

Bidirectional Deaf-Hearing Communication: a Modular Service- Oriented System Combining Multi-Purpose Artificial Intelligence and 3D Avatar Rendering

Telmo Adão

`telmoadao@dsi.uminho.pt`

ALGORITMI Center/LASI, University of Minho

Somayeh Shahrabadi

ALGORITMI Center/LASI, University of Minho

Bruno Ribeiro

ADiT-Lab, Polytechnic University of Viana do Castelo

Duarte Dias

ADiT-Lab, Polytechnic University of Viana do Castelo

Vasco Alves

ADiT-Lab, Polytechnic University of Viana do Castelo

Filipe Pereira

First Solutions

Tiago Oliveira

First Solutions

Fábio Araújo

First Solutions

Luís Romero

ADiT-Lab, Polytechnic University of Viana do Castelo

Pedro Miguel Faria

ADiT-Lab, Polytechnic University of Viana do Castelo

Luís Gonzaga Magalhães

ALGORITMI Center/LASI, University of Minho

Research Article

Keywords: Bidirectional Deaf-Hearing Communication, Deep Learning-based Sign Language Recognition, Gloss-based Large Language Model-Powered Sentence Generation, Natural Language-to-Gloss

Translation, 3D Avatar-based Sign Language Rendering, Service-Oriented Architecture

Posted Date: August 25th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10143194/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Bidirectional Deaf-Hearing Communication: a Modular Service-Oriented System Combining Multi-Purpose Artificial Intelligence and 3D Avatar Rendering

Telmo Adão^{a,*}, Somayeh Shahrabadi^a, Bruno Ribeiro^b, Duarte Dias^b, Vasco Alves^b, Filipe Pereira^c, Tiago Oliveira^c, Fábio Araújo^c, Luís Romero^b, Pedro Miguel Faria^b and Luís Gonzaga Magalhães^{a,b}

a) ALGORITMI Center/LASI, University of Minho, Campus de Azurém, Av. da Universidade, 4800-058, Guimarães, Portugal.

b) ADIT-Lab Polytechnic University of Viana do Castelo, Rua Escola Industrial e Comercial de Nun'Álvares, 4900-367, Viana do Castelo, Portugal.

c) First Solutions, Matosinhos Innovation Centre, Rua Conselheiro Costa Braga, 502 F, 4450-102, Matosinhos, Portugal.

d) CCG/ZGDV Institute, University of Minho, Campus de Azurém, 4800-058, Guimarães, Portugal.

*Corresponding author: telmoadao@dsi.uminho.pt

Acknowledgements: The authors would like to thank Portuguese Association of the Deaf (officially known as Associação Portuguesa de Surdos or APS), particularly Vânia Ferreira, Ângelo Costa, and the Portuguese Sign Language interpreters involved, for their valuable support in preparing, structuring, and validating the Portuguese Sign Language portfolio used in this work.

Abstract

Purpose: Communication between deaf and hearing individuals remains challenging due to the heterogeneity of communication modalities and the limited availability of integrated systems supporting effective bidirectional interaction. This study aims to address this gap by investigating how a modular, service-oriented architecture (SOA) can support two-way communication between both user profiles.

Methods: The proposed Sign Language System was designed as a decoupled architecture composed of three interoperable modules: a Sign Language Interpretation Module for landmark-based sign recognition and natural-language sentence construction; an Avatar-Generated Sign Language Module for natural-language-to-gloss translation and 3D avatar-based rendering; and a Chatroom Frontend Module responsible for user interaction, profile-aware orchestration, landmark acquisition, and avatar visualization. The system was implemented and assessed using European Portuguese and Portuguese Sign Language as a case study.

Results: An integrated functional prototype featuring computer vision, deep learning, language-model-based sentence construction, natural-language processing, avatar-based sign language rendering, was developed under SOA principles. Building on previously reported module-level evaluations, it achieved 95.6% sign recognition accuracy and an average semantic similarity of 0.81 in sentence construction across commerce, tourism, and healthcare contexts. This work also reports an expert assessment of the avatar-based sign synthesis component, which reached an average gesture comprehensibility of 84%, and the integration of sign-language-to-natural-language and natural-language-to-sign-language flows with response times compatible with near real-time interaction.

Conclusion: The results demonstrate the feasibility of a loosely coupled, service-oriented approach for bidirectional deaf-hearing communication, while preserving modularity, scalability, and extensibility to additional vocabularies, interaction scenarios, and sign languages.

Keywords: Bidirectional Deaf-Hearing Communication; Deep Learning-based Sign Language Recognition; Gloss-based Large Language Model-Powered Sentence Generation; Natural Language-to-Gloss Translation; 3D Avatar-based Sign Language Rendering; Service-Oriented Architecture.

1 Introduction

Since the early developments of real-time remote communication systems, voice telephony has stood out as a major milestone, enabling the conversion of electrical signals into sound and vice versa, thus allowing conversational interaction over distance [1].

With the evolution of communication networks and the Internet, real-time remote communication solutions have significantly diversified. More specifically, it has enabled the establishment of synchronous voice and video teleconferencing systems and platforms supporting the involvement of two or more participants [2].

More recently, several applications have incorporated real-time idiom conversion capabilities, allowing individuals speaking different languages to communicate seamlessly, wherein artificial intelligence (AI)-driven machine translation plays a vital role [3]. From a service perspective, this problem can be considered fairly addressed, with real-time performance being reasonably ensured. However, for deaf individuals, the biophysical nature of communication necessarily diverges from conventional speech-based interaction. Instead, communication relies on structured systems of signs, combining sequences of upper limb movements and gestures with complementary body pose and facial expressions [4]. On the other hand, written/reading skills may not be viable in scenarios where one or both participants (deaf and/or hearing) experience literacy limitations [5]. In such cases, it becomes necessary to support bidirectional translation between inherently incompatible communication systems, requiring intermediary representations that enable mutual understanding. This includes converting spoken or written language into gestures for deaf users, and transforming gestures into written or synthesized speech for hearing users.

The capture, analysis, interpretation, and synthesis of gesture-based communication involve complex visual computing processes [6], [7]. When implemented as-a-service, concerns regarding the consumption of computational and networking resources [8] must be considered. These demands pose significant challenges to maintaining fluid real-time interaction in systems with such stringent communication requirements. These challenges are addressed in this work by proposing an approach grounded on the following research objectives:

- i. the integration of specialized communication components within an end-to-end service-oriented architecture – gesture-to-natural-language and natural-language-to-gesture – enabling seamless communication between deaf and hearing individuals, including scenarios involving limited literacy; and
- ii. the design of a modular platform operating in a distributed computing environment, leveraging both graphics processing and artificial intelligence resources, with the ability to dynamically mitigate computational bottlenecks arising from visual data processing, AI-based interpretation, gesture synthesis, and data transmission.

While existing digital solutions addressing deaf–hearing communication are predominantly unilateral, the proposed work consolidates bidirectionality by integrating previously independent contributions: i) sign language interpretation using both predictive and generative AI models [8], [9]; and ii) gesture synthesis based on linguistic machine learning and 3D avatar rendering [5], [10], [11]. Therefore, this paper documents the integration efforts leading to a unified platform, hereinafter referred to as the Sign Language Digital

Interpretation System (SLaDIS), as a full-stack solution capable of bridging deaf and hearing users through two-way communication.

The remainder of this document is organized as follows: Section 2 presents the background and related work on deaf–hearing communication systems. Section 3 introduces SLaDIS as a service-oriented architecture for deaf–hearing communication. Section 4 describes the system implementation. Section 5 presents and discusses the main results at both the module level and the integrated application layer. Finally, Section 6 concludes the paper and outlines directions for future work.

2 Background and Related Work

Gesture recognition has become a central topic in computer vision and human–computer interaction (HCI), enabling natural communication between humans and machines in domains such as virtual reality [12], robotics[13], smart environments[14]. The field has evolved significantly from traditional handcrafted feature-based methods toward deep learning–based architectures that provide higher accuracy and better generalization[15].

Early approaches relied on manually engineered features such as histogram of oriented gradients (HOG)[16], optical flow[17], and hidden Markov models (HMMs)[18]. HOG-based methods were widely used to capture spatial appearance information of hand shapes by extracting gradient orientation histograms, often combined with classifiers such as support vector machines (SVMs) for gesture classification[19], a temporal gesture recognition system based on HOG and motion trajectory features which demonstrated effective hand localization and classification, but required careful feature design and preprocessing steps. Similarly, optical flow–based approaches[20] modelled motion by estimating pixel-wise displacement between consecutive frames, allowing the extraction of dynamic motion patterns for gesture recognition. More advanced works extended this idea to 3D optical flow representations combined with motion context descriptors to improve view-invariant gesture recognition [21]. In addition, hidden Markov models (HMMs) were extensively used to model the temporal evolution of gestures as probabilistic state transitions. For instance, early systems combined spatial features (Fourier descriptors) with motion information and used HMMs to recognize dynamic gestures from sequential data [22]. HMM-based approaches were particularly effective for continuous gesture recognition tasks due to their ability to model time-series data; however, they required significant computational resources and careful parameter tuning, limiting scalability in complex scenarios [23]. Despite their success, these traditional handcrafted approaches exhibited several limitations. Many methods depended heavily on assumptions such as controlled environments, static backgrounds, or consistent lighting conditions. For example, motion-based and color-based hand detection techniques were shown to be sensitive to illumination changes and background clutter, reducing robustness in real-world settings [19]. Furthermore, optical flow–based methods are computationally expensive and rely on accurate motion estimation, which can degrade under noise or rapid movement [24].

With the emergence of deep learning, convolutional neural networks (CNNs) became widely adopted for static gesture recognition due to their strong ability to learn spatial representations directly from RGB and depth images. Recurrent neural networks (RNNs)[25], particularly long short-term memory (LSTM) networks [26], extended these capabilities by modeling temporal dependencies in dynamic gestures, enabling sequence-

based recognition of hand and body movements. Cifuentes et. al [27] proposed the use of LSTM networks for gesture classification, enabling the modeling of temporal dynamics in sequential data. Their approach showed enhanced recognition accuracy compared to conventional methods by capturing long-term dependencies in gesture sequences. Abraham et.al [28] used Long Short-Term Memory (LSTM) networks for real-time Indian sign language translation by modeling temporal dependencies between gesture frames. Similarly, in [29] was introduced a hierarchical LSTM architecture that captures both local gesture-level and global sentence-level temporal dependencies for continuous sign language translation. Several studies combined convolutional and recurrent architectures to improve spatiotemporal feature extraction. A hybrid CNN-RNN framework was proposed by [30] for American Sign Language recognition, where convolutional layers extract spatial gesture features and recurrent layers model temporal motion patterns. Likewise, Zhu et. al [31] employed 3D convolutional neural networks together with convolutional LSTM layers to jointly learn spatial and temporal representations from multimodal gesture data. To address continuous gesture segmentation and recognition, a proposed method in [32] combined 3D convolutional neural networks with ConvLSTM models for simultaneous gesture segmentation and classification from continuous video streams. Their method improved temporal consistency and reduced segmentation errors in dynamic gesture sequences. While Xiao et al. [33] also address continuous sign language neural machine translation through spatial-temporal LSTM-based fusion of visual information, SCOPE [34] extends this direction by explicitly incorporating conversational context and Large Language Model (LLM)-based embeddings to improve both gloss recognition and natural language translation.

A transfer learning framework developed by [35] used convolutional neural networks on surface electromyography (sEMG) signals for gesture recognition. Their approach leveraged pretrained CNN models to extract discriminative muscle activation features, improving recognition accuracy with limited training data.

Recent research has shifted toward more advanced architectures, particularly transformer-based models, which have demonstrated superior performance in modeling long-range temporal dependencies without the limitations of recurrent structures. Transformers are increasingly used in continuous gesture recognition tasks due to their scalability and effectiveness in processing sequential data [36]. At the same time, skeleton-based gesture recognition has emerged as a dominant direction, where human joints are estimated using pose estimation algorithms and represented as graphs. These skeleton representations are typically processed using graph convolutional networks (GCNs) [37], which capture both spatial relationships between joints and temporal motion patterns across frames. According to recent surveys, skeleton-based methods significantly improve robustness in complex environments such as occlusions, cluttered backgrounds, and varying illumination conditions [38]. For instance, in [39], [40] is shown that graph-based models outperform traditional RGB-based methods in dynamic gesture recognition due to their structural modelling of human motion.

In addition to skeleton-based methods, multimodal fusion has become an important trend in state-of-the-art systems. These approaches combine RGB video, depth maps, inertial measurement units (IMUs), and surface electromyography (sEMG) signals to improve robustness and reduce ambiguity in gesture interpretation. Multimodal systems

consistently outperform single-modality approaches, particularly in real-world applications where noise and occlusion are common [41], [42].

Another significant development is the optimization of gesture recognition models for real-time and edge computing environments. Lightweight architectures such as MobileNet [43], EfficientNet [44], and compressed transformer models [45] are widely used to reduce computational cost while maintaining high accuracy. This is particularly important in applications such as augmented reality (AR), virtual reality (VR), robotics, and wearable devices, where latency and power consumption are critical constraints.

Sign language communication systems using avatars have emerged as an important research direction for improving accessibility between deaf and hearing communities. Unlike traditional sign language recognition systems that only convert signs into text or speech, avatar-based systems [46], [47] focus on synthesizing sign language visually through animated virtual humans capable of performing both manual and non-manual signing components.

To bridge the gap between spoken text and avatar-based sign synthesis, multiple studies investigated gloss representation and intermediate translation systems. In [48] an XML-based gloss annotation framework was proposed for encoding sign language grammar and motion information in machine-readable form. Similarly, Aouiti et al [49] introduced an Arabic gloss representation system for Arabic sign language synthesis and translation pipelines. Avatar-based sign language generation systems also became increasingly important for accessibility applications. VirtualSign [50] proposed a bidirectional Portuguese sign language translation platform that combines speech/text processing with real-time avatar animation. Their system translates Portuguese text into animated sign language sequences using avatar-based rendering for interactive communication.

Early avatar systems primarily relied on rule-based animation and predefined motion sequences, which often resulted in unnatural movements and low comprehensibility [51]. Kipp et al. [46] discussed the challenges of creating understandable signing avatars and highlighted that many early systems lacked realistic facial expressions, prosody, and smooth motion transitions, limiting communication effectiveness. The authors proposed a gloss-based animation workflow and emphasized the importance of evaluating avatar comprehensibility against human signers. As the field progressed, researchers began focusing on the linguistic representation of sign language and the integration of computer animation techniques into communication systems. A Survey [47] provided a comprehensive overview of signing avatar synthesis approaches, including keyframe animation, procedural animation, and data-driven techniques. Their study showed that modern avatar systems increasingly rely on concatenative and motion-driven synthesis methods to generate more natural and expressive sign language utterances.

More recent work has shifted toward intelligent sign language translation pipelines that combine computer vision, machine translation, and avatar synthesis [52], [53]. Coster et al. [54] highlighted that modern sign language communication systems increasingly integrate deep learning-based recognition and translation models with 3D avatar generation frameworks. The authors emphasized that avatar-based synthesis plays a crucial role in spoken-to-sign translation systems because sign languages do not possess a universally accepted written representation, requiring visual rendering through animated avatars. Text2Sign [55] proposed a neural machine translation framework combined with generative

adversarial networks (GANs) to synthesize continuous sign language pose sequences from spoken language text. Their approach generates intermediate skeletal representations which are then rendered through signing avatars, improving motion continuity and realism compared to rule-based animation systems. Forte et al. proposed SGNify, a reconstruction framework that uses monocular RGB videos together with SMPL-X body modeling, inverse kinematics, and sign language linguistic priors to reconstruct expressive 3D signing avatars. The method combines 2D keypoint detection, 3D pose optimization, and hand articulation constraints to recover detailed body, hand, and facial movements from sign language videos. Recent research has increasingly focused on neural and diffusion-based avatar synthesis frameworks for sign language production. Baltatzis et al. [56] proposed a diffusion-based sign language production system trained on large-scale 4D signing avatar datasets. Their approach uses a graph neural network built on the SMPL-X skeleton to generate realistic 3D signing motions directly from text sequences. Another recent direction focuses on modeling both manual and non-manual signing components simultaneously. A transformer-based framework was introduced by [57] using discrete motion tokenization for body, hand, and facial movements. Their method combines SMPL-X with FLAME facial modelling and modality-specific vector quantization to generate linguistically accurate signing avatars with improved facial expression synthesis and grammatical non-manual markers. Lakhan et al. [58] proposed a latent diffusion model for generating photorealistic signer avatars. Their approach explicitly separates manual features (hands and body motion) from non-manual features (facial expressions and emotions) using a sign feature aggregation module, enabling more expressive and personalized avatar synthesis. Another personalized framework was introduced by [59] which combined diffusion-based framework with federated learning for privacy-preserving sign language production. Their method transforms textual input into temporally coherent skeleton sequences using cluster-specific diffusion models and motion-aware signer adaptation mechanisms. Low et al. [60], in their sparse keyframe-based sign language production framework proposed a sign segmentation and Conditional Flow Matching (CFM) to generate smooth 3D signing sequences in SMPL-X and MANO spaces. Their method synthesizes multilingual signing avatars from key motion anchors and employs 3D Gaussian Splatting for photorealistic avatar rendering. A real-time text-to-sign translation framework [61] employed neural sequence models and avatar rendering pipelines to generate animated sign language outputs for accessibility applications.

