gives the user an economic approach to create signed sequences by selecting signs from the lexicon and then modifying them with the assistance of specialised editors (that focus on different aspects of the sign's phonetics and morphology) where necessary [122][125]. The tool is delivered for Windows 32bit and MacOS platforms in three user interface languages (German, English, and Dutch). Modules to support features specific to individual sign languages have been implemented. In the realm of free-text translation, the approach adopted by eSIGN revolves around supporting human translators to streamline the process of converting spoken language into animation input, optimizing economic viability [122]. Central to this endeavor is the eSIGN Editor a pivotal tool designed to facilitate this translation process. The core principle underlying the program involves selecting sign-by-sign sequences from a lexicon and implementing morphological adjustments to individual signs or sign sequences as needed [125]. Various "assistants" are available within the editor to address diverse morphological and contextual modifications, including pronunciation databases to aid in depicting mouth movements and gestures [122][125].

During the composition phase, users have the capability to instantly validate the content by transmitting the text to the Avatar [125]. Depending on the operational environment at the content creator's location, lexicon maintenance functions can be performed within the eSIGN Editor or through external applications[122]. Documents generated through the editor can be exported to SIGML format[122]. However, the integration of SIGML into web pages remains a manual undertaking at present.

Given the expanded scope of the tool's usage across multiple content sites, as opposed to the originally envisioned single site, the focus has shifted towards developing a cross-platform solution that accommodates specific editors for each target language, such as fingerspelling [125][122]. While the current tool offers considerable flexibility in combining parameterized signs in linguistically meaningful ways, ongoing efforts are directed towards enhancing its efficiency [122]. These enhancements will be incorporated into future iterations of the tool, reflecting our commitment to continuous improvement[125].

Steps in the Translation Process

The intricacies of the translation process necessitate the expertise of individuals specifically trained in translation and/or the development of signed content, such as sign language interpreters or relay interpreters [126][122]. Despite its apparent complexity to the untrained eye, a cohort of interpreters has undergone successful training and adeptly navigates the utilization of the Editor [126].Within the

eSIGN framework[2], the initial text is inputted into the designated editor, subsequently segmented into discrete sentences or phrases [126]. Concurrently, the corresponding sign language translation is meticulously inputted alongside the original text, gloss by gloss [126].

Figure 2.20: Original text and gloss transcription in the eSIGN editor [126]

These glosses, accompanied by their respective HamNoSys strings, are sourced directly from the lexicon database tailored to the pertinent language [126]. Given the multifaceted nature of sign language, inclusive of dialectal variations and individual idiosyncrasies, multiple entries often exist for a singular gloss [126][122]. To ascertain the precise sign, several methods are available [126]:

- Observation of accompanying video clips, where available, depicting each sign within the database.
- Utilization of an avatar to demonstrate the sign.
- Reference to the HamNoSys string elucidating the manual components of the sign.

In instances where signs are absent from the database, transcription into HamNoSys is undertaken and subsequent integration into the database is facilitated [126].

Esign Virtual Human Avatar Technology

In the nascent stages of the preceding ViSiCAST endeavor -As mentioned in the previous Sections-, the utilization of motion capture technology stood as the primary method for crafting signed content. This methodology involved the application of magnetic and video-based tracking techniques to meticulously record the intricate movements of a live human signer [126]. The captured data underwent segmentation, yielding a compendium of motion data files representing a lexicon of distinct signs [124]. These files could then be seamlessly integrated and reproduced through an animated signing avatar as required. Notably, the TESSA system [124], a product of the ViSiCAST

[2] https://www.visicast.cmp.uea.ac.uk/eSIGN/

initiative aimed at facilitating interactions between a service clerk and a deaf patron within a postal environment, was built upon this foundational technology [126][122].

However, the substantial resource overheads associated with employing motion capture technology in this manner, coupled with the inherent constraints on the adaptability and reusability of signs from preexisting sequences, spurred the exploration of an innovative paradigm shift within the ViSiCAST project [126]. This paradigm shift involved the synthesis of signed animations directly from an input script encoded in SIGML notation [126].

The foundational framework for avatar animation and rendering within the eSIGN project is provided by Televirtual [126]. This encompasses the animation and rendering software integrated into Televirtual's Mask-2 system, complemented by the character specifications of the VGuido and Visia4 avatar personas [126]. Each avatar definition meticulously encapsulates a comprehensive depiction of the interrelation between the avatar's skeletal structure and key anatomical landmarks on the surface mesh—a critical component for the precise rendition of signing data [126]. Notably, VGuido was meticulously crafted for eSIGN following informal consultations with sign language practitioners in Germany and the UK. While conceivable, the creation of alternate avatars with distinct attire, hairstyles, etc, would entail considerable expenses [126][122].

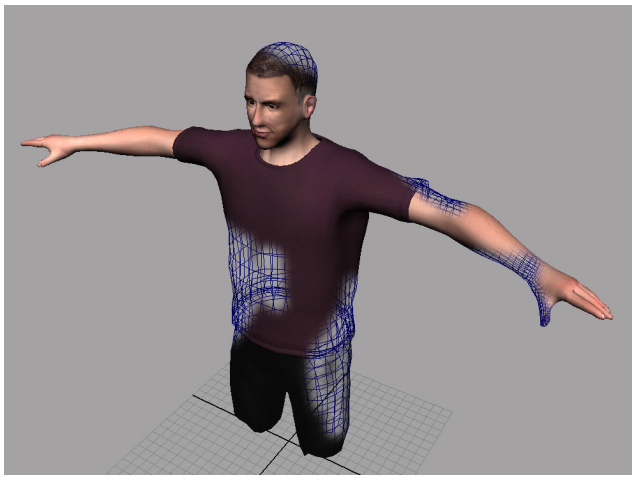

Figure 2.21: The VGuido avatar, developed by Televirtual for the eSIGN project. [126]

Augmenting the core animation and rendering technology provided by Televirtual is the SIGML-Signing software, a product of UEA's development efforts [126]. This software furnishes an ActiveX control and SIGMLToSigningAvatar module, which can be seamlessly integrated into HTML environments [126]. These components furnish a user-friendly interface designed to facilitate: the input of signing sequences encoded in SIGML, the generation of corresponding animation frames, and the scheduling of frame rendering utilizing Televirtual's Mask-2 ActiveX controls

[126]. emarkably, SIGMLSigning is versatile enough to interface with any avatar system that supports a rendering interface similar to the one provided by the Mask-2 system.[126].

Beyond the foundational animation capabilities elucidated above, the SIGMLSigning and Mask-2 systems bestow upon HTML developers scriptable control over system features such as virtual camera positioning, viewing parameters, lighting configurations, and the avatar's ambient motions, thereby fostering a dynamic and immersive user experience [126].

2.6 Related Works

Despite the advanced capabilities of modern computer systems, there is a notable lack of research focused on the development of machine translation (MT) systems for sign language, particularly in Algeria. Several types of MT systems have been employed for other sign languages, each with distinct methodologies and applications:

- **Automated Arabic-Arabic sign language translation system based on 3D avatar technology [127]:** A machine translation system translating Arabic into Arabic Sign Language (ArSL) has been developed utilizing advanced avatar technologies. Initially, a comprehensive ArSL dictionary was constructed using the eSign Editor Software. This dictionary comprises three thousand signs and can be integrated into the translation system, facilitating the conversion of written text into sign language. This dictionary will be freely accessible to researchers, acknowledging that its creation is intricate and time-consuming but a crucial milestone towards the machine translation of complete Arabic texts into ArSL with 3D animations. Subsequently, the translation system itself was developed. It conducts syntactic and morphological analyses of Arabic text and then employs a set of predefined rules to convert the text into ArSL, adhering to the linguistic structure and grammar of ArSL. The system's efficacy was evaluated using a parallel corpus of 180 sentences and the METEOR (Metric for Evaluation of Translation with Explicit ORdering) metric, where it achieved a relative score of 86

- **Automatic translation of English text to Indian sign language synthetic animations [127]:** This article introduces a prototype system for converting English text into Indian Sign Language (ISL) using synthetic animations in a real-world context. The translation system comprises several modules. Initially, a parsing module analyzes the input English sentence to generate a phrase structure grammar representation. Indian Sign Language grammar rules are then applied to reorder the words, addressing the grammatical differences between English and ISL.

An elimination module subsequently removes unnecessary words from the reordered sentence. Lemmatization is employed to convert words to their root forms, as ISL does not utilize word inflections. Each word is then checked against a lexicon containing English words and their corresponding HamNoSys notations. Words not found in the lexicon are replaced with synonyms. The words are converted into their respective HamNoSys codes; if a word is not present, HamNoSys codes are generated for each of its alphabets.

These HamNoSys codes are then translated into SIGML tags, which are sent to an animation module. This module converts the SIGML tags into synthetic animations using an avatar. Unlike existing systems that rely on videos, the proposed system is innovative in its use of synthetic animations. Furthermore, while current systems are limited to converting individual words and predefined sentences into ISL, the system is capable of translating full English sentences into ISL in real-time.

- **3D Avatar Approach for Continuous Sign Movement Using Speech/Text [128]:** a 3D avatar-based sign language learning system that converts input speech or text into corresponding sign movements for Indian Sign Language (ISL). The system comprises three main modules. First, the input speech is transcribed into an English sentence. Next, this English sentence is translated into the corresponding ISL sentence using Natural Language Processing (NLP) techniques. Finally, the 3D avatar's motions are generated based on the ISL sentence. The translation module achieves a Sign Error Rate (SER) score of 10.50.

- **ES2ISL: An Advancement in Speech to Sign Language Translation using 3D Avatar Animator[129]:** This study introduces a model and an initial implementation of a robust system designed to convert English Speech into Indian Sign Language (ES2ISL) animations. Such a system holds promise for significantly improving the quality of life for individuals with hearing impairments, particularly in facilitating interaction and information exchange with others. The primary objective of this system is to bridge the communication gap between hearing-impaired individuals in India and the broader community.

  The system leverages the semantics of Natural Language Processing (NLP), the Google Cloud Speech Recognizer API, and a predefined sign language database. Experimental results demonstrate that the proposed system surpasses existing models, achieving an average accuracy rate of 77%. Furthermore, it excels in terms of processing time, completing the conversion process in approximately 0.85 seconds. These findings underscore the potential of the proposed system to significantly enhance communication accessibility for individuals with hearing impairments.

2.7 Conclusion

This chapter has provided a detailed examination of the various tasks and technologies associated with sign language (SL) translation. Through this literature review, we have highlighted the significant strides made in the detection, identification, segmentation, recognition, translation, and production of SL. The advancements in machine learning and deep learning have played a pivotal role in enhancing the accuracy and efficiency of SL translation systems, making them more accessible and reliable for the Deaf and Hard of Hearing (DHH) community.

One of the key insights from this chapter is the critical role of 3D avatar technology in sign language synthesis and translation. These avatars not only facilitate more accurate and natural translations but also serve as effective tools for sign language education. The integration of motion capture technology and artificial intelligence has further improved the realism and precision of these avatars, enabling more immersive and interactive learning experiences.

As we look to the future, it is evident that ongoing research and technological innovations will continue to enhance the capabilities of SL translation systems. The development of larger and more diverse datasets, coupled with advancements in neural network architectures, will likely lead to even greater improvements in SL recognition and translation. Ultimately, these advancements will contribute to bridging the communication gap between the DHH community and hearing individuals, fostering a more inclusive and accessible society.

For further progress, it is essential to continue interdisciplinary collaborations, involving linguists, computer scientists, and educators, to develop comprehensive and culturally sensitive SL translation systems. By doing so, we can ensure that the benefits of these technological advancements are realized across diverse contexts and communities, supporting the linguistic and communicative needs of all users.

Chapter 3

Conception

3.1 Introduction

This chapter serves as a comprehensive overview of the methodologies, phases, and technologies involved in the development of our project, which aims to create a 3D avatar capable of performing Algerian Sign Language (ASL) gestures. We will outline the work process, highlight the steps involved, and emphasize the significance and potential impact of the project.

The chapter provides a detailed description of the system architecture, including the two primary methods employed: the Notation System Method (NSM) and the Gesture Animation via Motion Capture (GAMC). We will delve into the phases, requirements, and design of each method. Additionally, we will explore the technological choices underpinning our project, examining the software tools utilized and the rationale for their selection. Finally, we will review related works in the field, comparing various approaches and emphasizing the unique contributions of our project.

3.2 Overview

This project embarks on a multifaceted journey to establish a communication solution facilitating seamless interaction between Arabic-speaking individuals and users of Algerian Sign Language (ALSL) through an adaptable 3D avatar. With our overarching aim of dismantling prevailing communication barriers to foster inclusivity and efficiency, the initial phase involves conducting an exhaustive review of existing sign language translation systems while compiling a diverse dataset of Algerian Sign Language (ALSL) signs alongside their corresponding Arabic text translations. As we progress, our focus shifts towards the development of a lifelike 3D avatar adept at accurately performing ALSL signs. This stage integrates natural and expressive gestures, alongside realistic textures, facial expressions, and body movements, thereby enhancing user engagement. Advancing further, we leverage Computer Vision models to recognize ALSL signs and devise an algorithm for translating them into Arabic text. Concurrently, we integrate a user interface for Arabic text input and to display the avatar's sign language responses. Throughout the project's duration, we conduct continuous testing with both ALSL users and Arabic speakers to gather feedback aimed at refining the avatar's accuracy and responsiveness. Additionally, there is a dedicated emphasis on ensuring cultural sensitivity, particularly regarding Algerian Sign Language (ALSL) and Algerian Deaf culture, with ongoing collaboration with experts in these fields to maintain cultural authenticity and respect cultural nuances. This endeavor not only underscores our imperative of enhancing communication accessibility for ALSL users, thereby empowering autonomy and self-expression within the Deaf community but also showcases the potential for innovative technological applications in sign language interpretation, adaptable to diverse linguistic contexts and global regions.

The project is expected to be completed within a six-month timeframe, with the following tentative schedule:

- **Months 1-2:** Research and Data Collection
- **Months 3-4:** 3D Avatar Development
- **Months 5-6:** ALSL Recognition and Translation

This schedule allows for a structured development process, with ample time allocated for research, avatar development, and the implementation of recognition and translation algorithms. User testing and feedback will be ongoing throughout the project to ensure continuous improvement.

To solve the project's main problem, we employ two primary methods: the notation system method (NSM) and the Gesture Animation via Motion Capture method (GAMC). For each method,

we will delve into each phase in more detail in the upcoming sections, examining its nuances and subtleties.

3.2.1 Significance and Impact

The field of supporting deaf communities around the world attracts significant attention from research and scholarly publications. Deaf and hard of hearing individuals often encounter difficulties in various daily experiences. The language barrier constitutes a big obstacle for them, hindering their ability to perform daily tasks within the hearing community. This often leads to isolation and marginalization of deaf and hard of hearing individuals in closed communities away from the hearing community. The deaf and hard of hearing community in Algeria is no exception, as it requires attention compared to the research advancements in other sign languages. In this regard, our research aims to overcome these barriers to empower deaf and hard of hearing individuals to integrate with the rest of the community and achieve the principle of equal opportunities in education, employment, and other forms of interaction with society. We aspire to develop an innovative solution to overcome the aforementioned barriers by developing a high-precision dual translation system between Arabic and Algerian Sign Language, utilizing frameworks and artificial intelligence models specialized in recognizing hand signals, movements, and facial expressions. Additionally, we aim to leverage the advancements in 3D avatar technology, in collaboration with experts in Algerian Sign Language, to achieve the following effects:

1. Facilitating communication between the deaf and hard of hearing community and the hearing community smoothly.

2. Enriching datasets for Algerian Sign Language to train intelligent models by translating Arabic words and sentences into sign language.

3. Augmenting sources of deaf education by training avatars in a unified sign language.

4. Delivering promotional, awareness, and religious campaigns to the deaf and hard of hearing community.

5. Facilitating navigation and interaction with various web technologies.

This research not only addresses immediate challenges faced by the deaf community in Algeria but also sets a precedent for similar accessibility projects worldwide, promoting inclusivity and improving communication for Deaf communities globally.

3.3 System Architecture

3.3.1 System Design

The project initially aimed to develop a globally recognized 3D avatar technology for representing Algerian Sign Language (ALSL) while integrating AI capabilities to animate gestures realistically. Our approach naturally bifurcated into two distinct methods: the Gesture Animation via Motion Capture (GAMC) and the Notation System Method (NSM). GAMC focuses on creating a lifelike 3D character that performs ALSL gestures through advanced motion capture techniques, ensuring visual fidelity and user immersion. Concurrently, our goal was to establish a product scalable for international use and future research integration. Upon thorough research, we discovered the HamNoSys notation system, widely adopted across various sign languages globally and easily translatable into Sign Language Markup Language (SiGML). Opting for NSM enables compatibility with international standards and facilitates seamless integration with evolving tools and applications. However, focusing solely on NSM risked deviating from our primary goals of achieving realism and leveraging AI for gesture interpretation. By employing both methods, we maintain a balanced approach: NSM ensures linguistic accuracy and interoperability, while GAMC enhances visual realism and user interaction. This dual-method strategy ensures our system meets global standards, evolves with technological advancements, and fulfills our original objectives of creating a realistic, AI-driven ALSL representation.

1. **Notation System Method (NSM):** This method focuses on translating ALSL gestures into a structured textual format using established linguistic notation systems.

2. **Gesture Animation via Motion Capture (GAMC):** This method utilizes motion capture technology to animate a 3D avatar that performs ALSL gestures based on human motion data.

These methodologies encompass several phases, each crucial to the system's success.

- **NSM Method**
  - **Phase 01: ALSL Data Collection**
  - **Phase 02: ALSL Translation**
  - **Phase 03: From HamNosys To SiGML**
  - **Phase 04: Text processing**

  - **Phase 05: UI Based on CWASA**
  - **Phase 06: Test the System**
  - **Phase 07: Collaboration with ALSL Experts**

- **GAMC Method**
  - **Phase 01: ALSL Data Collection**
  - **Phase 02: Create a 3D avatar**
  - **Phase 03: Creating motion capture files**
  - **Phase 04: Enrich the 3D avatar**
  - **Phase 05: Set up an environment**
  - **Phase 06: Develop text treatment algorithm**
  - **Phase 07: Deploying a Prototype on Mobile**

3.3.2 Functional Requirements

- The system is required to accurately process Arabic texts as inputs.
- Utilize a comprehensive dictionary for translating Arabic textual words into Algerian Sign Language (ALSL).
- The system must ensure the accuracy of translating Arabic text into Algerian Sign Language gestures represented by the avatar.
- Incorporate a 3D avatar capable of executing ALSL gestures.
- The system must ensure that the embedded avatar is capable of translating the Arabic textual inputs to ALSL gestures derived from the dictionary.
- The system must ensure that the embedded avatar movements and textures exhibit realism.

3.3.3 Non-Functional Requirements

- The system in its applied model should prioritize user-friendliness.
- The system should guarantee efficient processing of system functions with low complexity.
- Ensure responsiveness across diverse computer devices.

3.3.4 Interaction Design

- **User Interface:** The user interface is designed to be intuitive, allowing users to input Arabic text easily and view the 3D avatar's sign language output. The user interface can be divided into three main sections:

    1. **Avatar Interaction Area:** A designated space is allocated for the avatar character, facilitating the accurate and visually comprehensible display of its movements. This configuration ensures the visual accuracy and comprehensibility of the translated sign language. Details and movements are presented in a manner that faithfully represents the translated texts or vocabulary, thereby ensuring clarity and precision for viewers. Furthermore, the viewing angle can be adjusted to examine movements from various perspectives.
    2. **Input and Output Submission:** This encompasses a user-input method where texts or phrases are entered into a designated text area for translation into sign language. Users can submit these inputs using interface buttons for display.

- **User Experience:** The system aims to provide a positive user experience by ensuring smooth and accurate translations, realistic avatar animations, and minimal latency. The project emphasizes a seamless user experience during interaction with various elements of the web page. The avatar area occupies the largest space on the page to facilitate clear observation of avatar movements. Users can adjust the viewing angle to enhance visibility of sign language gestures. A vocabulary lexicon is organized into categorized lists to ensure flexible access to desired words. Buttons, input spaces, and vocabulary lists are uniformly and ergonomically distributed to enable comfortable interaction with diverse elements. The project ensures the integration of different interactive elements on the page for a problem-free experience devoid of undesired disruptions. Additionally, algorithms are employed to process entered texts, optimizing the translation system's performance and enabling users to translate texts easily and clearly, thereby meeting user needs effectively.

3.3.5 Step-by-Step Workflow

NSM Method

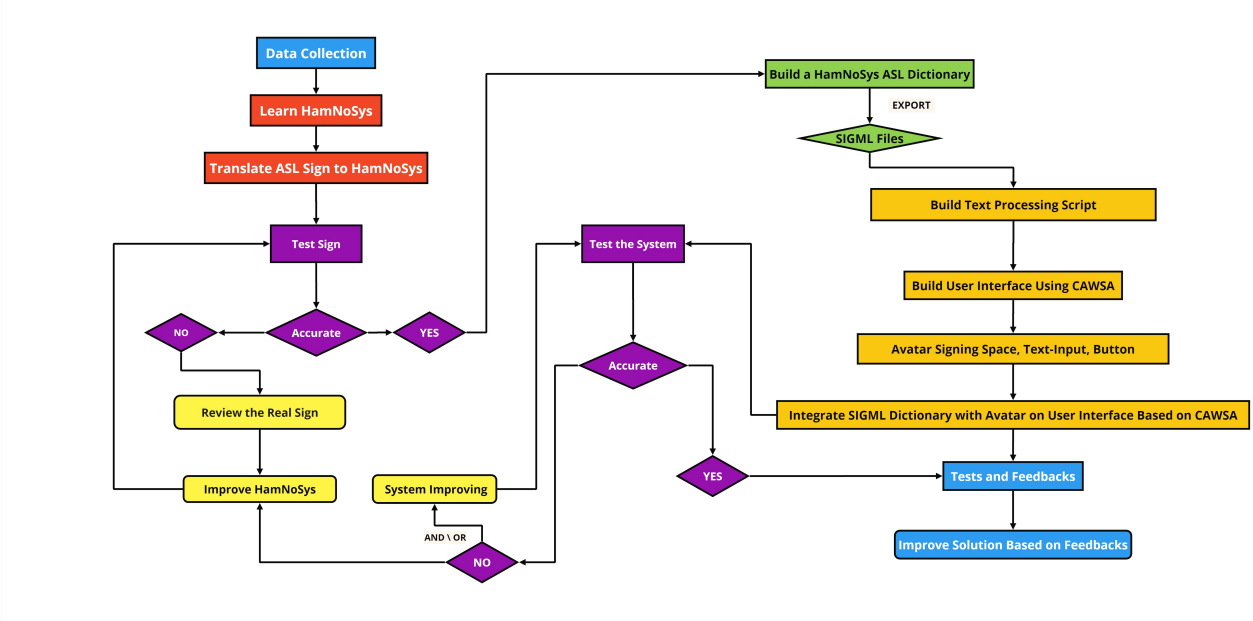

Figure 3.1: Notation System Method Flowchart

1. **Phase 01: ALSL Data Collection**

   - Gather a comprehensive dataset of Algerian Sign Language (ALSL) signs and their corresponding Arabic text translations.
   - Acquire datasets pertaining to French Sign Language (LSF) and engage in collaborative efforts with experts in Algerian Sign Language translation to discern data alignment between Algerian Sign Language (ALSL) and LSF, along with their corresponding Arabic text translations and their corresponding HamNoSys notation translations.

