MACHINE LEARNING FOR AMERICAN SIGN LANGUAGE RECOGNITION AND ACQUISITION IN ARTIFICIAL AGENTS

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN THE FACULTY OF SCIENCE AND ENGINEERING

2023

Federico Tavella

School of Engineering
Department of Computer Science

Contents

Word Count: 24618

List of Tables

List of Figures

Abstract

Artificial agents and, in particular, humanoid robots can interact with the surrounding environment, objects, and people using their cameras, actuators, and embodiment. They can employ verbal and non-verbal communication methods to engage with other agents. However, the behaviour of these robots is typically pre-programmed, limiting their capabilities to a predetermined set of actions. One alternative approach is to teleoperate the robot using additional devices, which provide precise measurements of pose and speed. Nonetheless, these devices can be costly and require expertise to operate effectively. Another option is imitation, where a system can replicate observed actions using only the robot's camera, akin to how humans learn. Consequently, one intriguing avenue of research is the acquisition of non-verbal communication skills through learning from demonstrations. This approach holds promise for applications such as teaching machines to comprehend and express sign language.

The overall aim of this thesis is to study imitation learning in a human-like fashion for artificial agents. As a case study, we teach a simulated humanoid agent American Sign Language (ASL) by imitating videos of people performing different signs. We use computer vision and imitation learning techniques, namely deep learning and reinforcement learning, to extract information from pre-recorded videos and teach the agent how to replicate the observed action. When compared to other approaches, our proposal removes the need for additional hardware (like motion capture suits or virtual reality headsets) to collect information necessary for imitation. Additionally, our approach shows how to take advantage of data without ground truth for regression via weak validation through classification.

We perform a first study to evaluate the data extracted from the videos using pre-trained vision models. We base our evaluation on subunits (i.e., phonological classes), as we believe they provide more fine-grained information when compared to lemmas. Consequently, we expand our findings to a novel large-scale dataset that we generate,

in order to test the generalisability of our approach. We apply state-of-the-art techniques for generating animation based on motion capture data to our scenario, teaching a virtual agent how to fingerspell and perform signs. The results we collected show that keypoints extracted from videos provide a good reference for recognising phonological classes of sign language. Additionally, we demonstrate how, albeit with some limitations, an imitation learning approach based on reinforcement learning and motion data extracted from videos provides a possible way of acquiring sign language. In particular, we show how our methodology is able to learn fingerspelling for 6 different letters and 5 different signs involving the whole upper body.

To summarise, we generate two novel datasets in which, for each sign, we associate phonological properties. Moreover, we demonstrate how to automate the recognition of such properties in signs, even on unseen signs, by carrying out the first large-scale attempt of phonological properties. Finally, to the best of our knowledge, we showcase for the first time how to acquire sign language (both fingerspelling and full signs) in an embodied fashion using only videos as input data.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to express my sincere gratitude to my supervisory team, Prof. Angelo Cangelosi and Dr. Aphrodite Galata, for their unwavering support, guidance, and encouragement throughout my PhD journey. Their expert advice, insightful feedback, and constructive criticism have been instrumental in shaping this thesis.

I am also grateful to my lab colleagues over the past years for their camaraderie, support, and stimulating discussions. Their insights, feedback, and suggestions have contributed significantly to the development and refinement of my research career. Thanks to Ana, Baris, Carlo, Chuang, Daniel, Federico M., Francesco L., Francesco S., Gabriella, Giannis, Hazel, Haodong, Hongbo, Ioanna, Jacopo, Luca, Marta, Martina, Mehdi, Mohammad, Radu, Rahul, Samuele, Sandro, Sophia, Wenjie, Wenxuan, Wiktor and Wolodymyr. I hope I did not forget anyone, but if I did rest assured that you did leave a mark. A special thanks goes to my social bubble during COVID: Alessio, Marta, Mauricio and Viktor. When the whole world stopped, we somehow managed to keep it rolling. A very special thanks goes to Marta for being such a good friend throughout good and bad times.

I am forever indebted to my family for their love, encouragement, and unwavering support throughout my academic journey. Their belief in me and their sacrifices have been a source of inspiration and motivation. Thank you to my father Claudio and mother Elena for shaping me in who I am today. In your own different, personal ways and according to your possibilities, you showed me what it really means to wear your heart on your sleeve.

Finally, I would like to thank my friends, both old and new, for their support, encouragement, and understanding throughout this journey. Their friendship, laughter, and encouragement have been a source of strength and motivation. As always, a special thanks goes to Chiara and Davide who, after so many years and kilometres, are still by my side.

To all those who have contributed to the success of this thesis, I extend my deepest

gratitude and appreciation. This achievement would not have been possible without your support and encouragement. Thank you all.

Chapter 1

Introduction

1.1 Motivations

The development of artificial agents like humanoid robots has opened up a wide range of possibilities in various scenarios, spanning from industrial to domestic settings. These artificial agents have the potential to revolutionize our daily lives by assisting us in numerous tasks. From manufacturing processes in factories to household chores, humanoid robots are being designed to enhance efficiency and convenience. To achieve their full potential, it is crucial to explore novel approaches for enabling robots to perform diverse actions and interact effectively with humans. Traditionally, the problem of implementing these actions has been approached by hard coding numerous primitives into the robot's programming. However, this approach has its limitations. It requires significant resources and meticulous consideration of various environmental and interaction factors. Moreover, it is exceedingly difficult to account for all possible variations that can occur in real-world scenarios through hard coding alone. One notable success in the field of hard coding is the Atlas robot (Atlas robot). This impressive creation can perform acrobatic movements without any learning component. However, such accomplishments are rare, and the challenges of scalability and adaptability remain prevalent. Developing a comprehensive set of hard-coded actions that can account for all possible variations is an ongoing struggle. Another approach to robot control involves teleoperation, where a person controls the robot remotely. This method utilises a wide array of sensors to capture the demonstrator's movements and translate them into corresponding actions for the robot. While teleoperation can provide more versatility in actions, it necessitates the presence of a human operator and

a complex sensor setup (Penco et al., 2019). This reliance on a human operator limits its autonomy and poses practical challenges for widespread implementation. A more practical and efficient approach to robot control involves learning from human demonstration. By observing a tutor performing a specific action, a robot can learn to reproduce it autonomously. This method, inspired by how humans acquire skills, holds promise for imparting a wide range of actions to robots without explicitly hard coding each one. Machine learning techniques, such as imitation learning and reinforcement learning, enable robots to acquire and refine their skills through repeated demonstrations and feedback. While effective action execution is vital, communication is equally crucial for human-robot interaction, whether verbal (Mavridis, 2015; Bonarini, 2020) or non-verbal (Breazeal et al., 2005; Mavridis, 2015; Saunderson and Nejat, 2019; Bonarini, 2020). Verbal communication is the default mode for most users, but it is not always suitable or accessible for everyone. Deaf individuals, for instance, rely heavily on non-verbal communication methods such as sign language. This necessitates the development of **robotic sign language** acquisition, where robots are designed to understand and speak sign languages. This field aims to bridge the communication gap between the deaf community and the broader population, allowing seamless interaction and engagement. By combining the challenges of action implementation and communication, robotic sign language acquisition serves as a compelling instance where humanoid robots can learn sign languages and interact effectively with both the deaf community and individuals proficient in sign languages. This integration of learning from human demonstration and accommodating diverse communication modalities has the potential to revolutionise the accessibility and inclusivity of robots in our society.

Sign language is a natural language that conveys a message through visual means, as opposed to spoken language which uses sound. As such, it requires people to perform specific gestures and facial expressions. Considering that, according to the World Federation of Deaf “there are more than 70 million deaf people worldwide” (UN), and that according to WHO “by 2050 nearly 2.5 billion people are projected to have some degree of hearing loss and at least 700 million will require hearing rehabilitation” (WHO), it is clear that also machines need to learn how to convey messages using a medium different from sound. The societal benefits of having robots that can speak sign language are extensive and hold immense potential across various fields. In the field of education, these robots can assist in classrooms by providing real-time sign language interpretation, enabling deaf students to fully participate and engage in lessons.

They can also serve as interactive language tutors, helping both deaf and hearing individuals learn sign language effectively. In healthcare settings, sign language-speaking robots can improve communication between healthcare providers and deaf patients, enhancing the quality of care and ensuring that medical information is accurately conveyed. Furthermore, in customer service and hospitality industries, robots capable of sign language can offer personalised assistance and support to deaf customers, ensuring their needs are met without any communication barriers. By integrating sign language into these diverse fields, the societal benefits of these robots extend beyond accessibility and inclusivity, transforming the way we communicate, learn, and interact across various sectors.

Recently, researchers started approaching the problem of robotics sign language, focusing on imitation (Zhang et al., 2022; Hosseini et al., 2019) or translation (Gago et al., 2019a). However, neither of these approaches enables the robot to build an *embodied* representation of different signs based on visual data, as they focus on retargeting rather than learning, or using additional hardware. As an alternative approach to retargeting, Learning from Demonstrations (LfD) (Billard et al., 2008) or Learning from Observations (LfO) (Bentivegna et al., 2004) has gained huge popularity in the robotics community. For example, Yang et al. (2015) teach a Baxter robot how to cook by "watching" YouTube videos or Peng et al. (2020) make a robotic dog learn how to move by imitating an actual robot. Similarly, Peng et al. (2018a,c) perform virtual character animation by teaching a virtual humanoid agent different skills based on motion capture data. One of the main differences between these approaches is that some of them focus on an objective, while others on copying the movements (i.e., one focuses on *what* to perform while the other on *how* to perform it). In addition, the agent **acquires** an **embodied** representation of such skills.

In conclusion, our work is motivated by a gap in the current state-of-the-art in the field of robotics sign language. In particular, there is no research which approaches the problem of sign language acquisition in artificial agents based on imitation.

1.2 Research questions

Having framed the scope and motivation that drive this line of research, here we formulate the main questions that we wish to address through this thesis:

RQ1 Given the lack of 3D annotations for sign language, can a pose estimation model

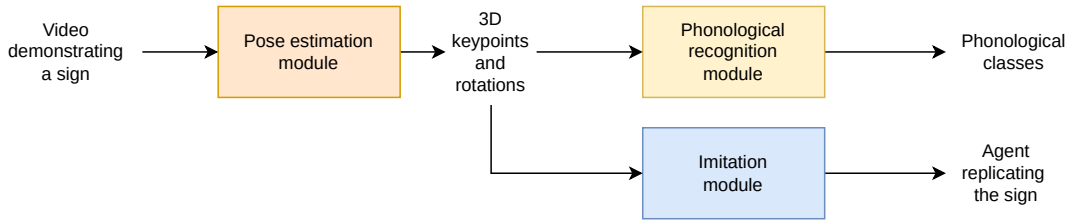


Figure 1.1: Overall architecture of our proposal. We combine a pose estimation module with a classification module to evaluate the poses extracted from a video. Then, we use the data extracted by the pose estimator to enable the imitation module to replicate the demonstrated sign.

extract data which can be recognised in an automated fashion as high-level properties of sign language?

RQ2 Can classification algorithms generalise to a larger set of signs and properties? If so, can they also recognise the same properties over unseen signs?

RQ3 Does information extracted by the pose estimation model constitute a good source to learn fingerspelling on a robotic hand based on imitation?

RQ4 Can the approach adopted to learn fingerspelling generalise to a more complex task like learning signs involving the whole upper body?