Recent advances in avatar-based sign language communication systems have increasingly focused on real-time neural rendering, multimodal interaction, and expressive conversational avatars for virtual communication and meeting environments. Deep learning translation pipelines are now commonly integrated with animated avatar systems to convert speech or text into continuous sign language sequences in real time. In [62], neural translation models are employed to transform spoken and textual Arabic input into animated avatar gestures, enabling smoother conversational interaction during live communication scenarios. Personalized convolutional neural networks (CNNs) are also employed to model non-manual signing components such as lip, cheek, and tongue movements, while UDP-based streaming enables low-latency avatar communication in immersive VR environments [63]. Picron et al [64] proposed a mobile sign language avatar communication platform combined with a user-centered evaluation methodology for accessibility applications. Their approach integrates avatar rendering and interaction assessment pipelines to evaluate comprehensibility, usability, and end-user

communication effectiveness in real-world scenarios. In [65], two complementary avatar synthesis systems were proposed, based on Behaviour Markup Language (BML), HamNoSys, SiGML, inverse kinematics, and machine learning-based webcam motion capture. Their method combines procedural avatar animation with ML-driven 3D motion reconstruction to generate scalable sign language animations without specialized motion-capture hardware. Zuo et al. [66] introduced an autoregressive multilingual sign language generation framework built on pretrained language models and discrete sign tokenization. Their method converts continuous body and hand motions into token sequences and uses transformer-based autoregressive generation to synthesize 3D signing avatars from text input. A mixed-reality communication framework [67] used generative AI avatars for deaf-hearing collaboration. Their approach integrates AI-generated signing, facial expression synthesis, voice synchronization, and avatar customization within immersive mixed-reality environments such as Apple Vision Pro. A Spoken2Sign translation pipeline [68] composed a Text2Gloss translator, a sign connector module, and a 3D avatar rendering system. Their method generates intermediate 3D sign representations from spoken language text and renders them through animated signing avatars using gloss-guided neural translation.

Table 1 and Table 2 summarize representative works identified in the literature concerning the two main addressed contexts: sign/gesture recognition (encompassing but not confined to human interpersonal communication), and avatar-based sign language communication systems.

Table 1 – Comparison of methods used in gesture and sign language recognition systems, including handcrafted, machine learning, and deep learning approaches.

Method	References	Input Modality	Main Techniques	Key Characteristics
Handcrafted feature-based methods	[18], [19], [20], [21], [22].	RGB Video.	HOG, Optical Flow, HMM.	Manual feature engineering with temporal probabilistic modelling.
Temporal probabilistic models	[18], [20], [22].	Feature sequences (from video)	Hidden Markov Models (HMM).	Sequence modeling using probabilistic state transitions.
CNN-based methods	[24], [30], [35], [44].	RGB / sEMG.	Convolutional feature learning.	Strong spatial representation learning.
RNN / LSTM-based methods	[27], [28], [29], [33].	Sequential video / labelled videos.	Recurrent temporal modelling; motion encoder.	Captures sequential dependencies in gestures. In [33], the interpretation of such dependencies produce sentences.
CNN-RNN Hybrid models	[31], [32].	RGB / multimodal video.	Spatiotemporal fusion.	Joint spatial-temporal feature learning.
Transformer-based methods	[35], [45], [34].	Skeleton / sequence / labelled videos.	Self-attention mechanism; LLM-tuned sentences.	Long-range dependency modelling; in [34], signs/words provide context for LLM-based sentence construction.
Skeleton-based GCN methods	[37], [39], [40].	Skeleton joints.	Graph convolution networks.	Structural body representation.
Multimodal learning methods	[42], [43].	RGB + IMU + sEMG.	Feature fusion / lightweight CNN.	Robust real-world recognition.

Table 2 – Comparison of avatar-based sign language communication systems.

Method	References	Pipeline Type	Avatar Technique	Key Characteristics
Rule-based/early avatar systems	[46], [50].	Text → predefined gestures.	Keyframe / procedural animation.	Limited expressiveness, rigid motion.
Gloss-based systems	[48], [49], [61], [68].	Text → Gloss → Sign.	Rule + structured translation.	Linguistically structured sign generation.
GAN-based systems	[55].	Text → Pose.	Generative adversarial networks.	Continuous motion synthesis.
Diffusion-based systems	[56], [58], [59].	Text → Motion.	Diffusion models.	High realism and personalized synthesis.
Transformer / token-based models	[57], [66].	Text → motion tokens.	Autoregressive transformers.	Multilingual scalable generation.
Keyframe / flow-based systems	[60].	Text → Keyframes → Motion.	Conditional Flow Matching + SMPL-X / MANO + keyframe learning.	Efficient multilingual sign language production with sparse motion anchors.

3 SLaDIS Proposal

This section presents the overall architecture of the proposed system, covering the requirements elicitation stage, the corresponding system specification, and the service-oriented design perspective that supports its modular integration.

3.1 Main requirements

The requirements of SLaDIS aim to establish the main goals for developing a digital platform that supports bidirectional communication between deaf and hearing users through automatic sign recognition, natural-language translation, and avatar-based sign language – namely, Portuguese sign language, officially known as *Linguagem Gestual Portuguesa* or LGP – visualization. To enable such functional diversity, the system is designed in modules combining computer vision, AI, NLP, 3D avatar visualization, a mobile frontend, and support for future extensibility across different sign languages.

3.1.1 Functional requirements

The functional requirements define the main capabilities required for SLaDIS to support bidirectional deaf–hearing communication through gesture recognition, translation, avatar-based visualization, and frontend interaction, and are listed as follows:

- Bidirectional communication support: the system shall support communication from LGP to spoken/written Portuguese and from Portuguese to LGP, enabling interaction between deaf and hearing users.
 - Deaf sign capture: the system shall capture the movements of users communicating in LGP using video cameras available on mobile devices or dedicated equipment.
 - Sign recognition and interpretation: the system shall process captured visual data and recognize LGP signs using computer vision and AI-based methods, considering arms/hands/fingers movements and facial expressions.
 - LGP-to-natural-language conversion: the system shall convert recognized sets of LGP signs into natural-language text and/or audio understandable by hearing users.
 - Natural-language-to-LGP conversion: the system shall process natural Portuguese language and map it into LGP-compatible glosses.
 - Avatar-based LGP visualization: the system shall provide a 3D avatar capable of rendering LGP signs, including arms/hands/fingers, facial, and body expression animations.
- Mobile and fixed-device access: the system shall allow access through mobile devices and fixed equipment, supporting a front-end application.
 - Hearing-user interface (HUI): the system shall provide an interface where hearing users can receive translated content, read text, and send responses.
 - Deaf-user interface (DUI): the deaf-user interface shall provide visual cues to support user awareness during LGP sign formulation, gloss validation, sentence validation, and message-sending confirmation. It shall also

include an intuitive avatar-based viewport, enabling deaf users to receive natural-language messages translated into animated LGP gestures.

- Service integration: the system shall integrate heterogeneous modules, including computer vision, translation, avatar rendering, frontend interaction, and backend services, through explicit communication interfaces.

3.1.2 Non-functional requirements

The non-functional requirements define the quality attributes that guide SLaDIS operation, and are glimpsed next:

- Near real-time operation: the system should support low-latency interaction suitable for conversational communication between deaf and hearing users.
- Modularity: the system should be decomposed into independent components or services, allowing the recognition, translation, avatar rendering, frontend, and backend layers to evolve separately.
- Scalability: the system should support the progressive addition of new signs, glosses, vocabulary, inference models, and communication scenarios while preserving modular independence, so that extensions can be introduced with minimal or no impact on unaffected system components.
- Interoperability: the system should expose communication interfaces that allow internal modules and external third parties to exchange information consistently.
- Multi-device compatibility: the system should operate on smartphones and fixed setups, potentially involving conventional cameras, depth cameras, screens, and dedicated equipment.
- Context-aware interface adaptation: the user interfaces (UI) should be adequately adjusted to each user profile (deaf and hearing), encompassing intuitiveness.
- Maintainability: the system should support updates to individual components, such as recognition models, translation models, animation libraries, or frontend interfaces, without requiring full system redesign (in line with the scalability requirement).
- Performance awareness-driven design: architecture should consider the computational cost of video processing, AI inference, translation, animation retrieval, and avatar rendering, especially in communication scenarios requiring low response times.

3.2 Service-oriented General Architecture

The proposed architecture for SLaDIS is organized into three main modules: the Sign Language Interpretation Module (SLIM), the Avatar-Generated Sign Language Module (ASLM), and the Chatroom Frontend Module (CFM), as shown in Figure 1.

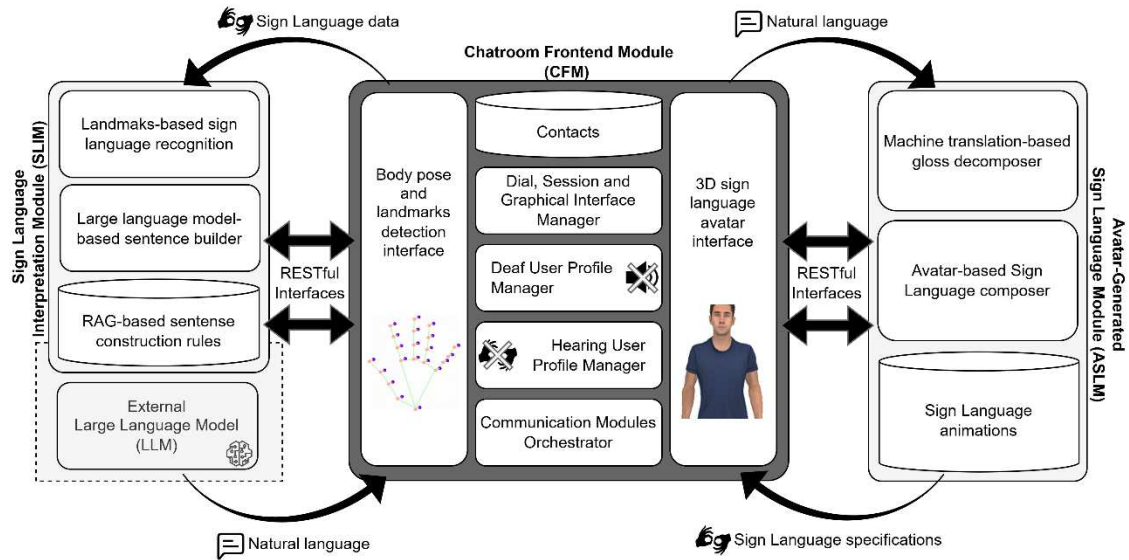


Figure 1 – SLaDIS architecture proposal, composed of 3 main modules (from left to right): the Sign Language Interpretation Module (SLIM), the Avatar-Generated Sign Language Module (ASLM).

SLIM supports the LGP-to-natural-language flow by processing landmark-based sign data, inferring signs/glosses, and constructing natural-language sentences with the support of language-model-based mechanisms. ASLM supports the opposite flow, converting natural-language input into LGP glosses and rendering them through avatar-based sign language animations. The CFM operates as the frontend and orchestration layer, managing user interaction, sessions, profiles, landmark acquisition, and avatar visualization. The modules communicate through RESTful interfaces, reinforcing a decoupled service-oriented architecture that integrates heterogeneous AI, NLP, and visual computing components into a unified bidirectional communication system.

3.3 System Specification

A set of complementary modeling artifacts is used to provide a more detailed specification of SLaDIS. These include use case diagrams, sequence diagrams, and process flowcharts, which together describe the main user interactions, service communication mechanisms, and operational workflows supporting bidirectional deaf-hearing communication.

For starters, Figure 2 presents the main SLaDIS use cases for deaf and hearing users. Both profiles access the system through a virtual session after login. Deaf users can compose messages using LGP signs and consult received natural-language messages either as text or through avatar-based graphical rendering. Hearing users can compose messages in natural language and consult received messages by reading them or through text-to-speech playback. In both cases, users may confirm message submission or clear/cancel the message during composition.

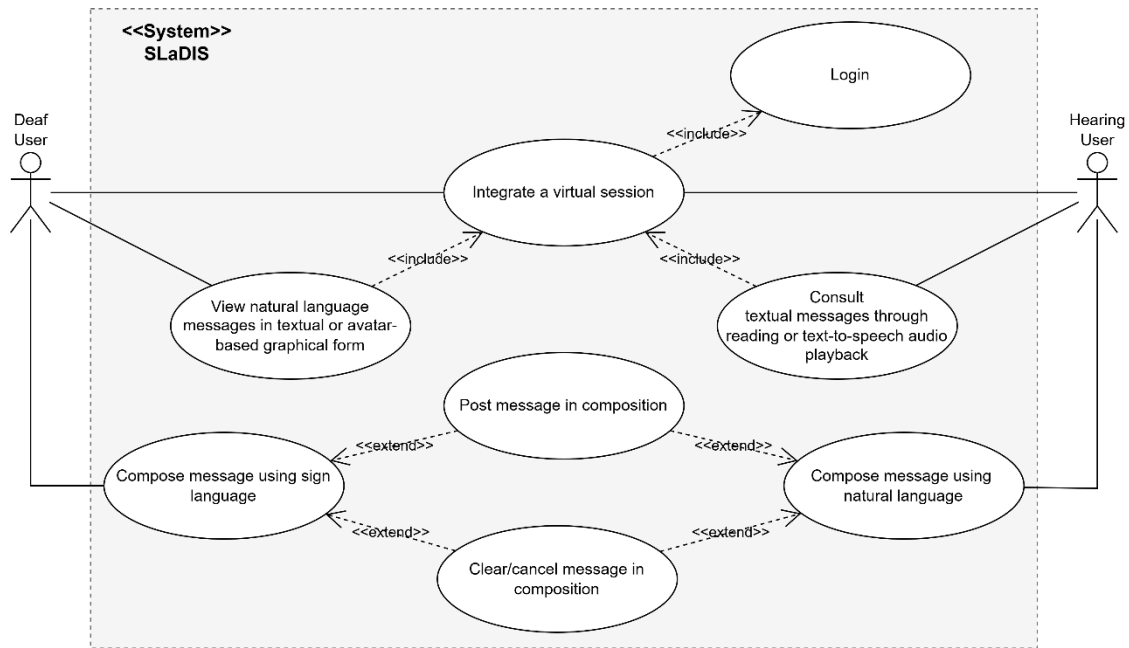


Figure 2 – General use-case for Deaf and Hearing users, depicting functionalities inherently associated to each profile.

A simplified sequence diagram representing deaf-user operational context is presented in Figure 3, adapted from [8]. The hearing-impaired user expresses a message through LGP signs, which are captured as camera frames and processed by the pose analyzer to extract body and facial keypoints. Once a sufficient buffer is collected, the sign language interpreter converts the gesture data into gloss-like units, which are then used by the sentence builder to generate a natural-language message proposal. The user can then visually validate the proposal and either cancel or confirm message submission.

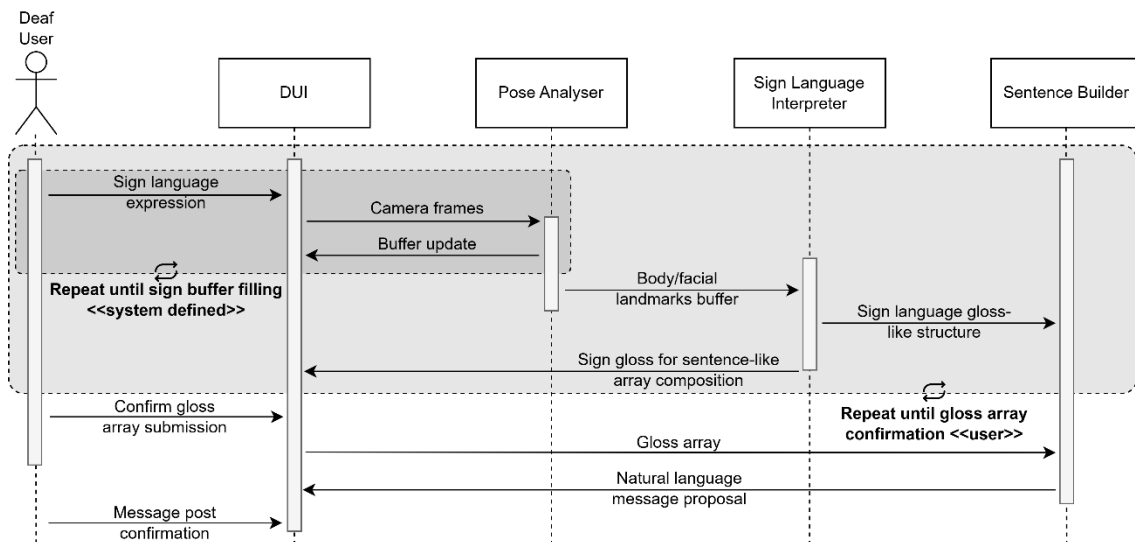


Figure 3 – Deaf-side sequence diagram from message composition to post confirmation.