2. **Phase 02: ALSL Translation**

   - **Learn HamNosys:**
     - Acquire knowledge and proficiency in the Hamburg Notation System (HamNosys) for notating sign language gestures accurately.
     - Practicing notation of Algerian Sign Language gestures and reviewing notation of other sign languages based on the Hamburg Notation System to achieve accurate notation.

- **Translate ALSL Sign to HamNosys:**
  - Utilizing the dataset for a deeper understanding of the structure of Algerian Sign Language (ALSL) gestures and collaborating with (ALSL) translation experts to notate (ALSL) gestures using the Hamburg Notation System.
  - Convert each (ALSL) sign in the dataset to HamNosys notation and linking it with its corresponding Arabic textual translations.
  - Each sign from the Algerian Sign Language (ALSL) dataset is converted into HamNosys notation and associated with its corresponding Arabic textual translations.
- **Test Sign:**
  - Each sign from the Algerian Sign Language (ALSL) dataset is implementing the generated outcomes on avatar and overseeing their application for sign representation.
  - Validate the accuracy of the HamNosys notation by testing it to ensure it matches the actual ALSL sign in the dataset.
- **Is the Translation Accurate?**
  - If No:
    - Review the real sign and compare it with the notation.
    - Improve the HamNosys notation based on the review and retest the sign.
  - If Yes: Proceed to the next step.

3. **Phase 03: From HamNosys To SiGML**

- **Build a HamNosys ALSL Dictionary:**
  - Compile all the accurate HamNosys notations into a comprehensive ALSL dictionary.
- **Export to SiGML Files:**
  - Convert the HamNosys ALSL dictionary into SiGML (Sign Markup Language) files for digital processing.
  - Each ALSL sign from the Algerian Sign Language (ALSL) dataset is represented in the Dictionary by a SiGML File associated with its corresponding Arabic textual translations based on glosses in HamNoSys.

4. **Phase 04: Text Processing**

- **Build Processing Script for Text-Inputs:**
  - Developing custom input rules to prevent issues when users input Arabic text.

5. **Phase 05: UI Based on CWASA**
   - **Build User Interface Using CWASA:**
     - Create a user interface utilizing the CWASA Software.
   - **Design Avatar Signing Space, Text-Input, and Button:**
     - Using the CWASA features to design and integrate the signing space for the avatar, along with text-input fields and interactive buttons for user commands based on our specific purposes.
   - **Integrate SiGML Dictionary with Avatar on User Interface:**
     - Connect the SiGML dictionary to the avatar within the user interface to enable translation and signing.

6. **Phase 06: Test the Translation System**
   - **Test The System:**
     - Validate the accuracy of the Translation System via comprehensive testing procedures to ensure its alignment with the genuine ALSL sign in the dataset.
   - **Is the Translation Accurate?**
     - If No:
       * Review the real sign and compare it with the Avatar Signing.
       * Improve the HamNosys notation based on the review and retest the sign.
       * Improve The Translation System including the HamNosys-SiGML conversion and CWASA installation.
     - If Yes: Proceed to the next step.

7. **Phase 07: Collaboration with ALSL Experts**
   - **Tests and Feedback:**
     - Conduct thorough testing sessions with ALSL users and Arabic speakers.
     - Gather feedback to identify areas for improvement in the system's accuracy and responsiveness.

- **Improve Solution Based on Feedback:**
  - Refine and enhance the solution based on user feedback and ongoing testing results.
- **Final Implementation and Deployment:**
  - Implement the final solution and prepare it for deployment to end-users.

Notes:

- Throughout this process, we ensure continuous collaboration with experts in ALSL and Arabic languages to maintain accuracy and cultural sensitivity.
- We ensure to regularly update the HamNosys dictionary and SiGML files to accommodate new signs and improvements.
- We maintain detailed documentation of each step to facilitate future enhancements and troubleshooting.

GAMC Method

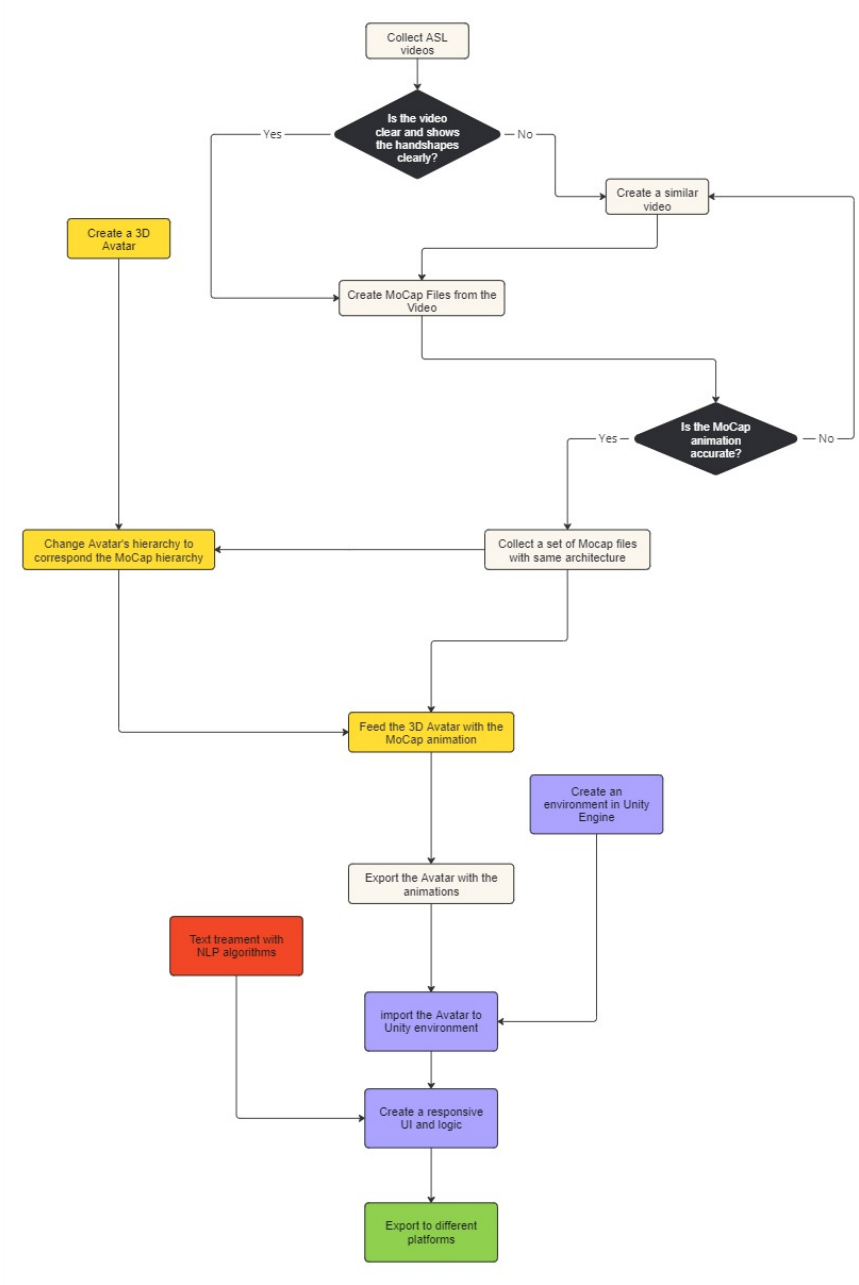

Figure 3.2: Gesture Animation via Motion Capture method Flowchart

1. **Phase 01: ALSL Data Collection**

    - Gather a set of videos of people performing ALSL gestures, each video represents a word or a sentence, the videos must be clear and show the hands and gestures in all the frames.

- Some videos might be re-performed by us if they are not clear.
- Include criteria for video quality and frame rate.

2. **Phase 02: Create a 3D avatar**

    - Create a 3D avatar using a 3D software along with various tools that might facilitate the process.
    - Consider using existing 3D models to save time, ensuring they meet the detail requirements.
    - The 3D avatar must have an armature so it can perform gestures.
    - The upper body of the 3D avatar must be highly detailed, especially the hands and fingers so the gesture can be clearly seen.
    - Test the avatar's movements early to ensure the rigging is functional.
    - Emphasize the importance of realistic skin and texture details for the hands and face.

3. **Phase 03: Creating motion capture files**

    - Using Computer Vision libraries and scripts, create motion capture files that can be understood by the 3D avatar software.
    - Explore and compare different computer vision libraries (e.g., OpenPose, MediaPipe) for accuracy.
    - Perform tests to validate the motion capture accuracy against real gestures.
    - The motion performed by the 3D avatar when fed by the motion capture file must be identical to the video in the first step, otherwise a video of the same gesture must be recorded again.
    - The motion capture file can be in different formats such as: fbx, collada, Alembic, Wavefront, SVG, Stanford PLY, STL, Bio-Vision...

4. **Phase 04: Enrich the 3D avatar**

    - Each library, script, or software that generates animation file from a video uses a custom armature hierarchy, the first modification must be done to the avatar is to match its armature to the finally selected hierarchy.
    - Test the compatibility of the animation files with the 3D software.
    - Feed the animation files to the 3D avatar to have all the animation we have prepared.

- Perform iterative testing and refinements to ensure smooth animations.

5. **Phase 05: Set up an environment**

    - Prepare the environment and lighting that will surround the avatar.
    - Ensure the environment is adaptable for different scenarios (e.g., classroom, outdoors).
    - Set up a flexible camera with different angles and zooming levels for clear visibility of the avatar.
    - Import the created avatar with the proper settings.
    - Create a user interface.

6. **Phase 06: Develop text treatment algorithm**

    - Create an algorithm that uses natural language processing (NLP) techniques to preprocess text.
    - Ensure the algorithm can handle synonyms and variations in sentence structure, and selects the right animations to play.
    - Link the algorithm with the user interface then make sure it interacts well with the UI elements.

7. **Phase 07: Deploying a Prototype**

    - Optimize the 3D model and animations for different devices performance.
    - Configure the main settings such as adjusting the resolution, quality settings, and performance optimizations.
    - Implement controls for both mobile and desktop with the necessary user interface adjustments.
    - Release the prototype to a selected group of users for feedback, then use this feedback to make necessary improvements and updates to the application.

Notes:

- We prioritize the use of open-source tools and libraries to ensure accessibility and reproducibility of our work.
- While collecting data we must obtain consent from individuals being recorded and ensure ethical considerations.

- To make a good 3D avatar, we must choose a 3D software that supports detailed modeling and rigging, such as Blender or Maya, we must also consider using existing 3D models to save time, ensuring they meet the detail requirements.
- We maintain detailed documentation of each step to facilitate future enhancements and troubleshooting.

3.3.6 Components

The two methods have the same goal, therefore they have identical components.

Input Module

Text Capture: Arabic text is captured using standard input methods such as keyboard input.

Processing Module

- **Text Processing Techniques:** Text processing techniques preprocess Arabic text inputs, facilitating the accurate translation of texts into ALSL gestures. These techniques ensure the system can manage synonyms and sentence variations while linking smoothly with the user interface for seamless interaction.
- **Translation Algorithms:** The system uses rule-based translation algorithms to convert Arabic texts into ALSL. These algorithms are designed to handle the dictionary of Algerian Sign Language.

Output Module

- **3D Avatar Technology:** The 3D avatar is capable of performing ALSL signs with lifelike movements. The avatar's design includes realistic textures, facial expressions, and body movements to enhance user engagement.
- **Rendering and Display:** The project's culmination is manifested in a web page featuring a dedicated space for an avatar to display textual input results. This setup incorporates avatar control tools for avatar manipulation, facilitating avatar switching or result display through specified buttons. Furthermore, a text input area enables the entry of Arabic natural language texts intended for translation into Algerian Sign Language (ASL). Additionally, an Arabic vocabulary lexicon is available to facilitate the selection of words required for translation.

3.4 Technological Choices

3.4.1 NSM Method

Technology Selection

The selection of technologies and tools is crucial for the project's success.

- **3D Avatar Technology:** CWASA technology was chosen for its robust capabilities in performing ALSL signs with lifelike movements based on SiGML files. The avatar's design includes realistic textures, facial expressions, and body movements to enhance user engagement.
- **Translation System:**
    1. We selected the Hamburg Notation System (HamNoSys) for transcribing gestures in Algerian Sign Language (ASL). The choice was motivated by its inherent flexibility, defined by a collection of 200 symbols that cover handshapes, movements, movement direction, hand positioning in relation to the body or head, non-manual movements, and other aspects. This inclusive framework enables accurate encoding of Algerian Sign Language gestures with exceptional fidelity.
    2. In order to streamline the transformation of notations into a format comprehensible to avatars, Signing Gesture Markup Language (SiGML) was chosen for its ability to convert HamNoSys notations into tags. This functionality enables avatars to generate sign language movements and differentiate between various components including palm orientation, finger direction, hand position, and other parameters, utilizing the information encoded within these tags.
- **Web-based Interface:** For project demonstration purposes, we selected CWASA technology from among others. This technology was chosen for its compatibility with the outputs of the translation system employed in the project. It facilitates the display of the eSign 3D Avatar on a web page, ensuring seamless translation of SiGML file contents and the generation of avatar movements based on this data. Moreover, it allows for the addition or removal of speed control tools, presentation of input results, and management of pre-configured input spaces.

Comparison of Alternatives

1. **Sign Writing Notation Systems**

   Sign writing notations play a crucial role in converting words into a format suitable for automated solutions and animation translation. The most renowned and extensively used notations include Stokoe, Gloss, SignWriting, and HamNoSys. Each notation method exhibits distinct merits and demerits [1]. Table 3.1 presents a comparative analysis of these notations, with HamNoSys identified as the optimal choice for lightweight solutions.

Notation name	Language dependency	Non-manual signs support	Used for	Computer supported encoding	Signs arrangement
Stokoe	Yes	No	Academic	ASCII Codes	Linear
Gloss	Yes	Yes	Academic	ASCII Codes	Linear
SignWriting	Yes	A few	Public	ASCII and Unicode	Pictures
HamNoSys	No	Yes	Academic	Unicode based font	Linear

Table 3.1: Comparison of famous sign writing notation systems [1]

3.4.2 GAMC Method

Technology Selection

1. **3D Avatar creation Technology:**

   Our 3D avatar will represent ASL gestures based on human footage, necessitating a skeletal structure similar to that of a human being. Therefore, the chosen technology must support detailed rigging and animation capabilities. Various tools are available for creating and animating 3D avatars, each with its own set of features, strengths, and limitations. Here, we will explore some of the most prominent alternatives and provide a comparative analysis.

   - **Suggested tools**
     - Blender
     - Autodesk Maya
     - Cinema 4D
     - Houdini
     - 3ds Max

Comparative Analysis

Tool	Pros	Cons
Blender	- Free and open-source - Strong community support - Comprehensive rigging and animation tools - Supports various export formats	- Very steep learning curve for beginners - Sometimes less optimized for very high-poly models
Autodesk Maya	- Industry standard for animation and rigging - Extensive toolset and plugins - Strong support for complex animations	- Expensive licensing costs - High system requirements - Complex interface for new users
Cinema 4D	- User-friendly interface - Excellent motion graphics capabilities - Strong community and resources	- Expensive - Limited advanced character rigging compared to Maya and Blender
Houdini	- Powerful procedural generation - Excellent for complex simulations - Strong support for VFX	- Very steep learning curve - Expensive - Overkill for simple rigging and animation needs
3ds Max	- Strong modeling tools - Good for architectural visualization - Extensive plugins and scripts	- Expensive - Less focused on character animation compared to Maya - High system requirements

Table 3.2: Comparative Analysis of 3D Avatar Creation Tools

Based on this analysis, Blender emerges as a strong candidate for our project. It offers comprehensive rigging and animation tools, supports various export formats, and has a strong community that can provide support and resources. Additionally, being free and open-source makes it an accessible and cost-effective solution, especially for projects with budget constraints.

- **Decision**

  Given the needs of our project, including the creation of a detailed and lifelike 3D avatar that can accurately perform ASL gestures, Blender is the optimal choice. It balances robust features with accessibility, making it a suitable tool for both beginners and experienced animators.

2. **Selection of Motion Capture Format**

Selecting the appropriate motion capture (mocap) format is crucial for ensuring seamless integration with our chosen 3D avatar software and achieving accurate representation of ASL gestures. The chosen format must support detailed motion data and be compatible with Blender, our selected 3D avatar creation tool.

- **Criteria for Selection**

  The following criteria will guide our selection of the mocap format:
  - **Compatibility:** The format must be compatible with Blender.
  - **Detail and Accuracy:** The format should capture intricate movements, especially of the hands and fingers.
  - **Ease of Use:** The format should be easy to import and manipulate within Blender.
  - **Community and Support:** Formats with strong community support and documentation are preferred.

  Available Formats

  Here are some of the most widely used mocap formats:

-
  - **FBX (Filmbox):** A versatile format widely supported by most 3D applications, including Blender.
  - **BVH (BioVision Hierarchy):** A format specifically designed for motion capture data, widely used and supported.
  - **COLLADA (Collaborative Design Activity):** An open standard XML schema for exchanging digital assets.
  - **Alembic:** A format for exchanging animated scenes between different software packages.
  - **MDD (Motion Designer's Data):** A format that stores vertex animation data.

- **Comparative Analysis**

Format	Pros	Cons
FBX (Filmbox)	- Widely supported across many 3D applications - Handles complex animations and detailed motion capture data - Strong industry standard	- Can be proprietary with some features not supported in all applications - Larger file sizes
BVH (BioVision Hierarchy)	- Specifically designed for mocap data - Well-documented and widely used - Easy to edit and manipulate	- Less support for complex rigging and non-standard skeletons - Limited detail compared to newer formats
COLLADA	- Open standard with strong interoperability - XML-based, making it easy to parse and edit - Supports a wide range of data types	- Can be complex to set up and configure - Performance issues with very large files
Alembic	- Excellent for complex animations and simulations - Highly efficient with data compression - Supports both geometry and animation	- Steeper learning curve - Not as widely supported for rigging and skeletal animation
MDD (Motion Designer's Data)	- Great for vertex animation - Lightweight and efficient for specific use cases - Easy to integrate with various software	- Limited to vertex data only - Not suitable for skeletal animations

Table 3.3: Comparative Analysis of Mocap Formats

- **Decision**

  All the evaluated mocap formats meet the necessary criteria and can be seamlessly integrated into our workflow. Since we have tools that can convert videos to various mocap file formats, there is no significant difference between them in terms of usability for our project. We can import any of these formats into Blender without difficulty.

  However, if we plan to develop a Python script using a computer vision library to create mocap files, we should consider formats that are more understandable and easier to work with, such as BVH and COLLADA (which is XML-based). These formats offer better readability and ease of manipulation, making them suitable for custom development and further processing.

3. **Selection of Game Engine**

   Choosing the right game engine is crucial for developing an application that features a 3D avatar performing ASL gestures in response to user text input. The game engine must support advanced 3D graphics, animations, and integration with Natural Language Processing (NLP) libraries.

   - **Available Game Engines**

     Here are some of the most prominent game engines that can be considered for this project:

     – **Unity:** A highly popular game engine known for its ease of use, extensive documentation, and strong community support. Unity supports C# scripting and has robust tools for 3D animation and integration with external libraries.

     – **Unreal Engine:** Known for its high-fidelity graphics and powerful visual scripting system (Blueprints). Unreal Engine supports C++ and has a steep learning curve but offers unparalleled rendering quality.

     – **Godot:** An open-source game engine that is lightweight and highly customizable. Godot supports multiple scripting languages, including GDScript, C#, and C++.

     – **CryEngine:** Renowned for its stunning graphics and realistic physics. CryEngine uses C++ and offers a suite of powerful tools for game development.

     – **Blender Game Engine:** Although not as powerful as the others, Blender Game Engine is integrated within Blender, making it convenient for projects that heavily

rely on Blender's modeling and animation tools.

Evaluation Criteria

The following criteria will guide our evaluation of these game engines:

- **Ease of Use:** The game engine should have a user-friendly interface and be accessible to developers with varying levels of experience.
- **3D Graphics and Animation Support:** The engine must support advanced 3D graphics and provide robust tools for character animation.
- **Scripting and Extensibility:** The ability to integrate NLP libraries and other external tools via scripting.
- **Community and Support:** Strong community support and comprehensive documentation to assist with development.
- **Cost:** Preferably free or low-cost to reduce the financial burden on the project.

- **Comparative Analysis**

Game Engine	Pros	Cons
Unity	- User-friendly interface - Extensive documentation and community support - Supports C sharp scripting and external libraries - Robust 3D animation tools	- May require additional plugins for high-fidelity graphics - Licensing cost for advanced features
Unreal Engine	- High-fidelity graphics and rendering quality - Powerful visual scripting (Blueprints) - Strong community support	- Steep learning curve - Requires more powerful hardware - Licensing cost for commercial use
Godot	- Open-source and free to use - Lightweight and customizable - Supports multiple scripting languages - Active community	- Less mature compared to Unity and Unreal - Limited high-end graphics capabilities
CryEngine	- Stunning graphics and realistic physics - Comprehensive suite of tools - Free to use with royalty-based model	- Steep learning curve - Smaller community compared to Unity and Unreal
Blender Game Engine	- Integrated with Blender - Convenient for projects using Blender for modeling/animation - Open-source and free	- Less powerful compared to dedicated game engines - Limited community support and documentation

Table 3.4: Comparative Analysis of Game Engines

- **Decision**

All the evaluated game engines have their strengths and can be used to develop an application featuring a 3D avatar performing ASL gestures. However, based on our criteria,

Unity emerges as the most suitable game engine for this project. Its user-friendly interface, robust 3D animation tools, support for C# scripting, and extensive community support make it ideal for integrating NLP libraries and developing a comprehensive application.