1.3 Aims and Objectives

On the one hand, to address RQ1 and RQ2, we propose a set of automated classification models to recognise phonological properties (i.e., the smallest units and the minimal properties that can be used to create signs). The research methodology seeks to:

1. Review the current literature on sign language and computational approaches to sign language, including the available datasets from the literature,
2. Generate a novel dataset in which, for different instances of the same sign, phonological properties are associated,
3. Identify pose estimation models that can be used to extract information from videos of people speaking sign language,
4. Design and implement classification models to recognise in an automated way phonological properties from information extracted from videos, and

5. Evaluate the automated recognition process.

On the other hand, to address RQ3 and RQ4, we propose to use information extracted by the pose estimation models exploited to assess RQ1 and RQ2 to replicate the observed signs. In particular, the research methodology aims to:

1. Survey the current literature on imitation learning and sign language in robotics,
2. Identify the current limitation about robotics sign language approaches,
3. Define an alternative approach to the problem of sign language acquisition in artificial agents based on approaches from computer animation,
4. Model the learning problem for both fingerspelling and signs involving the whole upper-body, and
5. Evaluate the learning process and its generalisability through quantitative and qualitative measures.

1.4 Contributions to Knowledge

The main aim of this thesis is to advance the field of sign language acquisition via imitation learning for robotics by studying whether a simulated agent can learn sign language by observing and imitating people. The contributions to knowledge in this thesis are summarised as follows:

- We introduce a new dataset in which, for each lemma (i.e., dictionary form of a word), a set of phonological properties are associated. Such a dataset is innovative for a computational approach to sign language, as it enables researchers to train models able to recognise properties and automatically annotate new videos. Our dataset is then expanded to test generalisation.
- We conduct the first large-scale attempt at the recognition of phonological classes in sign language based on videos and pre-trained pose estimation models. This work serves as a baseline on which future attempts can build upon to compare and improve automated phonological recognition tools.
- We provide the first attempt to create an embodied representation of fingerspelling and upper-body signing in sign language based exclusively on visual data and reinforcement learning.

- To the best of our knowledge, we create the first artificial agent which **learns** an **embodied representation** – using imitation learning – for each different sign based solely on videos demonstrating such sign.

1.5 Publications

A list of publications, either published or in the process of review, that relate to the contributions of the thesis is given below.

1.5.1 Conference proceedings

1. **Tavella, Federico**, Aphrodite Galata, and Angelo Cangelosi. "Phonology Recognition in American Sign Language." ICASSP 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022 (Tavella et al., 2022a). This paper introduces the methodology to create our dataset and the first attempt at performing recognition of phonological properties of sign language.

2. **Tavella, Federico**, Viktor Schlegel, Marta Romeo, Aphrodite Galata, and Angelo Cangelosi. "WLASL-LEX: a Dataset for Recognising Phonological Properties in American Sign Language." Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 2022 (Tavella et al., 2022b). This paper contains the large-scale version of the dataset introduced in the previous paper, plus additional models and evaluation methodologies on the new large-scale dataset. The authors contributed in the following parts: Tavella conceived the idea, contributed to modelling the problem, retrieved and pre-processed the data, performed part of the experiments and the analysis of the results, and wrote part of the paper; Schlegel contributed to modelling the problem, performed part of the experiments and the analysis of the results, and wrote part of the paper; Romeo performed part of the experiments and wrote part of the paper; Galata and Cangelosi supervised and wrote part of the paper.

3. **Tavella, Federico**, Aphrodite Galata, and Angelo Cangelosi. "Signs of Language: Embodied Sign Language Fingerspelling Acquisition from Demonstrations for Human-Robot Interaction" IEEE RO-MAN 2023 (Tavella et al., 2023). This paper provides a reference implementation for a simulated hand and the use of reinforcement learning to perform fingerspelling based on imitation learning.

1.6 Thesis structure

The thesis organises into 7 chapters. The first two chapters introduce the research problem and the state of the art. The next two chapters describe our methodology for automated recognition of phonological properties in sign language, while the fifth and sixth chapters describe the methodology for sign language acquisition based on imitation learning. Finally, the last chapter summarises our contributions and describes limitations and possible future works. Here follows a more detailed description of each chapter.

Chapter 1 gives an introduction, motivates the research problem we are addressing, frames the fundamental research questions driving our approach. Moreover, it lists the contributions we expect to provide by the end of our research. Such contributions are also provided as a list of selected publications produced during this period.

Chapter 2 provides a literature review on computational approaches to sign language and learning approaches for physically simulated characters. We provide both a brief introduction to sign language from a linguistic perspective, and from a computational point of view, enabling the reader to understand the basics of sign language properties and the methodology used to study such properties. Then, we describe the current state-of-the-art for generating animations for simulated characters. In the end, we specify how our methodology aims to address limitations in the current literature.

Chapter 3 details our methodology for developing a new dataset for the problem of phonological properties recognition from RGB videos. We create a dataset in which each lemma is represented by a single instance, avoiding overlapping between training and test data in terms of signs. Additionally, we illustrate how such data can be used in a supervised learning scenario to perform automated classification of phonological properties.

Chapter 4 expands Chapter 3 by introducing an expanded dataset and more advanced models for the classification task. We include multiple instances of the same lemma. Each repetition of the same sign is performed by a different individual. Moreover, we train and evaluate all the models on two different sets, one that contains the same lemmas in the training and test set and one that does not.

Chapter 5 describes an imitation learning model based on reinforcement learning and 3D motion capture data for fingerspelling sign language acquisition. We develop a simulated hand with a degree of freedom for each joint and estimate the parameters for the controllers. Additionally, we train multiple policies to replicate fingerspelled letters. This experiment serves to verify the transferability of existing approaches to different scenarios.

Chapter 6 brings together our results from previous experiments to teach a simulated agent to perform signs. By leveraging the data extracted from videos and validated using phonological properties, we train a humanoid character with dexterous hands to replicate the observed signs. This final step aims at demonstrating an imitation learning approach for skills that require fine-grained movements.

Chapter 7 concludes the thesis by providing a summary of what we achieved and the contributions to knowledge and the scientific community provided by our research. Finally, we discuss some of its limitations and the potential impact, with possible future directions aimed at providing ideas for additional research by fellow researchers.

Chapter 2

Background

2.1 Introduction

This chapter introduces the theoretical background of this research and the relevant literature domain. However, for the sake of clarity, some topics that are related to specific experiments are presented in the methodological section of their respective chapter. The following sections are organised as follows. **Section 2.2** provides a brief introduction to sign language from a general and linguistics point of view, while **Section 2.3** illustrates different computational approaches to sign language, with an in-depth overview of works on phonological properties. **Section 2.4** focuses on imitation learning for robotics and for generating animations based on motion capture data, and **Section 2.5** provides a summary of the limitations of the state-of-the-art and presents our plan to overcome them.

2.2 Sign language

Sign language (SL) is a natural language which uses visual and motor elements to deliver a message. Around 200 languages in the world are signed rather than spoken (UN), featuring their own vocabulary and grammatical structures. For example, American Sign Language (ASL) is not a mere translation of English into signs and is unrelated to British Sign Language (BSL). From now on, when we talk about sign language, we will actually be referring to ASL, unless differently specified. As with the other forms of natural languages, sign language contains all the fundamental features of a language, like word order/formation and pronunciation. Valli and Lucas (2000)

produced a text used by many researchers as one of the main references for sign language linguistics. Similarly, Sandler (2012) created an international handbook about sign language. In this text, one can find several chapters on all the different properties of language, such as

- **Phonetics:** the low-level production and perception of manual and non-manual signals;
- **Phonology:** study of how the language organises the constituent parts of signs;
- **Prosody:** the part of language that determines how we say what we say. By manipulating timing, prominence, and intonation, we separate constituents from one another and indicate ways in which constituents are related to one another;
- **Morphology**: words, the relationship between words, and the means for creating new words are affected by the particular modality of the language;
- **Syntax:** words order and its functional and articulatory aspects;
- **Lexical semantics:** the field of linguistics that is concerned with the meanings of lexical items or words, and how they mean what they mean;
- **Psycholinguistics** and **neurolinguistics:** relation between linguistic factors and psychological aspects, how the language is processed and represented in the mind and brain. For the sake of completeness, an example of studies on electrical brain activity during signing can be found in Leonard et al. (2020).

Sandler (2012) provide some guidelines on handling sign language data, from data collection to computer modelling. Petitto and Marentette (1991) are among the first to argue for an additional commonality between spoken and signed languages: babbling. In spoken languages, babbling – the attempt at producing articulated sounds – has been associated with the maturation of the articulatory apparatus responsible for spoken language production. Additionally, Petitto and Marentette (1991) argue that manual babbling has an equivalent role for sign language. By analysing deaf and hearing infants' manual activity, they identified two types of manual activity, syllabic manual babbling and gestures (e.g., raising arms to be picked). On a similar note, Mayberry and Squires (2006) present their findings on sign language acquisition. Manual babbling occurs at the same age as vocal babbling. They describe manual babble as a reduced set of phonological parameters that follows the syllabic organisation of sign languages. In

addition, the body location where the sign is executed seems to change on a personal basis. Sandler (2017), while comparing spoken and sign language phonologies (i.e., the study of how languages systematically organise their constituent parts), describes the differences between the articulatory systems of the two types of languages. The most obvious difference is that the articulatory system of sign language can be seen by the naked eye, but the one of spoken language requires an MRI to see the articulators of sign language.

2.2.1 Sign language phonology

Over the past 60 years, linguistics interested in sign language produced interesting results regarding the phonology of this non-spoken language. In 1960, Stokoe (2005) published one of the most influential works regarding sign language phonology. In their work, they provided evidence for a compositional rather than holistic view of sign language. Hence, we could say that sign language can be studied as the sum of the parts, rather than a whole. They proposed three main components to describe signs: location of execution, handshape and type of movement. Moreover, their work demonstrated that sign language has a level of structure that corresponds to the phonological level – i.e. the language consists of a finite list of meaningless units, which combined form all the lexical items in the language. Bellugi and Fischer (1972) provide a comparison of signed and spoken language. On top of providing an intuitive description of the parameters involved in ASL (i.e., location or relationship between hands and movement), they discuss the speech rate for these two forms of language. Interestingly, they found that spoken language is almost twice as fast when compared to sign language on word production, but the rate turns out to be equivalent for propositions due to the possibility of providing information in parallel with signs. In fact, Brentari et al. (2018) attribute this to the fact that spoken language uses sequentially ordered units, while sign languages tend to layer morphological units – i.e., the smallest meaningful unit within a sign – simultaneously. Liddell and Johnson (1989) outline the phonological structures and properties of ASL. They state that a considerable number of signs are produced with changes in hand shape. In addition, they show how it is very common for the hand to move between different locations during the execution of a single sign. For example, a lot of verbs are marked by subject-object agreement and move from one location to another. Brentari (1999) provides one of the main books on sign language phonology. The main contribution of this book is the division of phonological features

of ASL into two categories: inherent and prosodic. Prosodic features are all those features whose sequential articulation results in movement in a sign syllable, such as open and then closed positions of the fingers or making contact with the signing hand on the cheek and then on the chin. Inherent features are all the remaining, characterising, for example, the fingers selected to make the handshape (e.g. index and middle) or the general body area (e.g. the head). Another interesting fact from Brentari et al. (2018) is that people can temporally resolve auditory stimuli when they are separated by an interval of only 2 milliseconds, as opposed to the visual system, which is much slower and requires at least 20 milliseconds to resolve visual stimuli presented sequentially. Additionally, sign language can transmit multiple visual events simultaneously, and there are two hands and arms involved in signing, while speech is transmitted through a single stream of an acoustic signal (Meier, 2002).