Figure 4 presents a set of gesture recognition workflows, adapted from [8]. This specific process starts by initializing an inactivity timer and a body/facial landmark buffer. Camera frames are continuously analysed to detect skeletal keypoints, which are added to the buffer whenever valid landmark data are available. If no keypoints are detected for longer than the defined tolerance time, both the buffer and the inactivity timer are reset. Once the

buffer contains enough landmark sets, it is sent to the AI-based sign language interpretation model (Figure 4-a), which infers the corresponding LGP gloss (Figure 4-b). When a sufficient set of glosses has been accumulated to compose a sentence, they are sent to an LLM-based sentence builder, together with LGP-oriented generation rules (Figure 4-c). This component returns a natural language sentence that can be interpreted by the hearing user.

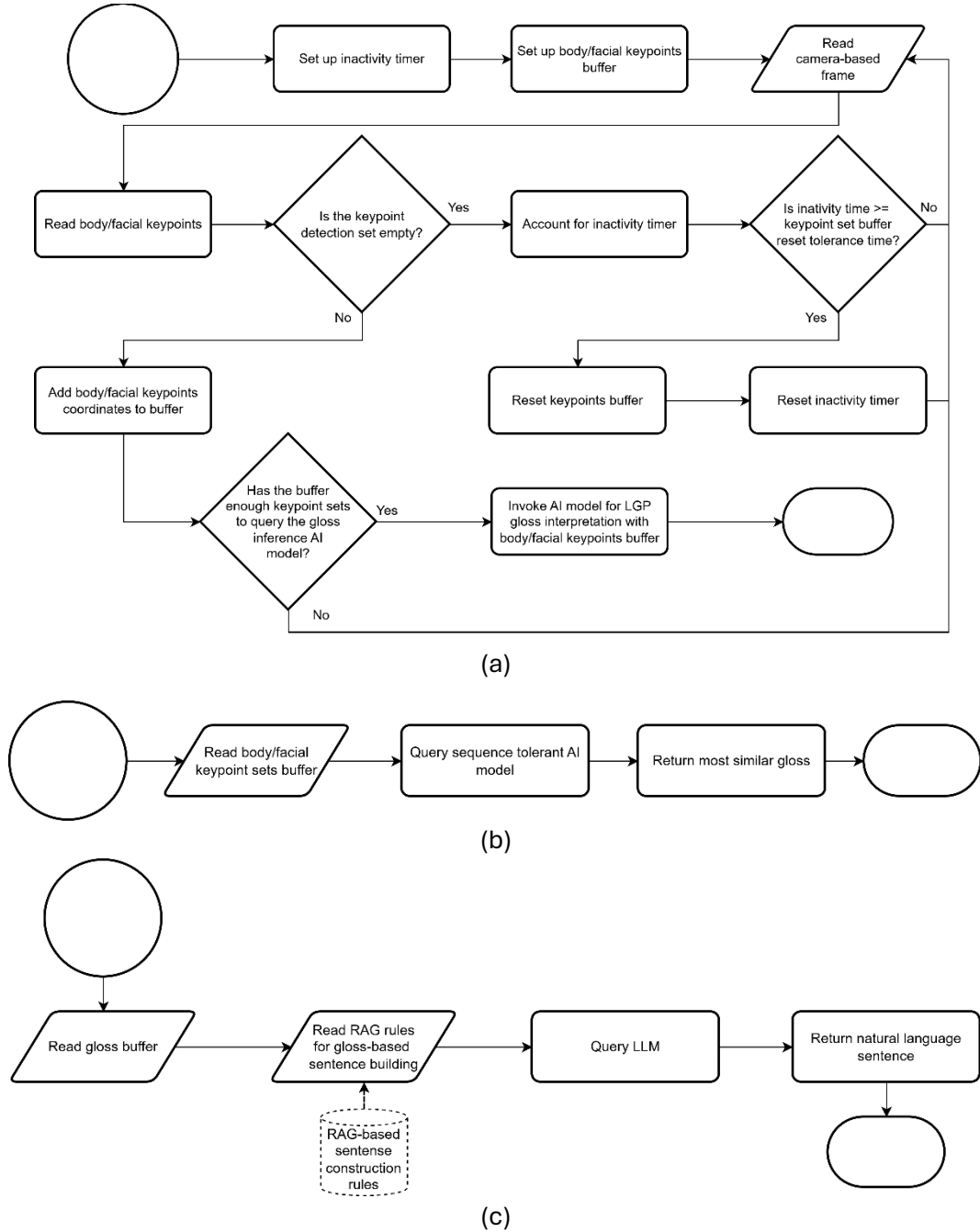


Figure 4 – Flow diagrams depicting the LGP-to-natural-language interpretation pipeline: a) body/facial keypoints sign queuing; (b) keypoints-to-gloss inference performed by a specialized AI model; and (c) gloss set conversion into a natural-language sentence using a RAG-enabled LLM.

On the hearing-user side, messaging works as depicted by the diagram of Figure 5. A user with this profile may compose a message through text or speech and confirms its submission in the interface. The natural-language input is then sent to the translation component, which decomposes it into LGP glosses. These glosses are forwarded to the

avatar-based sign language engine, which queries the animation repository, retrieves the corresponding gesture animations, and renders the synthesized sign-language message in the DUI. Although the rendered output is available within the interface, it is primarily intended to make the hearing user's message accessible to the deaf user.

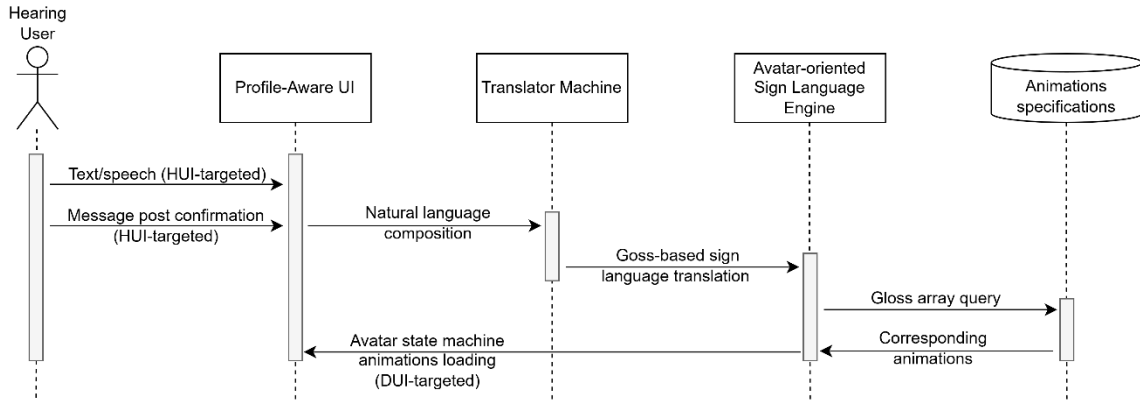


Figure 5 – Hearing-side sequence diagram.

The natural-language-to-avatar-based LGP communication workflow is Figure 6 illustrates the process through which messages produced by hearing users are converted into avatar-rendered LGP signs. When a hearing user sends a message to a deaf user, the natural-language content is forwarded to a translation service, where it is morphologically analysed, adapted according to LGP linguistic constraints, normalized, and optionally refined through an encoder–decoder neural network. The resulting LGP glosses are then returned to the main application (Figure 6-a). Subsequently, these glosses are used to query the animation retrieval service, which provides the corresponding gesture animations. The selected animations are loaded into the 3D avatar state machine and rendered locally through viewport-like component, allowing the deaf user to receive the original message as avatar-based LGP signs (Figure 6-b).

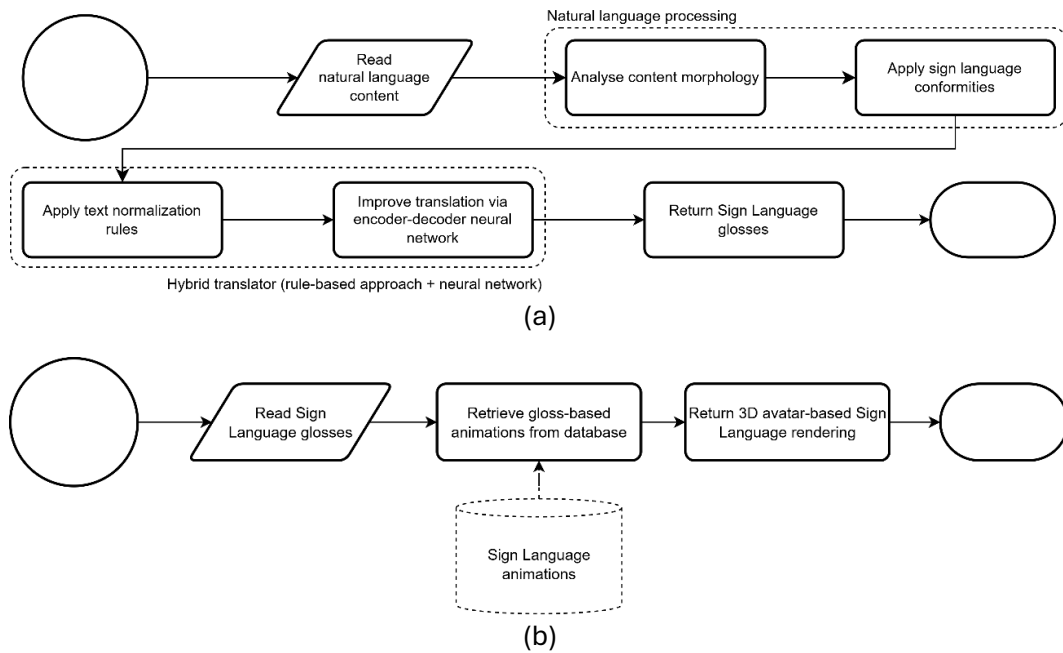


Figure 6 - Flow diagrams depicting the natural-language-to-avatar-based LGP communication pipeline: a) natural-language-to-LGP gloss conversion through a hybrid translator combining rule-based linguistic

processing and encoder-decoder neural enhancement; and b) gloss-driven retrieval of sign animations and their local rendering through a 3D avatar.

Having defined the system requirements, overall architecture, and functional specification of SLaDIS, the next section focuses on its implementation, providing details on the technological artifacts that materialize the proposed specification as a whole solution.

4 SLaDIS implementation

This section presents the implementation of the SLIM, the ASLM, and the CFM, within the context of Portuguese Sign Language (LGP) as case study. It also describes how the SLIM and ASLM are encapsulated as service-based components to support their seamless integration within the CFM.

4.1 SLIM and Services Implementation

Inspired in [8], the SLIM is responsible for converting continuous LGP expressions into natural-language content understandable by hearing users. The stages that supported its development involve: raw data acquisition, dataset construction, machine learning for sign language training based in anatomic keypoint sets, and LLM interfacing.

At the acquisition level, labeled videos of dynamic LGP signs were recorded with the support of LGP experts and collaborators of the Portuguese Association of the Deaf (officially known as *Associação Portuguesa de Surdos* or APS), resulting in a dataset of 50 words/terms, each represented by 33 examples, for a total of 1650 videos. For model development, five examples per sign were reserved for testing and the remaining 28 were used for training/validation.

The preprocessing and training pipeline is based on anatomical keypoints rather than raw RGB frames, as the former significantly reduces input dimensionality, computational requirements, as well as bandwidth demands – which is critical for enabling near real-time operation in RESTful service-based architectures. The first processing steps involve augmenting video samples through shear transformations and horizontal flipping, followed by the extraction of 88 anatomical landmarks using MediaPipe¹, covering torso, arms, hands, and facial regions, resulting in 176 xy keypoints per frame. With regard to the combination of landmark data processing and modelling strategies, the best-performing configuration employed point-based augmentation through random radial shifts, followed by sequence length normalization to ensure consistency across samples, and was combined with a hybrid Convolutional LSTM architecture. Prepared to handle 10 sequences of keypoint sets at time, this configuration achieved an accuracy of 95.6%, supporting the feasibility of a gloss-like sign-to-word inference pipeline in near real-time conditions.

To accommodate the transformation of the buffer-accumulated inferred glosses, the module delegates sentence construction to an LLM-based component. Therefore, ChatGPT 3.5² was used through an application programming interface (API). The model receives the inferred vocabulary together with a rule set designed to constrain generation. The dynamic rule instructs the model to generate a concise sentence from the provided token set,

¹ Google. “MediaPipe Solutions guide.” Google AI Edge. Available at: <https://ai.google.dev/edge/mediapipe/solutions/guide> (Accessed: 22 June 2026).

² OpenAI. “Introducing ChatGPT.” OpenAI, 30 November 2022. Available at: <https://openai.com/index/chatgpt/> (Accessed: 22 June 2026).

optionally considering punctuation. The static rules include ignoring repeated words, minimizing the introduction of new words, assuming present indicative by default, conjugating in first-person singular when no pronoun is given, performing only minimal lexical transformations, correcting spelling, resolving gender/number agreement, and interpreting numbers as quantifiers. The result is a natural-language sentence proposal that is semantically and grammatically more meaningful to a person that has no experience in sign language than the original word cloud.

The service layer that isolates and exposes these capabilities was implemented as a centralized web service using Python-based FastAPI³, deployed in a asynchronous server gateway Interface (ASGI) known as Uvicorn. This layer acts as the integration point between a third-party frontend, the trained TensorFlow/Keras⁴ LSTM-based models, and the external ChatGPT API. After startup and model loading, the service exposes endpoints such as:

- config – to provide expectable settings for MediaPipe landmark formats;
- mp_estimator – to receive landmark coordinates and return predicted terms/expressions;
- and sentence_GPT – to transform an input list of words into a semantically and grammatically consistent sentence through ChatGPT.

In practice, this service layer decouples the application-facing communication logic from the underlying inference and generation models, making the interpretation module consumable by an independent frontend or mobile application.

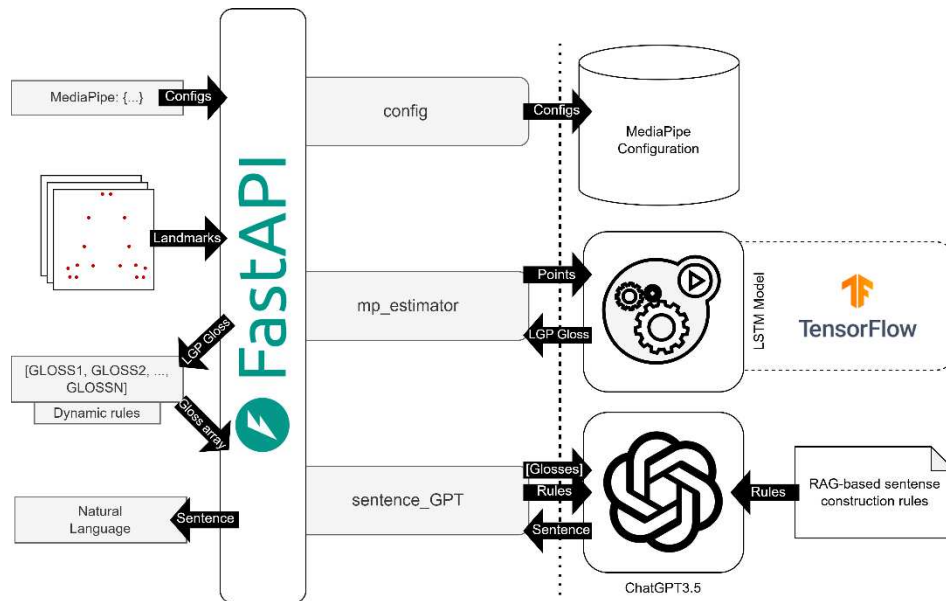


Figure 7 – Technological architecture of SLIM, aggregating LSTM-based LGP sign estimators and LLM-based sentence building services.

4.2 ASLM and Services Implementation

The Avatar-Generated Sign Language Module (ASLM) – based on [10] – Is responsible for transforming natural language input into LGP through a 3D avatar. To that end, an intensive preparatory stage involving building a natural language to LGP translator, gloss-based LGP

³ FastAPI. “FastAPI.” FastAPI. Available at: <https://fastapi.tiangolo.com/> (Accessed: 22 June 2026).