We also need to highlight that none of the propositions support Python libraries integration, therefore, there is no possibility to use Python NLP libraries such as NLTK. The only techniques we can use are the ones that can be developed with C#.

While Unreal Engine offers superior graphics, its steep learning curve and hardware requirements make it less suitable for rapid development. Godot and CryEngine, though powerful, have smaller communities and may lack some of the ease-of-use features present in Unity. Blender Game Engine, while convenient for Blender users, lacks the advanced features and support needed for this project.

In conclusion, Unity's balanced feature set, strong community support, and extensibility make it the best choice for developing our ASL gesture application.

Related Works and Sign Representations Comparison

1. **Different Media for the Sign Representation**

Algerian Sign can be represented as real human video, sign pictures, coded sign language text, or 3D Avatar animation. Each approach has its own advantages and disadvantages, but synthetic animations are particularly suitable for translating spoken language into sign language. A comparative analysis of all these media types is presented in the table as depicted:

Kind of Media	Pros	Cons
Video Signs	• Realistic • Easy to create	• Time consuming to create • High memory consumption • Not supported by translation system
Pictures	• Very less memory consumption	• Time consuming to create pictures • Not realistic as compared to videos • Not supported by translation system
Coded Sign Language Text	• Minimal memory consumption • Supported by translation system as it is the written form and can be processed very easily	• Very difficult to read and understand • Required to be learnt
3D Avatar Animations	• Very less memory consumption • Can be easily reproduced • Supported by translation system • Avatar can be made different according to choice	• Not as realistic as human videos

Table 3.5: Comparison of Different Media for Sign Language Representation[2]

2. Related Works Comparison

The majority of research proposes various systems that effectively assist the deaf community with sign language processing. This section discusses the research conducted on sign language processing, which is summarized in Tables 3.4 and 3.5. This work is analyzed and compared based on several identified parameters, as depicted in Table 3.5.

Sr.	Ref.	Year	Sign language	Prog. language	Sign presentation	Avatar	Coverage/features	Limitation/drawbacks
1	[130]	2015	Sinhala	Python	Expr. Of Signs Def. in Files	Yes	Word, Phrase, Finger Spelling	MB, DSC, DC
2	[131]	2013	American	Java	Sign Writing Markup Language	Yes	Under Development	DSC
3	[132]	2017	American	-	XML-Gloss	No	Words, Finger Spell	DSC
4	[133]	2017	Arabic	-	-	Yes	Words	DSC
5	[108]	2016	Indian	Java	SiGML	Yes	Words	DSC
6	[134]	2015	Brazil	C++	XML	Yes	Words	MB, DC, DSC
7	[135]	2019	Arabic	-	Gloss System	Yes	Arabic Words, Stored Sentences	MB, DSC
8	[136]	2019	Vietnam	-	HamNoSys	Yes	Words	DSC
9	[137]	2019	Arabic	Rule Based	Video Sequences	Yes	Words	DSC
10	-	2024	Algerian (NSM)	JavaScript	SiGML	Yes	Arabic Words, Arabic Phrases	DSC
11	-	2024	Algerian (GAMC)	JavaScript, C#	SiGML, BVH	Yes	Arabic Words, Arabic Phrases	DSC

Table 3.6: Comparison between sign languages, tools and coverage of signs[1]

Sr.	Avatar mesh	3D software	Animation software	Animation engine	Multiplatform
1	Low poly	MakeHuman	Blender	Blender game engine	No
2	Low poly	-	-	WebSign player	No
3	No	No	No	No	No
4	Low poly	-	-	-	No
5	Low poly	-	-	JA SiGML App	No
6	Low poly	Blender	Blender	Irrlicht	No
7	-	-	-	-	No
8	Low poly	-	-	-	No
9	Low poly	-	-	-	No
10	Low poly	-	-	CWASA SiGML App	No
11	Low poly	Blender	Unity	Unity game engine	No

Table 3.7: Comparison of sign language translation systems based on software platform [1]

Evaluation Parameters:

- **Software Platform, Technology, Sign Language, Tools, and Sign Coverage:** Compare system features and support.
- **Avatar Mesh and Avatar:** Distinguish between low poly (fast render) and high poly characters.
- **Signs Coverage:** Identify system support for sentences and language features.
- **3D Software, Animation Software, and Animation Engine:** Determine if animations are manual or runtime-generated.
- **Multi-Platform:** Assess if the system runs on various platforms like mobile and PC.
- **Sign Presentation:** Identify notation used for sign language representation and highlight weaknesses.

System Developments:

- **Sinhala Sign Language (SSL) :** Developed by Punchimudiyanse et al [130]. using Python and Blender, animates sentences without video or motion capture.
- **American Sign Language (ASL):** Bouzid et al [131]. created a web-based system using SignWritingmarkup Language and low poly 3D avatars.
- **American Sign Language Transcription [132]:** Othman et al. developed a system for machine translation with a new XML representation.

- **Arabic Sign Language (ArSL):** Al-Barahamtoshya et al [133]. developed a system translating Arabic text to ArSL using a speech module and transformational rules.
- **Indian Sign Language:** Kaur et al [108]. used SiGML technology and JA Signing App for Indian sign language words.
- **Brazilian Sign Language:** Gonçalves et al [134]. developed a system with a 3D avatar using C++ and Irrlicht animation engine.
- **Arabic Sign Language in Gloss System:** Luqman et al [135]. represented Arabic signs and sentences.
- **Viennese Sign Language:** Da et al [136]. converted television news to 3D animations using Hamburg Notation.
- **Arabic Sign Language Animation:** Brour et al [137]. used video sequences and a rule-based system.
- **Arabic Algerian Sign Language:** in our NSM Method we used SiGML technology and CWASA Technologies for Algerian sign language Translation.

Limitations in Related Works:

- **Dependency Chain (DC):** Requires cost-effective 3D software, animation experts, and sign language experts for recording.
- **Memory Bound (MB):** High storage memory and bandwidth usage.
- **Dynamic Sentence Creation (DSC):** Inability to extract gestures for new sentences, relying on pre-recorded words/sentences.
- **Multi-Platform (MP):** Platform-dependent development, e.g., Android, Windows, Linux.

3.5 Conclusion

This chapter has presented an in-depth examination of the conceptual and architectural framework for the Automated Arabic Algerian Sign Language Translation System. By meticulously exploring the system design, we have elucidated the critical phases involved in constructing a robust and precise translation mechanism. The incorporation of the Notation System Method (NSM) and the Gesture Animation via Motion Capture Method (GAMC) ensures accurate and lifelike sign language representation, addressing fundamental communication obstacles faced by the deaf community. Additionally, this chapter has highlighted the importance of user-friendly interaction design and the judicious selection of technologies to enhance the system's performance and cultural authenticity. The insights and methodologies detailed in this chapter will underpin the subsequent implementation and refinement of the translation system, ultimately empowering the Algerian deaf community and advancing inclusivity in communication.

Chapter 4

Implementation

4.1 Introduction

The implementation of the Automated Arabic Algerian Sign Language (ALSL) Translation System combines two main methods: the Notation System Method (NSM) and the Gestures Animation via Motion Capture (GAMC) Method. This chapter discusses the development environment, hardware, programming languages, software, and tools used for each method. We then delve into the implementation details and results of each phase, showcasing how these components were integrated to achieve a functional and effective ALSL translation system. The following sections provide a comprehensive overview of our development process, highlighting the challenges faced and the solutions implemented.

4.2 Development Environment

To implement our solution, we require a set of hardware and software tools. This section presents an overview of these tools that have been used for both methods.

4.2.1 Hardware Specifications

In order to implement our solution, we used:

Computer Configuration:

For NSM method:

- Lenovo Ideapad 330s
- CPU: i5-8250U
- RAM: 12 GB
- GPU: Intel(R) UHD Graphics 620

For GAMC method:

- Lenovo Legion y520
- CPU: i5-7300HQ
- RAM: 16 GB
- GPU: Nvidia Geforce GTX 1060 6GB

Phone Configuration:

- Samsung Galaxy Note 8
- Camera: 12 MP Full HD 30 fps
- RAM: 6 GB

4.2.2 Software Specifications

Operating Systems

- **Windows 10:** Windows 10 was selected based on its compatibility with the hardware and the used tools.

Development Tools:

Notation System Method (NSM):

- **HamNoSys:** The Hamburg Notation System (HamNoSys) is used for encoding sign language gestures. It provides a standardized way to document the movements and expressions used in ASL, making it easier to convert these notations into visual representations [138].
- **SigML:** The Sign Gesture Markup Language (SigML) converts HamNoSys notations into animations displayed by the 3D avatar. This bridge between textual descriptions and visual representation ensures accurate and consistent gesture animations [139].
- **eSIGN Editor:** This tool is used to create and manage sign language dictionaries, vital for ensuring the accuracy and comprehensiveness of the ASL gestures performed by the avatar [125].
- **CWASA SIGML Player:** SIGML Player is a tool used to animate sign language data encoded in SiGML (Signing Gesture Markup Language). This player forms part of the broader CWASA (CWA Signing Avatars) system, which synthesizes natural sign language performance using virtual human avatars. The CWASA system is a successor to the earlier JASigning and SiGMLSigning systems developed in projects like ViSiCAST and eSIGN [125].
- **CWASA Virtual Signing System:** The CWASA system enhances the visual signing capabilities of the avatar, allowing it to perform complex ASL gestures fluidly and naturally. This system ensures that the gestures are lifelike and expressive, adhering to the nuances of sign language [140].
- **Visual Studio Code:** An integrated development environment (IDE) used for coding and scripting within the project, particularly for tasks involving SigML and CWASA integration [141].
- **JSON (JavaScript Object Notation):** A lightweight data-interchange format [142].

- **Wampserver:** A Windows-based web development platform for creating dynamic web applications.[143]
- **Vercel:** A frontend cloud platform for creating high-performance and customized websites [144].

Gesture Animation via Motion Capture (GAMC):

- **Blender:** A powerful open-source 3D modeling software used to create detailed character, rigging, and rendering, which are essential for producing realistic 3D Avatars.
- **Blender Add-ons:**
  - **MBLab:** A Blender plugin that assists in creating anatomically accurate humanoid characters.
  - **BlendARMocap:** Used for motion capture integration within Blender, enhancing the realism of the avatar's movements.
  - **BVH:** Biovision Hierarchy (BVH) files add-on, used for importing motion capture data into Blender to animate the 3D avatar.
  - **Rokoko:** Rokoko add-on is created to be used with Rokoko studio software, but we are using it for its ability to copy motion to different metarigs.
- **Dollars Mocap MONO:** A low-cost motion capture solution that provides basic movement data to enhance the avatar's gestures.
- **PyCharm:** An IDE used for developing and debugging the Python scripts necessary for integrating various components of the project.
- **Unity engine:** A real-time development platform used to test and deploy the 3D avatar in interactive environments.

Programming Languages:

- **Notation System Method (NSM):** || JS, HTML, CSS
  - **HTML5:** The markup language used to structure and give meaning to our web content, for example defining paragraphs, headings, and data tables, or embedding images and videos in the page.
  - **CSS3:** A language of style rules used to apply styling to our HTML content, for example setting background colors and fonts, and laying out our content in multiple columns [1].

[1] http://w3c.org

- **JavaScript:** A scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else [2].

- **Gesture Animation via Motion Capture (GAMC):**

  - **Python:** Python was utilized to develop a script responsible for generating BioVision Hierarchy (BVH) files from captured ALSL gesture videos. Python's versatility and extensive libraries for video processing and data manipulation made it ideal for this task, enabling efficient conversion of raw motion capture data into a format compatible with 3D animation tools.

  - **C#:** C# is the programming language used in Unity, the game engine chosen for developing the virtual environment and integrating the 3D avatar technology. C# is well-suited for Unity development due to its robust object-oriented features, strong integration capabilities with the Unity Editor, and support for creating interactive and visually compelling applications.

[2] http://w3c.org

4.3 Implementation Details and Results

In the preceding chapter, we outlined our dual approach to addressing the problem at hand. In this section, we will comprehensively detail the implementation of each method: Notation System Method (NSM) and Gesture Animation via Motion Capture method (GAMC), providing in depth explanations and procedural steps.

4.3.1 Notation System Method (NSM)

Phase 01: ALSL Data Collection

- **Data Sources:** The approach primarily relied on the DictaSign dataset. This corpus includes 1,000 lexical items translated into four different sign languages, with our research focusing on the equivalents in French Sign Language. Collaborating with a team of experts, we translated over 300 words from the dataset, aligning sign performances between Algerian Sign Language and French Sign Language. Figure 4.2 shows the official logo of the DictaSign dataset, while Figure 4.3 displays the web page interface of the dataset, which includes a list of over 1,000 words [145].

Figure 4.1: The logo of the DictaSign dataset

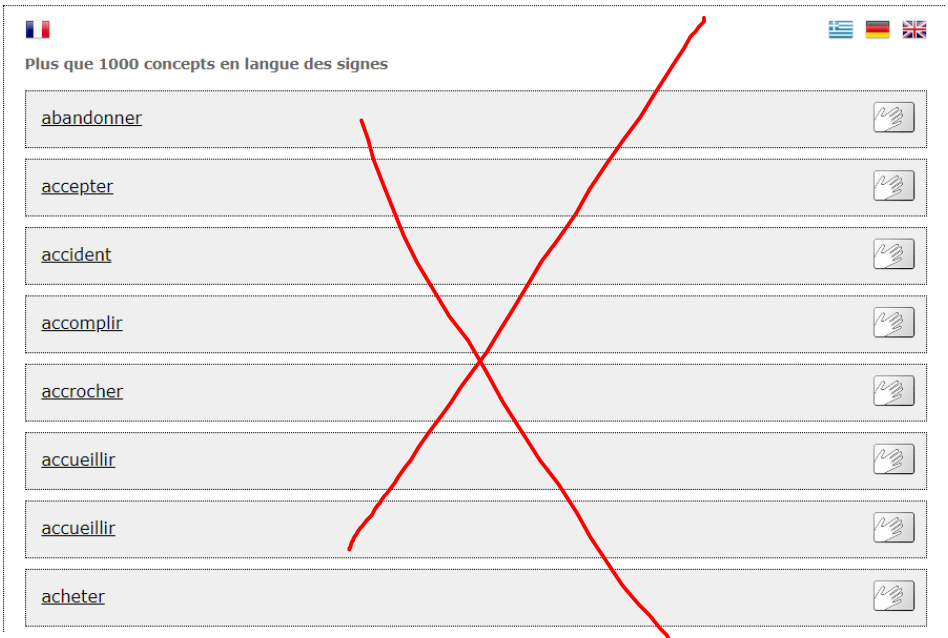

Figure 4.2: The web page interface of the DictaSign dataset

The dataset features video clips of sign performances for each word along with notations in HamNoSys notation as illustrated in Figure 4.3.

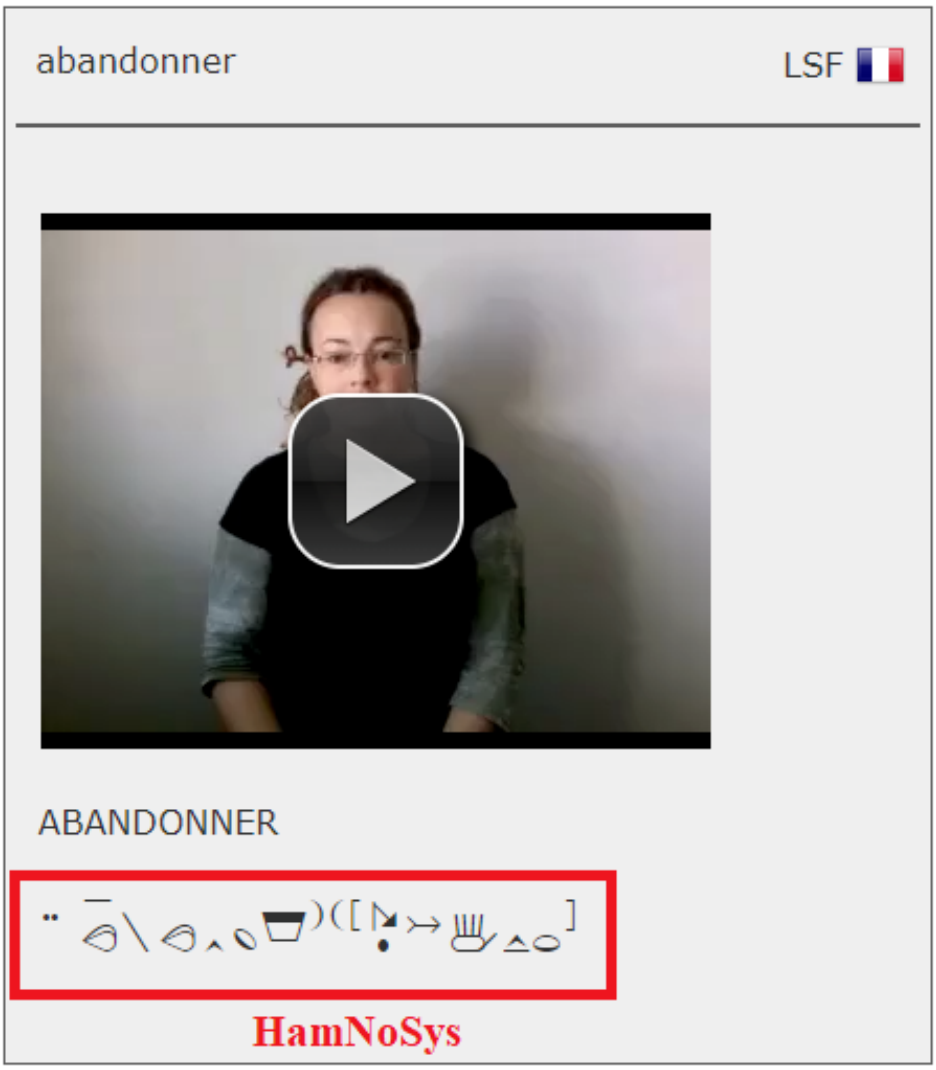

Figure 4.3: LSF representation in a video & in HamNoSys for the word ”Abondonner”

We used videos of native performers of Algerian Sign Language for words that are not included in the DictaSign dataset. For example, we sourced videos from the ”لغة الإشارة الجزائرية ” Facebook page [146], which regularly showcases Arabic words performed alongside their corresponding signs in Algerian Sign Language as showed in figure 4.4.

Figure 4.4: Algerian Sign Language gestures of "Eid al-Adha"

- **Phase Outputs:** LSF selected words list with its according HamNoSys Notation.

Phase 02: ALSL Translation

Dictionary Creation In this stage, the focus is on the first step of building the ALSL dictionary, which involves translating vocabulary entries (representing Arabic textual words in our case) based on HamNoSys system using eSign editor. We adopted two methods for this:

1. **Import Method:** Importing HamNoSys notations from the DictaSign dataset and linking them to the Arabic word that represents the Arabic translation of the corresponding French word associated with the imported HamNoSys notation. These entries are then added to the dictionary.

2. **Manual Translation Method:** This method relies on two fundamental pillars that complement each other:

   - Understanding the HamNoSys coding system thoroughly.
   - Translating the videos that are made by Algerian sign language performers.

The Figure 4.5 shows a HamNoSys Translation for the word ”أسرة ” in ALSL using the manual translation method, and the figure 4.6 shows the imported HamNoSys Translation for the word ”Abandonner” in LSF using the importing method.

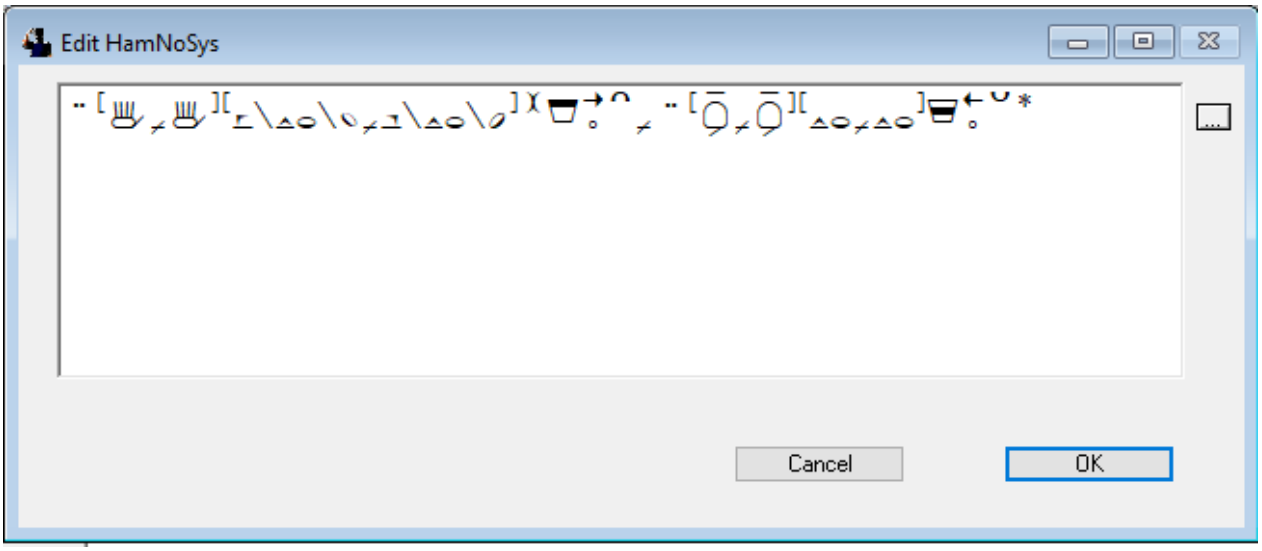

Figure 4.5: Manual HamNoSys Translation for the word ”أسرة ” in ALSL

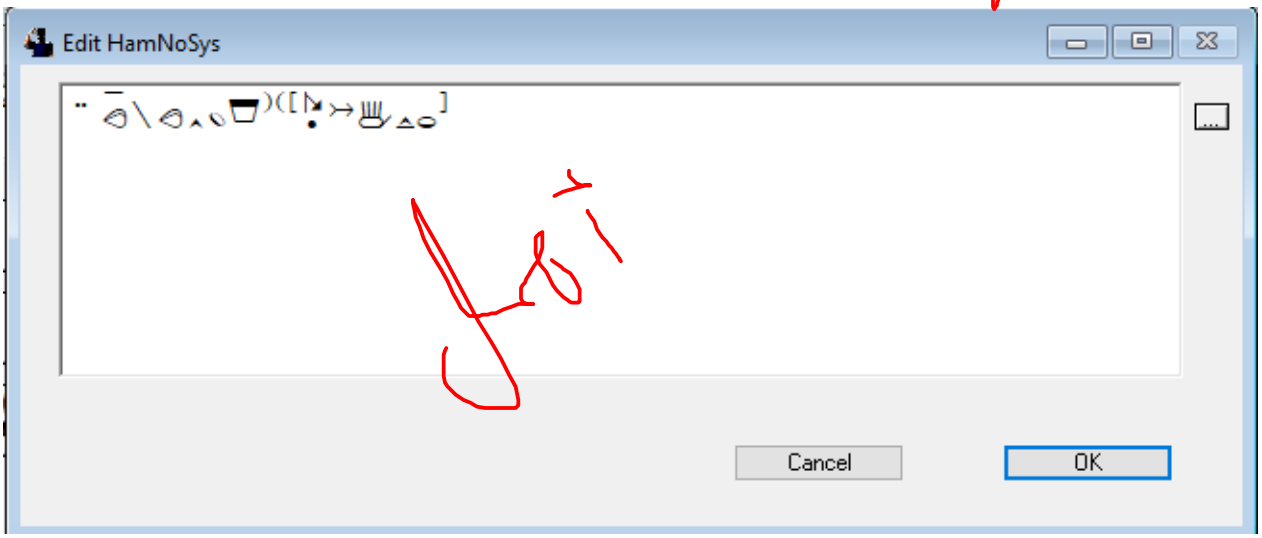

Figure 4.6: The imported HamNoSys Translation for the word ”Abandonner” in LSF

To construct the dictionary, we started with the use of the eSign editor application, depicted in Figure 4.7. Initially, the Arabic word ”أعلم ” (I know) is entered into the application to integrate it into the dictionary under the ”Spoken Language text” section. Figure 4.8 illustrates the entry of ”أعلم ” into the dictionary.