2.2.2 ASL datasets

All these studies helped define the different properties of sign language, but one of the limits they all share is that they present only a few examples for each of the identified properties. To foster research in sign language linguistics, especially for a computational approach, researchers need accessible resources with data annotated by experts. Until 2017, there were only two small-scale databases. On one hand, Mayberry et al. (2014) created a list of subjective frequency ratings for 432 signs, but none of them are coded for lexical and phonological properties. On the other hand, Morford and Macfarlane (2003) produced a set of over 4000 signs for frequency in ASL, but it is not publicly available. Caselli et al. (2017) produced a database of lexical and phonological properties of American Sign Language obtained through a study involving both deaf and hearing participants who were asked to assign some properties to videos of people performing signs. For each of the words and/or lemmas in the database, several different features are available: video clip duration, sign length, grammatical class and phonological properties, such as sign type, selected fingers, flexion, major/minor location, or movement. Additionally, they built an online tool[1] which shows phonological sign similarity using a graph. Figure 2.1 illustrates an example of this tool. Each sign is represented by a node, connected to phonologically similar signs. Complementary, Table 2.1 provides an intuitive description of each different phonological feature. Additional information about every single class can be found in Appendix B.

[1] https://ben-tanen.com/asl-lex/visualization/

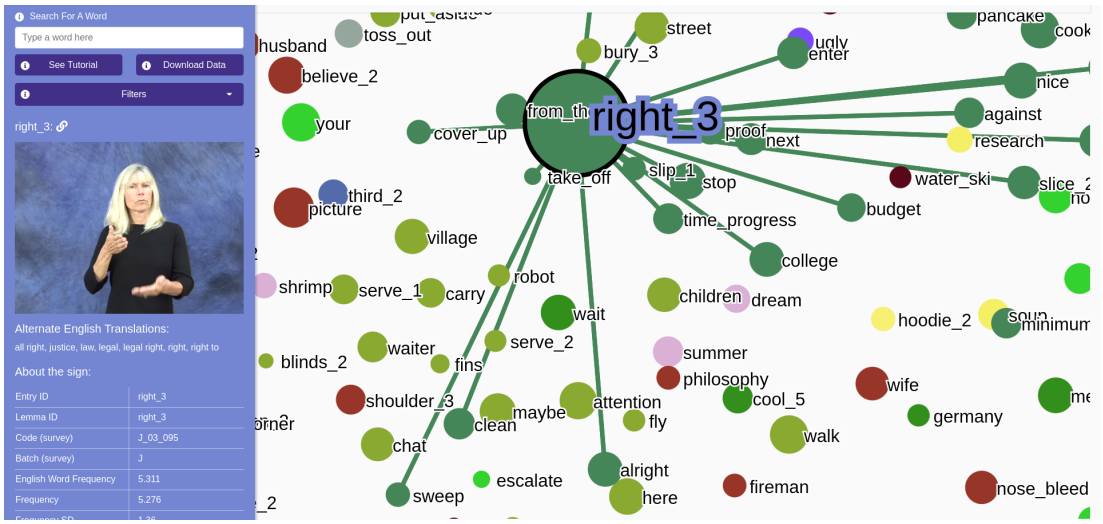


Figure 2.1: Screenshot of the ASL-Lex visualisation tool. For each different word, it is possible to visualise an example of that word in ASL, several different lexical properties and similar signs. Signs that share similar lexical properties are indicated with the same color and connected by lines.

Finally, Sehyr et al. (2021) introduced an updated version of ASL-Lex (namely, ASL-Lex 2.0) with additional signs and phonological properties.

2.3 Computational approaches to sign language

Research on Sign Language Processing (SLP) encompasses tasks such as sign language detection, i.e. recognising if a signed language is performed (Moryossef et al., 2020), and sign language recognition (Koller, 2020), i.e. the identification of signs either in isolation or in continuous speech. Other tasks concern the translation from signed to spoken (or written) (Camgoz et al., 2018) language or the production of signs from text (Rastgoo et al., 2021). The fact that sign languages have a non-textual nature introduces many challenges to their automated processing, compared with purely textual Natural Language Processing (NLP). With the recent success of deep learning-based approaches in computer vision (CV), as well as advancements in related tasks of action and gesture recognition (Asadi-Aghbolaghi et al., 2017), SLP is gaining more attention in the CV community (Zheng et al., 2017).

Metadata	Definition
Lexical Class	The sign's lexical class (Adjective, Adverb, Name, Noun, Number, Verb, and Minor: Preposition, Pronoun, Conjunction)
Compound	Compounds are signs that include more than one free morpheme, and morpheme boundaries are often indicated by a change in selected fingers or major location.
Initialized	Indicates that the handshape of the sign is the first letter of the English translation
Fingerspelled Loan Sign	Indicates that the sign includes more than one letter of the manual alphabet.
Sign Type	Symmetry of the hands according to Battison's sign types.
Major Location	General location of the dominant hand at sign onset
Minor Location	Specific location of the dominant hand at sign onset
Selected Fingers	Fingers that are moving or foregrounded in the first morpheme of the sign; Thumb is ignored unless it is the only selected finger in the sign
Flexion	Aperture of the selected fingers of the dominant hand at sign onset
Movement	Path movement of the first morpheme in the sign

Table 2.1: Metadata description from ASL-Lex (Caselli et al., 2017).

2.3.1 Vision-centric approaches to sign language processing

Cooper et al. (2011) argue that during the past decades, speech recognition has advanced to the point of being widely commercially available, but that is not the case for Sign Language Recognition. They also report how many of the initial approaches to SLR used datagloves and accelerometers to acquire data about the hands, but due to prohibitive costs, the use of vision became more popular. For example, combination of cameras like monocular (Zieren and Kraiss, 2004), stereo (Hong et al., 2007), orthogonal (Starner and Pentland, 1995). Moreover, they point out how Hidden Markov Models have been the go-to technique for SLR since the mid-90s. Zheng et al. (2017) provide a general framework for SLR based on three steps: sign gesture representation, feature extraction and classification. Additionally, they list the most used devices like data glove (Mohandes et al., 2017), accelerometer (Zhang et al., 2011), Leap motion controller (Chuan et al., 2014) and Microsoft Kinect (Lai et al., 2012). In their survey, Koller (2020) collected information about around 300 published sign language recognition papers. To summarise:

- Most of the studies focus on isolated rather than continuous sign language recognition,
- Most studies on isolated SLR analyse a dictionary with less than 50 words,
- after 2015, there were 40 results published tackling vocabularies larger than 1000 signs, but 34 of these 40 are about continuous SLR,
- before 2005, most studies used either gloves or mocap (short for motion capture) data, now most use RGB images, shifting from intrusive to non-intrusive tools,
- Most of the studies use the following four features
  - Location where the sign is executed,
  - Movement of the hands,
  - Hand shape,
  - Hand orientation;
- Recent papers prefer using end-to-end approaches rather than manually engineered features,
- From 2015 to 2020 the Word Error Rate (WER) on RWTH-Phoenix Weather 2014 dataset (Koller et al., 2015), the main dataset for continuous SLR, decreased from above 50 to around 20.

In addition, Rastgoo et al. (2021) provide an extensive survey on SLR. They provide both a taxonomy of deep models for sign language recognition (Figure 2.2). Moreover, they provide a list of sign language datasets containing images and videos, both for isolated and continuous sign language. Thus, we can notice how there are several different approaches to SLR based on when the recognition is performed (online vs offline) or the input data (cameras or wearable devices) (Starner et al., 1998; Vogler and Metaxas, 1999; von Agris et al., 2008). For example, fingerspelling recognition can be executed in real-time using RGB and/or depth data and different machine learning models (Pugeault and Bowden, 2011; Shi et al., 2019; Kang et al., 2015; Kim et al., 2013, 2016), or by using synthetic data to train the classificator (Tu et al., 2021). Moreover, recognition can be performed using Hidden Markov Models (HMM) (Grobel and Assan, 1997; Zafrulla et al., 2011), WiFi and CNNs (Ma et al., 2018), transformers (Camgoz et al., 2020), temporal graph neural networks (Li et al., 2020b) or increasing performance adding mouthing cues (Albanie et al., 2020).

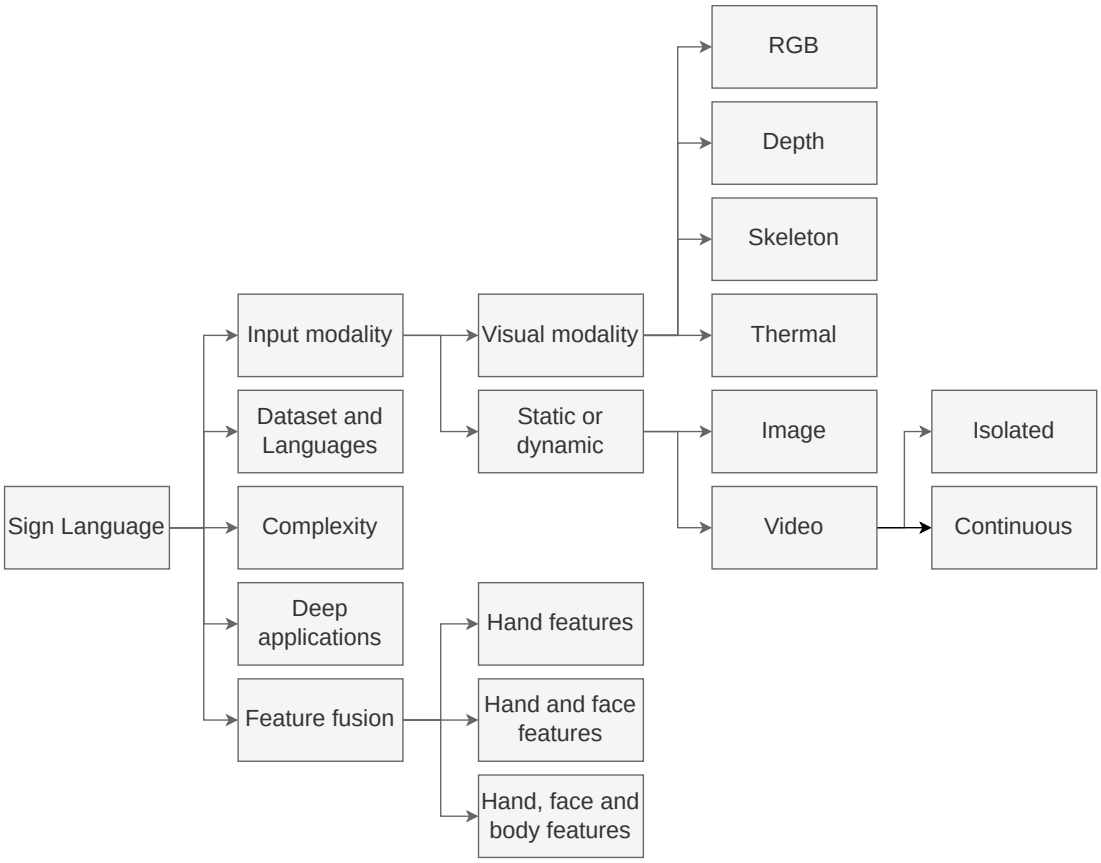


Figure 2.2: Taxonomy of deep learning models for sign language recognition from Rastgoo et al. (2021). Approaches can be divided based on input features, application, dataset, languages or type of data.

Most of the computational research in sign language has focused on the SLR of single letters - i.e., fingerspelling - or words from images or videos (Zheng et al., 2017; Lim et al., 2019; Ferreira et al., 2019), sentence level recognition (Pu et al., 2020; Li et al., 2020b), hand and facial features (Tornay et al., 2019; Kimmelman et al., 2020), iconicity (Östling et al., 2018), markers of questions (Kuznetsova, 2021), translation from sign to spoken language (Camgoz et al., 2018) and vice-versa (Stoll et al., 2020). This led to online tools based on machine learning that enable people to learn to fingerspell the alphabet using a webcam (Fingerspelling with Machine Learning) or to detect sign language in online videoconferences (Moryossef et al., 2020). Thus, SLR tools can significantly improve interaction for SL users who communicate remotely with non-sign language users.