⁴ TensorFlow. “tf | TensorFlow v2.5.0.” TensorFlow. Available at: https://www.tensorflow.org/versions/r2.5/api_docs/python/tf (Accessed: 22 June 2026).

signs motion capture for setting a database and LGP-enhanced 3D avatar implementation. This allowed to consolidate a suitable base for the implementation of a complete pipeline for bridging the communication from the hearing to the deaf user, namely, natural language processing, LGP gloss animation database retrieval and processing and the respective rendering of it through an Avatar-embedded environment.

4.2.1 Natural language to LGP translation

The translation component is implemented through a hybrid pipeline combining rule-based linguistic processing and neural machine translation, trained with several sentences mapped into LGP glosses. In the rule-based component, spaCy⁵ and Label_Lex_SW⁶ rules are employed to process input sentences, which are morphosyntactically analysed and restructured from Subject–Verb–Object (SVO) to Object–Subject–Verb (OSV) order, following LGP grammar rules, while applying transformations such as article removal, temporal adverb repositioning, number normalization, and articulation of gender-related nuances. The resulting representation may be used as the final translation or may be processed by a sequence-to-sequence LSTM-based neural model trained using an encoder-decoder architecture with embedding and time-distributed layers, aimed at translation enhancement purposes – powered by Keras⁷ framework. To improve generalization in a limited dataset scenario, named entities and numerical values are replaced with abstract tokens during training. The final system adopts a hybrid strategy, where the rule-based output provides a robust primary translation, and the neural network model is instrumentalized for refinement purposes. The translation service is exposed via a RESTful interface, returning both rule-based and neural-enhanced outputs in JSON format.

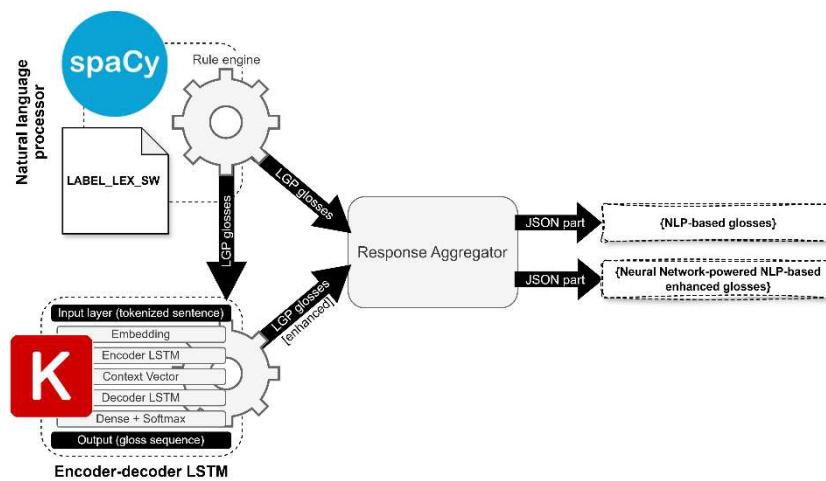


Figure 8 – Technological architecture of the Hybrid Translator, responsible for converting natural language to LGP glosses.

At this stage, both outputs are returned by the Hybrid Translator because the neural component is still being progressively trained with an expanded dataset within a continuous improvement/continuous development (CI/CD)-oriented workflow; therefore, the most

⁵ Explosion. “spaCy · Industrial-strength Natural Language Processing in Python.” spaCy. Available at: <https://spacy.io/> (Accessed: 22 June 2026).

⁶ LabEL. “LABEL-LEX.” LabEL, Instituto Superior Técnico. Available at: https://label.ist.utl.pt/en/labellex_en.php (Accessed: 22 June 2026).

⁷ Keras. “Keras: Deep Learning for humans.” Keras. Available at: <https://keras.io/> (Accessed: 22 June 2026).

appropriate translation output may be selected until a smooth transition toward the neural model becomes possible, as its performance becomes consistently superior to the rule-based baseline.

4.2.2 Decoupled LGP-Enhanced 3D Avatar Component

The ASLM relies on a motion-capture-based pipeline to build a reusable library of LGP gesture animations and render them through a 3D avatar. Since manually animating each gloss would be time-consuming and difficult to scale, the system adopts motion capture as the primary strategy for acquiring sign animations. Two motion capture systems were used to record gestures performed by an LGP speaker: Rokoko Studio⁸ and Perception Neuron's Axis Studio⁹. Rokoko relies on a sensor-based suit placed at key body points to capture position and rotation data, and also supports facial tracking through an Apple iPhone, converting facial motion into the standard 52 ARKit blend shapes¹⁰. Axis Studio provides a complementary motion capture setup, using body-worn sensors attached through straps. Both systems generate realistic body motion data in short acquisition sessions, enabling the creation of a large gesture animation library.

Because Rokoko and Axis Studio process motion data using different skeleton structures, a normalization and retargeting stage is required before the captured animations can be reused by a common avatar. To address this issue, the pipeline uses Autodesk Maya HumanIK¹¹, an inverse kinematics middleware that transfers animation data between humanoid skeletons with compatible structures. This step enables motion data captured from different systems to be mapped into a single standardized humanoid skeleton, ensuring that all animations can be applied consistently to the same avatar rig.

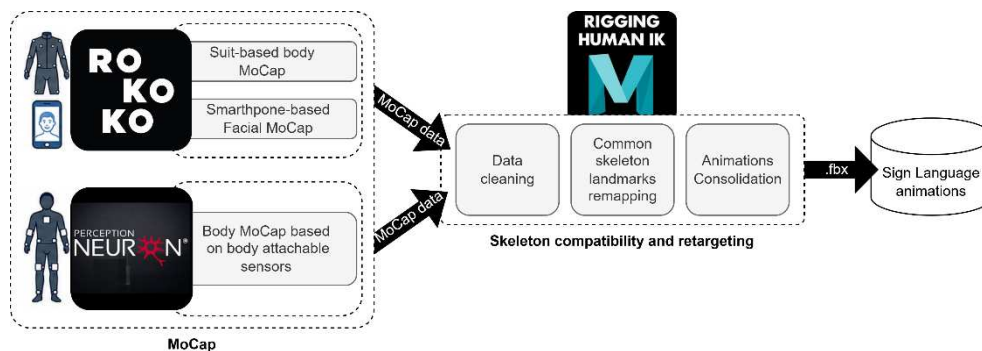


Figure 9 – General motion capture process resorting to Rokoko and Perception Neuron solution, uniformized by Maya's Rigging Human Inverse Kinematics before storage as Avatar-compatible animation.

The 3D avatar used for gesture synthesis was created using iClone 8¹² and Character Creator 4¹³. This tool chain allows the creation of multiple avatars with different meshes

⁸ Rokoko. "Rokoko Studio: one software for all your mocap & animations." Rokoko. Available at: <https://www.rokoko.com/products/studio> (Accessed: 22 June 2026).

⁹ Noitom. "Perception Neuron | Axis Studio Motion Capture Software." Noitom. Available at: <https://en.noitom.com.cn/axis-studio.html> (Accessed: 22 June 2026).

¹⁰ Apple. "ARKit." Apple Developer Documentation. Available at: <https://developer.apple.com/documentation/arkit> (Accessed: 22 June 2026).

¹¹ Autodesk. "HumanIK." Maya Help. Available at: <https://help.autodesk.com/view/MAYAUL/2024/ENU/index.html?guid=GUID-EDBDA3DB-4715-40EF-9ADF-412F78BFF98E> (Accessed: 22 June 2026).

¹² Reallusion. "3D Character Animation Software | iClone." Reallusion. Available at: <https://www.reallusion.com/iclone/> (Accessed: 22 June 2026).

¹³ Reallusion. "Character Creator – 3D Character Design & Animation Software." Reallusion. Available at: <https://www.reallusion.com/character-creator/> (Accessed: 22 June 2026).

while preserving the same underlying skeletal hierarchy. As a result, the same animation library can be reused across different avatar appearances without re-recording the gestures. This is important for scalability, since avatar customization becomes independent from the gesture acquisition process.

At runtime, gesture animation is handled in Unity¹⁴ through its animation system and state-machine logic. The Unity state machine includes an idle state and two animation states used alternately to play gloss animations. This design allows the system to animate sequences with an arbitrary number of glosses: while one state is playing a gesture, the next animation can be assigned to the other state. To avoid abrupt pauses between gestures, Unity's *ExitTime* is used so that the next animation starts before the current one fully ends, by default after approximately 75% of the current animation has been played. Unity then blends both animations by interpolating the position and rotation values of the avatar's skeleton bones, producing smoother transitions between consecutive LGP signs.

In operational terms, the ASLM receives a sequence of LGP glosses, maps each gloss to its corresponding animation asset, loads the required animations, and sequences them through the Unity-based avatar controller. Thus, the module transforms a symbolic gloss representation into an avatar-based visual output, enabling natural-language messages previously translated into LGP glosses to be rendered as understandable sign language gestures.

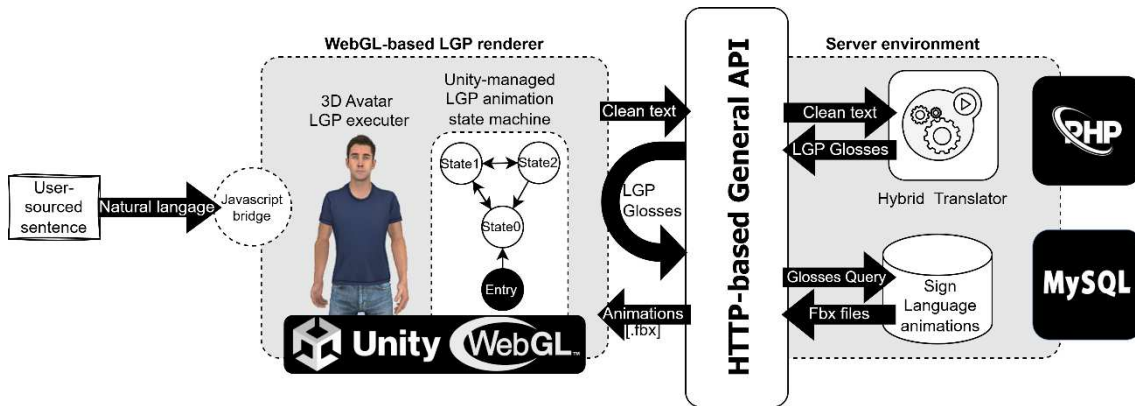


Figure 10 – Technological architecture of the LGP sign-making 3D Avatar rendering component.

Although developed within the scope of the ASLM, the Unity-based avatar component is deployed in SLADIS as a decoupled WebGL¹⁵ component, designed to be embedded and controlled by the CFM. This allows the avatar to locally render the gesture sequences associated with the received glosses, using the corresponding FBX animation assets retrieved through the service layer. By shifting the rendering process to the client side, the system avoids more burdensome and less responsive alternatives, such as remotely generating and transmitting pre-rendered avatar videos for each message. This reduces server-side processing and data transfer overheads, contributing to shorter response times in near real-time communication scenarios.

¹⁴ Unity Technologies. "Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine." *Unity*. Available at: <https://unity.com/> (Accessed: 22 June 2026).

¹⁵ Unity Technologies. "WebGL." *Unity User Manual*. Available at: <https://docs.unity3d.com/Manual/webgl.html> (Accessed: 22 June 2026).

4.3 CFM Implementation and Services Integration

The CFM is consolidated through a mobile application developed with Ionic, providing the interaction layer through which deaf and hearing users access the SLaDIS communication workflow. The application supports different user profiles, namely LGP and LP users, and provides communication modes such as asynchronous chat and real-time calls.

Real-time audiovisual communication is supported through OpenVidu¹⁶, which provides bidirectional video communication between devices using WebRTC¹⁷. In call-based interaction, participants can publish and receive media streams, enabling the visual capture of LGP communication and conventional audio/video communication between users. In parallel, the CFM communicates with backend services through REST and XMPP-based mechanisms¹⁸, which are used to coordinate/articulate with the different translation, recognition, and rendering services composing the system.

For LGP-to-LP communication, the CFM integrates a MediaPipe-based capture component that extracts relevant body landmarks from the user's video stream, including hands, face, shoulders, and elbows. These landmarks support the gesture interpretation workflow associated with SLIM, enabling the system to derive textual/natural-language representations from LGP input. For LP-to-LGP communication, the avatar component is embedded in the application as a WebGL component – originally developed within the ASLM scope and adapted to become a decoupled artifact to be embedded into CFM –, while maintaining independent communication with the server-side translation and animation-provisioning services. This enables natural-language input to be translated into LGP glosses and rendered as avatar-based gestures through the ASLM.

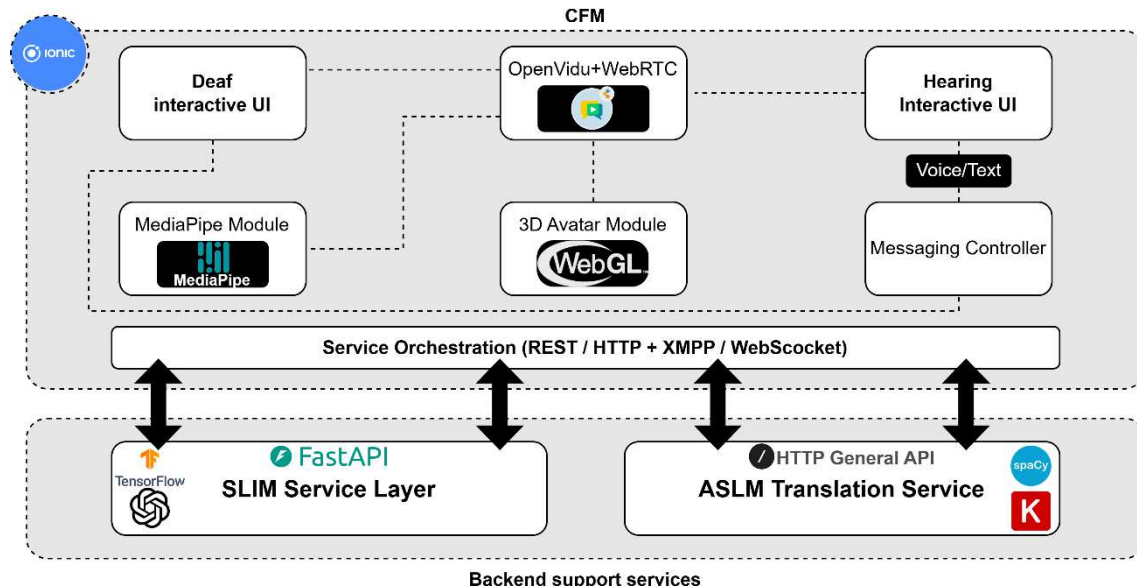


Figure 11 – CFM technological architecture, with backend orchestration.

¹⁶ OpenVidu. "OpenVidu: Self-hosted videoconferencing and custom real-time video applications." OpenVidu. Available at: <https://openvidu.io/> (Accessed: 22 June 2026).

¹⁷ WebRTC. "WebRTC." WebRTC. Available at: <https://webrtc.org/> (Accessed: 22 June 2026).

¹⁸ XMPP Standards Foundation. "XMPP — The universal messaging standard." XMPP. Available at: <https://xmpp.org/> (Accessed: 22 June 2026).

Summing up, rather than acting only as a video communication interface, the CFM operates as the orchestration layer that connects user interaction, real-time media exchange, gesture capture, translation services, and avatar-based rendering. This combination of WebRTC-based streaming with REST/XMPP service communication enables the integration of heterogeneous AI-driven modules into a unified conversational workflow.

The next section presents and discusses the main results obtained from the three implemented modules, focusing on their empirical outcomes, derived artifacts, and operational evidence.

5 Main results and discussion

This section presents the main empirical and visual results obtained from the development and integration of the SLaDIS modules. For each component, the most relevant outputs are reported, including quantitative performance indicators, derived artifacts, and illustrative examples of the modules/system in operation. Some of these results have already been reported in previous works [8], [10], [11], [69]; however, they are revisited and consolidated here to characterize the operational performance of each module and to support the discussion of the complete service-oriented system. Contributions and limitations of the proposed solution are also encompassed.

5.1 SLIM-related results and derived artifacts

The SLIM evaluation was supported by a prototype software artifact designed to reproduce the complete gesture-to-sentence workflow foreseen in SLaDIS. This prototype consists of a web-based interface integrating MediaPipe for landmark extraction and service interfaces for gesture recognition and sentence construction. In particular, the interface connects to AI-based gesture recognition services, including the best-performing ConvLSTM model, and to an LLM-driven sentence construction component responsible for transforming recognized gloss-like terms into natural-language outputs. The prototype interface is illustrated in Figure 12.