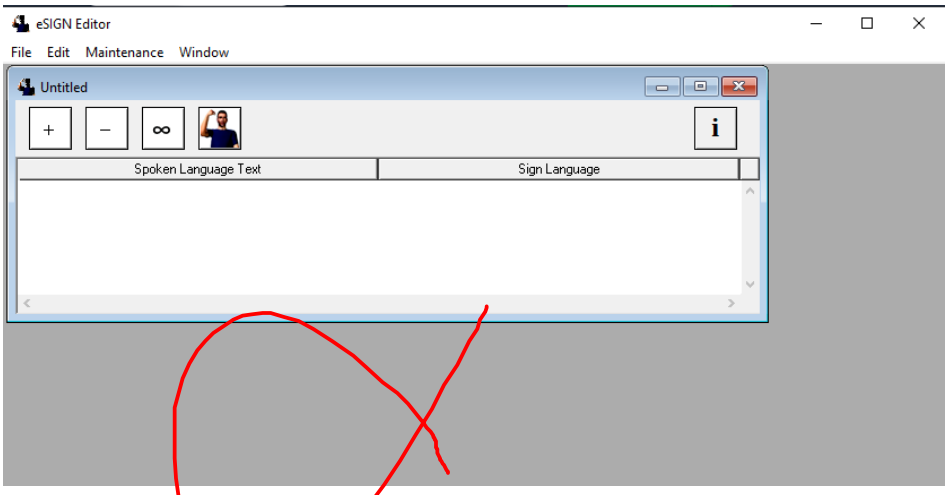

Figure 4.7: The eSign editor application interface

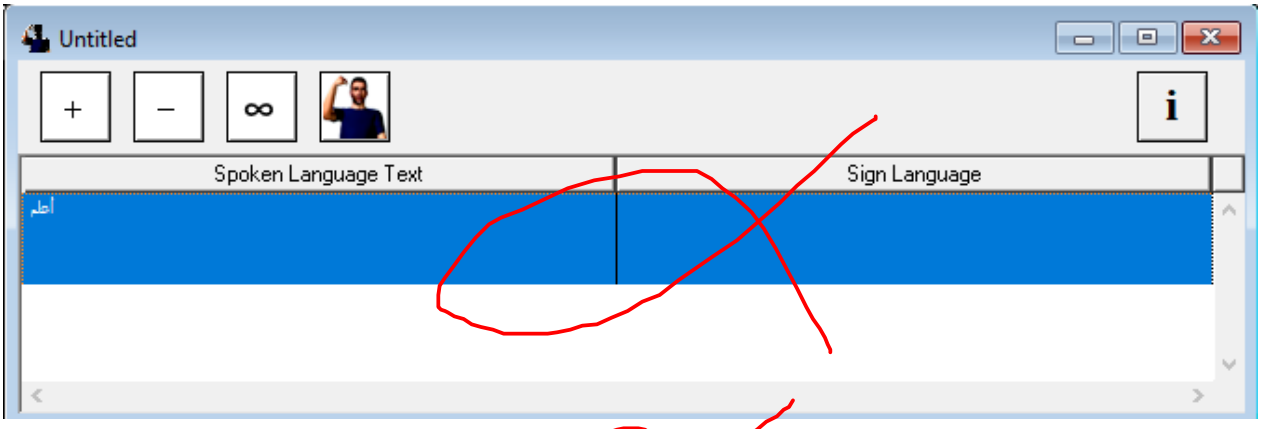

Figure 4.8: Entering the word ”أعلم ” (I know) into the ”Spoken Language text” section.

Following this, we add a new sign entry with the gloss ”أعلم” and the corresponding HamNoSys notation for Algerian Sign Language. The notation can be sourced either by importing directly from the DictaSign dataset or through the embedded HamNoSys input interface within the eSign editor application as shown in the figure 4.9.

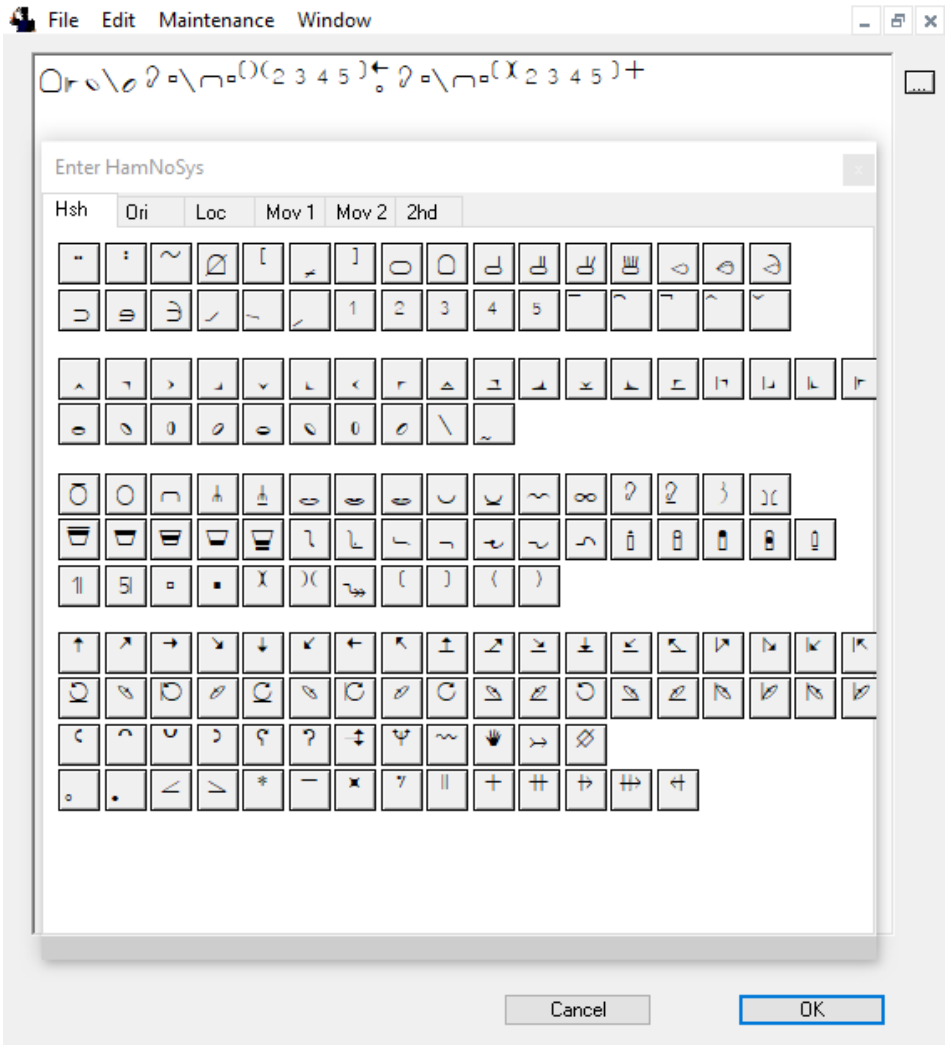

Figure 4.9: Manually inputting HamNoSys using the HamNoSys input panel

Additionally, the spatial movements of signs, lip movements and facial expressions are included to enrich the comprehensiveness of the entries as shown in figures 4.11, 4.12, and 4.13. Figure 4.14 illustrates the completed new sign entry after all parameters have been integrated.

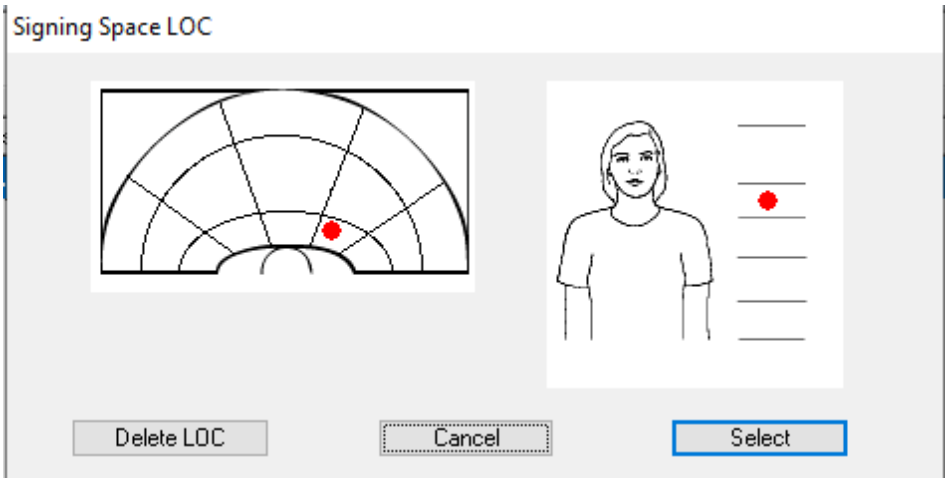

Figure 4.10: Signing space location interface

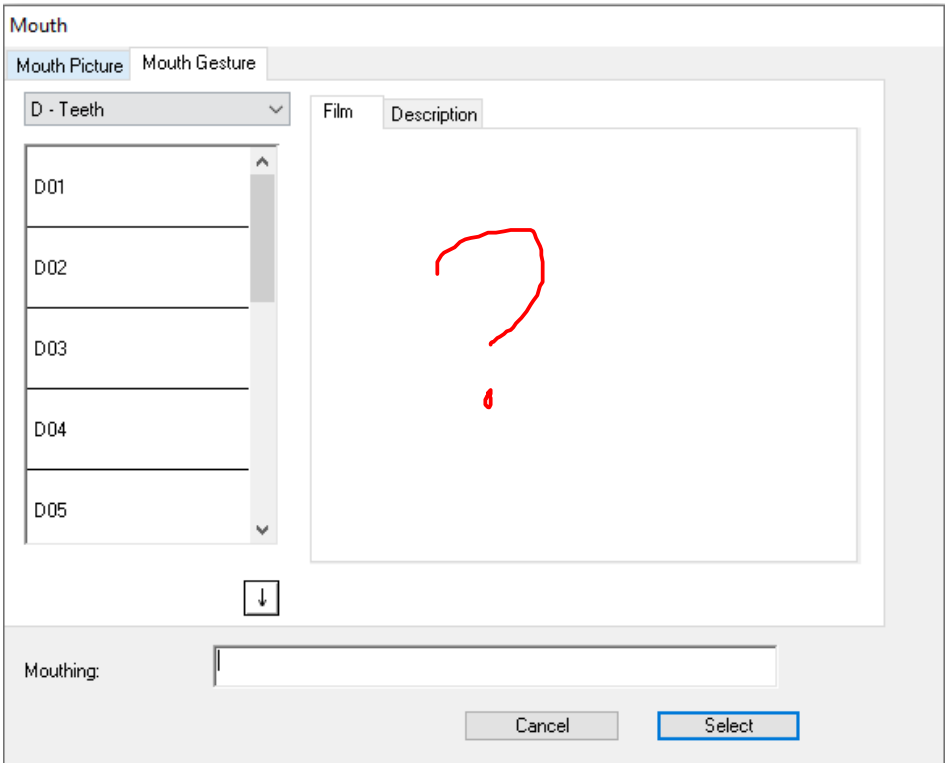

Figure 4.11: Mouth gestures interface

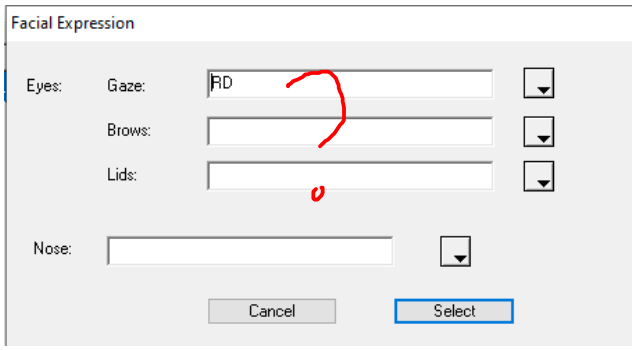

Figure 4.12: Facial expression interface

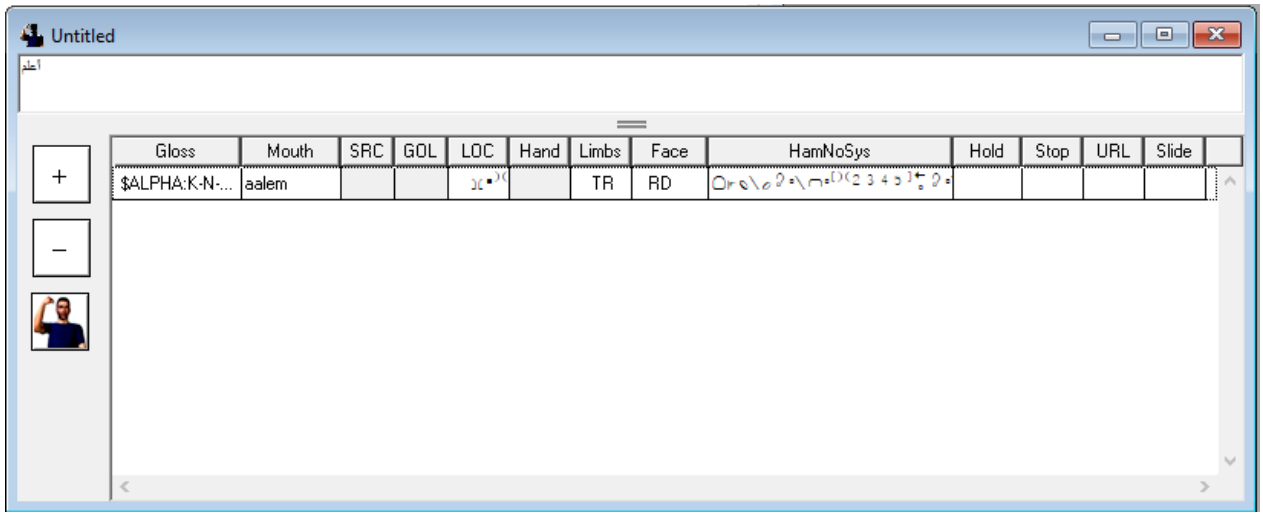

Figure 4.13: The completed new sign of the Arabic word "أعلم "

After completing these steps, we add the Arabic word into the dictionary through the eSign editor interface. Figure 4.15 displays the dictionary interface subsequent to saving the Arabic word with its translation in Algerian Sign Language.

Figure 4.14: The word "أعلم " paired with its translation in ASL in the dictionary.

Testing and Validation After incorporating the Arabic word into the dictionary, we assessed the avatar's performance in sign language motion using HamNoSys encoding associated with the word. This evaluation was conducted through the 'Sign' function in the eSign editor, presenting results via the CWASA Sigml player application. Subsequently, we validated translation accuracy by comparing video recordings of native Algerian Sign Language signers or by soliciting feedback

from experts in Algerian Sign Language. Figure 4.18 shows the ”Sign” button in the eSign editor interface while Figure 4.19 illustrates a comparative analysis between the sign gestures derived from translating the word ”أعلم ” using HamNoSys notation and the sign gestures performed by an Algerian Sign Language translation expert, as captured in a video recording.

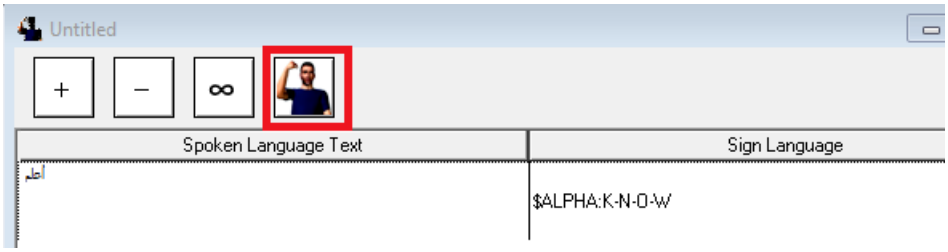

Figure 4.15: The ”Sign” button in the eSign editor

Figure 4.16: the translation of the word ”أعلم ” compared with a video of a signer.

Phase Outputs: A collection of Arabic vocabulary entries is annotated using the HamNoSys system.

ALSL Dictionary Construction Data Flow

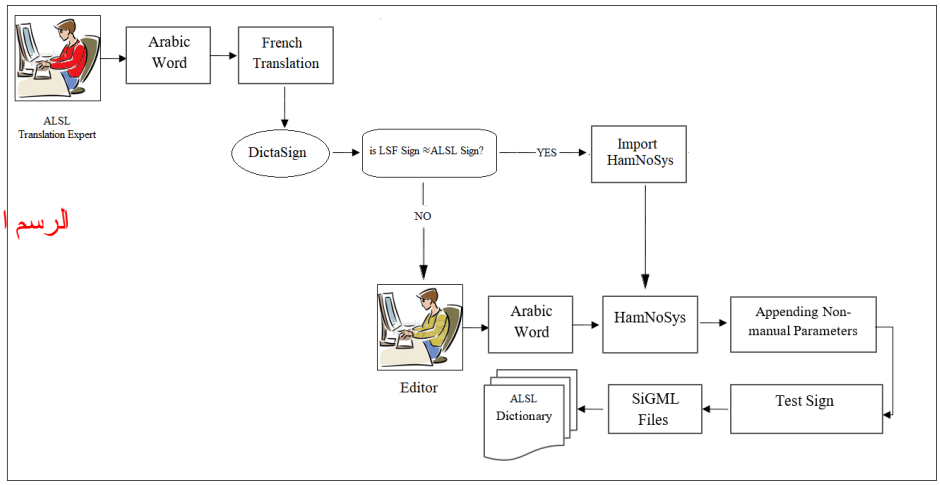

Figure 4.17: The dictionary construction

Detailed Steps:

- **Input Acquisition:**
  1. The French translation of the Arabic text is entered via keyboard.
  2. Algerian Sign Language translation expert researches the DictaSign dataset using French translations of the Arabic vocabulary intended to be added to the dictionary.
- **Translation Process:**
  1. Algerian Sign Language translation expert reviews the video clip depicting French sign language gestures corresponding to the French translation of the Arabic text.
  2. If the French sign language gestures associated with the French translation of the Arabic text align with Algerian Sign Language gestures, the HamNoSys notation for the French translation is imported as the notation for the Arabic vocabulary entry in the dictionary.
  3. If there is no alignment between the Algerian Sign Language (ALSL) gestures expressing the Arabic text and the French Sign Language (LSF), the HamNoSys notation for the Arabic vocabulary entry is manually inputted.
  4. Appending non-manual parameters such as facial expressions, head, and body movements.
  5. Test the sign accuracy.
  6. Exporting a SiGML file for each HamNoSys associated with an Arabic word.
- **Output Generation:** ALSL Dictionary, a collection of Arabic vocabulary entries is annotated using the HamNoSys system.

Phase 03: From HamNoSys to SIGML

After completing the first step of building the dictionary, the translation data in this phase is saved in the form of SiGML files based on HamNoSys, which is readable by the 3D avatar. When describing the process of translating HamNoSys annotations into SiGML files and the avatar's interaction with these files, it is important to address both aspects comprehensively:

Conversion Process:

- **SiGML Generation:** Once the HamNoSys notation is saved, it is converted and exported into SiGML, an XML-based format that represents these notations in a structured, digital form. This conversion is done by matching the HamNoSys symbols to corresponding SiGML tags. Each symbol in HamNoSys has a predefined SiGML tag that represents the same action or posture in a format that can be understood by 3D rendering software for sign language animation [108].

لخص

- **Playing Sign:** The SiGML file, now containing the structured representation of the sign language gestures, is used to animate these gestures through a virtual avatar. CWASA SiGML Player App reads the SiGML file and controls the avatar to perform the sign language gestures accurately [108].

صورة أصلية

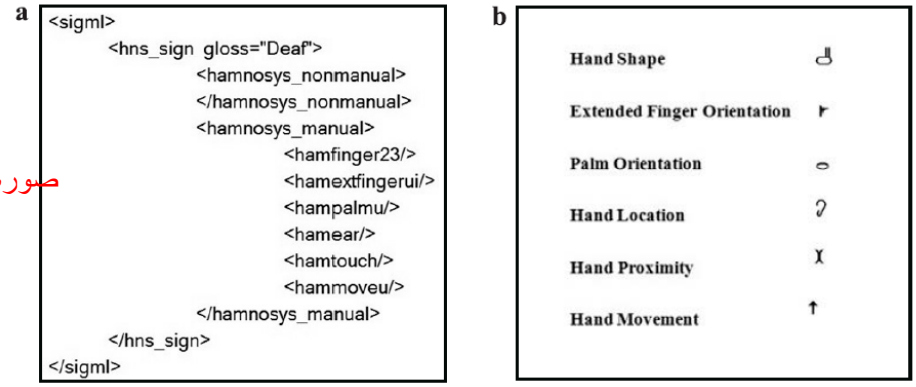

Figure 4.18: The SiGML and the HamNoSys Notation for Word "DEAF" [108]

Integration with Avatar: The CWASA (CWA Signing Avatars) system translates SiGML (Signing Gesture Markup Language) into animations using a protocol that operates over TCP sockets. This process is primarily driven by JavaScript and WebGL technologies, enabling the SiGML data to animate virtual avatars in real-time on web-based applications. When a client application connects to the SiGML Player, it sends SiGML data to the server via a TCP/IP connection on port 8052. The data can be sent either as a byte stream if an encoding is specified or as a UTF-8 stream

if no encoding is specified. Once the SiGML data is received, the server processes it to control the avatar's movements.