In a recent position paper for the computational linguistics community, Yin et al. (2021) pointed out how over the last 20 years there has been a steady increase of

publications in Computer Science referring to sign language in their title, as opposed to the ones in ACL Anthology, which remained mostly constant. They identify the tasks on which researchers focused so far, like

- Detection,
- Identification,
- Segmentation,
- Recognition,
- Translation,
- Production.

Moreover, they identify four macro areas on which researchers should focus to improve SLP:

1. The need for an efficient tokenisation method,
2. Development of linguistically-informed models,
3. Collection of real-world signed data, and
4. Inclusion of signed language communities as an active part in the direction of research.

In fact, it is quite trivial to notice how research in sign language has been driven by the computer vision community so far. A more linguistically-informed approach could provide more accurate and efficient models.

2.3.2 Linguistically-informed approaches

Some recent approaches to various SLP tasks implicitly rely on *phonological* features (Metaxas et al., 2018). Surprisingly, however, little work has been carried out on explicitly modelling the phonology of signed languages. This presents a timely opportunity to investigate signed languages from the perspective of computational linguistics (Yin et al., 2021). In the context of signed languages, phonology typically distinguishes between manual features, such as usage, position and movement of hands and fingers, and non-manual features, such as facial expressions. Sign language phonology is a

matured field with well-developed theoretical frameworks (Liddell and Johnson, 1989; Sandler, 2012). These phonological features, or *phonemes*, are drawn from a fixed inventory of possible configurations which is typically much smaller than the vocabulary of signed languages (Borg and Camilleri, 2020). For example, there is only a limited number of fingers that can be used to perform a sign due to anatomical constraints. Hence, different signs share phonological properties and well-performing classifiers can be used to predict those properties for signs unseen during training. This potentially holds even across different languages, because, while different languages may dictate different combinations of phonemes, there are also significant overlaps (Tornay et al., 2020). Finally, these phonological properties have a strong discriminatory power when determining signs. For example, in ASL-Lex (Caselli et al., 2017), a lexicon which also captures phonology information, the authors report that more than 50% of its 994 described signs have a unique combination of only six phonological properties and more than 80% of the signs share their combination with at most two other signs. By relying on this phonological information from resources such as ASL-Lex, many signs can be uniquely determined. This means that well-performing classifiers can leverage this information to predict signs without having encountered them during training. This is a capability that current data-driven approaches to SLR lack by design (Koller, 2020). Thus, in combination, mature approaches to phonology recognition can facilitate the development of sign language resources, for example by providing first-pass silver annotations for new sign languages based on their phonological properties. This is an important task for both documenting low-resource sign languages as well as the rapid development of large-scale datasets, and for fully harnessing data-driven CV approaches. Östling et al. (2018) propose a similar approach to ours, but they focus on iconicity, in particular plurality and body location, across 31 sign languages. More in detail, Östling et al. (2018) process 120000 videos of signs in 31 different languages automatically, looking for cross-linguistic patterns for two specific type of iconic-form meaning, namely plurality (and its relation to the number of hands involved) and location around specific part of the body (and its relation to what the sign represents). Thus, while they also conduct a large-scale recognition of properties based on keypoints extracted by a pose estimator, they do not use data annotated by linguistics to do so, but they rather define their own classes (i.e., singular vs plural) and visualise how different concepts are signed over related parts of the body (e.g., the ears for signs representing the concept of "hear" or the belly for signs related to the concept of "hungry". Moreover, our approach based on 3D mesh regression gives us an additional

advantage, which is robustness to pose variance. Cooper et al. (2011) perform sign language recognition based on lexical sub-units using a two-stage approach. They extract 2D and 3D features using a Kinect sensor and train Hidden Markov Models (HMMs) and Sequential Pattern Boosting (SPB) classifiers to recognise a sign based on the sub-units. To do so, the authors build an ad hoc dataset, which contains 984 Greek Sign Language signs with 5 examples of each performed by a single signer. Metaxas et al. (2018) use Conditional Random Fields (CRFs) to perform SLR, by creating a feature vector composed of 2D and 3D features extracted from videos via Convolutional Pose Machines and additional linguistically relevant information. Camgoz et al. (2018) train a combination of CNN, LSTM and Connectionist Temporal Classification (CTC) in an end-to-end fashion to recognise phonologically meaningful properties to perform SL handshape recognition. Similarly, Borg and Camilleri (2020) uses a combination of tracking algorithms and RNN to perform word-level recognition based on subunits of hand pose and motion. Interestingly, they point out how the number of subunits in sign language is much smaller than the number of words. Thus, performing sign language recognition based on subunits (i.e., phonological properties) instead of words is potentially much more efficient in terms of training data and the number of possible outcomes, similarly to what is done in Optical Character Recognition (OCR), where models are trained on letters instead of words. For example, training a classifier to distinguish 10 different signs could be done with two different phonological classes, where class A has 2 possible values and class B has 5. This would require 7 examples instead of 10. None of the aforementioned studies takes advantage of existing datasets containing phonological properties of sign language that were manually annotated by SL users. Consequently, we aim to address these gaps in research by exploiting pose estimation models which automatically extract keypoints coordinates from videos, and by recognising patterns in sequences of keypoints using as ground-truth phonological properties annotated by sign language researchers and users. These properties give a finer level of granularity about the performed movements than word, which on the contrary only label the whole sequence.

2.4 Robots and machines that learn from others

Peters et al. (2016) define robot learning as "a multitude of machine learning approaches in the context of robotics". The type of learning problem is usually characterised by the type of feedback, the process of data generation, and the type of data.

At the same time, the type of data will determine the robot learning approach which can be actually employed. Ideally, rather than programming an infinite amount of different actions, we would like machines to *learn* through examples, much like humans do. Often, this process is called *learning from demonstrations* (Schaal, 1996), *imitation learning* or *learning by imitation* (Byrne and Russon, 1998). Schaal (1999) has been one of the first to advocate for imitation learning in robotics. They theorise that studying imitation learning could help gain new insight into mechanisms of perceptual-motor control, which in turn would result in the creation of autonomous humanoid robots. Imitation learning focuses on three relevant issues:

1. Efficient motor learning,
2. The connection between action and perception, and
3. Modular motor control in the form of movement primitives.

However, according to Tomasello et al. (1993), we can talk about actual imitation if

- The imitated behaviour is new for the imitator,
- The same task strategy as that of the demonstrator is employed, and
- The same task goal is accomplished.

Schaal (1999) argues that for imitation a connection between the sensory systems and the motor systems is fundamental, such that percepts can be mapped to appropriate actions. In addition, they allude to movement primitives – i.e., sequences of actions that accomplish a goal-directed behaviour – as a possible way to map the demonstrator's movements to action. On a similar note, Bentivegna et al. (2004) explore how primitives can help to accelerate robot learning and enable robots to learn complex and dynamic tasks from observing demonstrations, like playing air-hockey and marble maze. They propose a 3 components approach: primitive selection, sub-goal generation and action generation. The former is a classifier that uses the current state to choose the primitive to execute. The second is a module that indicates a goal of performing the chosen primitive. The third specifies the actuator commands to achieve a sub-goal. Ratliff et al. (2007) present an innovative application of gradient-based technique for locomotion and manipulation, which, with the advent of machine learning, have been widely used in robotics, as opposed to the previous heavily engineered and hand-tuned techniques for dynamics based on control theory. In fact, reinforcement learning (Sutton and Barto, 2018) and its subbranches (among which some particular instances of

imitation learning) drove many innovations in robotics. For example, Ho and Ermon (2016) propose Generative Adversarial Imitation Learning (GAIL) as a potential solution to the problem of the high number of samples needed for imitation learning. They propose a one-shot imitation learning approach that is task-agnostic, hence able to generalise to new tasks based on demonstrations. Finn et al. (2017) also contribute to improving the efficiency of imitation with a meta-imitation learning that scales to raw pixels and acquires new skills from a single demonstration. Rajeswaran et al. (2018) provides a great example of how to combine reinforcement learning and learning from demonstrations in robotics. Using a 24 Degrees of Freedom (DoF) hand, they demonstrate that Deep Reinforcement Learning (DRL) can scale up to solve complex tasks and that they can reduce the complexity of the task by exploiting a small sample of human demonstrations (collected using special gloves and a headset). The use of demonstrations results in policies that exhibit very natural movements and are also significantly more robust. Furthermore, they find that their algorithm can be up to 30 times more efficient (in terms of samples) than RL from scratch with shaped rewards. Another example of how a headset can be used to instruct robots come from Zhang et al. (2018). They describe a system to teleoperate a robot based on a consumer-grade Virtual Reality (VR) headset, which can also be used to collect demonstrations that, fed to a behavioural cloning (i.e., a specific instance of imitation learning) algorithm, can in turn be used to generate a policy representing the acquired skills. Finally, Hussein et al. (2017) provides an extensive survey on imitation learning, while Fang et al. (2019) describes different approaches for robotic manipulation for imitation learning.

2.4.1 Sign language in robotics

Despite all the recent progress, the field of robotics sign language is still very scarce. Currently, there is a very limited amount of research on sign language acquisition in robotics. Most of the research so far focused on robot tutors to aid sign language acquisition in humans (Kose et al., 2011, 2012; Köse et al., 2015; Uluer et al., 2015; Zakipour et al., 2016; Scassellati et al., 2018; Zhi et al., 2018; Gago et al., 2019b; Meghdari et al., 2019; Luccio and Gaspari, 2020). In comparison, very few tried to make robots speak sign language (Lo and Huang, 2016; Hosseini et al., 2019; Gago et al., 2019a; Liang et al., 2021). Notably, Lo and Huang (2016) present an imitation system to teach a humanoid robot to perform sign language by replicating observed demonstrations. By using a Kinect camera and Leap motion sensor, the robot can

mimic demonstrated signs based on 3D keypoints and inverse kinematics. Alternatively, in Hosseini et al. (2019), the user wears a motion capture suit and performs a sign multiple times to train a set of parallel Hidden Markov Models to encode each sign. Additionally, the authors ensured signs comprehensibility and collision avoidance via a special mapping from the user's workspace to the robot's joint space. The performance of the system was assessed by teaching 10 signs in Persian Sign Language (PSL) to the robot and involving PSL users to investigate how easy was for them to recognise the signs. 8 participants out of ten managed to recognise the signs during the first attempt, while 10 out of 10 did so during the second attempt. Gago et al. (2019a) introduce a pipeline to carry out humanoid robot sign language using a sequence-to-sequence approach. By feeding a tokenised text sequence in the sequence-to-sequence model, they manage to generate Spanish sign language (LSE) tokens, which are in turn used as keys for a look-up table to retrieve the desired signs. They populate the look-up table in an offline fashion, by recording the joint space that corresponds to each LSE token. Liang et al. (2021) describe a novel motion retargeting approach based on optimisation and Dynamic Movement Primitives (DMPs) for two robotic arms. The proposed method consists of three different steps: modelling human arm movements, learning DMPs in a leader-follower manner from the instructor's demonstrations, and optimisation of initial and goal positions of DMPs. Zhang et al. (2022) propose a method for robots to "learn" sign language by combining learning and optimisation-based motion retargeting. They establish a mapping relationship between the latent space and the robot motion space with a graph decoder. Using the difference between the human and robot motion, they search for the optimal latent code that minimises the gap by gradient descent. Practically, they build an autoencoder and define multiple losses to minimise the difference between human and robot skeletons in terms of joints orientation and end effectors positions. Hence, despite the advancements in robotic technologies, there remains a glaring lack of research focused on robots that can acquire sign language through imitation. Imitation is a crucial learning mechanism for humans, allowing us to observe and mimic the actions of others to acquire new skills. However, this fundamental aspect of learning has not been extensively explored in the context of robotic systems and sign language acquisition. To address this gap, we draw inspiration from a related field, character animation in computer graphics based on reinforcement learning.