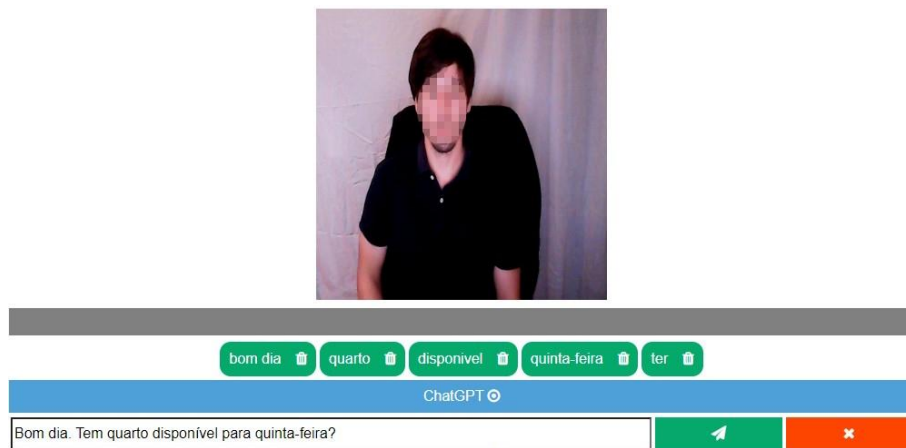


Figure 12 – Standalone prototype dedicated to test LGP signs to natural language translation (derived from [8]).

Using this interface, the gesture recognition component was first evaluated independently. As reported in [8], the best-performing ConvLSTM configuration achieved an accuracy of 95.60% over a set of 50 LGP signs, with each sign evaluated using five test videos. To recall the results in a compact and readable form, the 50 signs were grouped into five sets of ten

signs, as detailed in Table 3. Based on this grouping, Figure 13 presents the corresponding confusion matrix, summarizing the classification behavior of the model while avoiding an excessively dense 50-class visualization.

Table 3 – Grouping of the LGP words used for ConvLSTM training, with their English translations, into sets of ten words (rearranged from [8]).

Group	English-translated Terms
Group 1	1/one, 2/two, 3/three, blue, drink, good morning, hat, code, discount, since
Group 2	sports, available, migraine, I/me, try/experiment, weekend, chicken, fresh, big/large, exist/have
Group 3	hotel, internet, take/carry, show, very good, no/not, thank you, countries, bread, write prescription
Group 4	can/be able to, search/look for, when, how much does it cost, room, want, Thursday, book/reserve, leave/go out, sandals
Group 5	sardines, feel, yes, t-shirt, call/phone, have, take pill, you, all good, wine

The computational performance of the SLIM inference model was also assessed to verify its suitability for near real-time operation. Using the same computational environment reported in [8], specific tests were made to the trained LSTM-based model operating over ten independent sets of 25 LGP gesture samples to assess computational costs. As shown in Figure 14, the total inference time for each set remained below 0.12 s in all cases, with a global average of 0.095 s. This corresponds to an average inference time of approximately 0.004 s per LGP sign, indicating that the gesture recognition component support near real-time interaction.

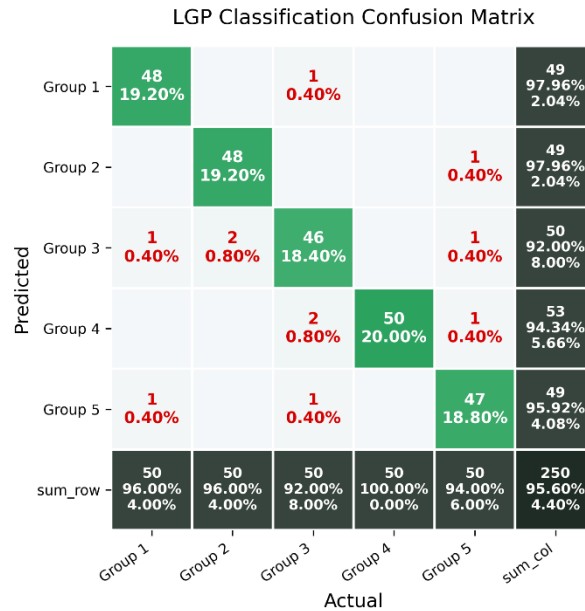


Figure 13 – ConvLSTM confusion matrix, obtained from test set inference (adapted from [8]).

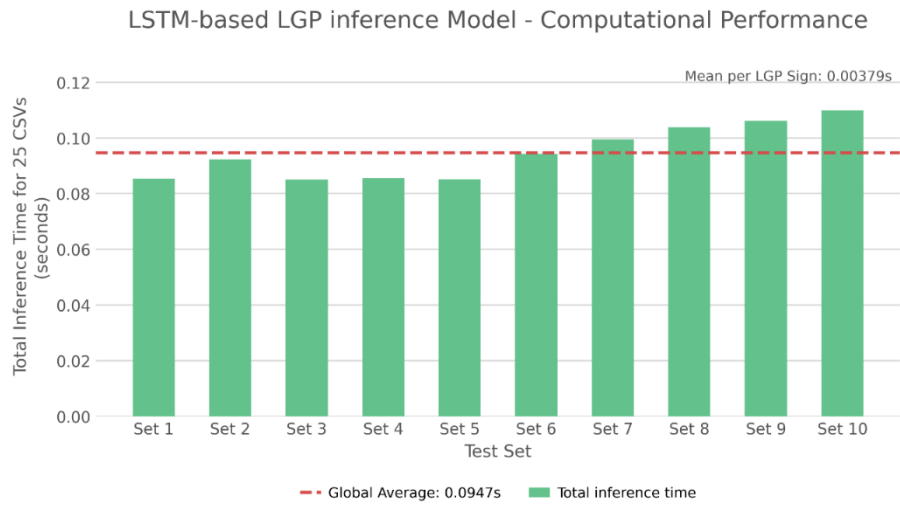


Figure 14 – Computational performance of the LSTM-based LGP inference model over ten test sets of 25 gesture samples.

Subsequently, the recognized signs were used to compose sets of gloss-like terms, which were submitted to a ChatGPT-3.5-based sentence construction component together with LGP-oriented generation rules. This component aimed to produce natural-language sentences understandable by hearing users. The resulting sentences were evaluated against expected reference sentences using the Universal Sentence Encoder [70], through an embedding-based semantic similarity metric. Across 19 assessed sentences covering commerce, tourism, and healthcare contexts, the system achieved an average semantic similarity score of 0.81, as previously reported in [8] and reformatted in Table 4.

Table 4 - Semantic similarity results for LGP gloss-to-natural-language sentence generation across the three assessed communication contexts – Commerce, Tourism and Healthcare (rearranged from [8]).

Context	LGP assessed sentences	Average semantic similarity
Commerce	7	0.797
Tourism	9	0.772
Healthcare	3	0.970
Overall	19	0.810

The computational performance of the LLM-based sentence construction service was also assessed, since this component is responsible for transforming accumulated LGP gloss-like terms into natural-language sentence proposals. The tests were conducted on the same computational environment reported in [8], using a 4G Internet connection provided by a Samsung M21 running Android 10, which are relatively modest specifications. To monitor the stability of the communication conditions, two Ookla Speedtest measurements were performed: one before and another after the LLM interaction tests. As summarized in Table 5, the observed variations were minimal, indicating that the network conditions remained sufficiently stable during the experiment.

Table 5 – Network conditions measured before and after the LLM-based sentence construction performance experiment using a 4G mobile hotspot connection.

Moment	Network test tool	Access type / Operator	Test server and location	Download speed	Upload speed	Idle latency	Latency under download	Latency under upload
Before experiment	Ookla Speedtest	4G mobile hotspot / NOS	MEO, Gaia, Portugal	39.28 Mbps	27.83 Mbps	20 ms	188 ms	288 ms
After experiment	Ookla Speedtest	4G mobile hotspot / NOS	MEO, Gaia, Portugal	33.46 Mbps	27.14 Mbps	19 ms	173 ms	280 ms

The LLM performance test itself consisted of three test sets, in which sentence construction requests were submitted through the API while varying the number of input glosses from 1 to 10, reproducing the interaction pattern between SLIM and the OpenAI service. As shown in Figure 15, most response times remained below 1 s, suggesting that the sentence construction component based in such LLM service introduces a limited delay into the overall communication workflow. These results support its adequacy for near real-time interaction scenarios, particularly when combined with the low computational cost previously observed for the LGP sign inference model.

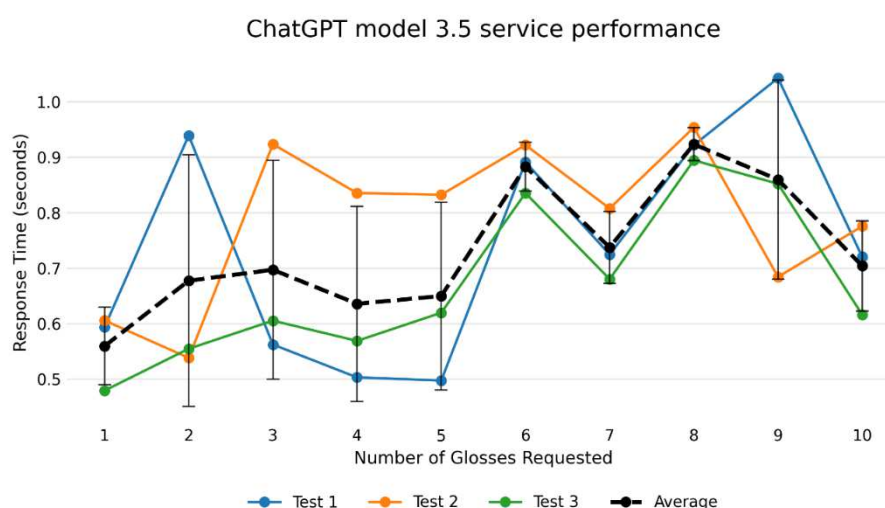


Figure 15 – ChatGPT 3.5 service response time for sentence construction requests with 1 to 10 LGP glosses, showing three test runs and their average response time.

A comparison of the SLIM results with works from literature encompassing technologies for sign language and, also, gesture recognition is summed up in Table 6. This comparison involves several factors from dataset context, acquisition and processing to utilized AI architectures, culmination in the main reported results.

Table 6 – Comparison of SLIM with relevant works from literature.

Work	Dataset	N.º of signs	Data acquisition	Dataset division	Test size	Augm. strategy	AI architectures	Main Results
SLaDIS / SLIM	LGP Dataset (proposed).	50 (dynamic).	1650 videos.	65% train / 35% validation.	250 videos.	Video variants + keypoint noise induction.	LSTM + RAG-LLM.	LGP Dataset → 95.6% accuracy (recognition), 0.81 semantic similarity.

[27]	BirthSIM obstetrical forceps dataset (instrumented laparoscopic forceps, 3D trajectories).	4 gestures (Right/Left × Novice/Junior).	392 trajectories.	Not explicitly specified in paper (training/validation split not clearly defined).	Not identified or detailed.	Not identified or detailed.	LSTM RNN (32 hidden units) + baseline Simple RNN comparison.	99.1% accuracy (LSTM), 96% accuracy (Simple RNN baseline).
[28]	Custom ISL sensor-glove dataset (flex sensors + MPU-6050 IMU).	26 ISL gestures (static + dynamic words).	1040 samples.	90% train / 10% validation-test split.	104 test samples (10% of dataset).	Not identified or detailed.	LSTM (50 units) + Dense (100 nodes) + Dropout (40%) + Softmax classifier.	98% accuracy (test set), ~0.076 loss.
[29]	Chinese Sign Language (CSL) continuous SLT dataset.	100 sentences (179-word vocabulary, ~4–8 signs per sentence).	5000 RGB videos (50 signers × 100 sentences).	Split I: 40 signers train / 10 signers test; Split II: 94 sentences train / 6 unseen sentences test.	Split I: 1000 videos; Split II: 300 videos.	Not identified or detailed.	3D CNN (C3D) + Hierarchical LSTM (3-layer encoder-decoder + attention).	Split I: 92.9% sentence accuracy, BLEU-4 ≈ 0.928; Split II: ~50.6% word accuracy (Acc-w), WER ≈ 0.185.
[30]	LeapMotion ASL kinematic dataset (TUAT, 25 subjects, 16890 samples, 100-frame sequences, 60 features from hand/forearm kinematics).	60 ASL signs (0–10 + 49 words).	LeapMotion sensor (hand + forearm skeletal tracking, 26 hand points, derived kinematic features).	User-independent nested k-fold cross-validation (outer loop subject-wise splits; inner loop 20 train/val subjects + 5 test subjects, K=4 CV).	5 subjects per outer loop test split.	Data warping (temporal distortion via spline interpolation), amplitude perturbation using sinusoidal noise; 40 augmentations per training sample.	k-NN, RF, SVM, MLP, CNN (ConvNet), LSTM RNN, DeepConvLSTM.	DeepConvLSTM: 91.1% ± 3.8 (test accuracy); ConvNet: 89.3% ± 4.0; LSTM: 87.2% ± 5.1; MLP: 83.7%; RF: 80.1%; SVM: 80.6%.
[31]	ChaLearn LAP IsoGD; SKIG (RGB-D gesture datasets).	isolated gesture data set : 249 gestures; Sheffield Kinect gesture: 10 gesture classes.	RGB-D video sequences (Kinect); multimodal RGB + depth.	IsoGD: official train/val/test split (subject-independent); SKIG: 3-fold cross-validation (subject-wise split).	IsoGD: validation set used (test labels unavailable); SKIG: cross-val test folds.	Uniform temporal sampling; temporal jitter augmentation; fixed-length clips (32 frames).	3D CNN + Convolutional LSTM + Spatial Pyramid Pooling + late fusion (RGB & Depth streams).	IsoGD: 51.02% (validation accuracy); SKIG: 98.89% (accuracy).
[33]	In-house CSL dataset; 500 CSL; RWTH-PHOENIX-Weather; RWTH-PHOENIX-Weather-2014T.	150 isolated signs arranged in 50 continuous sentences; 2,887 sign words.	RGB videos; in-house CSL for daily communication; 500 CSL with 25,000 videos, 100+ h, 50 actors.	70%, train and 15% validation for CSL datasets; 80%/10% for RWTH-PHOENIX.	15% for CSL; 10% for RWTH.	Not identified or detailed.	Faster R-CNN; CNN; ST-LSTM fusion; Bi-LSTM encoder-decoder with attention.	81.22% on 500 CSL; 76.12% on RWTH-PHOENIX; 75.32% on PHOENIX-2014T.
[34]	SCOPE; PHOENIX-2014; PHOENIX-2014T; CSL-Daily.	SCOPE: 5k glosses; 59,231 videos; 72.4 h.	CSL dialogue videos with gloss, text and context annotations; professional Deaf signers.	SCOPE: 80/5/15; original splits for external datasets.	15% for SCOPE.	Not identified or detailed.	DWPose; Transformer encoders; CTC + MWER; frozen LLM embeddings; Qwen2 + Q-LoRA.	SCOPE test: 27.4 WER; ROUGE-L 70.14; BLEU-4 38.59.
[35]	Source gesture dataset; TG_BS; TG_New; CapgMyo DB-a.	Source: 30 gestures; TG_BS: 30 gestures; TG_New: 10 gestures; DB-a: 8 gestures.	128-channel HD-sEMG acquisition using 2×48-channel forearm arrays + 2×16-channel upper-arm arrays; sampling at 1 kHz; band-pass	Subject-specific training/testing; random repetition-based split. For the source dataset: 7 repetitions for training + 1 validation. For target datasets: (N_u) repetitions	TG_BS: 8 repetitions/gesture; TG_New: 8 repetitions/gesture; DB-a: 10 repetitions/gesture.	Sliding-window segmentation (100 ms window, 50 ms stride); sEMG image reshaping (16×8); normalization (Z-score).	CNN-based transfer learning framework; Source ConvNet; CNN-only target network; CNN+LSTM target network with transferred convolutional layers.	CNN-only + TL: TG_BS 92.13%, TG_New 93.32%, DB-a 91.18% accuracy. CNN+LSTM + TL: TG_BS 93.73%, TG_New 97.34%, DB-a

			filtering (20–500 Hz).	for training and remaining repetitions for testing.				94.57% accuracy.
[37]	iMiGUE; Kinetics-Skeleton; NTU RGB+D60.	iMiGUE: 32 gesture/action classes; Kinetics: 400 action classes; NTU RGB+D60: 60 action classes.	Skeleton sequences extracted using OpenPose (2D) and Kinect v2 (3D skeletal joints).	iMiGUE: 70/30 train-test split; Kinetics: official train/test split; NTU RGB+D60: X-Sub and X-View benchmark splits.	iMiGUE: 4563 test clips; Kinetics: 20,000 test clips; NTU RGB+D60: 16,560 (X-Sub), 18,960 (X-View).	Coordinate normalization; pose correction offsets; multi-stream fusion (joint, bone, joint-motion, bone-motion streams).	PCCTR-GCN (Pose Correction + Channel Topology Refinement GCN) with multi-stream fusion.	iMiGUE: 55.08% (Top-1), 89.35% (Top-5); Kinetics: 36.5% (Top-1), 59.2% (Top-5); NTU RGB+D60: 89.7% (X-Sub), 95.4% (X-View).