Additionally, CWASA can utilize XMLHttpRequest to dynamically fetch SiGML files or update avatar settings without reloading the web page. This enhances user interaction by allowing seamless data exchanges and immediate updates to the avatar's performance in response to user inputs, maintaining a responsive and interactive user experience essential for the effective rendering of sign language animations [147].

Phase Outputs With the completion of this stage, the Algerian Sign Language dictionary becomes readable by the avatar and stored in SiGML files.

Data Flow

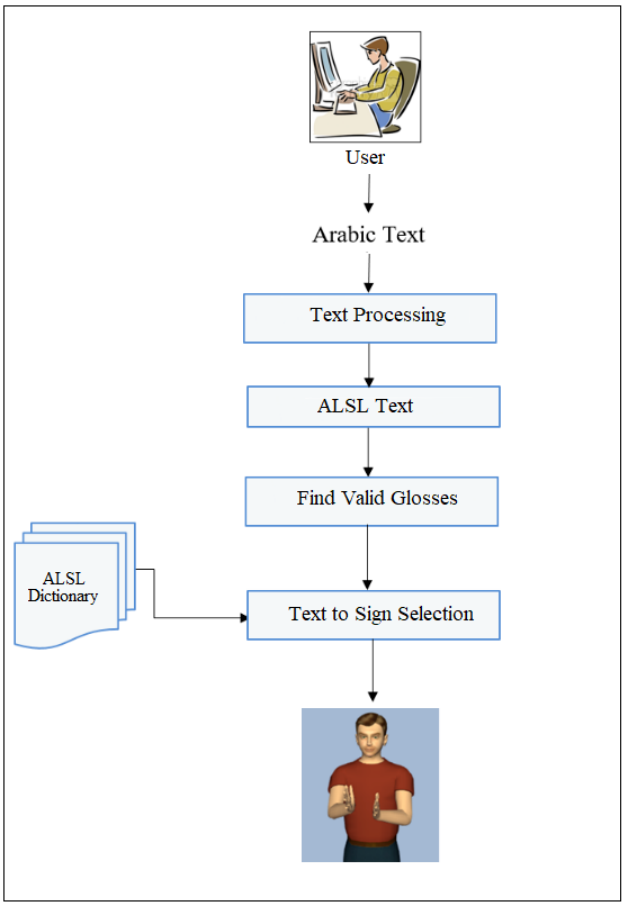

Figure 4.19: System architecture

Detailed Steps:

- **Input Acquisition:** Arabic text is entered via keyboard to the text-input area.
- **Translation Process:**

1. The input text is processed using regular expression techniques to get a valid ALSL text-input.
2. Validation algorithms used to validate the text-input glosses associated with ALSL Dictionary glosses.
3. Translation algorithms import the relevant translations from the ALSL dictionary based on the content of SiGML files to generate Algerian Sign Language (ALSL).

- **Output Generation:** The translated sign language is animated by the 3D avatar and displayed to the user.

Phase 04: Text Processing

To minimize issues for users, in this stage, we relied on developing specific algorithms to process Arabic textual inputs. This involved constructing linguistic and logical rules, which can be summarized as follows:

- **Regular Expression:** A regular expression (regex) algorithm is a powerful tool used for pattern matching and text manipulation tasks. It allows you to define a search pattern that can be employed to find, replace, or validate strings of text.

  In our case, we utilized a regular expression algorithm to process Arabic textual inputs. This algorithm searches for occurrences of specified patterns within the input text, such as special symbols, punctuation marks, or any non-alphanumeric characters, and removes these occurrences as necessary before using the cleaned texts as inputs for the application.

- **Sliding Window Algorithm:** To address the issue of accurately recognizing phrases such as "بعد غد " ("the day after tomorrow") in Arabic text inputs, which are separated by spaces and thus might be incorrectly treated as separate entities, we employ the sliding window algorithm. This algorithm systematically checks consecutive pairs of words by moving a window of two words across the text. Each pair (or window) is tested against entries in a glossary. If a pair matches a known gloss, it is treated as a single unit, improving the accuracy of understanding and translation by maintaining the correct grouping of words that form meaningful phrases. This method ensures that multi-word expressions are correctly identified and processed as whole units rather than fragmented parts.

Phase 05: UI Based on CWASA

User Interface Design The user interface (UI) for the Automated Arabic-Algerian Sign Language Translation System is designed to be intuitive and user-friendly, allowing users to input Arabic text and view the corresponding Algerian Sign Language (ALSL) signs performed by a 3D avatar. The UI is divided into four main sections: in figure

1. **Input Area:** This section allows users to input Arabic text either through a keyboard or via speech recognition.
2. **Avatar Display Area:** A designated space where the 3D avatar performs ALSL signs, ensuring the visual accuracy and comprehensibility of the translated sign language.
3. **Control Panel:** Provides options for users to start, pause, or reset the translation process, as well as adjust settings like avatar speed and display preferences.
4. **Drop-down Menus:** These are Arabic words categorized into drop-down lists based on linguistic categories, collectively representing dictionary entries eligible for translation. This facilitates user selection of suitable words for transmission to the text area for translation.

Development Tools The development of the UI leverages modern web technologies to ensure compatibility and responsiveness across various devices. The following tools and frameworks are used:

- **HTML5:** The backbone of the web page structure, providing the semantic elements necessary for a clean and accessible UI.
- **CSS3:** For styling the HTML elements and ensuring a visually appealing and responsive design.
- **JavaScript:** To handle the dynamic aspects of the UI, including user interactions and real-time updates.

CWASA Integration CWASA (Character and Word Alignment and Signing Avatar) is integrated into the UI to render the 3D avatar's sign language animations. The integration process involves the following steps:

- **Configuration:**

  - **Initialization:** Ensure CWASA is properly initialized when the web page loads. This involves setting up the CWASA library and ensuring all necessary assets (e.g., 3D models, animations) are loaded.
  - **Configuration Files:** Create and configure the CWASA configuration files, specifying parameters such as the path to the avatar models, animation settings, and initialization scripts. Relevant files include:
    * `cwa/cwacfg.json`
    * `cwa/clientcfg.json`

- **Avatar Initialization:**
  - **HTML Integration:** Add the necessary HTML elements to integrate the CWASA avatar into the web page.
  - **JavaScript Initialization:** Use JavaScript to initialize the CWASA avatar and bind it to the UI elements.

- **User Interaction Handling:**
  - **Event Listeners:** Add event listeners to handle user interactions, such as submitting text for translation and controlling the avatar's actions.

Functionality The key functionalities implemented in the UI include:

- **Text Input:** Allowing users to enter Arabic text manually.
- **Translation Trigger:** A button to trigger the translation process once the text is entered.
- **Avatar Display:** The 3D avatar dynamically performs the translated ALSL signs in the display area.
- **Control Options:** Options to control the playback of the avatar's animations, such as starting and resetting the animation.

Figure 4.22 shows the application user interface built on CWASA technology.

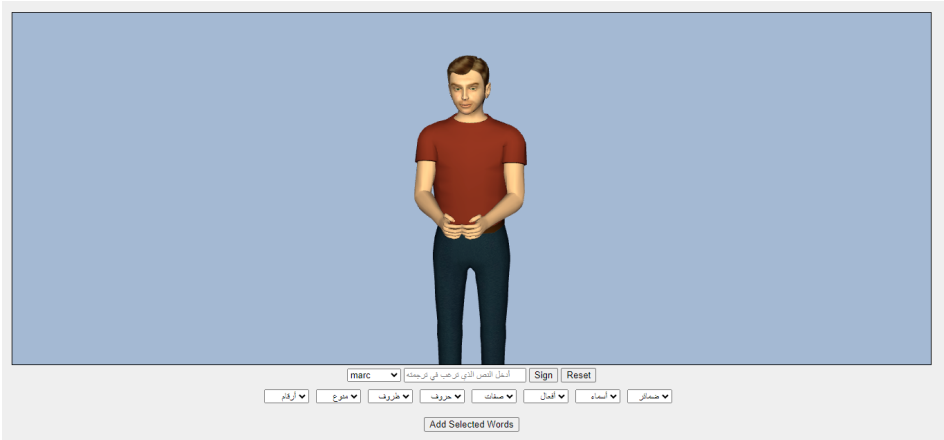

Figure 4.20: Arabic Algerian 3D Avatar Translator System UI Based on CWASA

CWASA Installation Settings The CWASA installation requires specific settings to be configured. These settings ensure that the CWASA library is properly set up and that all necessary components are available for the avatar to function correctly.

- **Library Installation:** Ensure that the CWASA library is correctly installed in the project directory.
- **Asset Loading:** Configure the paths to the avatar models and animation files in the configuration files.
- **Initialization Scripts:** Ensure that the initialization scripts are correctly set up to load the CWASA library and initialize the avatar.

Phase 06: Test the System

Testing Methodology The ALSL Translation System deployed on the Vercel platform so users could test it and see all its different pieces. Work was conducted in collaboration with specialists from the School for Deaf Children in Beni Slimane, who deal with Algerian Sign Language translations, and the association of Bouira province, which specializes in sign language translation. Problems in synchronization were detected at the integration test level for interaction between the translation engine and 3D avatar animation. Still, these were resolved by optimizing the communication protocols among the various modules. It is confirmed that the flow from Arabic text input to 3D avatar animation is smooth with the correct sign displays. The results from the user acceptance testing have proved that the interface is very natural and easy to use. Users left good comments on clarity and the actuality of translations. The test performance results were system responses in a

very short time, with acceptable translation time. Very slight changes have been made to the layout of the control panel based on user feedback to make this system more accessible and, by extension, user-friendly.

Results

All component-level tests passed for their corresponding unit tests. These confirmed that all the functions and modules work as expected individually. However, some early problems of Arabic text handling were resolved simply by refining input-handling code. Synchronization issues between the translation engine and 3D avatar animation were discovered in the integration tests. This was corrected through the optimization of communication protocols between modules. Validation was successful in the text input from Arabic to avatar animation, ensuring smooth transitions between signs and their appropriate representation. The acceptance tests showed an intuitive interface, well accepted by the users, with some positive comments about the clarity and accuracy of translations. Several performance tests had shown that the system response was fast, with acceptable translation times, and some minor changes on the layout of the control panel had been fed back by the user to enhance the ease of access and usability.

Phase 07: Collaboration with ALSL Experts

To ensure the accuracy and cultural relevance of the translation, we collaborated closely with experts in Algerian Sign Language (ALSL). This phase involved rigorous testing of the translation system using a comprehensive glossary of ALSL vocabulary. The experts' feedback was crucial for refining the system and improving its accuracy. The process involved the following steps:

- **Word Selection:** Experts selected specific words from a drop-down menu within the system.
- **Input and Observation:** Each selected word was input into a designated field, and the corresponding translation was rendered through the avatar's ALSL movements.
- **Accuracy Assessment:** The experts assessed the accuracy of each translation based on the avatar's performance.
- **Systematic Verification:** This process was repeated for every word category in the glossary, ensuring a thorough verification of the translation tool's effectiveness.

Feedback and Results:

The following table (Table 4.1) displays the results of the system tests based on the feedback from experts in Algerian Sign Language translation.

التصنيف	اختبار مترجم لغة الإشارة	نسبة الترجمات الصحيحة	عدد الترجمات الصحيحة	الترجمات غير الدقيقة
الضمائر	✓	100%	3/3	-
الأسماء	✓	96.48%	219/227	أستاذ ألم، أم، إشارة، اقتصاد، امرأة، يونيو، مكتبة
الأفعال	✓	97.44%	76/78	لا يعلم، يأخذ
الصفات	✓	90.91%	40/44	آخر، صعب، مريض
الحروف	✓	91.67%	11/12	كيف
الظروف	✓	82.35%	14/17	أمام، تحت، صباح
متنوع	✓	73.33%	11/15	فقط، مرة أخرى، مرحبا، من فضلك
الأرقام	✓	90.48%	19/21	90 40,

Table 4.1: The results of the system tests

The Figures 4.21, 4.22, and 4.23 show the percentage charts of: correct translations for each category, Correct Translations by Category, and Inaccurate Translations by Category, respectively.

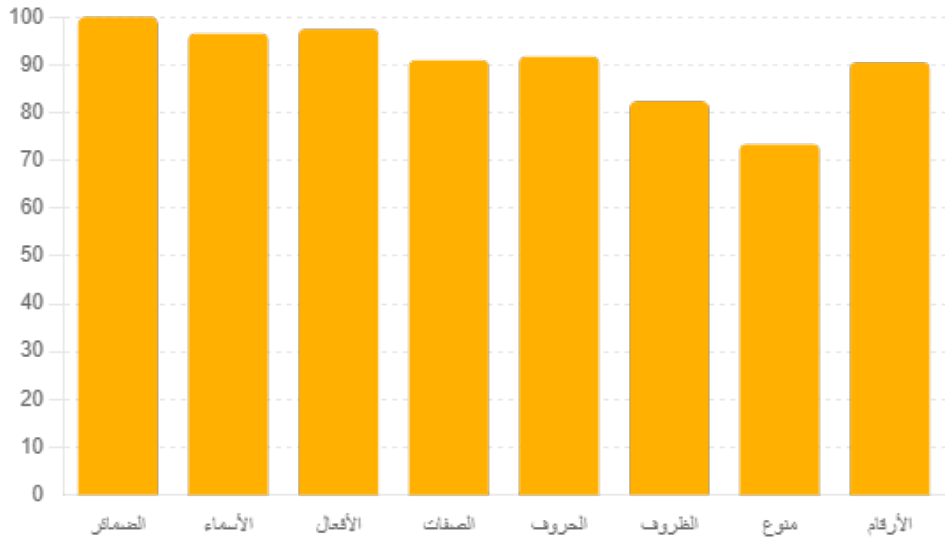

Figure 4.21: Percentage of Correct Translations by Category

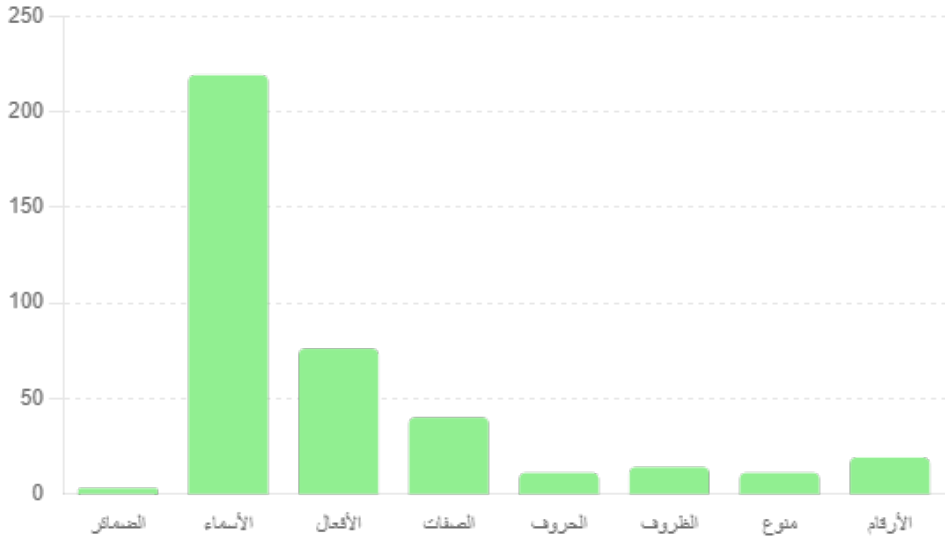

Figure 4.22: Number of Correct Translations by Category

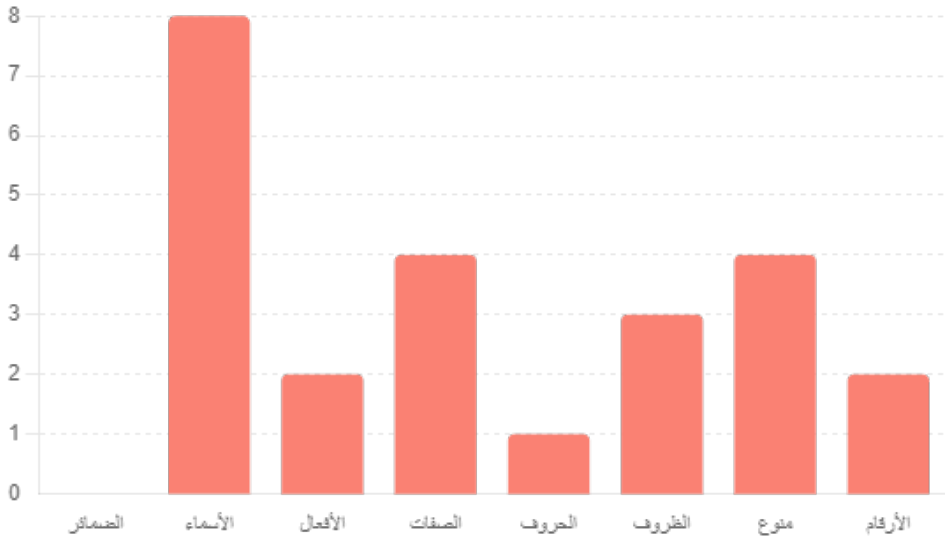

Figure 4.23: Number of Inaccurate Translations by Category

ALSL Translation System Accuracy and Enhancing

We measured the accuracy of the entire translation system based on the accuracy of category translations using a systematic approach that aggregates and analyzes performance data for each category to obtain a comprehensive view. Based on the results of this measurement, we periodically improve the accuracy of our translation system. The process can be explained as follows:

1. **Calculate Overall Accuracy:**

- **Weight Accuracy Based on Sample Size:** To calculate the overall system accuracy, we weighted the accuracy of each category based on the number of samples, as shown in the following equation:

$$\text{Overall Accuracy} = \frac{\sum(\text{Number of samples per category} \times \text{Category Accuracy})}{\sum(\text{Number of samples per category})}$$

The table 4.2 shows correct translations percentage by Category

Category	Number of Samples	Percentage of Correct Translations
Pronouns	3	100%
Nouns	227	96.48%
Verbs	78	97.44%
Adjectives	44	90.91%
Prepositions	12	91.67%
Adverbs	17	82.35%
Miscellaneous	15	73.33%
Numbers	21	90.48%

Table 4.2: Correct Translations by Category

After applying the equation to the table data, we found that the value of Overall Accuracy of our ALSL translation system = 94.2%

Figure 4.24 shows the chart of Overall Accuracy of our ALSL translation system.

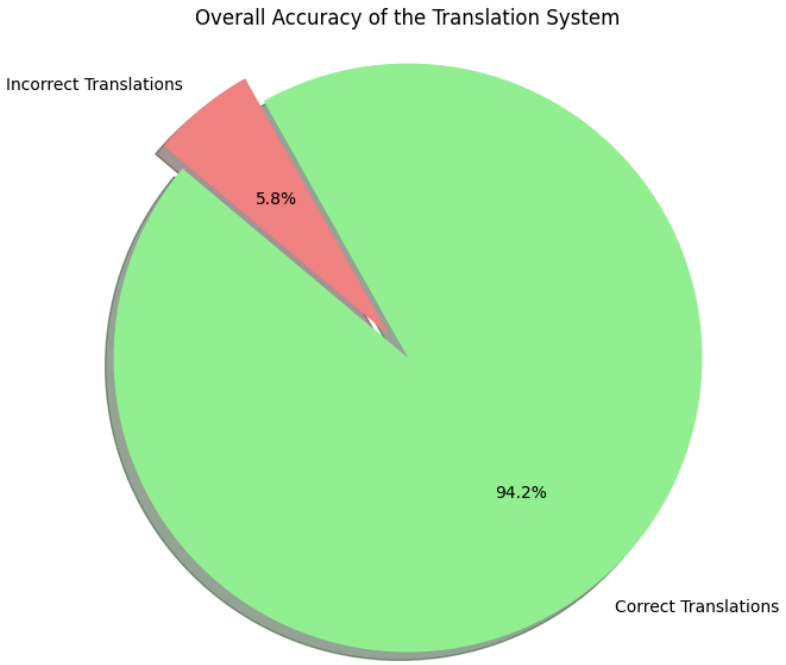

Figure 4.24: ALSL Translation System Overall Accuracy

2. **Analyze Results:**

    - **Identify Strong and Weak Categories:** Analyze the accuracy for each category to identify which categories need improvement.
    - **Feedback:** Use the results to provide feedback to the translation team or to improve the tools and techniques used.

3. **Continuous Improvement:**

    - **Repeat Testing:** Conduct periodic tests to measure improvements in the accuracy of the categories.
    - **Update System:** Based on the results, update the translation system and training to improve overall performance.

4.3.2 Gesture Animation via Motion Capture (GAMC)

Phase 01: ALSL Data Collection

The first step is to collect data, which is in this case a set of videos of people performing Algerian sign language gestures, each video must represent a word or a sentence. The videos must follow some criteria.

1. **Selection Criteria:**
    - The videos must be high resolution, enough for computer vision to detect body parts.
    - The person performing gestures must be inside the camera frame in every frame; otherwise, the detection will be unresponsive.
    - The upper body of the person is better be fully exposed to the camera; otherwise, the detection will show an inclined person.
    - The frame rate of the video must be between 24 fps and 30 fps; any less frame rate will cause the result to look disconnected, and higher frame rate will lead to unresponsive movement.
2. **Data Sources:** After following the criteria, we kept several data sources:
    - A Facebook page for Algerian sign language "لغة الإشارة الجزائرية " [146].
    - Some of the videos recorded by Deaf people school in Beni Slimane, Medea.
    - Some videos remade by us.

The total of videos collected was 61 videos; this number is enough to move to the next step.

Phase 02: Create a 3D Avatar

This section outlines the criteria, challenges, and solutions encountered during the process of creating a 3D avatar.

- **Criteria:** To ensure the 3D avatar can accurately performing ALSL gestures, the following criteria were established:
    - **Detailed Upper Body:** The avatar's upper body, particularly the hands and fingers, must be highly detailed to ensure gestures are clearly visible.
    - **Comprehensive Armature:** The avatar must have a complete armature, including bones for the upper body and fingers, to facilitate accurate gesture performance.

- **Realistic Textures:** Emphasis on realistic skin textures for the hands and face to enhance visual fidelity.
- **Early Movement Testing:** Early and frequent testing of the avatar's movements to ensure the rigging is functional and capable of performing complex gestures.