2.4.2 Agile characters based on demonstrations for computer animation

Peng et al. (2020) present an imitation learning system that enables legged robots to learn agile locomotion skills by imitating a real dog. By leveraging reference motion data, the authors demonstrate that their approach can automatically synthesise controllers for a diverse set of behaviours for legged robots. Additionally, they can learn adaptive policies in simulation that can be efficiently adapted for real-world scenarios by incorporating domain adaptation techniques into the training process. Thus, it is possible to use reference motion data from the real world to train virtual agents, and to transfer the simulated controllers to real-world use cases. Is it possible to use the same approach for a humanoid robot? In the recent literature, we can find several examples of learning-based approaches to control humanoid characters. For instance, Holden et al. (2017) present a real-time character control mechanism using a novel neural network architecture called a Phase-Functioned Neural Network. In their new approach, they provide as additional input the phase (i.e., a cyclic function like sine or cosine), so that the character can cyclically reproduce the behaviour. For example, walking forward is the repetition of two basic movements in repetition: alternating moving one foot forward after another. In addition to the phase, the authors provide as an input the user's control (e.g., move left, right) and the geometry of the scene to generate motions that reflect the desired control. Lee et al. (2018); Bergamin et al. (2019) provide additional examples of how motion capture data can be used to train a controller but use a supervised learning approach instead of reinforcement learning. Most notably, Peng et al. (2018a,c) developed a state-of-the-art approach to skills acquisition. In Peng et al. (2018a), the authors trained a policy (i.e., a phase-functioned neural network (Holden et al., 2017)) using reinforcement learning based on motion capture data. The motion capture data are acquired from an online library and include skills like running, jumping, doing a backflip, etc. Authors define ad-hoc weighted rewards for different components (e.g., pose, movement velocity, end effectors positions), which indicate to the agent the goodness of the chosen action. Moreover, they artefact a skill selector providing the motion identifier as an input to the network and selecting the clip returning the maximum reward, so the agent can switch between different skills according to the identifier. They also propose Random State Initialisation (RSI) and Early Termination (ET) to improve the efficiency of the learning algorithm. Finally, in Peng et al. (2018c), the authors included a pose estimator and a motion reconstruction component to extract data from RGB videos from YouTube and use them to train the control

policy. Lee et al. (2019) built a musculoskeletal model and its control system that reproduces realistic human movements driven by muscle contraction dynamics. In their work, the authors discuss simulating anatomical features and control of under-actuated dynamical systems based on deep reinforcement learning. In the end, they provide a learning algorithm to control a model with 346 muscles. Merel et al. (2019a) address the control problem by integrating perception, motor control and memory. They divide the problem in low-level motor control from proprioperception (i.e., the sensation of body position and movement (Tuthill and Azim, 2018)) and high-level coordination of low-level skills. Subsequently, Merel et al. (2019b) focus on learning a single motor module that can be used for a range of behaviours for the control of simulated humanoids, while Merel et al. (2020) evolve the scenario by involving tasks with objects interaction. They create an environment based on realistic actuation and first-person perception (including touch sensors and egocentric vision) with gaze direction. The whole framework is based on a motor primitive module, human demonstrations, and reinforcement learning. Nevertheless, the object interaction is not based on grasping skills, but on fetching objects using the hand fully open. Hasenclever et al. (2020) address the problem of learning reusable humanoid skills by imitating motion capture data and joint training with complementary tasks. They achieve to learn reusable skills through reinforcement learning on 50 times more motion capture data than prior work. Won et al. (2020) develop a technique for learning controllers for a large set of different behaviours. They divide the library of motions into clusters of similar motions and train a different policy for each cluster, which combined can reproduce the whole library. Then, Won et al. (2021) develop a policy model based on an encoder-decoder structure that incorporates an autoregressive latent variable, and a mixture-of-experts decoder. They use a two step-approach, learning first skills and then strategies, inspired by the way that people learn. They apply their approach to simulate two-players competitive sports, like boxing and fencing. Peng et al. (2021) try to obviate the need to manually design rewards and tools for motion selection by using an automated approach based on adversarial imitation learning. The adversarial RL procedure automatically selects which motion to perform, dynamically interpolating and generalising from the dataset. Finally, Peng et al. (2022) combine techniques from adversarial imitation learning and unsupervised reinforcement learning to develop skill embeddings that produce life-like behaviours. Most interestingly, they achieve significant speed-up by parallelising the training by leveraging a novel GPU-based simulator. Using a GPU-based simulator addresses one of the main shortcomings of the approaches based on reinforcement

learning, which is the extensive time required to train a single policy.

In conclusion, we aim to address the shortcomings of current approaches in robotics sign language acquisition, which we described in the previous sections, by drawing inspiration from the field of character animation. By doing so, we aim to address the acquisition problem as a learning problem which resembles how humans learn.

2.5 Research gaps and plan

Considering the current state-of-the-art, we identify the limitations which we aim to overcome:

- There is a clear lack of data for computational approaches to sign language (Section 2.2.2), as the field is severely understudied and sign language is treated as a minority language. Thus, progress in this field has been far slower than speech recognition or natural language processing. Additionally, the vast majority of studies regarding computational approaches to sign language recognition treat it as an action recognition problem, using a computer vision perspective rather than a linguistic one (Sections 2.3 and 2.3.2). A more linguistic-oriented approach could potentially take advantage of the properties of sign language. Currently, to the best of our knowledge, there are also no tools for the automatic annotation of such properties.

- Most of the current approaches for SLR focus on either fingerspelling recognition or end-to-end word-level recognition. While some of these approaches reach significant performance, they do not take advantage of the inherent properties of sign language, which are significantly studied by linguistics (Sections 2.3 and 2.3.2). Moreover, lots of studies focus on small and ad-hoc vocabularies.

- A significant chunk of research for robot learning has focused on either motion retargeting or learning based on goal-oriented tasks (Section 2.4.1). Very little research is based on the imitation of fine-grained movements based on human demonstration from RGB-only data, which in turn would address the problem with a learning approach more similar to human learning (i.e., only visual data) than approaches with multi-modal data.

- Even when considering both the fields of robotics and computer animation, there appears to be a very limited amount of work which involves control of both

body and hands. So far, the two tasks have been treated separately, but current approaches seem to have the potential to treat both body and hand control at the same time or in a parallel fashion.

Thus, given these open problems, we propose an imitation learning approach based on human demonstration from RGB videos. Sign language acquisition appears to be a perfect candidate to address these issues, as the exact replication of the movements is fundamental to convey the same message as the demonstrator. However, in order to be able to extract the necessary information for the imitation, we first study the properties of sign language from a computational perspective, in order to learn what can and cannot be recognised by machine learning models from the source RGB videos. Consequently, in this thesis, we approach the problem in the following way:

1. We create a preliminary dataset of phonological properties to study the recognisability of keypoints sequences,
2. We expand our finding to a larger dataset with additional phonological properties, to further investigate if fine-grained movements can be captured by pose estimation algorithms and recognised by classification models,
3. We devise an imitation approach for fingerspelling acquisition based on demonstration, and
4. We adapt such approach to signs involving the whole upper body.

In this way, we address the lack of data for computational approaches, and we show how such data can be used for linguistically-informed computational approaches. Moreover, we demonstrate how approaches from characters animation can be adapted to sign language acquisition, targeting the problem of human-like learning for sign language, as opposed to current retargeting approaches.

Chapter 3

Phonology recognition

3.1 Introduction

We discussed about sign language, computational approaches for sign language recognition and approaches in robotics and computer animation for learning from demonstration, as well as their application to sign language acquisition. A clear shortcoming in computational approaches to SL is that they do not take advantage of the properties that are inherent to the language, like phonological properties.

Figure 3.1 illustrates 4 different signs that share some common phonological features. Clearly, some of these signs can be distinguished by where the sign is executed (e.g., on the head or the torso) and how many hands they do involve. However, even if different signs are divided by one or more of these features, they usually share some similarities. Thus, multiple signs can be used as examples of the same properties, while they obviously represent examples of different lemmas. In addition, it is worth noticing that recognising specific fine-grained patterns in movements (like phonological properties) can also be a way to evaluate how good pose estimation models are at extracting such patterns from videos.

In this chapter, we propose a mesh regression approach (Kanazawa et al., 2019) to extract 3D temporal features from videos of people speaking American Sign Language (ASL) and a statistical and deep learning approach to perform a preliminary recognition of phonological classes based on the extracted features and ground truth assigned by ASL speakers. In particular, we use the mesh regression approach to extract 3D keypoints (i.e., joint coordinates) of the upper body from video of people performing signs. Then, we use such keypoints as an input for our supervised classification algorithms, which are trained to recognise phonological properties based on the sequences

(a) Sign for "remember". The sign is performed with one hand, and it is located on the head.

(b) Sign for "rest". The sign is performed symmetrically and it is located on the upper body.

(c) Sign for "compass". The sign is performed asymmetrically with different handshapes and it is located on the hand.

(d) Sign for "monster". The sign is performed symmetrically and it is not on any particular area of the body.

Figure 3.1: Comparison of four different signs and their phonological properties. We can see how different signs can be distinguished over different features (e.g., location of the sign), but also distinguished using other features (e.g., number of hands).

of 3D keypoints. Figure 3.2 illustrates an example of our approach.

We choose to infer phonological classes instead of words for the following reasons:

- Phonological properties of sign language are severely understudied in computational linguistics,
- Recognising phonological properties rather than whole signs gives a more precise indication of which movements are (not) tracked properly by the pose estimation algorithm,
- Training is more efficient in terms of resources (i.e., the number of phonological

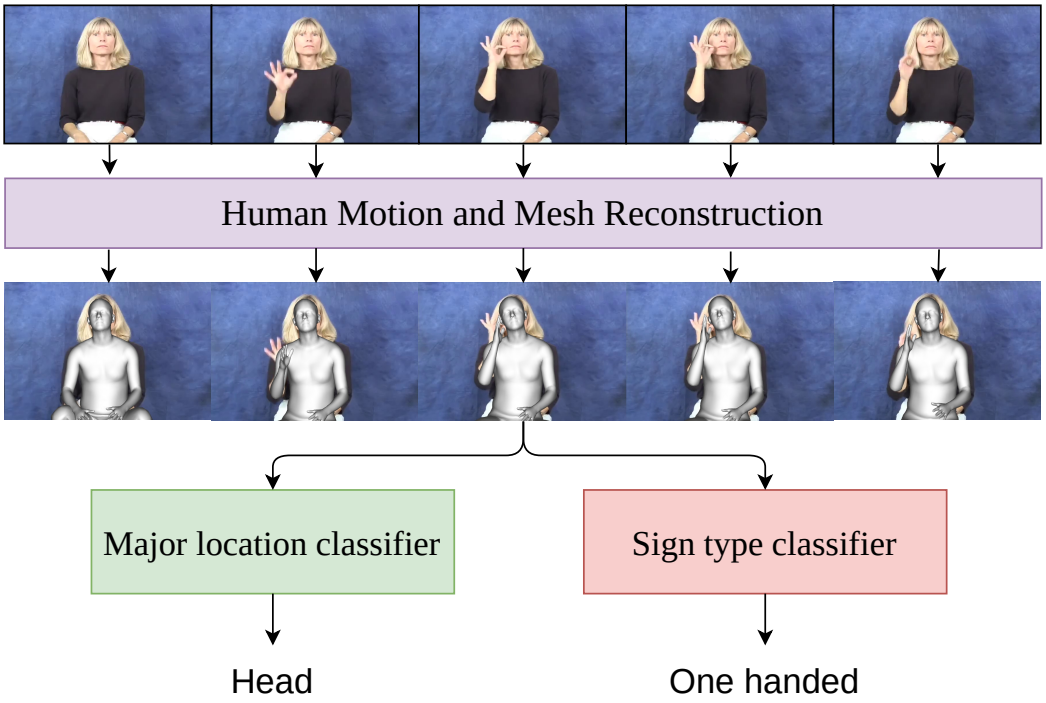


Figure 3.2: Our approach extracts a mesh (and 3D coordinates) from videos of people speaking ASL and uses the keypoints to classify the video according to 2 phonological classes. For example, the sign "cat" is composed of a back-and-forth movement executed with one hand near the head.

classes is much smaller than the number of words),

- It improves the interpretability of the SLR models by providing specific features for each sign,
- Being able to recognise phonological classes based on body movements can help to develop automated tools to teach people sign language by receiving automated feedback and improve tools for sign language generation, and
- Developing a new tool could help linguists to perform new studies on sign language.