From the surveyed works within the sign/gesture recognition scope, it is possible to observe a broad diversity of sensing and modeling strategies, including RGB/RGB-D video, skeleton-based pose estimation, Leap Motion tracking, sensor gloves, myographic sensors, and other specialized acquisition devices. The datasets also vary considerably, ranging from small gesture-classification datasets [27], [28] to large-scale sign-language, gesture, or action-recognition corpora [33], [34], [37]. Therefore, the comparison should be interpreted qualitatively, since the surveyed works differ in task formulation, sensing modality, dataset scale, evaluation protocol, and reported metrics.

Several approaches achieve high performance using dedicated sensing setups. For instance, [28] reports 98% accuracy for 26 ISL signs using a sensor glove, [30] reaches 91.1% accuracy for 60 ASL signs using Leap Motion data, and [35] obtains above 90% accuracy using high-density sEMG signals. Although relevant, these solutions depend on specialized or more intrusive devices. In contrast, SLIM relies on RGB-based landmark extraction through MediaPipe, using conventional video input and compact body/face keypoint representations, which is more aligned with mobile, web-based, and service-oriented communication scenarios.

Recurrent architectures, particularly LSTM-based models, remain common in dynamic gesture and sign recognition ([27],[28],[29],[30],[33]), due to their ability to model temporal dependencies. SLIM follows this direction through a ConvLSTM-based configuration, achieving 95.6% accuracy in the recognition of 50 dynamic LGP signs from a dataset of 1650 videos. In addition, SLIM extends recognition with LLM-based sentence construction, reaching an average semantic similarity score of 0.81, thus moving beyond isolated sign classification.

Works more directly related to stream-wise communication, such as [29], show the potential of continuous sign-language translation, but also expose limitations in unseen sentence translation and word-order handling. More recent approaches, such as [34], combine gloss detection with LLM-based refinement, reinforcing the relevance of language-model support in sentence construction. Overall, these works support the motivation behind SLIM: combining lightweight landmark-based recognition, temporal deep learning, and LLM-supported sentence generation as a feasible path toward interactive deaf–hearing communication.

5.2 ASLM-related results and derived artifacts

Similarly to SLIM, the ASLM was supported by a WebGL-based avatar prototype designed to validate the natural-language-to-avatar-based LGP communication flow (Figure 16). This prototype integrated the required interfaces to communicate with the hybrid gloss translation service and with the remote repository where the animations associated with each gloss are stored. Also, it established the basis for the avatar component that was later embedded into the CFM, enabling the frontend to locally render LGP gestures through the Unity/WebGL component.

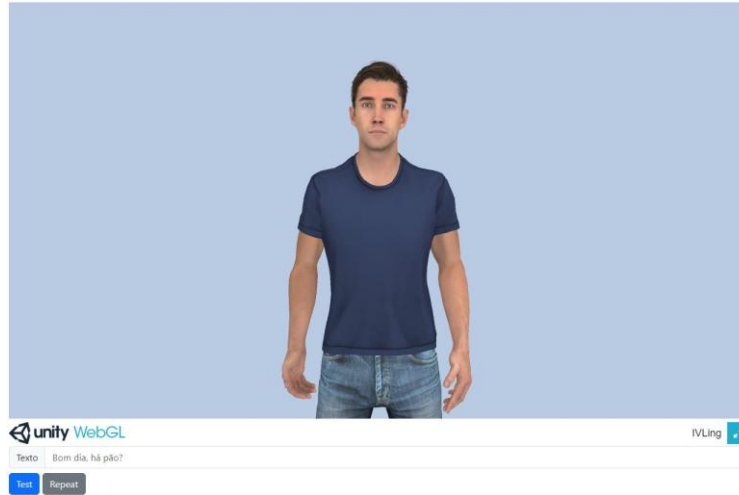


Figure 16 – Standalone prototype of the LGP-signing 3D avatar.

Regarding the hybrid translation component, Table 7 summarizes the results previously reported in [11], considering both the rule-based NLP approach and the neural-enhanced translation strategy. The comparison uses the Bilingual Evaluation Understudy (BLEU) and Metric for Evaluation of Translation with Explicit Ordering (METEOR) scores. As shown, the rule-based approach outperformed the neural-enhanced version at the reported development stage. This behavior can be explained by the still limited size of the dataset used to train the neural model. Nevertheless, the neural translation output was retained as part of the ASLM because of its potential for progressive improvement. Accordingly, the system currently prioritizes the rule-based translation while keeping the neural-enhanced output available, allowing a gradual transition as the neural model is retrained with larger datasets and becomes more competitive.

Table 7 – Natural-language to LGP-based gloss translation scores – based on previous report [11].

LGP Gloss translation method	BLUE score	METEOR score
Rule-based	0.72	0.80
Neural network enhanced	0.70	0.75

The avatar-based signs synthesis component was evaluated by two expert evaluators, who observed the avatar performing LGP signs and assessed their intelligibility. Figure 17 presents the distribution of the evaluated signs across different intelligibility levels, ranging from incomprehensible to perfect. The test results demonstrate that Evaluator 1 understood 90% of the words, while Evaluator 2 understood 78%. Regarding the quality of the animations, when they were classified as perfect, the evaluators always understood the

corresponding words. These results indicate that the avatar-based rendering component is capable of producing interpretable LGP gestures, while also showing that animation quality directly affects sign intelligibility.

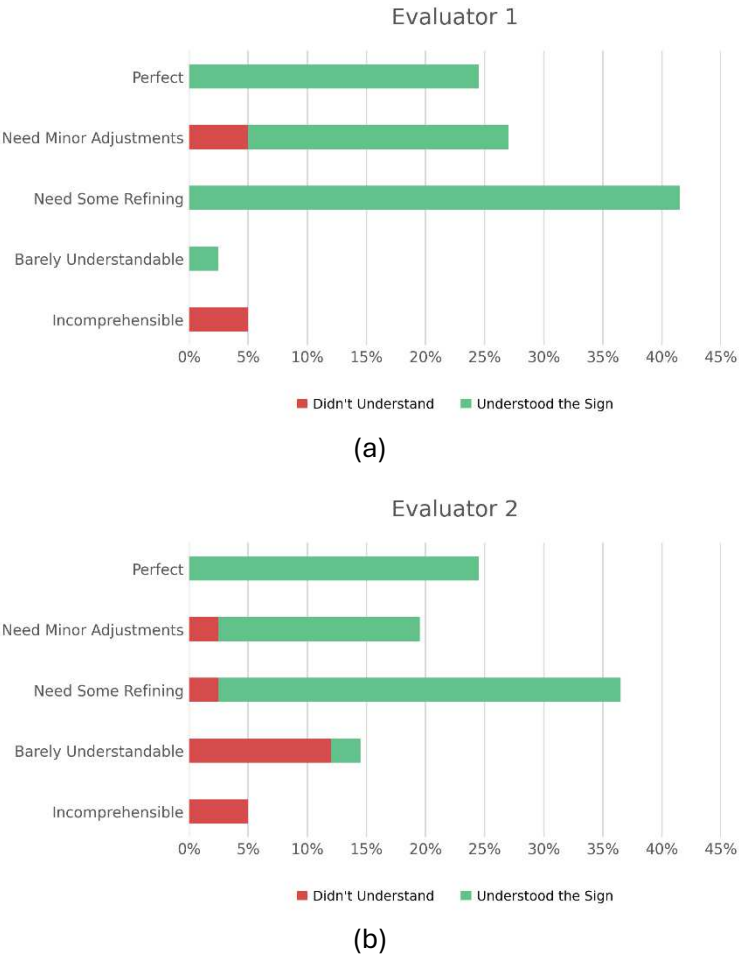


Figure 17 – Results of expert evaluation over the Avatar ability to produce understandable LGP signs: a) and b) correspond to Evaluator 1 and Evaluator 2, respectively.

Moreover, the responsiveness of the avatar-based rendering component was assessed through the direct WebGL-based solution used to display LGP signs through the 3D avatar. The test measured the elapsed time between sentence submission by the user and the moment when the corresponding avatar animation became ready for visualization. The conditions were aligned with the connectivity assumptions previously described for the SLIM interaction with the ChatGPT 3.5 API, considering a fast wireless connection scenario representative of favorable mobile-data usage. As shown in Figure 18, five sentences from tourism and commerce-related dialogues, ranging from four to eight LGP signs, were tested. The direct WebGL solution achieved an average response time of 1.10 s, with values ranging from 0.94 s to 1.42 s, supporting its adequacy for near real-time hearing-to-deaf communication scenarios.

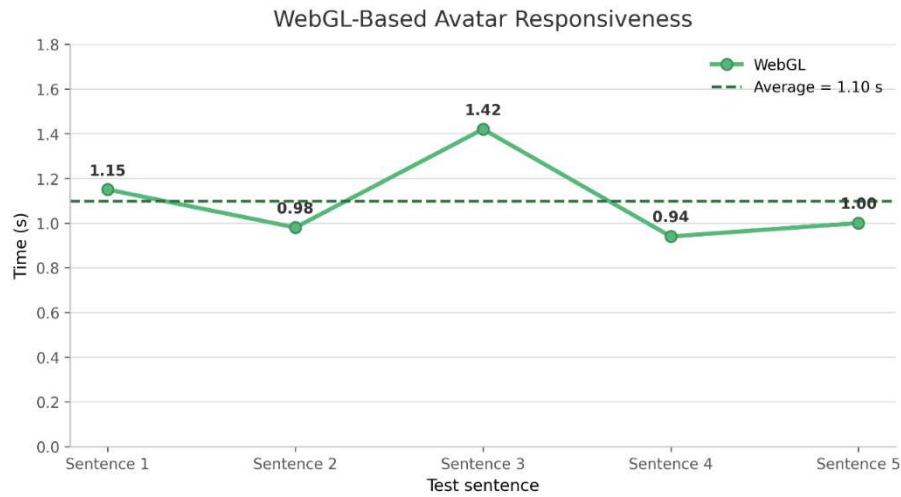


Figure 18 – Response time of the direct WebGL-based avatar rendering solution under fast network conditions, measured from sentence submission to animation readiness across five LGP gloss sequences.

A comparison of the ASLM results with state-of-the-art works is summarized in Table 8. This comparison considers several factors, namely language, input representation, number of signs/glosses, avatar or animation approach, deployment strategy, and main reported results.

Table 8 – Comparison of ASLM with relevant works from literature.

Work	Language	Input	No. of signs/glosses	Avatar/animation approach	Deployment	Main result(s)
SLaDIS / ASLM	LGP.	LGP gloss sequence.	>700 lexical entries; ~500 animated terms.	MoCap-based FBX animations; dynamic three-state machine	Embeddable WebGL artifact.	84% gesture intelligibility (preliminary user evaluation).
[46]	German Sign Language (DGS).	Gloss-based sign language utterances derived from annotated video recordings.	95 reusable glosses; 154 gloss instances across 11 utterances.	EMBR-based signing avatar with gloss-sequencing, k-pose animation, IK spine controls, morph-target facial expressions, gaze control, viseme/mouthing synthesis articulated with OpenMARY; cyclic refinement with deaf experts.	Research prototype using EMBR animation engine.	Avatar achieved ~58–59% relative to human signers.
[48]	American Sign Language.	English sentences → ASL gloss → XML-Gloss annotation.	Not explicitly fixed; rule-based tokens (gloss-based units, ASL signs in XML tokens).	XML structured annotation + API-driven generation + XSLT/XSL-FO rendering (no MoCap avatar claimed).	Web-based rendering (XML + XSLT → browser visualization).	~97% accuracy in validation against manual annotations; validated on >100 sentences.
[49]	Arabic Sign Language.	Arabic text → Arabic Sign Language (ArabSL) gloss.	Not identified or detailed.	Avatar-based synthesis mentioned (via WebSign project).	Web application pipeline (Arabic NLP → XML Gloss → avatar system integration in WebSign)	Not identified or detailed.
[50]	LGP.	Bidirectional: PSL gestures ↔ Portuguese text.	57 hand configurations; evaluation vocabulary of 10 PSL words.	Blender-based 3D avatar animations; Kinect skeletal tracking + 5DT sensor gloves; SVM classification + DTW sequence alignment	Desktop-based translation system with Kinect, gloves, Unity/Ogre integration.	91.7% real-time gesture translation accuracy for small vocabulary evaluation.

[55]	LGP	Bidirectional: PSL gestures $\leftrightarrow$ Portuguese text.	1,066 sign gloss vocabulary; 2,887 spoken language vocabulary (PHOENIX14T).	No avatar; deep generative approach using GAN conditioned on skeletal pose label maps (OpenPose keypoints); NMT encoder-decoder + Motion Graph for text-to-pose; pose-to-video via GAN generator-discriminator.	Research framework.	BLEU-4 of 16.34/15.26 (dev/test) for text-to-gloss translation; MS generation SSIM $\sim$ 0.94 across three signers; HD generation SSIM $\sim$ 0.933–0.934.
[56]	American Sign Language.	English text $\leftrightarrow$ ASL gloss representation.	16,608 mocap frames and 8,304 RGB frames, for 57 signs.	XML-based gloss annotation framework.	XML-based Sign Language Processing framework (XML Schema + XSLT renderer + Java API) integrated into the WebSign avatar translation pipeline.	API-generated annotations achieved 97% accuracy compared with manual gloss transcriptions on 100+ test sentences.

Considering the surveyed works focused on avatar-based sign language communication, it is possible to identify two main, differentiated and more clear technological directions. The first relies on explicit animation pipelines, usually supported by motion capture, gloss sequencing, inverse kinematics, facial expression control, and avatar rendering, as one can infer by [46], [50] and [56]. These approaches consist on expert sign acquisition resorting to motion capture suits, wearable sensors, camera systems, or controlled recording setups, followed by retargeting, animation refinement, and integration into an avatar-compatible representation. Their main advantage lies in the direct control over the produced animations and in the possibility of building reusable gloss-based animation units. However, they also involve complex and usually expensive and intrusive technological settings and initial learning efforts to operate with such setups and the workflow associated from sign acquisition to avatar-compatible computation. The second direction explores data-driven generation strategies, including deep generative models capable of producing synthetic signing motion or signer-like visual outputs from pose, gloss, or textual representations [55]. These approaches may reduce the need for manually assembling avatar animation libraries and may eventually produce more visually realistic signing outputs. Nevertheless, they usually require larger and well-annotated datasets, stronger computational resources, and more demanding training procedures to produce stable, controllable, and communicationally intelligible results. Besides, the generated outputs may be harder to inspect or correct at the level of individual signs when compared with explicit gloss-to-animation mappings.

ASLM adopts the first direction, following a motion-capture-based and avatar-rendering strategy, but organizes it within a modular service-oriented architecture. By design, the animation repository and the avatar rendering component are developed and maintained as separable elements: the database can progressively grow through the addition of new gloss-based animations, while the WebGL avatar can consume those assets through a service-mediated workflow. More specifically, the use of MoCap-based FBX animations, gloss-to-animation mapping, and a WebGL-embeddable avatar component provides a practical compromise between visual interpretability, reuse of animation assets, and frontend-side deployment. Compared with fully generative alternatives, this strategy is less dependent on large-scale training datasets and offers greater control over each rendered sign. However, its main scalability challenge lies in the continuous expansion, validation, and refinement of the remotely accessible gloss-based animation database, rather than in the rendering mechanism itself.

5.3 CFM in operation

The CFM mobile application represents the main integrative artifact resulting from the SLaDIS development efforts, demonstrating how the proposed modules can be brought together into an operational bidirectional communication prototype. The application provides two profile-oriented interfaces: one tailored to deaf users, supporting LGP-based message composition, visual feedback, and avatar-mediated reception of natural-language messages; and another tailored to hearing users, supporting natural-language message composition, reading, and audio-based interaction, as illustrated in Figure 19.

From an operational perspective, the CFM embeds MediaPipe as the body/face landmark extraction component and integrates a WebGL-based avatar engine for LGP sign synthesis and rendering. In the hearing-to-deaf communication flow, natural-language messages are sent to backend translation services, converted into LGP glosses, and used to retrieve the corresponding remote animation assets, which guide the 3D avatar state machine during local rendering. In the opposite direction, deaf users formulate LGP signs that are captured as landmark buffers, interpreted into gloss-like units by the SLIM services, and progressively accumulated into a term sequence. This sequence is then submitted to an LLM-based sentence construction service, configured with LGP-oriented generation rules, to produce a natural-language sentence that can be validated and sent to the hearing user.