- **Challenges:** The development of the 3D avatar presented several challenges:
  - **Blender Learning Curve:** Blender's learning curve is steep, especially given the limited time available.
  - **Creating Lifelike Avatars:** Creating a realistic human avatar requires expertise.
  - **Quality of Available Models:** Many available models online are either unrealistic, irrelevant, or require payment.

- **Solutions:** To overcome these challenges, the following solutions were implemented:
  - **Consulting a 3D Modeler:** We reached out to a professional 3D modeler for advice and assistance, which was invaluable in refining our model.
  - **Watching Tutorials:** Extensive tutorials were watched to better understand Blender and improve our modeling skills.
  - **Utilizing Plugins:** We discovered and installed plugins like MBLab, which significantly facilitated the creation of a realistic humanoid model.

We began by using the MBLab Blender plugin to generate a basic humanoid model. This plugin allowed for significant customization to meet our specific requirements.

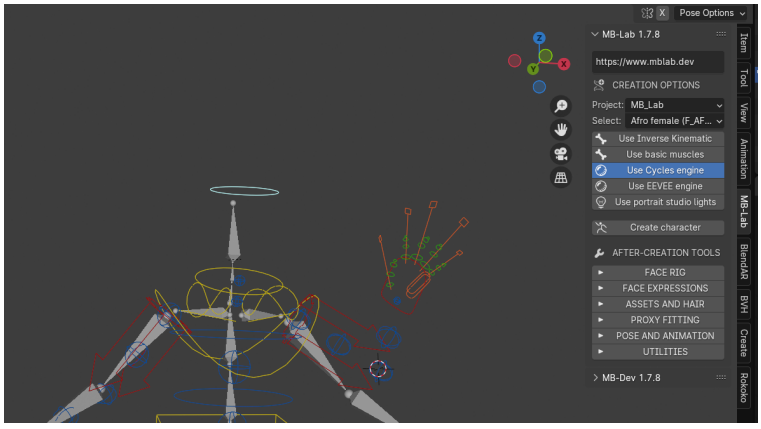

Figure 4.25: Blender with MBLab plugin tab

The next step was to focus on rigging. We created a comprehensive armature, ensuring that the avatar could perform detailed ALSL gestures. Frequent testing of the avatar's movements was conducted to identify and resolve any issues early in the process.

With the help of a professional 3D modeler, we refined the avatar further, resulting in a realistic model that met all our criteria.

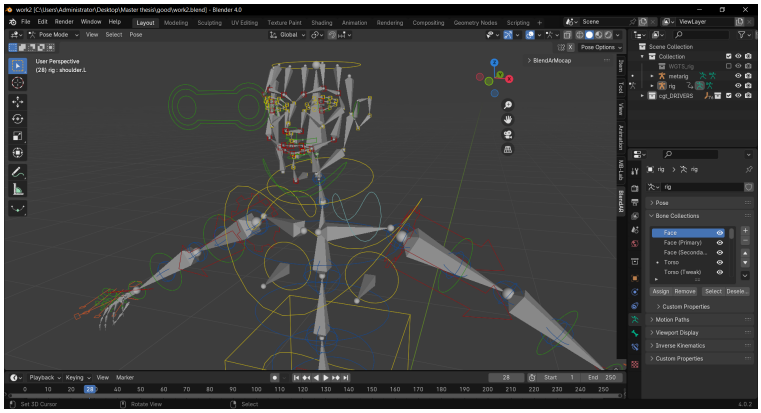

Figure 4.26: Rig of the 3D avatar

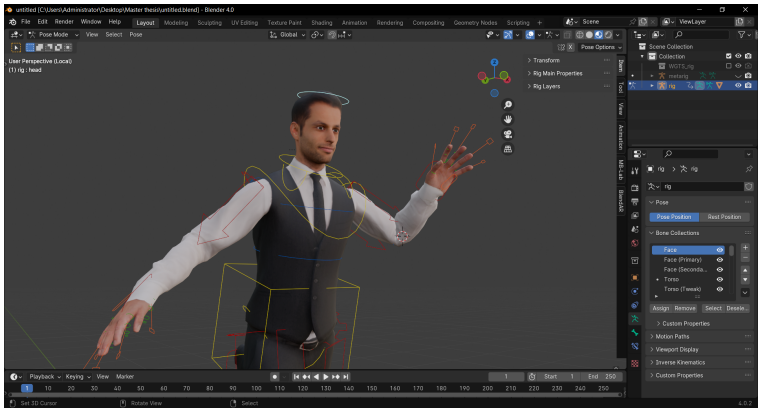

Figure 4.27: Final 3D avatar character

Finally, basic animations were performed to identify and fix any remaining bugs, ensuring that the avatar could perform all required gestures smoothly and accurately.

By following these steps, we developed a realistic and functional 3D avatar that serves as the backbone of our ALSL Translation System.

Phase 03: Creating Motion Capture Files

Motion capture (MoCap) is a crucial part of the ALSL Translation System. We explored various computer vision libraries to develop our own script that can capture motion from a video and convert it to a usable file.

- **Tools and Software Considerations:** Numerous computer vision libraries, such as OpenCV, HOLO and OpenPose, offer impressive accuracy. These tools were considered for developing our script. We also reviewed publicly shared scripts to understand the state-of-the-art approaches.
- **Challenges:** Despite the variety of available tools, predicting a 3D pose from 2D footage proved to be a significant challenge, even for well-performing models like MediaPipe. We tested several tools, one of which was FreeMocap, a MoCap tool that uses different cameras at various angles to accurately predict gestures. However, FreeMocap requires a calibration board and multiple cameras, which can't be provided.

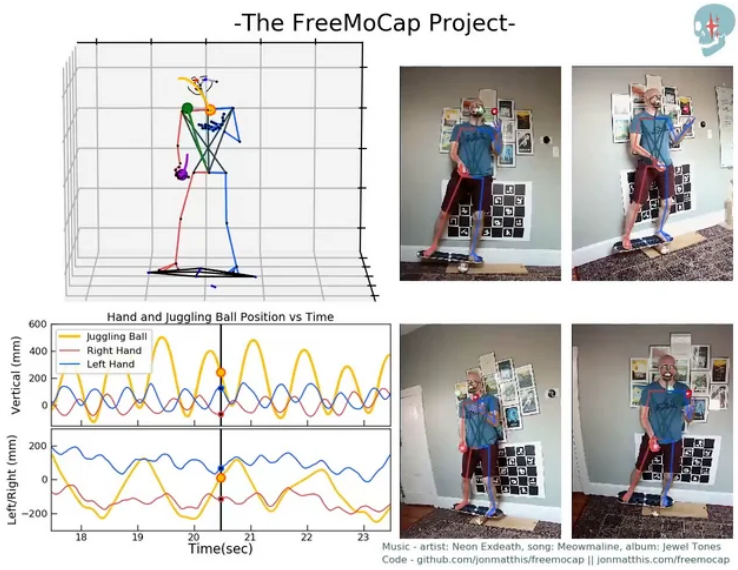

Figure 4.28: FreeMocap setup with multiple cameras and calibration board

- **Development and Testing:** We developed our script using OpenCV, achieving performance comparable to other tools using the same library. Additionally, we discovered BlendAR-Mocap, a Blender add-on that integrates MoCap functionalities. Although its precision was similar to other OpenCV-based tools.

  We explored other available tools, among them we found Dollars mocap which is an AI-powered motion capture solution designed to facilitate the creation of realistic animations

for characters in various digital applications. It can be used by animators, game developers, filmmakers, and VR developers to streamline the animation process, reduce production costs, and create high-quality, lifelike character movements without the need for expensive and complex traditional motion capture setups.

- **Final Tool Selection:** After evaluating several tools, including FreeMocap, Dollars Mocap and Rokoko studio, we found no tool that copies the input video perfectly. We decided to use Dollars Mocap MONO due to its superior output quality.

Figure 4.29: Dollars Mocap MONO trial version

- **File Format:** We chose the BioVision Hierarchy (BVH) format for our motion capture files. The BVH format is readable and allows for manual editing of small details, making it suitable for our needs.

  By carefully considering available tools, addressing challenges, and selecting the most appropriate software, we were able to create accurate motion capture files for around 20 of the videos we have, that serve as a foundation for animating our 3D avatar.

- **BVH File structure** A BVH file contains ASCII text, the first part of which provides the specifications for the initial pose of a human skeleton, and the rest a time-framed sequence of different specifications for subsequent poses.[148]

  Headed by the keyword HIERARCHY, the first part of a BVH file identifies the Hips joint as the ROOT, which is to say it has no parent-joint. This is the starting point of a nested-structure of parent-joints and child-joints.

أصلية

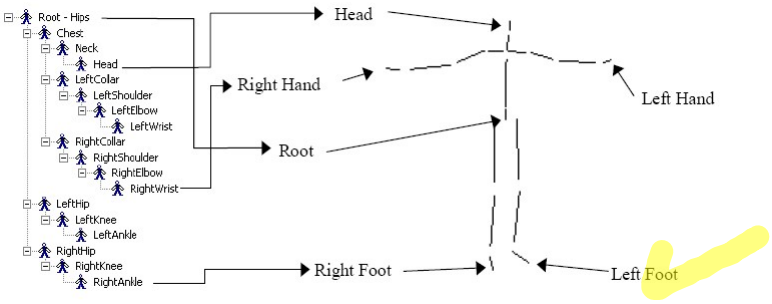

Figure 4.30: Bone hierarchy structure from a BVH file [149]

The ROOT section specifies the location of the Hips joint in a three-dimensional space. Below the ROOT section in the structure are JOINT sections, each containing information that specifies the location of the skeletal joint relative to its parent-joint. The relative location specifications for a parent-joint and its child-joint makes it possible to work out the length of the ”bone” between the two joints. Where a joint has no child-joint, it is linked to an End Site section.

The ROOT section and each of the JOINT sections also specifies the CHANNELS for processing the time-framed sequence of translation and/or rotational co-ordinates provided in the second part of the BVH file.

```
HIERARCHY
ROOT Hips
        {
                OFFSET 0.00 0.00 0.00
                CHANNELS 6 Xposition Yposition Zposition Zrotation Xrotation Yrotation
                JOINT Chest
                {
                        OFFSET 0.000000 6.275751 0.000000
                        CHANNELS 3 Zrotation Xrotation Yrotation
                        JOINT Neck
                        {
                                OFFSET 0.000000 14.296947 0.000000
                                CHANNELS 3 Zrotation Xrotation Yrotation
                                JOINT Head
                                {
                                        OFFSET 0.000000 2.637461 0.000000
                                        CHANNELS 3 Zrotation Xrotation Yrotation
                                        End Site
                                        {
                                                OFFSET 0.000000 4.499004 0.000000
                                        }
                                }
                        }
```

Figure 4.31: First part of a BVH file: Hierarchy [149]

The second part of a BVH file starts with the keyword MOTION, followed by information specifying, first, the number of FRAMES; second, the sampling rate per second after the keyword FRAME TIME; and third, a number of lines corresponding to the number of FRAMES, each line providing the tranlation and/or rotation co-ordinates to be processed according to the CHANNELS specifications in the first part of the file.

```
MOTION
Frames: 2
Frame Time: 0.04166667
-9.533684       4.447926        -0.566564       -7.757381       -1.735414       89.207932       9.763572
                6.289016        -1.825344       -6.106647       3.973667        -3.706973       -6.474916
                -14.391472      -3.461282       -16.504230      3.973544        -3.805107       22.204674
                2.533497        -28.283911      -6.862538       6.191492        4.448771        -16.292816
                2.951538        -3.418231       7.634442        11.325822       5.149696        -23.069189
                -18.352753      15.051558       -7.514462       8.397663        2.953842        -7.213992
                2.494318        -1.543435       2.970936        -25.086460      -4.195537       -1.752307
                7.093068        -1.507532       -2.633332       3.858087        0.256802        7.892136
                12.803010       -28.692566      2.151862        -9.164188       8.006427        -5.641034
                -12.596124      4.366460
-8.489557       4.285263        -0.621559       -8.244940       -1.784412       98.041962       8.849357
                5.557910        -1.926571       -5.487280       4.119726        -4.714622       -5.790586
                -15.218462      -3.167648       -15.823254      3.871795        -4.378940       22.399654
                2.244878        -29.421873      -6.918557       6.131992        4.521327        -18.013180
                3.059388        -3.768287       8.079588        10.124812       5.888083        -22.417845
                -15.736264      18.827469       -8.070700       9.689109        2.417364        -7.600582
                2.505005        -1.625679       2.430162        -27.579708      -3.852241       -1.830524
                12.520144       -1.653632       -2.688550       4.545600        0.296320        8.031574
                13.837914       -28.922058      2.077955        -9.176716       7.166249        -5.170825
```

Figure 4.32: Second part of a BVH file: Motion [149]

Phase 04: Enrich the 3D Avatar

After obtaining accurate motion capture data, the next step is to make the 3D avatar understand these files and perform the gestures contained within the BVH files. This phase involves several stages, addressing various challenges and implementing solutions to ensure smooth integration.

- **Stages:**
  - **Importing BVH Files:** The files can be imported via File -> Import -> BVH, however, this method requires a metarig identical to the one used during the motion capture process and we need to constantly replace the 3D avatar's metarig, to avoid this we used a community-made BVH add-on that automatically creates a new metarig to perform the BVH motion file.

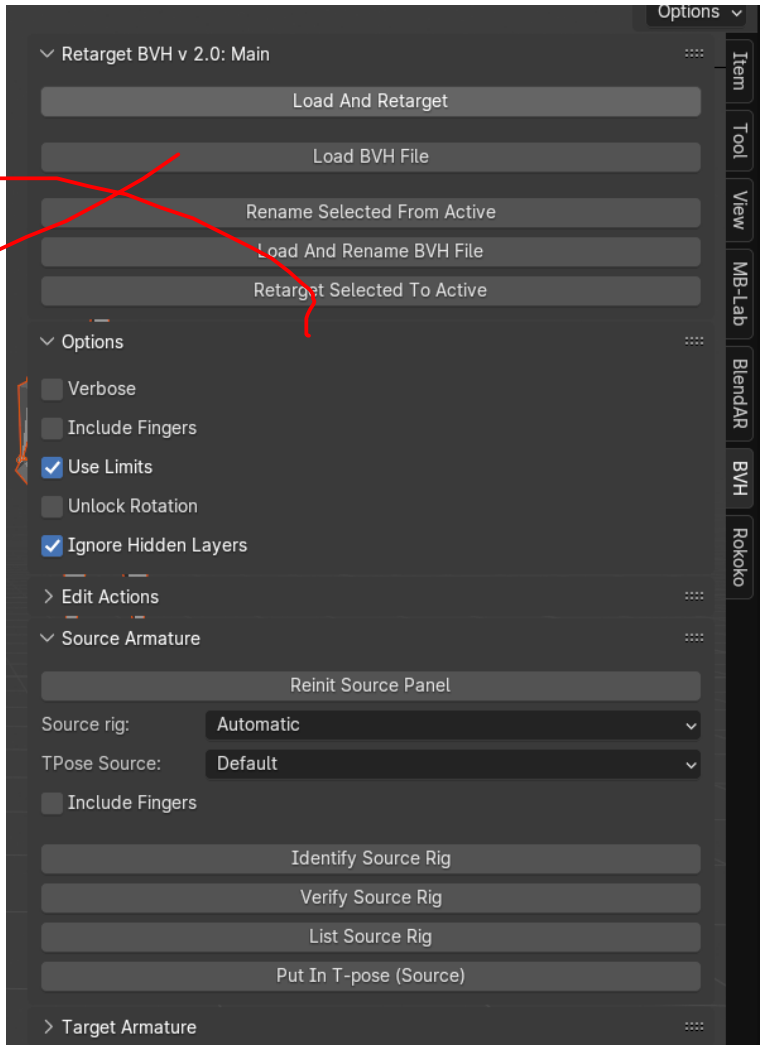

Figure 4.33: Community-made BVH add-on in Blender

– **Transferring Motion Data:** We employed the Rokoko add-on to transfer motion data from the newly created metarig to the original 3D avatar's metarig. The Add-on tries to match every two bones from the source and destination metarigs based on the rotation, location, name and number of bones connected to it, it can be manually edited as well.

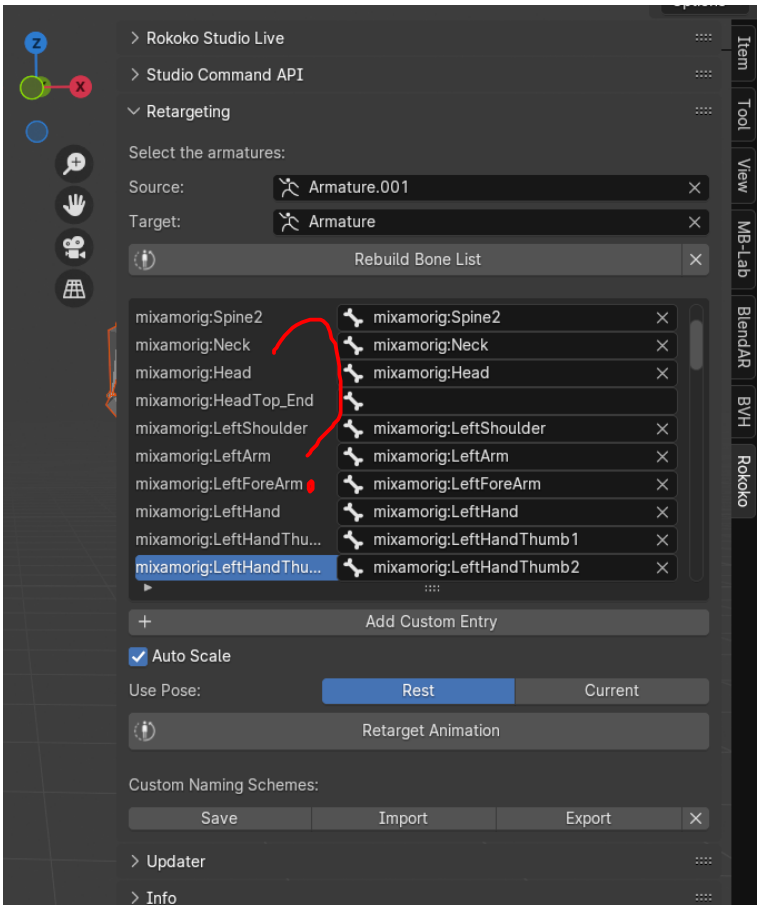

Figure 4.34: Rokoko add-on in Blender

– **Fine-Tuning Animations:** Additional adjustments were made, such as fixing starting and ending frames, and adjusting the speed of animations to ensure they appeared natural. Below is an image showing the process of transferring and adjusting the motion data.

Figure 4.35: Transferring and adjusting motion data

- **Initial Set of Animations:** By the end of this phase, our 3D avatar was capable of performing animations for 20 different sentences as a starting point.

- **Challenges:**
  - **Metarig Compatibility:** The default method of importing BVH files requires a metarig that is 100% identical to the one used in the motion capture tool, which is impractical.
  - **Motion Data Transfer:** Ensuring accurate transfer of motion data between different metarigs without losing fidelity or introducing errors.
  - **Animation Tweaks:** Adjusting animations to ensure they start and end correctly, and tweaking the speed for realism.

- **Solutions:**
  - **Community-Made BVH Add-On:** This add-on automatically generates a new metarig that matches the BVH motion file, eliminating the need to replace the avatar's metarig each time.
  - **Rokoko Add-On:** This tool was used to copy the motion data from the newly created metarig to the original avatar's metarig, ensuring compatibility and accuracy.
  - **Manual Adjustments:** Fixing starting and ending frames and adjusting the speed of animations were done manually to fine-tune the performance of the 3D avatar.

Through these stages and by overcoming the outlined challenges with effective solutions, we successfully enriched our 3D avatar, enabling it to perform a variety of ALSL gestures accurately.

This capability forms the basis for further expansion and refinement of the ALSL Translation System.

Phase 05: Set Up an Environment

After preparing our 3D avatar, the next step involves setting up the environment within the Unity game engine. The main goals of this phase are:

- Prepare the environment and lighting that will surround the avatar.
- Set up a flexible camera with different angles and zooming levels for clear visibility of the avatar.
- Import the created avatar with the proper settings.
- Create a user interface.

Exporting the Avatar: We exported our character as an FBX file, which was chosen because it preserves the rigging and animation data while ensuring compatibility with the Unity engine.

Importing into Unity: We imported the FBX file into an empty Unity project.

Setting Up the Environment The environment was designed to be flexible and adaptable for different scenarios. We focused on creating a classroom setting as our initial environment. We encountered an issue where all the textures appeared pink. This was due to missing or incompatible shaders. After consulting with a Unity community server on Discord, we resolved the issue by updating the materials and shaders to ensure proper rendering.After that we added appropriate lighting to enhance the visibility of the avatar. Adjustments were made to ensure that the lighting was both realistic and suitable for the gestures performed by the avatar.

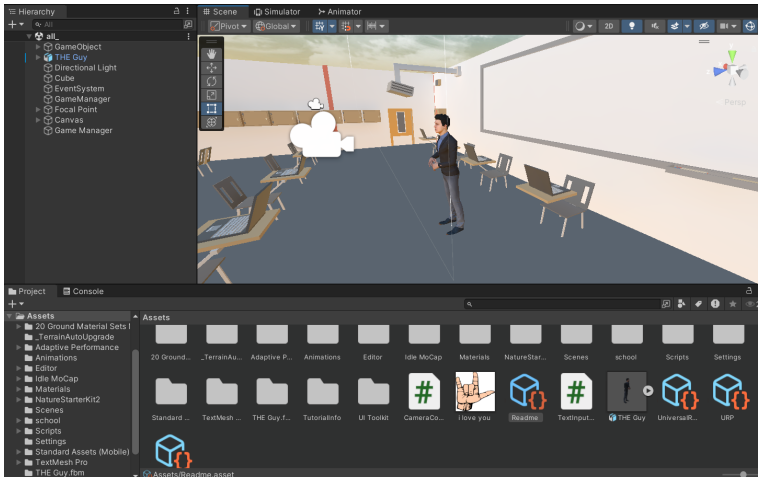

Figure 4.36: Unity environment showing the school setting and the 3D Avatar

Camera Setup We created a flexible camera system to allow different viewing angles and zoom levels. The camera was controlled using a script written in C#, which enabled both PC and mobile controls. Despite some lag in the zoom functionality, the script provided the necessary flexibility for viewing the avatar from various perspectives.