3.2 Methods

Our aim is to be able to assign phonological classes to each video. We divide our approach into different stages: data preparation, feature extraction, and model selection. We describe the available data and how we assign the labels for classification, and we

illustrate the process used to extract 3D temporal features from the videos and define our models used for the classification of temporal sequences.

3.2.1 Data

WLASL (Li et al., 2020a) is one of the largest datasets available composed of people demonstrating words in ASL. Each video has a *lemma* (i.e., dictionary form of a word) assigned as a label. However, we want to be able to identify phonological classes (e.g., the location where the sign is executed and the number of hands used to sign). Fortunately, ASL-Lex (Caselli et al., 2017) associates to each different lemma several lexical and phonological properties. As a preliminary study in this direction, we are interested in phonological classes that can be represented using the full body. As such, we picked the following 2 classes (i) *major location*, which indicates the general location of the dominant hand at sign onset, and (ii) *sign type*, which represents the symmetry of the hands according to Battison's sign types (Battison, 1978).

Combining information from WLASL and ASL-Lex, our dataset is composed of a set of lemmas and their phonological properties obtained by cross-referencing the two datasets. It is composed of 790 videos and 993 lemmas with phonological properties. By matching the lemmas with the labels of the videos, we obtain a final dataset composed of 725 unique entries. Figure 3.3 shows the number of lemmas and possible values for the selected classes. The fact that there are no duplicate lemmas in the dataset means that there are no videos representing the same sign. Consequently, there are no two identical 3D sequences in the training, validation and test sets, reducing the possibility of introducing a bias in our evaluation (i.e., data leakage).

3.2.2 Pose estimation

In order to extract 3D keypoints, we use the Human Mesh and Motion Recovery (HMMR) algorithm (Kanazawa et al., 2019), illustrated in Figure 3.4.

Given a video as input, HMMR extracts per-frame features ϕ_t using a ResNet (He et al., 2016) network. Then, the features are used to train a temporal encoder f_{movie} so that it learns a representation of the 3D human dynamics Φ_t over a temporal window centred at the current frame t. Then, we can regress the 3D human shape and pose Θ_t based on the Skinned Multi-Person Linear (SMPL) model (Loper et al., 2015), but also the change in the pose of the adjacent frames $t \pm \Delta t$. In addition, the authors learn a hallucinator $h : \phi_t \rightarrow \Phi_t$, whose goal is to hallucinate the movie strip representation

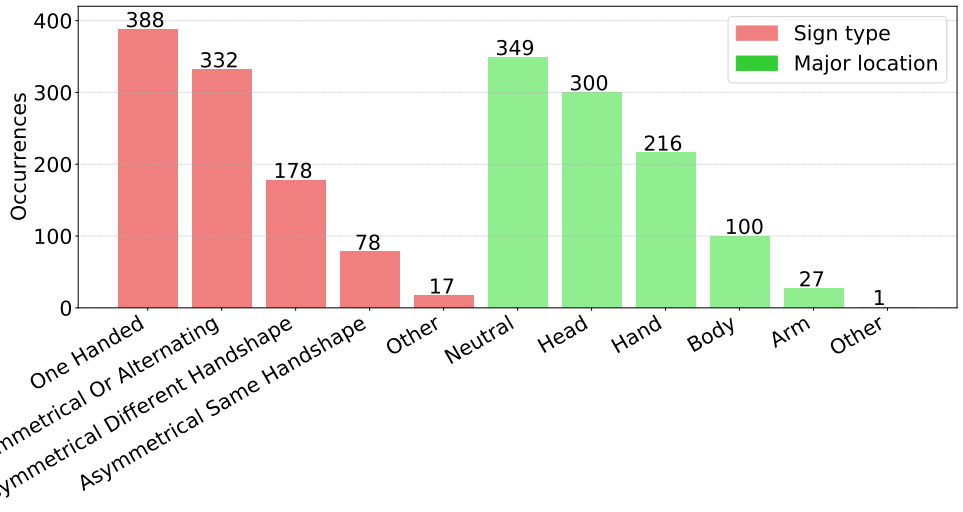


Figure 3.3: Number of samples for the phonological classes "major location" and "sign type" in ASL-Lex. For each class, we provide the number of samples for each different value.

from a static image feature ϕ_t at test time. Briefly, the SMPL model (Loper et al., 2015) is a state-of-the-art body model for human pose, shape, and motion. It is a highly-detailed and highly-realistic model that represents the human figure in 3D. The SMPL model is constructed from a combination of body shape parameters and poses parameters. The body shape parameters define the overall size and proportions of the body, while the pose parameters define the position and orientation of each body joint. One of the key features of the SMPL model is its ability to model a wide range of body shapes and poses. The model can be customised to closely match the appearance and motion of a real person, allowing it to be used in applications such as motion capture and animation.

Using a model such as SMPL brings the advantage of inferring parameters representing the human body, instead of directly regressing keypoints. Additionally, this enables users to compare different algorithms that use the same standardised output model (i.e., SMPL). Given the lack of 3D annotations regarding human poses, especially for videos, the HMMR is trained using heterogeneous datasets not related to ASL (Ionescu et al., 2014; Zhang et al., 2013; Fouhey et al., 2018; Kanazawa et al., 2019). For our purposes, we use HMMR in order to extract 3D coordinates for joints of the upper part of the body, namely the head, chest, shoulders, elbows and wrists. For example, Figure 3.5 shows the projected 3D coordinates with respect to the time of the sign "house", obtained via the HMMR model.

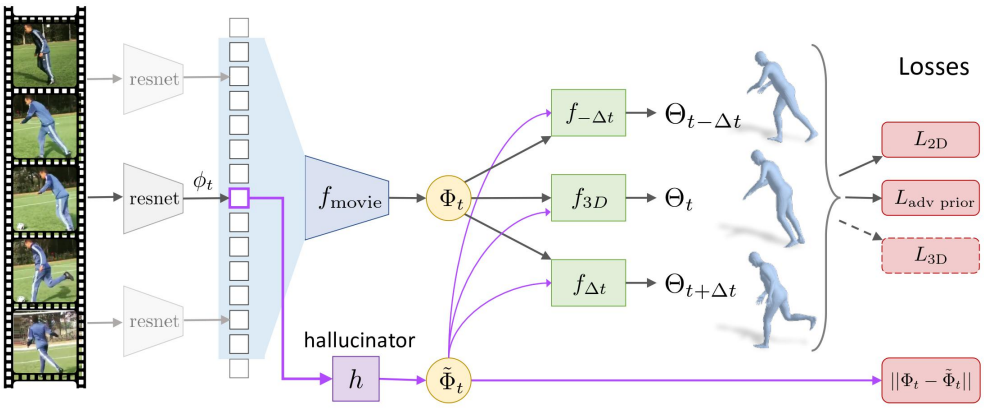


Figure 3.4: Architecture of the Human Mesh and Motion Recovery (HMMR) framework (Reprinted from (Kanazawa et al., 2019)). Taking different frames as input, a feature-extractor produces a representation for each frame, which are then combined to take into account the temporal component.

3.2.3 Classification algorithms

While assigning phonological labels can be - and has been - done manually (Caselli et al., 2017; Sehyr et al., 2021), an automated approach would be much more efficient, removing the labour-intensive manual labelling from researchers, who would just need to validate the data. As a baseline model, we use a classifier based on simple rules (e.g., predict the majority class or according to class distribution). The `DummyClassifier` from scikit-learn (Pedregosa et al., 2011) offers two possible strategies that fit our purpose:

1. *most frequent*: predicts the most frequent class in the training set,
2. *stratified*: predictions are generated based on training set classes distribution.

For each phonetic group, we want to be sure to pick the best strategy to establish a lower bound for the performance of our models. Thus, the baseline performance for each phonological group is the strategy with the highest score.

Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986) have been broadly used for sequence classification and achieved promising results, but they usually require datasets much larger than ours to be trained. From here on, we will use Recurrent Neural Networks (or RNNs) as an umbrella term to describe all the deep learning models that follow an RNN architecture. We use logistic regression and Support Vector Machines (SVMs) as standard statistical models, multilayer perceptron (MLP) as a deep learning model and Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRU) (Cho et al., 2014) as models capable of

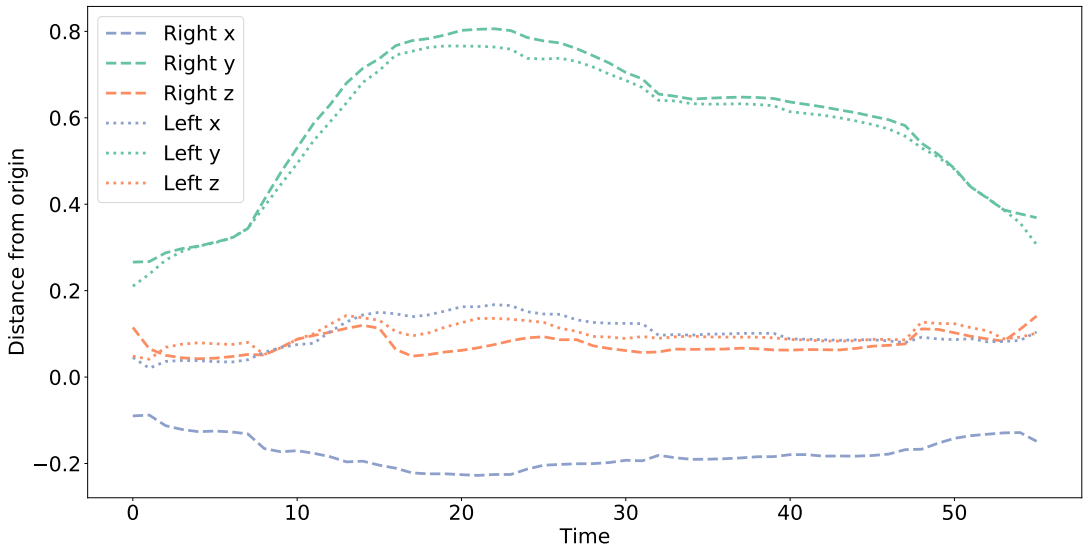


Figure 3.5: Coordinates projection of the left and right wrists during the execution of the sign ”house”. While movements along the Y and Z axis are similar, there is a constant offset for the X axis. This indicates a symmetrical sign, which can be visualised using ASL-Lex online tool.

capturing the temporal component of sequences. To summarise, we define a baseline and train 2 statistical models and 3 different neural networks models, namely

- Logistic regression,
- SVM,
- MLP,
- LSTM with a MLP attached in cascade,
- GRU with a MLP attached in cascade.