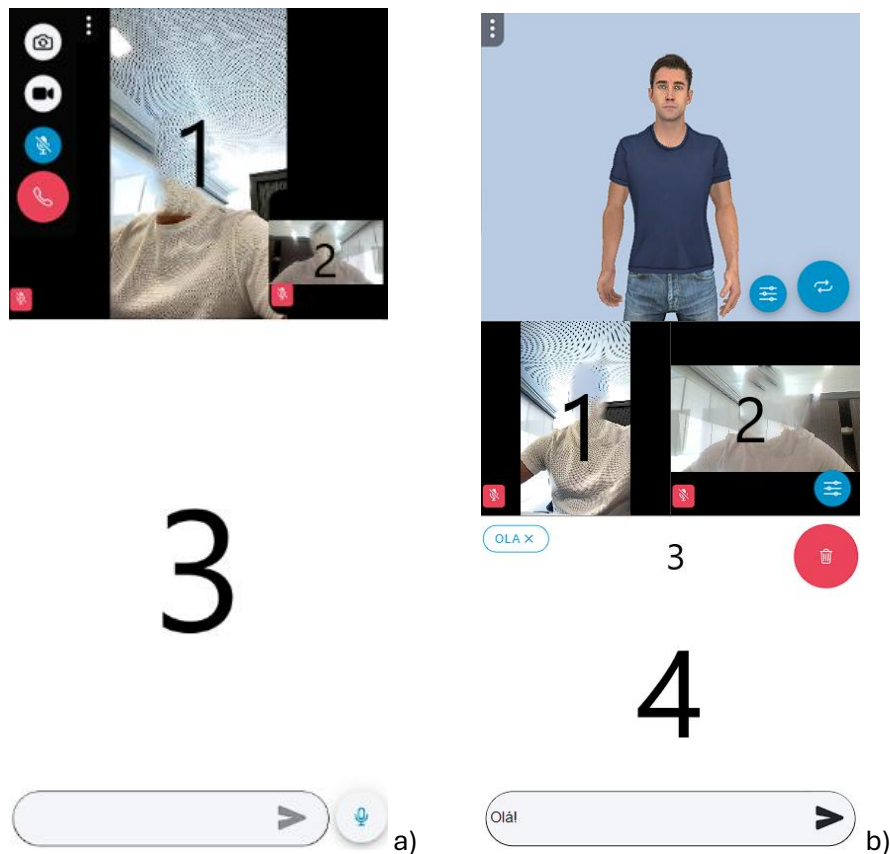


Figure 19 – CFM integrative module in operation: (a) hearing-user profile interface, including (a-1) deaf-user video viewport, (a-2) hearing-user self-video viewport, and (a-3) natural-language messaging area; (b) deaf-user profile interface, including (b-1) hearing-user video viewport, (b-2) deaf-user self-video viewport with skeleton/landmark detection capabilities, (b-3) AI-inferred LGP gloss, and (b-4) AI-generated sentence pending sending confirmation.

5.4 Discussion

After presenting SLaDIS main results, it is relevant to synthesize the main technological decisions that guided its implementation and integration. Table 9 summarizes the critical system aspects identified during development, the corresponding implementation choices, and the rationale behind each decision. This overview highlights how the proposed architecture balances computational efficiency, modularity, service interoperability, and fast operation across the complete SLaDIS workflow.

Table 9 - Summary of key technological decisions and rationale behind the implementation of the SLaDIS modules.

System aspect	Implementation decision	Rationale
AI-based LGP sign interpretation	Client-side MediaPipe-based landmark extraction	Landmark sequences provide a compact representation of body, hand, and facial motion, reducing data transmission requirements when compared with raw image or video sequences. Also, it allows to provide real-time intuitive user insights regarding the validity of the data acquired from LGP sign performing.
	Point-based AI inference	Inference over landmark sequences is computationally lighter than processing full image sequences, while preserving the spatio-temporal structure required for sign recognition.
LLM-based sentence construction	Replaceable external LLM-based service	Using a matured LLM (e.g. ChatGPT-3.5) avoids the need to manually implement complex sentence-generation rules or train a dedicated language model from scratch. The LLM component can, however, be replaced by alternative similar solutions, including conversational models directly deployed and hosted on a controlled server-side environment within SLaDIS ecosystem, while preserving the same service interface.
Sentence decomposition into LGP glosses	NLP-based and hybrid translation pipeline	NLP-based processing enables lightweight sentence cleaning, tokenization, morphological analysis, and structural transformation. Given the limited availability of LGP-specific resources in general-purpose LLM training data, a dedicated rule-based/neural hybrid approach provides more control over LGP-oriented gloss generation.
Avatar operation	Locally embedded Unity/WebGL avatar player	Client-side WebGL rendering avoids the need to remotely generate and transmit pre-rendered avatar data (e.g. videos) for each message. This reduces server-side processing and network overhead while supporting faster and more interactive gesture visualization.
Animation delivery	Gloss-to-animation mapping through remote assets	Associating glosses with reusable animation assets enables scalable avatar-based rendering, allowing new signs to be added by extending the animation repository rather than redesigning the whole avatar system.
Service integration	RESTful/service-based interfaces	Decoupled service interfaces allow SLIM, ASLM, CFM, recognition models, translation services, and animation repositories to evolve independently while remaining integrated into the same communication workflow.

Overall, SLaDIS can be positioned as a modular service-oriented system designed to support bidirectional deaf-hearing communication. Unlike most surveyed approaches, which tend to focus either on sign recognition (eg. [28]) or on sign synthesis (eg. [7]), SLaDIS integrates both communication directions into a unified full-stack workflow. From the literature analysis, it emerges as one of the few identified systems combining sign interpretation, natural-language sentence construction, gloss-based sign synthesis, avatar rendering, and frontend-mediated communication within the same service-oriented architecture.

For the deaf-to-hearing communication flow, SLaDIS relies on the SLIM, which combines MediaPipe-compatible landmark processing with deep learning-based sign recognition – similar to other proposals relying in skeleton detection (eg. [6]). The use of landmark representations allows LGP gestures to be converted into gloss-like terms while reducing

the amount of visual data that must be processed or transmitted. In the frontend, this process is supported by a progress-bar interaction mechanism that helps the deaf user regulate the timing of sign articulation and gesture capture. The inferred gloss-like terms are then submitted to an LLM-based sentence construction service – sharing communication goals with other approaches like [34] –, configured with LGP-oriented generation rules, to produce more cohesive natural-language sentences.

For the hearing-to-deaf communication flow, SLaDIS relies on the ASLM, which converts natural-language input into LGP glosses and uses those glosses to retrieve the corresponding animations from a remote repository. These animations are then rendered locally by a WebGL-based avatar – in alignment with other synthetic signaller-based solutions as [49] – embedded in the frontend. The avatar uses a state-machine-based animation mechanism to sequence the retrieved signs and support smoother transitions between consecutive gestures. This design follows the rationale of avatar-based sign language rendering systems, while distinguishing itself through the decoupling between translation services, the remote animation repository, and the frontend-side avatar component.

The CFM operates as the integrative layer that brings these services together in a profile-aware communication environment. It combines video communication functionalities with MediaPipe-based landmark acquisition, WebGL-based avatar rendering, and service-mediated access to the SLIM and ASLM components. Therefore, rather than acting only as a videoconferencing interface, the CFM orchestrates the recognition, translation, rendering, and communication resources required to support two-way interaction between deaf and hearing users.

A central design concern across the three modules was the mitigation of computational and communication bottlenecks. This concern is reflected in the use of compact landmark representations instead of raw video streams for sign recognition, in the encapsulation of AI-based components as services, in the local rendering of avatar animations through WebGL, and in the modular separation between frontend interaction, translation, inference, and animation retrieval. The computational performance results reported throughout this section support this design rationale, showing that the most critical components – including landmark-based sign inference, LLM-based sentence construction, and avatar animation readiness – can operate with response times compatible with near real-time interaction.

6 Future Work and Conclusions

This paper presented SLaDIS, an integrated service-oriented information system designed to support bidirectional communication between deaf and hearing users. The proposed solution is organized into three main modules: SLIM, ASLM and CFM. Together, these modules provide a full-stack communication workflow in which LGP signs can be interpreted as natural-language sentences, while natural-language messages can be translated into LGP glosses and rendered through a 3D sign-capable avatar.

The SLIM supports the deaf-to-hearing communication flow by combining computer vision, landmark-based gesture representation, deep learning inference, and LLM-based sentence construction. Its recognition component achieved promising results in the classification of

50 dynamic LGP signs, reaching an accuracy of 95.6%, while the sentence construction component obtained an average semantic similarity score of 0.81. The computational performance results also indicate that both the gesture inference model and the LLM-based sentence construction service introduce limited delays, supporting their adequacy for near real-time communication scenarios.

The ASLM supports the hearing-to-deaf communication flow by combining natural-language processing, hybrid translation into LGP glosses, motion-capture-based animation assets, and a loose Unity/WebGL-based avatar rendering (migrated to the frontend). The hybrid translator demonstrated the feasibility of combining rule-based linguistic processing with neural translation enhancement, while the avatar component showed its ability to render interpretable LGP signs through reusable animation assets. The WebGL-based rendering strategy proved particularly relevant from an architectural perspective, since it allows sign animations to be executed locally on the client-side, reducing the need for heavier remote rendering or video transmission approaches.

The CFM represents the integrative frontend artifact of SLaDIS. Developed as a profile-aware mobile application, it combines video communication, MediaPipe-based landmark extraction, service orchestration, and the embedded WebGL avatar component. By integrating SLIM and ASLM through service-based interfaces, the CFM demonstrates the operational feasibility of the proposed architecture and consolidates the bidirectional communication workflow between deaf and hearing users.

Despite these results, several improvements remain necessary. Future work should extend the LGP recognition dataset to cover a broader vocabulary and a wider diversity signers, while adapting the deep learning models to preserve or improve recognition accuracy as the number of supported signs increases. The LLM-based sentence construction component should also be further explored, including the possibility of deploying server-side locally hosted models, such as models served through Ollama or similar infrastructures, in order to reduce dependence on external services while balancing response quality and latency. Further developments are also required in the natural-language-to-LGP translation pipeline. The hybrid translator should evolve in terms of sentence cleaning, tokenization, morphological analysis, and structural transformation, either by strengthening the current NLP/AI enhancer-based approach or by exploring LGP-oriented LLM/RAG strategies capable of supporting more flexible, robust, and context-aware gloss generation. In parallel, the gloss-based animation repository should be expanded, since the current animation vocabulary remains limited when compared with the broader known LGP lexicon. The avatar itself should also be improved to reduce sign execution errors and increase naturalness.

Overall, SLaDIS constitutes an important step toward full-stack bidirectional deaf-hearing communication. Its main contribution lies not only in the individual AI, NLP, and visual computing components, but especially in their integration through a modular service-oriented architecture. While most existing approaches address either sign recognition or sign synthesis in isolation, SLaDIS demonstrates the feasibility of combining both directions into a unified and extensible communication platform.

Statements and Declarations

Funding

This work was co-financed by the project IVLinG – *Interprete Virtual de Língua Gestual* (Nº POCI-01-0247-FEDER-068605), financed by Portugal 2020, under the Competitiveness and Internationalization Operational Program (POCI), and by the European Regional Development Fund (ERDF).

Competing interests

The authors declare that they have no competing interests.

Ethics approval

The evaluation activities consisted of non-invasive prototype assessment tasks conducted with Portuguese Sign Language experts from the *Associação Portuguesa de Surdos*, which formally supported the IVLinG – *Intérprete Virtual de Língua Gestual* project (No. POCI-01-0247-FEDER-068605). Evaluators were informed about the purpose of the procedures, and all collected data were anonymized before analysis and reporting. No personally identifiable information is disclosed.

Consent

The expert evaluation activities were carried out within the formal collaboration framework established with the *Associação Portuguesa de Surdos* for the IVLinG – *Intérprete Virtual de Língua Gestual* project (No. POCI-01-0247-FEDER-068605). Evaluators participated in the assessment procedures in this context, and the collected data were anonymized and used only in aggregated form for data processing, scientific analysis, and publication purposes. No identifiable personal data is reported in this manuscript.

Data availability

The datasets developed and/or used during the current study are not publicly available due to legal constraints related to participants' ownership.

Code availability

The source code developed and/or used during the current study is not publicly available due to legal constraints related to participants' ownership.

Author contributions

Telmo Adão contributed to conceptualization, data curation, formal analysis, investigation, methodology, resources, project administration, software, supervision, validation, visualization, writing – original draft, and writing – review and editing. **Somayeh Shahrabadi**, **Bruno Ribeiro**, **Duarte Dias**, **Vasco Alves**, and **Fábio Araújo** contributed to investigation, data curation, software, validation, visualization, and writing – original draft. **Filipe Pereira**, **Tiago Oliveira**, **Luís Romero**, **Pedro Miguel Faria**, and **Luís Gonzaga Magalhães** contributed to conceptualization, methodology, project administration, resources, supervision, validation, and writing – review and editing. All authors read and approved the final manuscript.

References

- [1] R. Meucci, ‘The Forgotten Voice: Antonio Meucci and the Origins of Telecommunication Technology’, *Found Sci*, Nov. 2025, doi: 10.1007/s10699-025-10019-y.
- [2] A.-M. Suduc, M. Bizoi, and F. G. Filip, ‘Status, Challenges and Trends in Videoconferencing Platforms’, *INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL*, vol. 18, no. 3, May 2023, doi: 10.15837/ijccc.2023.3.5465.
- [3] Karunya S., Jalakandeshwaran M, Thanuja Babu, and Uma R, ‘AI-Powered Real-Time Speech-to-Speech Translation for Virtual Meetings Using Machine Learning Models’, in *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, Dec. 2023, pp. 1–6. doi: 10.1109/ICCEBS58601.2023.10448600.
- [4] N. Sarhan and S. Frintrop, ‘Unraveling a Decade: A Comprehensive Survey on Isolated Sign Language Recognition’, presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 3210–3219. Accessed: Apr. 15, 2026. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2023W/AMFG/html/Sarhan_Unraveling_a_Decade_A_Comprehensive_Survey_on_Isolated_Sign_Language_ICCVW_2023_paper.html
- [5] B. F. M. Ribeiro, ‘System for capturing and displaying portuguese sign language animations in a 3D environment’, Master Thesis, Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Viana do Castelo, Viana do Castelo, 2024. Accessed: Apr. 15, 2026. [Online]. Available: <http://repositorio.ipv.pt/handle/20.500.11960/4029>
- [6] G. H. Samaan *et al.*, ‘MediaPipe’s Landmarks with RNN for Dynamic Sign Language Recognition’, *Electronics*, vol. 11, no. 19, Oct. 2022, doi: 10.3390/electronics11193228.
- [7] P. Jayanthi, P. R. K. S. Bhama, M. Abitha, R. Shrijhaa, and D. Uva Shree, ‘3D Avatar-Based Sign Gesture Animation’, in *World Congress on Smart Computing*, P. K. Shukla, H. Sharma, and R. Mallipeddi, Eds., Singapore: Springer Nature, 2025, pp. 275–287. doi: 10.1007/978-981-97-9006-7_23.
- [8] T. Adão *et al.*, ‘Empowering Deaf-Hearing Communication: Exploring Synergies between Predictive and Generative AI-Based Strategies towards (Portuguese) Sign Language Interpretation’, *Journal of Imaging*, vol. 9, no. 11, Art. no. 11, Nov. 2023, doi: 10.3390/jimaging9110235.
- [9] Carlos Mayea, Dibet Garcia, Miguel A. Guevara Lopez, Emanuel Peres, Luís Magalhães, and Telmo Adão, ‘Building Portuguese Sign Language Datasets for Computational Learning Purposes’, in *Proceedings of the 2022 Edition of the International Conference on Graphics and Interaction (ICGI)*, Aveiro, Portugal: Eurographics Association}, Nov. 2022.
- [10] B. Ribeiro, D. Dias, V. Alves, P. M. Faria, and L. Romero, ‘A Translation System from European Portuguese to Portuguese Sign Language’, in *2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jun. 2023, pp. 1–5. doi: 10.1109/IWSSIP58668.2023.10180292.
- [11] V. Alves, J. Ribeiro, L. Romero, P. M. Faria, Â. Costa, and V. Ferreira, ‘A Gloss Based Translation From European Portuguese to Portuguese Sign Language’, in *2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jun. 2023, pp. 1–4. doi: 10.1109/IWSSIP58668.2023.10180304.
- [12] I.-B. Păvăloiu, ‘Leap Motion Technology In Learning’, *European Proceedings of Social and Behavioural Sciences*, vol. Education Facing Contemporary World Issues, May 2017, doi: 10.15405/epsbs.2017.05.02.126.
- [13] R. Jalayer, M. Jalayer, C. Orsenigo, and M. Tomizuka, ‘A review on deep learning for vision-based hand detection, hand segmentation and hand gesture recognition in human–robot interaction’, *Robotics and Computer-Integrated Manufacturing*, vol. 97, p. 103110, Feb. 2026, doi: 10.1016/j.rcim.2025.103110.
- [14] X. Ding *et al.*, ‘Robust gesture recognition method toward intelligent environment using Wi-Fi signals’, *Measurement*, vol. 231, p. 114525, May 2024, doi: 10.1016/j.measurement.2024.114525.