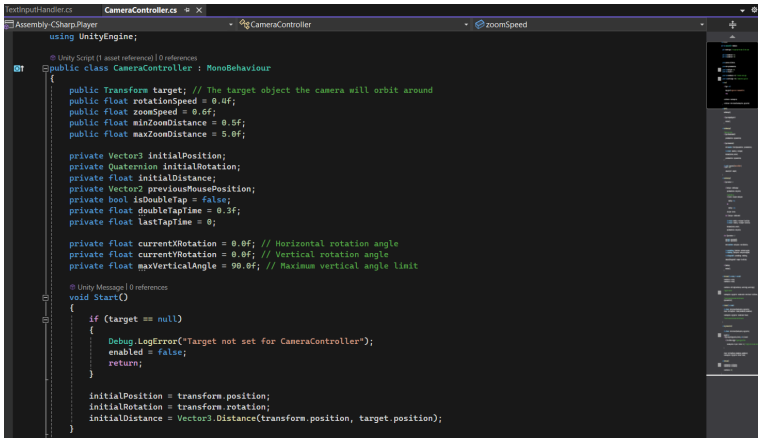

Figure 4.37: C# script used to control the camera in Unity

User Interface The user interface (UI) was created to facilitate interaction with the system. We included a text input field for users to enter text, and a button to start the animation. The UI design was kept simple to ensure ease of use, allowing users to quickly input text and trigger the corresponding animations.

Through careful planning and execution, we successfully set up the environment in Unity, imported our 3D avatar, and created a functional UI. These steps are crucial for enabling the avatar to perform ALSL gestures in various scenarios, providing a versatile platform for further development and testing.

Phase 06: Develop Text Treatment Algorithm

In this phase, our objective was to develop an algorithm that preprocesses text using natural language processing (NLP) techniques, ensuring the 3D avatar performs the correct animations based on user input. Despite some challenges, significant progress was made towards achieving this goal.

- **Tasks:**
  - Create an algorithm that uses NLP techniques to preprocess text.
  - Ensure the algorithm can selects the right animations to play.

  - Link the algorithm with the user interface, ensuring smooth interaction with the UI elements.

- **Progress and Challenges:** Initially, we focused on developing an algorithm capable of processing text input from the user interface. The goal was to ensure the algorithm could handle various synonyms and sentence structures, selecting the appropriate animations for the avatar to perform. However, due to time constraints and technical limitations, the full implementation of these tasks was not achieved within the project timeline.

- **Triggering Animations:** The first step towards achieving the text-to-animation functionality was to ensure the avatar could perform the correct animations in a specific order when the button in the UI is triggered. This part was successfully implemented, as shown in the animator setup.

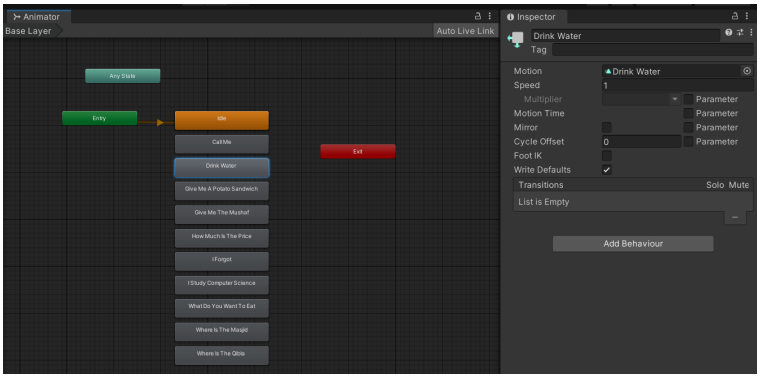

Figure 4.38: Animator setup in Unity for triggering animations

- **Natural Language Processing (NLP):** For the NLP component, we explored various tools and libraries. NLTK (Natural Language Toolkit) was considered but deemed unsuitable for our needs due to compatibility issues with C#. Instead, we explored alternative methods and algorithms available within the C# ecosystem.

- **Levenshtein Distance Algorithm:** The Levenshtein distance algorithm, also known as Edit Distance, is a measure used to quantify the dissimilarity between two strings. It calculates the minimum number of edits (insertions, deletions, or substitutions) required to change one string into the other. This metric is fundamental in various applications, including text editing, data cleaning, and clustering, among others. For instance, transforming the word ”kitten” into ”sitting” requires three edits: replacing ’k’ with ’s’, replacing ’e’ with ’i’, and adding ’g’ at the end, resulting in a Levenshtein distance of three. This algorithm is particularly useful

in applications such as spell-checking, DNA sequencing, and text preprocessing, where it helps in identifying and correcting errors by finding the closest match between strings. In the context of our project, the Levenshtein distance algorithm was employed to match user input with the closest animation name, enabling the avatar to perform the appropriate gesture based on the processed text.

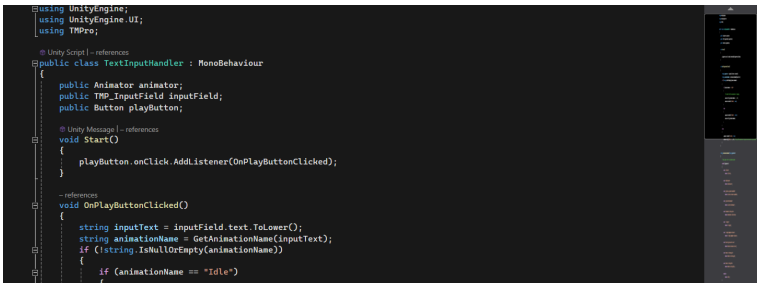

Figure 4.39: Text script implementation showing the use of Levenshtein distance algorithm

While the Levenshtein distance algorithm proved useful, it presented challenges when working with Arabic text. Arabic script has unique characteristics, such as complex morphology and contextual letter forms, which complicated the text matching process.

- **Solutions and Future Work:** To address these challenges, we considered various approaches:
  - Enhancing the text preprocessing step to better handle Arabic script, possibly integrating additional libraries or custom preprocessing steps.
  - Developing more sophisticated matching algorithms tailored to Arabic's linguistic nuances.
  - Collaborating with experts in Arabic NLP to refine and improve the algorithm's accuracy and performance.

Although the full integration of the text treatment algorithm with the user interface was not completed, the groundwork laid during this phase provides a solid foundation for future development. By leveraging advanced NLP techniques and further refining the algorithm, we aim to achieve seamless and accurate text-to-animation translation for ALSL gestures.

Phase 07: Deploying a Prototype

The final phase of our project involves deploying a working prototype of the ALSL translation system. This phase focuses on optimizing the 3D model and animations for performance across

different devices, configuring essential settings, implementing user controls, and gathering user feedback for further improvements.

- **Goals:**
  - Optimize the 3D model and animations for performance on various devices.
  - Configure main settings such as adjusting the resolution, quality settings, and performance optimizations.
  - Release the prototype to a selected group of users for feedback.
- **Optimization for Devices:** We started by ensuring that the 3D model and animations performed well across different devices. This involved reducing the polygon count where possible and optimizing textures and materials without compromising the quality of the animations. We also used Unity's optimization tools to adjust the level of detail (LOD) settings, ensuring that the application runs smoothly on devices with varying specifications.
- **Configuration of Main Settings:** Next, we configured the main settings of the Unity project. This included adjusting the resolution and quality settings to balance performance and visual fidelity. We enabled GPU instancing and adjusted the frame rate to ensure a consistent user experience. These settings were fine-tuned through iterative testing on the Unity emulator.

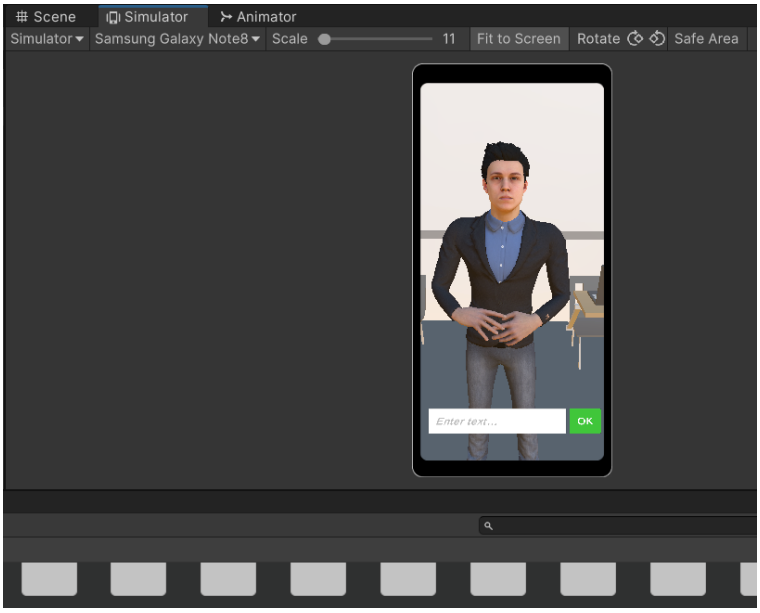

Figure 4.40: Testing the prototype on the Unity emulator

- **Initial Deployment:** After thorough testing in the Unity emulator, we deployed the prototype to a mobile device for real-world testing.

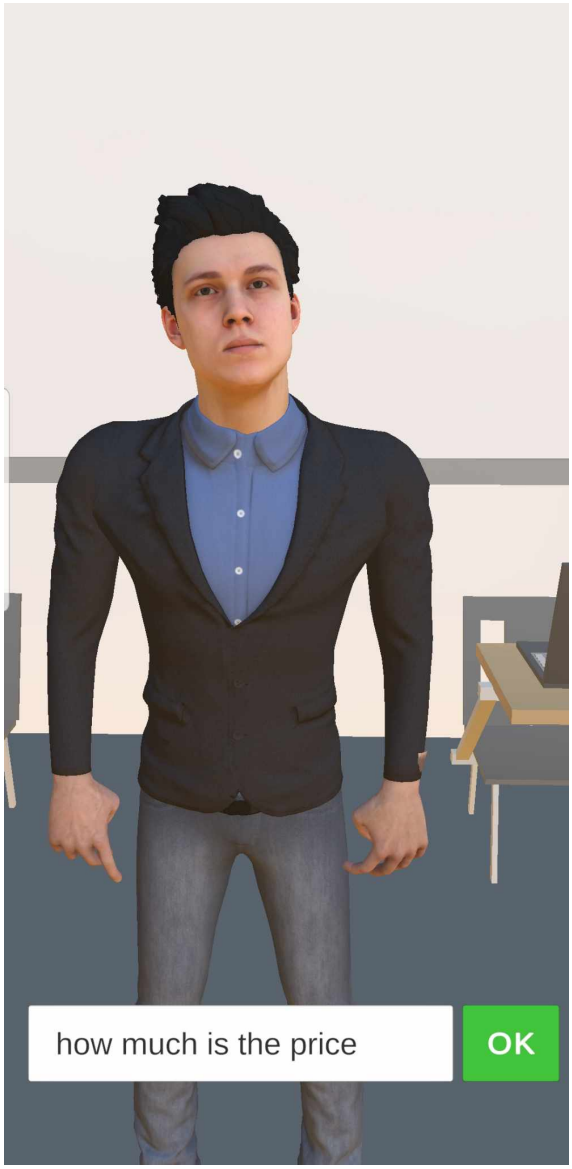

Figure 4.41: Prototype running on a mobile device

4.4 Results: Comparison

Notation System Method (NSM)

Programming Languages:

- HTML5, CSS3, JavaScript: Used to structure and style web content, and enable dynamic interaction.

Data Collection:

- Dicta-Sign Corpus: Utilized for collecting sign language data and translating it into HamNoSys notation.

Tools:

- HamNoSys: Encoding sign language gestures.
- SigML: Converting HamNoSys notations into animations.
- CWASA SIGML Player: Animating the gestures through a virtual avatar.

Challenges:

- Scarcity and lack of documentation for Algerian Sign Language (ALSL).
- Collaboration with experts to ensure accuracy.

Gesture Animation via Motion Capture (GAMC)

Programming Languages:

- Python: Developing scripts to generate BVH files from motion capture data.
- C#: Integrating the 3D avatar technology within the Unity engine.

Data Collection:

- Videos of ALSL: Recording high-resolution videos for motion capture.

Tools:

- Blender: Creating and animating the 3D avatar.

- FreeMocap, Dollars Mocap: Tools for recording and analyzing human movements.
- BVH: Format for motion capture files, allowing manual editing.

Challenges:

- Technical difficulties in using motion capture tools.
- Ensuring realistic animations and accurate gesture performances.

4.5 Discussion: Future Works

4.5.1 Enhancing NLP Algorithms

Improvement in Text Processing:

- Further development and refinement of the text preprocessing algorithm to handle the complexities of Arabic script, including its unique morphological and syntactical structures. This may involve integrating additional libraries or developing custom preprocessing steps tailored to Arabic.

Advanced Matching Algorithms:

- Develop more sophisticated matching algorithms to better address the linguistic nuances of Arabic, ensuring accurate and contextually appropriate translations. Collaboration with experts in Arabic NLP can be instrumental in refining these algorithms.

4.5.2 Expansion of ALSL Dataset

Data Collection:

- Continuously expand the ALSL dataset to cover a broader range of vocabulary and gestures. This involves recording and annotating more videos of sign language performances.

Quality Enhancement:

- Ensure the collected data is of high quality, with clear and accurate annotations. Regularly update the dataset to include new signs and variations used within the Algerian Sign Language community.

4.5.3 Optimization for Various Devices

Performance Tuning:

- Optimize the 3D model and animations to improve performance on various devices, including low-end smartphones and tablets. This includes reducing the polygon count, optimizing textures, and adjusting the level of detail settings.

Extensive Testing:

- Conduct comprehensive performance testing across multiple devices and browsers to ensure compatibility and smooth operation. Implement adaptive techniques to maintain performance without compromising visual quality.

4.5.4 User Interface Improvements

Usability Enhancements:

- Gather continuous user feedback to refine and enhance the UI design, ensuring it is intuitive, user-friendly, and accessible. This may include redesigning interface elements, improving navigation, and providing better feedback mechanisms.

Feature Expansion:

- Implement additional features based on user needs and feedback, such as customizable avatar settings, enhanced control options, and support for various input methods (e.g., voice input, gesture-based controls).

4.5.5 Advanced Gesture Animation

Sophisticated Motion Capture:

- Explore and integrate more advanced motion capture technologies to improve the realism and accuracy of the 3D avatar's gestures. This may involve using multi-camera setups, higher frame rates, and more precise tracking systems.

AI-Driven Animation:

- Investigate the use of artificial intelligence and machine learning techniques to generate more fluid and natural animations. AI-driven models can learn from large datasets of sign language performances, enabling the avatar to produce lifelike gestures and expressions.

4.5.6 Collaboration and Community Involvement

Partnerships:

- Establish partnerships with academic institutions, research centers, and sign language organizations to foster collaboration and knowledge exchange. Such partnerships can provide valuable resources, expertise, and validation for the project.

Community Engagement:

- Engage with the ALSL community through workshops, seminars, and online platforms to gather insights, validate the system, and encourage contributions to the dataset. Community involvement is crucial for ensuring the system remains relevant and accurate.

By addressing these future works, the Automated Arabic Algerian Sign Language Translation System can be significantly enhanced, providing a more accurate, efficient, and user-friendly tool for bridging communication gaps between Arabic speakers and the deaf community. The continued development and refinement of the system will contribute to its adoption and effectiveness in various real-world applications.

4.6 Conclusion

The development of the Automated Arabic Algerian Sign Language Translation System represents a significant step forward in improving communication for the Deaf community in Algeria. By leveraging the Notation System Method (NSM) and Gesture Animation via Motion Capture (GAMC), this project successfully created a tool that translates Arabic text into accurate and fluid ALSL gestures performed by a 3D avatar.

Throughout the implementation, various challenges were addressed, including the scarcity of high-quality ALSL datasets, the complexity of motion capture technology, and the intricacies of accurately translating and animating sign language gestures. Despite these challenges, the project achieved its primary goals, providing a functional and user-friendly translation system.

Future work will focus on expanding the ALSL dataset, enhancing the realism and accuracy of avatar animations, and improving the system's performance across different devices. Collaboration with the ALSL community and further refinement of the user interface will also be essential to ensure the system's continued relevance and effectiveness.

Ultimately, this project not only contributes to the field of sign language translation but also holds the promise of making a tangible difference in the lives of many Deaf individuals in Algeria, fostering greater inclusion and accessibility in communication.

General Conclusion

In conclusion, the development of an automated Arabic Algerian Sign Language translation system based on 3D avatar technology represents a significant step towards enhancing communication and inclusivity for the deaf community in Algeria. This project has demonstrated the potential of combining advanced systems engineering and web technologies to create an effective and user-friendly platform for sign language translation. The use of 3D avatars not only ensures accurate and expressive representation of ALSL but also provides a visually engaging interface that can be easily understood by users.

Chapter 1 provided a foundational understanding of sign languages, their structure, and the unique aspects of Algerian Sign Language, establishing the necessity of this project. Chapter 2's literature review highlighted the technological advancements and methodologies essential for developing a robust translation system. Chapter 3 detailed the conceptual framework, including the design and architecture, while Chapter 4 presented the implementation process and the practical realization of the system.

The successful implementation of this system highlights the importance of interdisciplinary collaboration in addressing complex social challenges. By integrating knowledge from computer science, linguistics, and user experience design, this project has paved the way for future innovations in sign language translation and accessibility technologies. Moving forward, further research and development can expand the capabilities of the system, incorporating more languages and refining the avatar technology to achieve even greater levels of accuracy and naturalness in sign representation. Ultimately, this work contributes to the broader goal of breaking down communication barriers and fostering a more inclusive society for all.

Bibliography

[1] Muhammad Sanaullah, Babar Ahmad, Muhammad Kashif, Tauqeer Safdar, Mehdi Hassan, Mohd Hilmi Hasan, and Norshakirah Aziz. A real-time automatic translation of text to sign language. *Computers, Materials and Continua*, 70(2), 2022.

[2] Lalit Goyal and Vishal Goyal. Automatic translation of english text to indian sign language synthetic animations. In *Proceedings of the 13th International Conference on Natural Language Processing*, pages 144–153, 2016.

[3] Ceil Lucas. *The Sociolinguistics of Sign Languages*. Cambridge University Press, 2002.

[4] A. C. Hamill and C. H. Stein. Culture and empowerment in the deaf community: An analysis of internet weblogs,. *Journal of Community and Applied Social Psychology*, page 388–406, 2011.

[5] D. M. Perlmutter. What is sign language. *Linguistic Society of America, Accessed*, 2023.

[6] P. A. Judéaux. French sign language: a language in its own right. *TradOnline, [Online] Available: French Sign Language: a language in its own right - Tra-donline*, 2023.

[7] D.F. Moores. Partners in progress: The 21st international congress on education of the deaf and the repudiation of the 1880 congress of milan. *American Annals of the Deaf*, 155(3):309–310, 2010.

[8] Donald F Moores. Partners in education: The 21st international congress on education of the deaf: Vancouver, british columbia july 2010. *American Annals of the Deaf*, 155(1):3–4, 2010.

[9] Varin Mathilde. Comparaison lexicale entre la langue des signes française (lsf) et la langue des signes britannique (bsl): le vocabulaire des nouvelles technologies, 2010.

[10] Thomet Julien. Une vue d'ensemble de la reconnaissance de gestes. In *Séminaire Gesture recognition*, 2009.

[11] Amal Djama. Les points communs entre la langue des signes algérienne (lsa) - dialecte de laghouat, sud de l'algérie - et la langue des signes française (lsf). Dossier, licence SCL « Acquisition et dysfonctionnement » (SCL F14), Licence 3, AMU, Faculté ALLSHS d'Aix-en-Provence, 2016.

[12] Foudil. Nekkaa. "détection automatique de la main: Application à la reconnaissance de la langue des signes arabe.". *Spécialité: Systèmes Distribués et Méthodes Formelles (SDMF), Univer-sité Abdelhamid Mehri-Constantine 2*, 2014/2015.

[13] Kouar El Bachir Kouar Imene. Deep-rsl: Deep learning for sign language recognition from a video sequence. Master's thesis, Yahia Fares University of Médéa, 2022–2023. Advisor: Dr. Kheldoun Ahmed.

[14] M. A. Abdel-Fattah. Arabic sign language: A perspective. *Journal of Deaf Studies and Deaf Education*, 2005.

[15] Kreeft J. Wilcox S. P. American sign language as a foreign language. eric digest. *https://eric.ed.gov/?id=ED429464*, 2004.

[16] FrancoSourd. Francosourd. "accueil." francosourd, 2023.

[17] Hicham. Abdelouaf. "deaf education in algeria: Is it a sustainable approach?". *Sociology Review Volume:05 / N°:02(2021)*, pages 417–429, 2021.

[18] S. Lanesman. Algerian jewish sign language: Its emergence and survival (master. *Ishara Press*, 2013.

[19] Sara Lanesman and Irit Meir. Algerian jewish sign language: A sociolinguistic sketch. *Sign languages in village communities: Anthropological and linguistic insights*, pages 361–364, 2012.

[20] Ulrike Zeshan and Connie De Vos. *Sign languages in village communities: Anthropological and linguistic insights*. de Gruyter, 2012.

[21] Meir I. Lanesman S. The survival of algerian jewish sign language alongside israeli sign language in israel. anthropological and linguistic insights. *Sign Languages in Village Communities*, October 2012.

[22] M. S. Mansour. Langage et surdité, descriptive de la langue des signes des sourds oranais (magistère). *Université d'Oran Es-Sénia.*, page 124, 2007.

[23] Bruno Bossard. *Problèmes posés par la reconnaissance de gestes en Langue des Signes*. PhD thesis, LIMSI-CNRS - Université Paris XI, Orsay, June 2002. Soutenu 24–27 juin 2002. 10 p.

[24] Abdelaziz Lakhfif. Un environnement de traduction automatique du texte arabe vers la langue des signes algérienne (lsa). Magistère en informatique, Université Badji Mokhtar-An-naba, 2009. Soutenu le Juillet 2009. 134p.

[25] Hicham Abdelouafi. Teaching sign language to the deaf children in adrar, algeria: A case study of the school of hearing-impaired children in hai graoui, 2018.

[26] Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. Including signed languages in natural language processing. *arXiv preprint arXiv:2105.05222*, 2021.

[27] P Ilanchezhian, I Amit Kumar Singh, M Balaji, A Manoj Kumar, and S Muhamad Yaseen. Sign language detection using machine learning. In *Semantic Intelligence: Select Proceedings of ISIC 2022*, pages 135–143. Springer Nature Singapore, 2023.

[28] Chai X. Min Y., Hao A. and Chen X. Visual alignment constraint for continuous sign language recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11542–11551, 2021.

[29] Mazen S. Nada B.I., Hala Z. Advances, challenges, and opportunities in continuous sign language recognition. *ARPN Journal of Engineering and Applied Sciences*, 15(5):1205–1227, December 2019.