3.3 Experiment

We test two different approaches, namely statistical models and deep learning, to infer the phonological classes associated with each video. Initially, we normalise the input features (i.e., 3D keypoints) so that all the values are included in the range $[-1, 1]$, which should help to improve the stability of our models. In addition, we deploy a zero-padding strategy in order to make sure that all time series have the same duration and there is no loss of information (when compared to alternatives like subsampling). We

use zero-padding for both statistical models and deep learning models, as both require for all the different inputs to have the same input lenght. We then divide our data in a stratified fashion (i.e., according to the distribution of the labels) into two different subsets: training and test, with 85% of the data dedicated to the former and 15% to the latter. We choose these proportions because our dataset is small and not having enough training data could lead to bad performances. We train our 3 different models for 50 epochs using the Adam (Kingma and Ba, 2017) optimisation algorithm and a learning rate equal to 10^{-4}. We use 5-fold stratified cross-validation for hyperparameter tuning and we identify the micro-averaged F1-score as the best metric to train and measure the performance of our classifier, as the classes distribution of our dataset is uneven. As a first approach, we want to maximise the number of correct predictions, regardless of the dataset distribution. Micro-averaged F1-score gives us an idea of how well the classifier is performing *overall*, by calculating the total number of correctly classified instances instead of averaging the scores of the different classes. Moreover, we report the macro-averaged F1-score because it is an indicator of how the score is affected by the imbalance across different classes. Each of the final evaluation runs is repeated 10 times using different seeds to calculate the mean and standard deviation.

Multinomial logistic regression is the first algorithm that we are exploring for classification. A simple method is ideal as a first approach to understand how much can be learned from the data as there is a small number of parameters that we can modify. In particular, we investigate different techniques of regularisation, like L2 regularisation and early stopping, in order to prevent overfitting. Support Vector Machines (SVMs) (Cortes and Vapnik, 1995) are a powerful tool to perform nonlinear classification. Their power relies on the flexibility they provide using the *kernel function*, which makes them operate efficiently in high-dimensional space. Thus, the kernel function is certainly among the most important parameters to select for a good performance. We want our models to perform better than the baselines, but at the same time, they should be flexible enough to generalise. For logistic regression and SVM, this means considering the number of iterations used to train the models and the regularisation magnitude. We do not consider methods such as k-nearest neighbour classifiers due to the nature of the dataset - i.e., a small number of samples and strongly unbalanced classes. In order to take advantage of the temporal component of our data, we can use deep learning architectures that are explicitly designed to do so, such as RNN (Rumelhart et al., 1986). In particular, we are going to use LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Cho et al., 2014). The main differences between these

architectures are briefly illustrated in Figure 3.6.

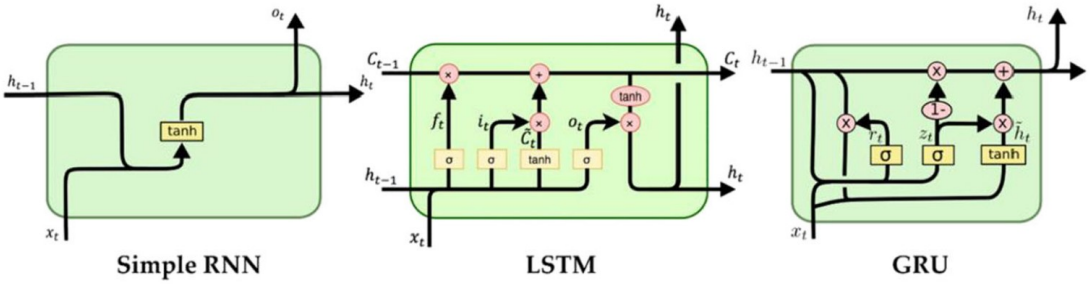


Figure 3.6: Comparison of the cells in RNN, LSTM and GRU (Reprinted from (Tembhurne and Diwan, 2021)). RNN is a basic type of recurrent model, while LSTM and GRU are advanced variants that address the vanishing gradient problem and enable better long-term dependencies modeling in sequential data.

When using neural networks, every single parameter and hyperparameter can make an impact on the model performance. Thus, we focus our experiments on the neural network architecture and methods to avoid overfitting, such as dropout and batch normalisation, given the small size of our dataset. We also train a multi-layer perceptron as a middle ground between statistical and deep learning methods that take into account the temporal component, like RNNs. Table 3.1 and Table 3.2 provide the list of different parameters for statistical and deep models respectively.

Model	Parameter	Value
Logistic regression	multi_class	ovr, multinomial
	class_weight	None, balanced
	solver	liblinear, lbfgs, newton-cg
	C	10^{-4}, $3.16 \cdot 10^{-3}$, 10^{-1}, 3.16, 10^{2}
	max_iter	1, 34, 67, 100, 133, 167, 200, 233, 266, 300
Support Vector Machine	kernel	linear, poly, rbf
	class_weight	None, balanced
	gamma	scale, auto
	decision_function_shape	ovo, ovr
	C	10^{-4}, $3.16 \cdot 10^{-3}$, 10^{-1}, 3.16, 10^{2}
	max_iter	1, 28, 56, 84, 111, 139, 167, 194, 222, 250

Table 3.1: Hyperparameters explored for each different statistical model. For each attempt, we use cross-validation to ensure the significance of our attempt given the small size of the dataset.

Parameter	Value
Temporal units	None, LSTM, GRU
Number of temporal layers	0, 1, 2, 3
Hidden units temporal layers	128, 256, 512
Number of linear layers	0, 1, 2, 3
Hidden units linear layers	128, 256, 512
Dropout	0.0, 0.3, 0.5, 0.8
Linear dropout	0.0, 0.3, 0.5, 0.8
Loss weights	Uniform, Class weight
LR scheduler step size	100, 150

Table 3.2: Explored parameters for deep models. The only permutations of these parameters which are not tested are when both numbers of temporal and linear layers are 0.

3.4 Results and discussion

We provide the results of the set of hyperparameters leading to the highest score, which are presented in Table 3.3. We further analyse the erroneous classifications made by the best classifiers using confusion matrices, in order to understand which classes get misclassified and how.

In order to calculate a baseline for our current set-up, we take into account the metrics used to measure the performance and the data distribution. Given that we chose the micro-averaged F1-score, which is directly proportional to the number of correct predictions, the baseline which yields the highest score is the one that generates predictions based on the majority class. In addition, we provide a baseline and results for the macro-averaged F1-score as an additional evaluation metric. The macro-averaged baseline is calculated using the statistical distribution (i.e., the number of times a certain value appears compared to the total number of samples) of the different labels. For the major location, the micro and macro F1 scores are respectively 34.9% and 20.7%, while for the sign type are 38.5% and 20.0%.

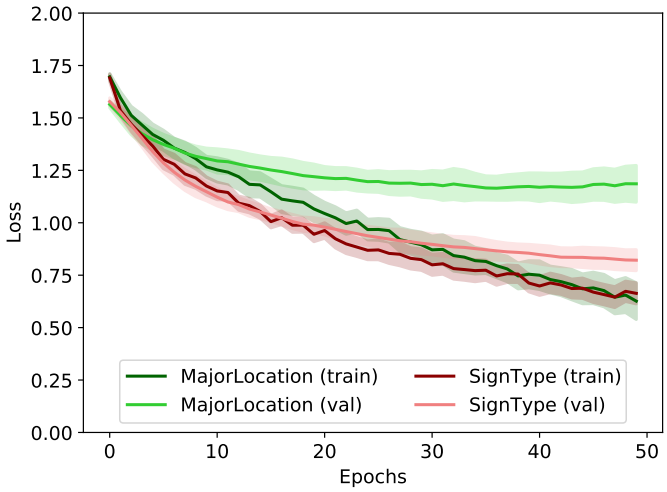
Figure 3.7 illustrates the learning curves (i.e., training and validation loss) of our best-performing deep model. We found these configurations to be a good trade-off between training speed and fluctuations in the loss. The sign type is clearly the easier class to learn for our models, while the movement class is the hardest. We can also see that the multilayer perceptron is the model with the largest gap between training and validation loss, meaning that it overfits more easily when compared to LSTM or GRU. Based on the validation loss and early stopping, we found out that 50 epochs are

Model	Parameters	Major location	Sign type
Logistic regression	multi_class	one-vs-rest	multinomial
	class_weight	None	None
	solver	lbfgs	lbfgs
	C	0.003	0.003
	max_iter	100	100
SVM	kernel	poly (degree 3)	poly (degree 2)
	class_weight	None	None
	gamma	scale	scale
	decision_function_shape	one-vs-one	one-vs-one
	C	3	3
	max_iter	250	250
MLP	number of linear layers	3	3
	number of hidden units	256	256
	dropout	0.5	0.5
	batch size	64	128
LSTM	number of temporal layers	1	1
	number of linear layers	2	2
	number of hidden units	64	256
	dropout temporal layers	0	0
	dropout linear layers	0	0.5
	batch_size	8	64
GRU	number of temporal layers	1	2
	number of linear layers	2	2
	number of hidden units	256	1024
	dropout temporal layers	0	0.5
	dropout linear layers	0.5	0.5
	batch_size	64	128

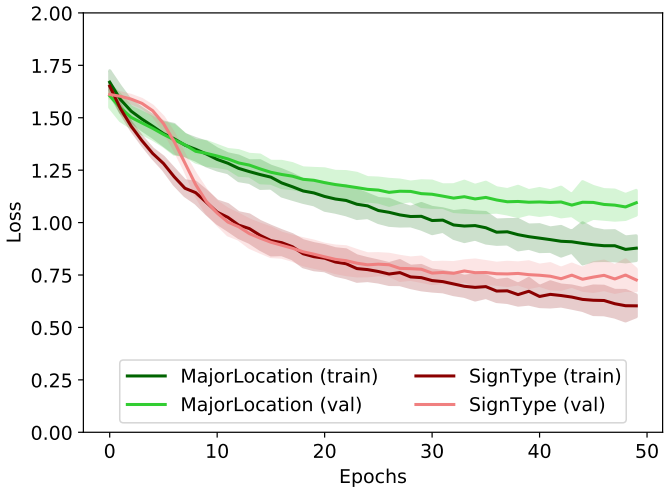
Table 3.3: Best hyperparameters for each model over different phonological properties.

enough to make the models converge (to the point in which further training did not led to significant improvements) despite the limited amount of data. Thus, we conclude that the models we propose do not limit themselves in predicting the most frequent class, nor predict randomly based on the distrubution of the classes, but actually learn to recognise the pattern in the data.

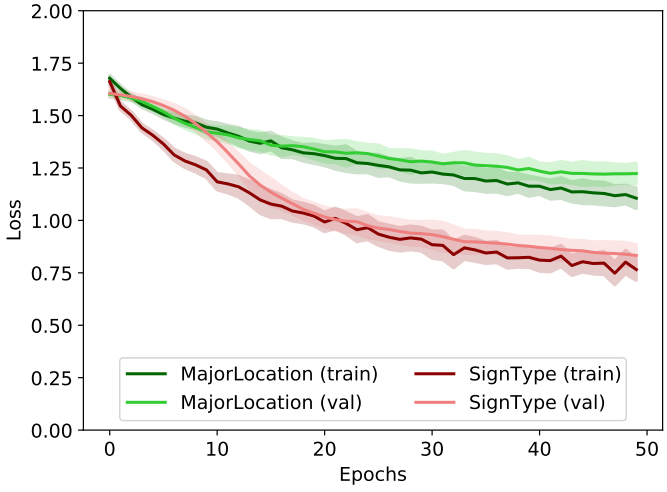
Table 3.4 summarises the results for each different model and compares them with the baseline. Overall, most of the models perform similarly over the same class, with neural networks-based models that usually achieve slightly higher scores than standard models. For the sign type, statistical models achieve around 53% of correct predictions, while deep learning models measure around 57% on average, both accounting for an



(a) MLP



(b) LSTM



(c) GRU

Figure 3.7: Learning curves (train and validation) for "major location" and "sign type" features. By comparing training a validation loss, we aim at limiting the overfitting caused by the limited amount of data.