- [15]N. S. Kumar *et al.*, ‘HARNet in deep learning approach—a systematic survey’, *Sci Rep*, vol. 14, no. 1, p. 8363, Apr. 2024, doi: 10.1038/s41598-024-58074-y.
- [16]F. Jafari and A. Basu, ‘Saliency-Driven Hand Gesture Recognition Incorporating Histogram of Oriented Gradients (HOG) and Deep Learning’, *Sensors*, vol. 23, no. 18, Sep. 2023, doi: 10.3390/s23187790.
- [17]S. Zhong, A. Albini, G. Caroleo, G. Cannata, and P. Maiolino, ‘Improving Tactile Gesture Recognition with Optical Flow’, Aug. 06, 2025, *arXiv*: arXiv:2508.04338. doi: 10.48550/arXiv.2508.04338.
- [18]X. Wang, M. Xia, H. Cai, Y. Gao, and C. Cattani, ‘Hidden-Markov-Models-Based Dynamic Hand Gesture Recognition’, *Mathematical Problems in Engineering*, vol. 2012, no. 1, p. 986134, 2012, doi: 10.1155/2012/986134.
- [19]J. Lin and Y. Ding, ‘A temporal hand gesture recognition system based on hog and motion trajectory’, *Optik*, vol. 124, no. 24, pp. 6795–6798, Dec. 2013, doi: 10.1016/j.ijleo.2013.05.097.
- [20]S.-Y. Park and E.-J. Lee, ‘Hand Gesture Recognition using Optical Flow Field Segmentation and Boundary Complexity Comparison based on Hidden Markov Models’, *Journal of Korea Multimedia Society*, vol. 14, no. 4, pp. 504–516, Apr. 2011, doi: 10.9717/kmms.2011.14.4.504.
- [21]M. B. Holte, T. B. Moeslund, and P. Fihl, ‘View-invariant gesture recognition using 3D optical flow and harmonic motion context’, *Computer Vision and Image Understanding*, vol. 114, no. 12, pp. 1353–1361, Dec. 2010, doi: 10.1016/j.cviu.2010.07.012.
- [22]F.-S. Chen, C.-M. Fu, and C.-L. Huang, ‘Hand gesture recognition using a real-time tracking method and hidden Markov models’, *Image and Vision Computing*, vol. 21, no. 8, pp. 745–758, Aug. 2003, doi: 10.1016/S0262-8856(03)00070-2.
- [23]J. Qi, L. Ma, Z. Cui, and Y. Yu, ‘Computer vision-based hand gesture recognition for human-robot interaction: a review’, *Complex Intell. Syst.*, vol. 10, no. 1, pp. 1581–1606, Feb. 2024, doi: 10.1007/s40747-023-01173-6.
- [24]G. Benitez-Garcia *et al.*, ‘Improving Real-Time Hand Gesture Recognition with Semantic Segmentation’, *Sensors*, vol. 21, no. 2, Jan. 2021, doi: 10.3390/s21020356.
- [25]D. E. Rumelhart, G. E. Hinton, and R. J. Williams, ‘Learning representations by back-propagating errors’, *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.
- [26]S. Hochreiter and J. Schmidhuber, ‘Long Short-Term Memory’, *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [27]J. Cifuentes, P. Boulanger, M. T. Pham, F. Prieto, and R. Moreau, ‘Gesture Classification Using LSTM Recurrent Neural Networks’, in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Jul. 2019, pp. 6864–6867. doi: 10.1109/EMBC.2019.8857592.
- [28]E. Abraham, A. Nayak, and A. Iqbal, ‘Real-Time Translation of Indian Sign Language using LSTM’, in *2019 Global Conference for Advancement in Technology (GCAT)*, Oct. 2019, pp. 1–5. doi: 10.1109/GCAT47503.2019.8978343.
- [29]D. Guo, W. Zhou, H. Li, and M. Wang, ‘Hierarchical LSTM for Sign Language Translation’, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Art. no. 1, Apr. 2018, doi: 10.1609/aaai.v32i1.12235.
- [30]V. Hernandez, T. Suzuki, and G. Venture, ‘Convolutional and recurrent neural network for human activity recognition: Application on American sign language’, *PLOS ONE*, vol. 15, no. 2, p. e0228869, Feb. 2020, doi: 10.1371/journal.pone.0228869.
- [31]G. Zhu, L. Zhang, P. Shen, and J. Song, ‘Multimodal Gesture Recognition Using 3-D Convolution and Convolutional LSTM’, *IEEE Access*, vol. 5, pp. 4517–4524, 2017, doi: 10.1109/ACCESS.2017.2684186.
- [32]G. Zhu, L. Zhang, P. Shen, J. Song, S. A. A. Shah, and M. Bennamoun, ‘Continuous Gesture Segmentation and Recognition Using 3DCNN and Convolutional LSTM’, *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 1011–1021, Apr. 2019, doi: 10.1109/TMM.2018.2869278.

- [33]Q. Xiao, X. Chang, X. Zhang, and X. Liu, ‘Multi-Information Spatial–Temporal LSTM Fusion Continuous Sign Language Neural Machine Translation’, *IEEE Access*, vol. 8, pp. 216718–216728, 2020, doi: 10.1109/ACCESS.2020.3039539.
- [34]Y. Liu, W. Zhang, S. Ren, C. Huang, J. Yu, and L. Xu, ‘SCOPE: Sign Language Contextual Processing with Embedding from LLMs’, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 6, pp. 5739–5747, Apr. 2025, doi: 10.1609/aaai.v39i6.32612.
- [35]X. Chen, Y. Li, R. Hu, X. Zhang, and X. Chen, ‘Hand Gesture Recognition based on Surface Electromyography using Convolutional Neural Network with Transfer Learning Method’, *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 4, pp. 1292–1304, Apr. 2021, doi: 10.1109/JBHI.2020.3009383.
- [36]C. Plizzari, M. Cannici, and M. Matteucci, ‘Spatial Temporal Transformer Network for Skeleton-Based Action Recognition’, in *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part III*, Berlin, Heidelberg: Springer-Verlag, Jan. 2021, pp. 694–701. doi: 10.1007/978-3-030-68796-0_50.
- [37]Y. Gao, X. Duan, and Q. Dai, ‘Skeleton-Based Action Recognition Using Graph Convolutional Network with Pose Correction and Channel Topology Refinement’, *Computers, Materials and Continua*, vol. 83, no. 1, pp. 701–718, Mar. 2025, doi: 10.32604/cmc.2025.060137.
- [38]M. Feng and J. Meunier, ‘Skeleton Graph-Neural-Network-Based Human Action Recognition: A Survey’, *Sensors*, vol. 22, no. 6, Mar. 2022, doi: 10.3390/s22062091.
- [39]W. Peng, J. Shi, T. Varanka, and G. Zhao, ‘Rethinking the ST-GCNs for 3D skeleton-based human action recognition’, *Neurocomputing*, vol. 454, pp. 45–53, Sep. 2021, doi: 10.1016/j.neucom.2021.05.004.
- [40]Y. Jiang and H. Deng, ‘Lighter and faster: A multi-scale adaptive graph convolutional network for skeleton-based action recognition’, *Engineering Applications of Artificial Intelligence*, vol. 132, p. 107957, Jun. 2024, doi: 10.1016/j.engappai.2024.107957.
- [41]M. Kaseris, I. Kostavelis, and S. Malassiotis, ‘A Comprehensive Survey on Deep Learning Methods in Human Activity Recognition’, *Machine Learning and Knowledge Extraction*, vol. 6, no. 2, pp. 842–876, Apr. 2024, doi: 10.3390/make6020040.
- [42]J. Gu, F. Wang, K. Li, Y. Wei, Z. Wu, and D. Guo, ‘MM-Gesture: Towards Precise Micro-Gesture Recognition through Multimodal Fusion’, Aug. 05, 2025, *arXiv*: arXiv:2507.08344. doi: 10.48550/arXiv.2507.08344.
- [43]B. Wang, L. Yu, and B. Zhang, ‘AL-MobileNet: a novel model for 2D gesture recognition in intelligent cockpit based on multi-modal data’, *Artif Intell Rev*, vol. 57, no. 10, p. 282, Sep. 2024, doi: 10.1007/s10462-024-10930-z.
- [44]X. Li, Y. Zhou, L. Huang, and C. Shao, ‘Efficientnet Model Based Gesture Recognition Method Research’, in *Proceedings of the 2021 4th International Conference on Artificial Intelligence and Pattern Recognition*, in AIPR ’21. New York, NY, USA: Association for Computing Machinery, Feb. 2022, pp. 255–259. doi: 10.1145/3488933.3489024.
- [45]M. Aly and I. S. Fathi, ‘Recognizing American Sign Language gestures efficiently and accurately using a hybrid transformer model’, *Sci Rep*, vol. 15, no. 1, p. 20253, Jun. 2025, doi: 10.1038/s41598-025-06344-8.
- [46]M. Kipp, A. Heloir, and Q. Nguyen, ‘Sign Language Avatars: Animation and Comprehensibility’, in *Intelligent Virtual Agents*, H. H. Vilhjálmsson, S. Kopp, S. Marsella, and K. R. Thórisson, Eds., Berlin, Heidelberg: Springer, 2011, pp. 113–126. doi: 10.1007/978-3-642-23974-8_13.
- [47]L. Naert, C. Larboulette, and S. Gibet, ‘A survey on the animation of signing avatars: From sign representation to utterance synthesis’, *Computers & Graphics*, vol. 92, pp. 76–98, Nov. 2020, doi: 10.1016/j.cag.2020.09.003.
- [48]A. Othman and M. Jemni, ‘An XML-gloss annotation system for sign language processing’, in *2017 6th International Conference on Information and Communication Technology and Accessibility (ICTA)*, Dec. 2017, pp. 1–7. doi: 10.1109/ICTA.2017.8336054.
- [49]N. Aouiti, M. Jemni, and S. Semreen, ‘Arab gloss and implementation for Arabic Sign Language’, in *2017 6th International Conference on Information and Communication Technology and Accessibility (ICTA)*, Dec. 2017, pp. 1–6. doi: 10.1109/ICTA.2017.8336055.

- [50] P. Escudeiro *et al.*, ‘Virtual Sign – A Real Time Bidirectional Translator of Portuguese Sign Language’, *Procedia Computer Science*, vol. 67, pp. 252–262, Jan. 2015, doi: 10.1016/j.procs.2015.09.269.
- [51] K. Bergmann and S. Kopp, ‘Modeling the Production of Coverbal Iconic Gestures by Learning Bayesian Decision Networks’, *Applied Artificial Intelligence*, vol. 24, no. 6, pp. 530–551, Jul. 2010, doi: 10.1080/08839514.2010.492162.
- [52] T. Cravens and S. Rashad, ‘Advances in Automatic Sign Language Translation: A Survey of Techniques and Systems’, in *2026 IEEE 16th Annual Computing and Communication Workshop and Conference (CCWC)*, Jan. 2026, pp. 0900–0909. doi: 10.1109/CCWC67433.2026.11393701.
- [53] S. Tan, N. Khan, Z. An, Y. Ando, R. Kawakami, and K. Nakadai, ‘A review of deep learning-based approaches to sign language processing’, *Advanced Robotics*, vol. 38, no. 23, pp. 1649–1667, Dec. 2024, doi: 10.1080/01691864.2024.2442721.
- [54] M. De Coster, D. Shterionov, M. Van Herreweghe, and J. Dambre, ‘Machine translation from signed to spoken languages: state of the art and challenges’, *Univ Access Inf Soc*, vol. 23, no. 3, pp. 1305–1331, Aug. 2024, doi: 10.1007/s10209-023-00992-1.
- [55] S. Stoll, N. C. Camgoz, S. Hadfield, and R. Bowden, ‘Text2Sign: Towards Sign Language Production Using Neural Machine Translation and Generative Adversarial Networks’, *Int J Comput Vis*, vol. 128, no. 4, pp. 891–908, Apr. 2020, doi: 10.1007/s11263-019-01281-2.
- [56] V. Baltatzis, R. A. Potamias, E. Ververas, G. Sun, J. Deng, and S. Zafeiriou, ‘Neural Sign Actors: A diffusion model for 3D sign language production from text’, Apr. 05, 2024, *arXiv: arXiv:2312.02702*. doi: 10.48550/arXiv.2312.02702.
- [57] A. Symeonidis-Herzig, J. Low, O. M. Sincan, and R. Bowden, ‘M3T: Discrete Multi-Modal Motion Tokens for Sign Language Production’, Mar. 24, 2026, *arXiv: arXiv:2603.23617*. doi: 10.48550/arXiv.2603.23617.
- [58] M. I. Lakhal and R. Bowden, ‘Diverse Signer Avatars with Manual and Non-Manual Feature Modelling for Sign Language Production’, Aug. 21, 2025, *arXiv: arXiv:2508.15988*. doi: 10.48550/arXiv.2508.15988.
- [59] R. Rastgoo, K. Kiani, and S. Escalera, ‘Personalized continuous sign language production via a motion-aware federated diffusion model’, *Neurocomputing*, vol. 677, p. 133047, May 2026, doi: 10.1016/j.neucom.2026.133047.
- [60] J. Low, A. Symeonidis-Herzig, M. Ivashechkin, O. M. Sincan, and R. Bowden, ‘SignSparK: Efficient Multilingual Sign Language Production via Sparse Keyframe Learning’, Mar. 23, 2026, *arXiv: arXiv:2603.10446*. doi: 10.48550/arXiv.2603.10446.
- [61] M. Sanaullah *et al.*, ‘A Real-Time Automatic Translation of Text to Sign Language’, *CMC*, vol. 70, no. 2, pp. 2471–2488, 2021, doi: 10.32604/cmc.2022.019420.
- [62] F. M. Talaat, W. El-Shafai, N. F. Soliman, A. D. Algarni, F. E. Abd El-Samie, and A. I. Siam, ‘Real-time Arabic avatar for deaf-mute communication enabled by deep learning sign language translation’, *Computers and Electrical Engineering*, vol. 119, p. 109475, Oct. 2024, doi: 10.1016/j.compeleceng.2024.109475.
- [63] K. Waldow, A. Fuhrmann, and D. Roth, ‘Facial Feature Enhancement for Immersive Real-Time Avatar-Based Sign Language Communication Using Personalized CNNs’, in *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, Mar. 2024, pp. 919–920. doi: 10.1109/VRW62533.2024.00256.
- [64] F. Picon *et al.*, ‘The EASIER Mobile Application and Avatar End-User Evaluation Methodology’, in *Proceedings of the LREC-COLING 2024 11th Workshop on the Representation and Processing of Sign Languages: Evaluation of Sign Language Resources*, Torino, Italy: ELRA Language Resources Association (ELRA) and the International Committee on Computational Linguistics (ICCL), May 2024, pp. 276–281. Accessed: Jun. 08, 2026. [Online]. Available: <https://www.sign-lang.uni-hamburg.de/lrec/pub/24021.html>
- [65] V. Ubieto, J. Pozo, E. Valls, B. Cabrero-Daniel, and J. Blat, ‘Sign Language Synthesis: Current Signing Avatar Systems and Representation’, in *Sign Language Machine Translation*, A. Way, L. Leeson, and D. Shterionov, Eds., Cham: Springer Nature Switzerland, 2024, pp. 247–266. doi: 10.1007/978-3-031-47362-3_10.

- [66]R. Zuo, R. A. Potamias, E. Ververas, J. Deng, and S. Zafeiriou, ‘Signs as Tokens: A Retrieval-Enhanced Multilingual Sign Language Generator’.
- [67]S. Chen *et al.*, ‘Customizing Generated Signs and Voices of AI Avatars: Deaf-Centric Mixed-Reality Design for Deaf-Hearing Communication’, *Proc. ACM Hum.-Comput. Interact.*, vol. 9, no. 2, p. CSCW055:1-CSCW055:31, May 2025, doi: 10.1145/3710953.
- [68]R. Zuo, F. Wei, Z. Chen, B. Mak, J. Yang, and X. Tong, ‘A Simple Baseline for Spoken Language to Sign Language Translation with 3D Avatars’, in *Computer Vision – ECCV 2024*, vol. 15107, A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, and G. Varol, Eds., in Lecture Notes in Computer Science, vol. 15107. , Cham: Springer Nature Switzerland, 2025, pp. 36–54. doi: 10.1007/978-3-031-72967-6_3.
- [69]B. Ribeiro, D. Dias, P. M. Faria, and L. Romero, ‘Capturing and Processing Sign Animations to a Portuguese Sign Language 3D Avatar’, in *2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Jun. 2023, pp. 1–5. doi: 10.1109/IWSSIP58668.2023.10180233.
- [70]D. Cer *et al.*, ‘Universal Sentence Encoder’, Apr. 12, 2018, *arXiv*: arXiv:1803.11175. Accessed: Aug. 04, 2023. [Online]. Available: <http://arxiv.org/abs/1803.11175>