[30] Gouiffès M. Chaaban H. and Braffort A. Automatic annotation and segmentation of sign language videos: Base-level features and lexical signs classification. In *16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021)*, volume 5, pages 484–491, February 2021.

[31] Müller M. Ebling S. Moryossef A., Jiang Z. and Goldberg Y. Linguistically motivated sign language segmentation. *arXiv preprint arXiv:2310.13960*, 2023.

[32] Zheng Yu Tan, Shafriza Nisha Basah, Haniza Yazid, and Muhammad Juhairi Aziz Safar. Performance analysis of otsu thresholding for sign language segmentation. *Multimedia Tools and Applications*, 80:21499–21520, 2021.

[33] S. Sabharwal and P. Singla. Alphabet-level indian sign language translation to text using hybrid-ao thresholding with cnn. *Intelligent Automation and Soft Computing*, 37(3), 2023.

[34] Müller M. Ebling S. Moryossef A., Jiang Z. and Goldberg Y. Linguistically motivated sign language segmentation. *arXiv preprint arXiv:2310.13960*, 2023.

[35] Pushan Kumar Datta, Abhishek Biswas, Ahona Ghosh, and Nilanjana Chaudhury. Creation of image segmentation classifiers for sign language processing for deaf and dumb. In *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pages 772–775. IEEE, June 2020.

[36] Sunar M.S. Kolivand H., Joudaki S. and Tully D. A new framework for sign language alphabet hand posture recognition using geometrical features through artificial neural network (part 1). *Neural Computing and Applications*, 33(10):4945–4963, 2021.

[37] Gary J Grimes. Digital data entry glove interface device, 1983.

[38] Ibrahim Adepoju Adeyanju, Oluwaseyi Olawale Bello, and Mutiu Adesina Adegboye. Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, 12:200056, 2021.

[39] Khalid Al-Qurishi and Souissi. Deep learning for sign language recognition: Current techniques, benchmarks, and open issues. *IEEE Access*, 9:126917–126951, 2021.

[40] Akçay Arslan, Şahin. Deep learning-based isolated sign language recognition: a novel approach to tackling communication barriers for individuals with hearing impairments. *Journal of Scientific Reports-A*, (055):50–59, 2023.

[41] Rauers Blanke and Riediger. Nice to meet you—adult age differences in empathic accuracy for strangers. *Psychology and Aging*, 30(1):149, 2015.

[42] Bellard M. Berke L. Boudreault P. Braffort A. Caselli N. Huenerfauth M. Kacorri H. Verhoef T. Bragg D., Koller O. and Vogler C. Sign language recognition, generation, and translation: An interdisciplinary perspective. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, pages 16–31, October 2019.

[43] Zhuang L. Zhou W. Zhang Z., Pu J. and Li H. Continuous sign language recognition via reinforcement learning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 285–289. IEEE, September 2019.

[44] Geetha M. Aloysius N. and Nedungadi. Incorporating relative position information in transformer-based sign language recognition and translation. *IEEE Access*, 9:145929–145942, 2021.

[45] Zhu P. Wang Z. Wang Y. Qian J. Hou J., Li X.Y. and Yang P. Signspeaker: A real-time, high-precision smartwatch-based sign language translator. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–15, August 2019.

[46] Roy P.P. Balasubramanian R. Mittal A., Kumar P. and Chaudhuri B.B. A modified lstm model for continuous sign language recognition using leap motion. *IEEE Sensors Journal*, 19(16):7056–7063, 2019.

[47] Hong R. Tang S., Guo D. and Wang M. Graph-based multimodal sequential embedding for sign language translation. *IEEE Transactions on Multimedia*, 24:4433–4445, 2021.

[48] Ventura L. Ghadiyaram D. DeHaan K. Metze F. Torres J. Duarte A., Palaskar S. and Giro i Nieto X. How2sign: A large-scale multimodal dataset for continuous american sign language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2735–2744, 2021.

[49] Yin K. and Read J. Better sign language translation with stmc-transformer, 2020. arXiv preprint arXiv:2004.00588.

[50] Khan Nabeela, Tan Sihan, Itoyama Katsutoshi, and Nakadai Kazuhiro. Back translation in sign language generation. *Materials of the second research meeting of the Society for Artificial Intelligence*, 2023(Challenge-063):05, 2023.

[51] Maucher J. Baumgärtner L., Jauss S. and Zimmermann G. Automated sign language translation: The role of artificial intelligence now and in the future. In *Proceedings of the International Conference on Computer-Human Interaction Research and Applications (CHIRA)*, pages 170–177, November 2020.

[52] Woll B. Fox N. and Cormier K. Best practices for sign language technology research. *Universal Access in the Information Society*, pages 1–9, 2023.

[53] Adrián Núñez-Marcos, Olatz Perez-de Viñaspre, and Gorka Labaka. A survey on sign language machine translation. *Expert Systems With Applications*, 2023.

[54] K. Kamata, T. Yoshida, M. Watanabe, and Y. Usui. An approach to japanese-sign language translation system. In *Conference Proceedings., IEEE International Conference on Systems, Man and Cybernetics*, pages 1089–1090 vol.3, 1989.

[55] Tony Veale and Aisling Conway. Cross modal comprehension in zardoz: An english to sign-language translation system. In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 249–252, 1994.

[56] Jintae Lee and Tosiyasu L Kunii. Visual translation: From native language to sign language. In *Proceedings IEEE workshop on visual languages*, pages 103–109. IEEE, 1992.

[57] Masaru Ohki, Hirohiko Sagawa, Tomoko Sakiyama, Eiji Oohira, Hisashi Ikeda, and Hiromichi Fujisawa. Pattern recognition and synthesis for sign language translation system. In *Proceedings of the first annual ACM conference on Assistive technologies*, pages 1–8, 1994.

[58] H. Sagawa, M. Ohki, T. Sakiyama, E. Oohira, H. Ikeda, and H. Fujisawa. Pattern recognition and synthesis for a sign language translation system. *Journal of Visual Languages and Computing*, 7(1):109–127, 1996.

[59] M. Tokuda and M. Okumura. Towards automatic translation from japanese into japanese sign language. *Assistive technology and artificial intelligence*, pages 97–108, 1998.

[60] B. Bauer, S. Nießen, and H. Hienz. Towards an automatic sign language translation system. In *Proceedings of the 1st International Conference*. Citeseer, 1999.

[61] A.B. Grieve-Smith. English to american sign language machine translation of weather reports. In *Proceedings of the Second High Desert Student Conference in Linguistics (HDSL2)*, pages 23–30, Albuquerque, NM, 1999.

[62] Yin K. and Read J. Better sign language translation with stmc-transformer. *arXiv preprint arXiv:2004.00588*, 2020.

[63] Perez-de-Viñaspre O. Núñez-Marcos A. and Labaka G. A survey on sign language machine translation. *Expert Systems with Applications*, 213:118993, 2023.

[64] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

[65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, and et al. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[66] T. Miyazaki, Y. Morita, and M. Sano. Machine translation from spoken language to sign language using pre-trained language model as encoder. In *Proceedings of the LREC2020 9th Workshop on the Representation and Processing of Sign Languages: Sign Language Resources in the Service of the Language Community, Technological Challenges and Application Perspectives*, pages 139–144, 2020.

[67] B. Fang, J. Co, and M. Zhang. Deepasl: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, pages 1–13, 2017.

[68] M.E.M. Ahli. Towards a reliable machine learning-based model designed for translating sign language videos to text, 2023.

[69] Sarigiannidis P. Papatsimouli M. and Fragulis G.F. A survey of advancements in real-time sign language translators: Integration with iot technology. *Technologies*, 11(4):83, 2023.

[70] Dimitar Shterionov, Mirella De Sisto, Vincent Vandeghinste, Aoife Brady, Mathieu De Coster, Lorraine Leeson, Josep Blat, Frankie Picron, Marcello Scipioni, Aditya Parikh, et al. Sign language translation: Ongoing development, challenges and innovations in the signon project. In *23rd Annual Conference of the European Association for Machine Translation*, pages 325–326. European Association for Machine Translation, 2022.

[71] He Y. Rahmani H. Gong J., Foo L.G. and Liu J. Llms are good sign language translators. *arXiv preprint arXiv:2404.00925*, 2024.

[72] N. Berdic, S. Mihic, and D. Dragan. 3d full body avatar applicability in consumer products. In *The proceedings of international conference on Mass Customization and Personalization in Central Europe MCP-CE*, pages 24–29, 2016.

[73] Eliane Schlemmer, Deividi Trein, and Cátia Oliveira. The metaverse: Telepresence in 3d avatar-driven digital-virtual worlds. *@ tic. revista d'innovació educativa*, (2):26–32, 2009.

[74] Frank A Salamone. *Encyclopedia of religious rites, rituals, and festivals*. Routledge, 2004.

[75] Brian Dear. *The Friendly Orange Glow*. Pantheon Books, first edition, 2017.

[76] Michael Gerhard, David Moore, and Dave Hobbs. Embodiment and copresence in collaborative interfaces. *International Journal of Human-Computer Studies*, 61(4):453–480, 2004.

[77] Peter H.Diamandis. What is an avatar, really? Available at: https://www.xprize.org/articles/what-is-an-avatar-really, Dec 18 2020.

[78] Christian Timmerer, Jean Gelissen, Markus Waltl, and Hermann Hellwagner. Interfacing with virtual worlds. *Network and Electronic Media Summit*, 2009.

[79] Benard Gachanja Wanjiru. *Analysis of methods for creation of human 3D avatars*. PhD thesis, ETSI_Informatica, 2022.

[80] Marius Preda and Blagica Jovanova. Avatar interoperability and control in virtual worlds. *Signal Processing: Image Communication*, 28(2):168–180, 2013.

[81] Xiang Suo, Weidi Tang, and Zhen Li. Motion capture technology in sports scenarios: a survey. *Sensors*, 24(9):2947, 2024.

[82] Ron Fischer. Motion capture process and systems. *M. Jung, R. Fischer, M. Gleicher (Eds.), Motion capture and editing: Bridging principle and practice. Natick, MA: AK Peters*, 2002.

[83] Roya Haratian. Motion capture sensing technologies and techniques: A sensor agnostic approach to address wearability challenges. *Sensing and Imaging*, 23(1):25, 2022.

[84] Emma Harvey, Hauke Sandhaus, Abigail Z Jacobs, Emanuel Moss, and Mona Sloane. The cadaver in the machine: The social practices of measurement and validation in motion capture technology. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–23, 2024.

[85] Clémence Mertz, Vincent Barreaud, Thibaut Le Naour, Damien Lolive, and Sylvie Gibet. A low-cost motion capture corpus in french sign language for interpreting iconicity and spatial referencing mechanisms. In *Language Resources and Evaluation Conference*, 2022.

[86] Mohamed-El-Fatah Benchiheub, Bastien Berret, and Annelies Braffort. Collecting and analysing a motion-capture corpus of french sign language. In *sign-lang@ LREC 2016*, pages 7–12. European Language Resources Association (ELRA), 2016.

[87] Pavel Jedlička, Zdeněk Krňoul, Miloš Železnỳ, and Luděk Müller. Mc-trislan: A large 3d motion capture sign language data-set. In *Proceedings of the LREC2022 10th Workshop on the Representation and Processing of Sign Languages: Multilingual Sign Language Resources*. European Language Resources Association, 2022.

[88] Yutong Gu. *Sign Language Translation Using Wearable Motion Capture System and Machine Learning Methods*. PhD thesis, □□□□□, 2022.

[89] Ready Player Me. Animation, motion capture, and ai software: 3d full body avatar creator, 2024. Accessed: 2024-05-27.

[90] Deusens. Avatars with motion capture technology for businesses, 2024. Accessed: 2024-05-27.

[91] Design4Real. Ai motion capture, 2024. Accessed: 2024-05-27.

[92] Top AI Tools. Motion capture tools, 2024. Accessed: 2024-05-27.

[93] Saiwa AI. Openpose vs mediapipe, 2024. Accessed: 2024-05-27.

[94] Roboflow. What is openpose?, 2024. Accessed: 2024-05-27.

[95] Ikomia. Complete openpose guide, 2024. Accessed: 2024-05-27.

[96] OpenFuture AI. Dollars mocap, 2024. Accessed: 2024-05-27.

[97] Rokoko. Richer animations start with rokoko mocap. https://www.rokoko.com/, Accessed 2024.

[98] Laser Scan. 3d scanning technical information. Available at: http://www.3dscanco.com/about/3d-scanning/, 2015.

[99] D. Dragan, S. Mihic, Z. Anisic, and I. Lukovic. Role of background subtraction in creating human body point clouds from photos. In *The 6th International Conference and Exhibition on 3D Body Scanning Technologies*, pages 210–217, Lugano, Switzerland, October 27-28 2015.

[100] S. Hermena and M. Young. Ct-scan image production procedures. *StatPearls*, 2022. PMID 34662062, retrieved 2023-11-24.

[101] Donald W McRobbie, Elizabeth A Moore, Martin J Graves, and Martin R Prince. *MRI from Picture to Proton*. Cambridge University Press, Cambridge, 2007.

[102] Fabian. Top 25: Most popular 3d modeling and design software for 3d printing, 2016.

[103] Brian Mallon. 3d avatar revolutionises skin cancer diagnosis, 2015.

[104] Bouzid and Jemni. The effect of avatar technology on sign writing vocabularies acquisition for deaf learners. *2016 IEEE 16th International Conference on Advanced Learning Technologies (ICALT)*, pages 441–445, July 2016.

[105] Kiriakos Stefanidis, Dimitrios Konstantinidis, Athanasios Kalvourtzis, Kosmas Dimitropoulos, and Petros Daras. 3d technologies and applications in sign language. *Recent advances in 3D imaging, modeling, and reconstruction*, pages 50–78, 2020.

[106] M. J. Davidson. PAULA: A computer-based sign language tutor for hearing adults. In *Intelligent Tutoring Systems 2006 Workshop on Teaching with Robots, Agents, and Natural Language Processing*, 2006.

[107] Kerem Yorganci, Ahmet Ata Kindiroglu, and Hatice Kose. Avatar-based sign language training interface for primary school education. In *Workshop: Graphical and Robotic Embodied Agents for Therapeutic Systems*, 2016.

[108] Kumar Kaur. Hamnosys to sigml conversion system for sign language automation. *Procedia Computer Science*, 89:794–803, 2016.

[109] Maria Papadogiorgaki, Nikos Grammalidis, Dimitrios Tzovaras, and Michael G Strintzis. Text-to-sign language synthesis tool. In *2005 13th European Signal Processing Conference*, pages 1–4. IEEE, 2005.

[110] Ferreira. *3D Character Animation Using Sign Language*. PhD thesis, Universidade do Porto (Portugal), 2017.

[111] Y. Bouzid and M. Jemni. A virtual signer to interpret signwriting. In K. Miesenberger, D. Fels, D. Archambault, P. Pevnáz, and W. Zagler, editors, *Computers Helping People with Special Needs*, pages 458–465. Springer International Publishing, 2014.

[112] I. E. Murtagh. A linguistically motivated computational framework for irish sign language. *Trinity College*, 2019.

[113] Matt Huenerfauth. Learning to generate understandable animations of american sign language. In *Proceedings of the 2nd Annual Effective Access Technologies Conference*, 2014.

[114] Maarif, Akmeliawati, and Gunawan. Survey on language processing algorithm for sign language synthesizer. *International Journal of Robotics and Mechatronics*, 4(2):39–48, 2018.

[115] Ralph Elliott, John RW Glauert, JR Kennaway, and Ian Marshall. The development of language processing support for the visicast project. In *Proceedings of the fourth international ACM conference on Assistive technologies*, pages 101–108, 2000.

[116] Hanke Thomas. Hamnosys–representing sign language data in language resources and language processing contexts. In *sign-lang@ LREC 2004*, pages 1–6. European Language Resources Association (ELRA), 2004.

[117] Robert Smith. Hamnosys 4.0 user guide. *Institute of technology Blanchardstown*, 2013.

[118] Ralph Elliott, John RW Glauert, JR Kennaway, and Ian Marshall. The development of language processing support for the visicast project. In *Proceedings of the fourth international ACM conference on Assistive technologies*, pages 101–108, 2000.

[119] Margriet Verlinden, Corrie Tijsseling, and Han Frowein. A signing avatar on the www. In *Gesture and Sign Language in Human-Computer Interaction: International Gesture Workshop, GW 2001 London, UK, April 18–20, 2001 Revised Papers*, pages 169–172. Springer, 2002.

[120] Televirtual LTD. The visia avatar, 2000. Accessed: 2024-05-30.

[121] Virtual Humans Group. Jasigning, 2021. Accessed: 2024-05-30.

[122] Thomas Hanke and Hortensia Popescu. esign -essential sign language information on government networks, 2005.

[123] Virtual Humans Group. Cwa signing avatars, 2024. Accessed: 2024-05-30.

[124] J Andrew Bangham, SJ Cox, Ralph Elliott, John RW Glauert, Ian Marshall, Sanja Rankov, and Mark Wells. Virtual signing: Capture, animation, storage and transmission-an overview of the visicast project. In *IEE Seminar on speech and language processing for disabled and elderly people (Ref. No. 2000/025)*. IET, 2000.

[125] University of East Anglia. esign project. https://www.visicast.cmp.uea.ac.uk/eSIGN/index.html, 2024. [Online; accessed 19-June-2024].

[126] University of East Anglia. Animating sign language: The esign approach. Accessed: 2024-05-30.

[127] Ahmed A Alethary, Ahmed Hussein Aliwy, and Nabeel Salih Ali. Automated arabic-arabic sign language translation system based on 3d avatar technology. *Int J Adv Appl Sci*, 11(4):383–396, 2022.

[128] Debashis Das Chakladar, Pradeep Kumar, Shubham Mandal, Partha Pratim Roy, Masakazu Iwamura, and Byung-Gyu Kim. 3d avatar approach for continuous sign movement using speech/text. *Applied Sciences*, 11(8):3439, 2021.

[129] Bhavinkumar Devendrabhai Patel, Harshit Balvantrai Patel, Manthan Ashok Khanvilkar, Nidhi Rajendrakumar Patel, and Thangarajah Akilan. Es2isl: an advancement in speech to sign language translation using 3d avatar animator. In *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5. IEEE, 2020.

[130] Malinda Punchimudiyanse and Ravinda GN Meegama. 3d signing avatar for sinhala sign language. In *2015 IEEE 10th international conference on industrial and information systems (ICIIS)*, pages 290–295. IEEE, 2015.

[131] Yosra Bouzid and Mohamed Jemni. An avatar based approach for automatically interpreting a sign language notation. In *2013 IEEE 13th International Conference on Advanced Learning Technologies*, pages 92–94. IEEE, 2013.

[132] Achraf Othman and Mohamed Jemni. An xml-gloss annotation system for sign language processing. In *2017 6th International Conference on Information and Communication Technology and Accessibility (ICTA)*, pages 1–7. IEEE, 2017.

[133] Omar H Al-Barahamtoshy and Hassanin M Al-Barhamtoshy. Arabic text-to-sign (artts) model from automatic sr system. *Procedia Computer Science*, 117:304–311, 2017.

[134] Diego Addan Gonçalves, Eduardo Todt, and Laura Sanchez Garcia. 3d avatar for automatic synthesis of signs for the sign languages. In *WSCG 2015 Conference on Computer Graphics, Visualization and Computer Vision*. Václav Skala-UNION Agency, 2015.

[135] Hamzah Luqman and Sabri A Mahmoud. Automatic translation of arabic text-to-arabic sign language. *Universal Access in the Information Society*, 18(4):939–951, 2019.

[136] Quach Luyl Da, Nguyen Hua Duy Khang, and Nguyen Chi Ngon. Converting the vietnamese television news into 3d sign language animations for the deaf. In *Industrial Networks and Intelligent Systems: 14th EAI International Conference, INISCOM 2018, Da Nang, Vietnam, August 27–28, 2018, Proceedings*, pages 155–163. Springer, 2019.

[137] A Mb and ATLASLang MTS Ab B. 1: Arabic text language into arabic sign language machine translation system-sciencedirect. *Procedia Computer Science*, 148:236–245, 2019.

[138] Academy of Sciences and Humanities in Hamburg. Dgs-korpus project. https://web.dgs-korpus.de/hamnosys-97.html, 2024. [Online; accessed 19-June-2024].

[139] Ralph Elliott, John Glauert, Vince Jennings, and Richard Kennaway. An overview of the sigml notation and sigmlsigning software system. *sign-lang@ LREC 2004*, pages 98–104, 2004.

[140] University of East Anglia. CWA Signing Avatars. https://vh.cmp.uea.ac.uk/index.php/CWA_Signing_Avatars, 2024. [Online; accessed 19-June-2024].

[141] Microsoft Corporation. Visual Studio Code Learn. https://code.visualstudio.com/learn#:~:text=Visual%20Studio%20Code%20is%20a,programming%20language,%20without%20switching%20editors., 2024. [Online; accessed 19-June-2024].

[142] JSON.org. JSON: The JavaScript Object Notation Website. https://www.json.org/json-en.html, 2024. [Online; accessed 19-June-2024].

[143] AviaTechno. WampServer. https://wampserver.aviatechno.net/, 2024. [Online; accessed 19-June-2024].

[144] Vercel Inc. Vercel Documentation. https://vercel.com/docs, 2024. [Online; accessed 19-June-2024].

[145] Eleni Efthimiou, Stavroula-Evita Fotinea, Thomas Hanke, John Glauert, Richard Bowden, Annelies Braffort, Christophe Collet, Petros Maragos, and François Lefebvre-Albaret. Sign language technologies and resources of the dicta-sign project. In *5th Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon. Satellite Workshop to the eighth International Conference on Language Resources and Evaluation (LREC-2012)*, 2012.

[146] Algerian sign laguage page on facebook. Algerian sign language page on facebook. https://www.facebook.com/profile.php?id=100093996740140, 2024. [Online; accessed 19-June-2024].

[147] VisiCast Project. Driving the SiGML player app. https://vh.cmp.uea.ac.uk/index.php/Driving_the_SiGML_Player_App, 2024. Accessed: 2024-06-22.

[148] LI Chengqing ZHENG Weicheng Vincent CHAN Ka Chun, CHEUNG Pan Yeung and LO Terence. Bvh motion capture data animated, 2007. ©2007 CS4185/5185 Multimedia Technologies and Applications (Semester A, 2007/2008), Accessed: 2024-06-22.

[149] Maddock Meredith, Steve Maddock, et al. Motion capture file formats explained. *Department of Computer Science, University of Sheffield*, 211:241–244, 2001.