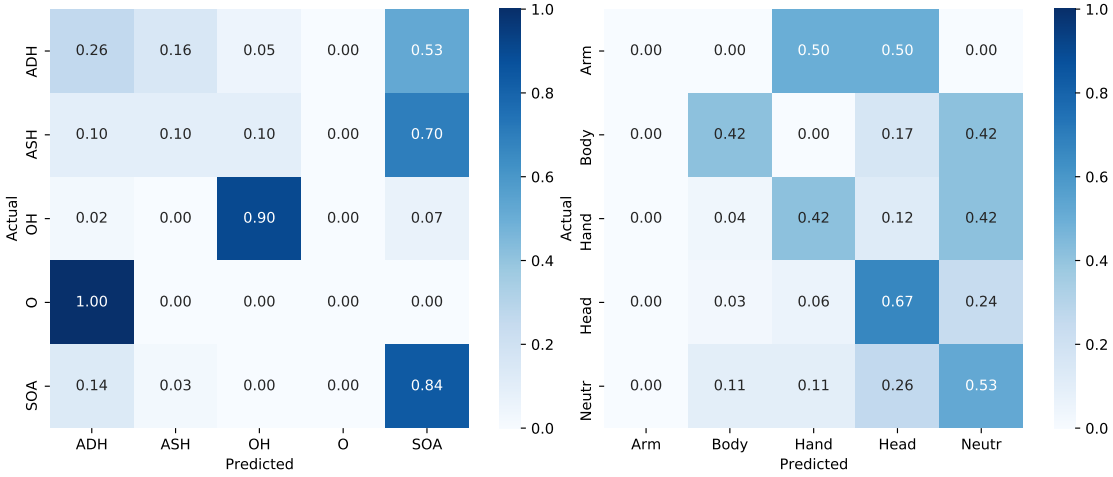
increment of around 55% when compared to the baseline. For the major location, there is no relevant difference between statistical and deep models, with both scoring approximately 68% for the micro F1 score, an increment of 74% compared to the baseline. Finally, all the models experience a 100% gain compared to the baseline when considering the macro F1 score, indicating that not only do the models learn to predict the classes that are most represented, but also according to the dataset distribution despite being trained to maximise the number of correct predictions.

	Major location		**Sign type**	
Model	**micro F1**	**macro F1**	**micro F1**	**macro F1**
Baseline	34.9 ± 0.0	20.7 ± 3.1	38.5 ± 0.0	20.0 ± 3.0
Logistic	53.2 ± 2.5	43.1 ± 7.4	67.2 ± 2.6	42.5 ± 4.2
SVM	52.8 ± 3.3	43.9 ± 5.9	68.6 ± 1.9	40.5 ± 2.5
MLP	$\mathbf{58.1 \pm 3.6}$	42.7 ± 3.8	69.1 ± 4.2	41.8 ± 4.3
LSTM	56.8 ± 2.9	42.8 ± 4.1	$\mathbf{70.2 \pm 3.5}$	42.2 ± 2.5
GRU	56.1 ± 3.9	$\mathbf{43.2 \pm 5.3}$	69.4 ± 3.8	$\mathbf{42.7 \pm 2.4}$

Table 3.4: Summary of the results for each model trained with the hyperparameters that respectively lead to the best micro-averaged F1 score. Test scores are calculated over 10 different seeds.

We analyse the misclassifications from the MLP, as it provides high scores for both major location and sign type while being simpler than RNNs algorithms. Figure 3.8a illustrates the confusion matrix for a single run of the MLP trained to recognise the sign type. We can see that the algorithm mainly distinguishes one vs two-handed signs. Moreover, removing the strongly under-represented value ”Other” would lead to an increase in the actual score of the model. Lastly, asymmetrical signs with different and same hand shapes (ADH and ASH) are classified as symmetrical or alternating (SOA) due to the lack of features regarding the hand pose. For the major location, most of the incorrect values are classified as “neutral”. This reflects the fact that the “neutral” value is the majority for the major location. Similarly, most of the examples are classified as “back and forth” when it comes to the movement class, which is once again the majority class. Moreover, the confusion matrix clearly shows that almost no value is predicted correctly, given that the diagonal of the matrix is mostly composed of values close to zero. Finally, we can see that the values predicted for the sign type fall into three different categories: “other”, “asymmetrical different handshape” and “symmetrical or alternating”. While it is quite obvious that it is impossible for the model to distinguish between “asymmetrical different handshape” and “asymmetrical same

handshape" given that there is no information about the hand, it is interesting to notice how the three predominant predictions correspond to one-handed signs, asymmetrical signs and symmetric/alternating signs (both of which involve two hands), which are patterns easy to distinguish.



(a) Sign type (b) Major location

Figure 3.8: Confusion matrices for the multilayer perceptron over different output classes. We choose MLP as it provides high scores for both classes while being simpler than RNNs algorithms. ADH = Asymmentrical Different Handshape, ASH = Asymmetrical Same Handshape, OH = One Handed, O = Other, SOA = Symmetrical or Alternating.

3.5 Conclusions

We propose a novel approach to the recognition of phonological classes of ASL based on data validated by ASL users. We extract temporal features from the video using a 3D tracking algorithm based on the SMPL model and perform recognition using statistical and deep models. Our experiments suggest that it is possible to extract phonological classes from videos using pre-trained tracking algorithms. We believe this work opens many possibilities for additional research. Challenges regarding this task include, but are not limited to, distinguishing similar phonological classes (e.g., the difference between chest and neutral space) and the reliability of the tracking algorithm. In the next chapter, we will address our approach shortcomings by expanding the dataset, including data augmentation techniques to compensate for imbalance and replacing the current tracking algorithm with one that includes hands features.

Chapter 4

Large scale phonology recognition

4.1 Introduction

In the previous chapter, we describe how pre-trained pose estimation models can be used to extract 3D information from videos and recognise the phonological properties of sign language. However, we pointed out how our approach has some significant limitations, like the size of the dataset and the lack of pose estimation regarding the hands. In particular, the latter is clearly a major drawback when studying sign language.

Chapter 4 aim is to describe how we overcame such limitations, namely by

1. Expanding the current dataset,
2. Deploying pose estimation models capable of extracting additional features, and
3. Training classification models which can take advantage of the intrinsic properties of sign language.

4.2 Methods

We address the recognition of phonological properties as a classification problem based on features extracted from videos of people speaking SL. Although manual annotation approaches are widely adopted, they are time-consuming and require expert knowledge. Instead, we rely on automated dataset construction. On a high level, to do so we cross-reference a large-scale SLR dataset with an ASL Lexicon and annotate videos of signs with their corresponding phonological properties. We then extract skeletal

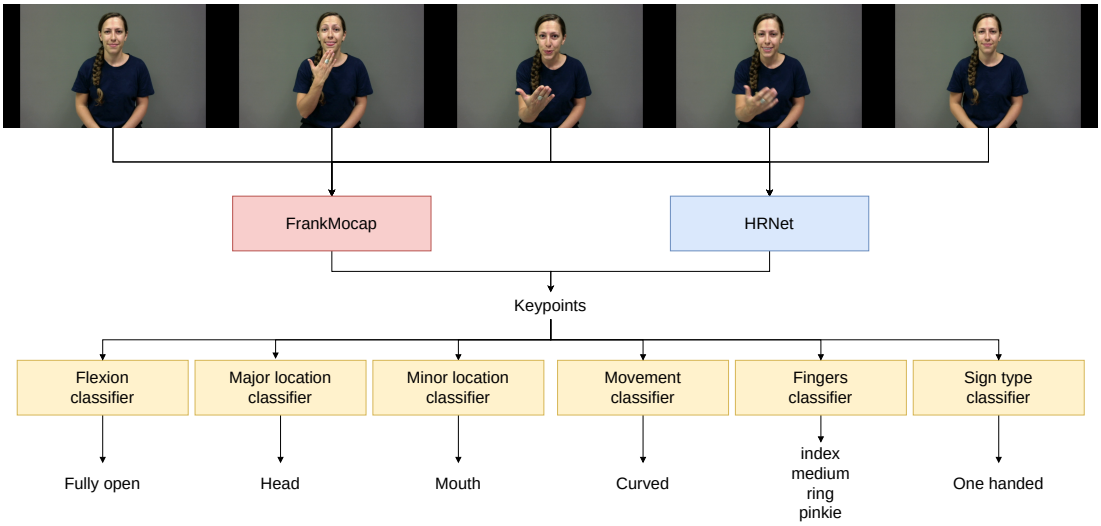


Figure 4.1: Our approach extracts coordinates from videos of people speaking ASL using two different models as alternatives. Then, we use the keypoints to classify the video according to 6 phonological classes. For example, the sign "thank you" involves one hand, with all the fingers fully open, with a curved movement, executed next to the head, specifically next to the mouth.

features, by taking advantage of pre-trained deep learning models from the computer vision community (Rong et al., 2021; Wang et al., 2020). Finally, we train several deep models to classify them as phonological classes.

4.2.1 Data

As previously mentioned, ASL-Lex (Caselli et al., 2017) contains phonological features of American Sign Language, such as where the sign is executed, the movement performed by the hand and the number of hands and fingers involved. The latter properties were coded by 3 ASL-versed people. In our work, we are interested in recognising phonological properties from videos of people speaking ASL. Consequently, we aim to construct a dataset, suitable for supervised learning, containing videos labelled with six phonological properties. Specifically, we choose the manual properties with the strongest discriminatory power to determine signs based on their configuration (Caselli et al., 2017)

- **Flexion:** aperture of the selected fingers of the dominant hand at sign onset,
- **Major location:** general location of the dominant hand at sign onset,
- **Minor location:** specific location of the dominant hand at sign onset,

- **Movement:** the first movement path of the sign,
- **Selected fingers:** fingers that are moving or are foregrounded during that movement, and
- **Sign type:** symmetry of the hands according to Battison (1978).

A detailed description of all the properties is provided in the Appendix B.

One of the limitations of ASL-Lex is the small number of examples and lack of variety: its first iteration (ASL-Lex 1.0) contains less than 1000 videos, all signed by the same person. While sufficient for educational purposes, these videos are of limited suitability for developing robust classifiers that can capture the diversity of ASL speakers (Yin et al., 2021). To this end, we source videos from WLASL (Li et al., 2020a) (Word Level-ASL), one of the largest available SL datasets, featuring more than 2000 glosses i.e., the transcribed version of the sign/lemma) demonstrated by over 100 people, for a total of more than 20000 videos. There are several subsets of WLASL based on the number of glosses. WLASL*N* indicates a subset of WLASL containing *N* different glosses. Each sign is performed by at least 3 different signers, which implies greater variability compared to having one gloss performed by only one user. By cross-referencing ASL-Lex and WLASL2000 based on corresponding glosses, we can increase the number of samples available to train our models.

Finally, to leverage state-of-the-art SLR architectures that operate over structured input, we enrich each raw video with its extracted keypoints that represent the joints of the speaker. To do so, we use two pre-trained models, FrankMocap (Rong et al., 2021) and HRNet (Wang et al., 2020), which we further describe in Section 4.2.2. While these tracking algorithms follow different paradigms, the former extracting 3D coordinates based on a predicted human model and the latter predicting keypoints as coordinates from videos directly, they produce similar outputs. An important distinction is that while FrankMocap estimates the 3D keypoints, HRNet outputs 2D keypoints with associated prediction confidence scores. We use these different models to explore whether different tracking algorithms affect the recognition of phonological classes. Thus, they both produce 3D outputs. We select a subset of features of the upper body, namely: nose, eyes, shoulders, elbows, wrists, thumbs and first/last knuckles of the fingers. These manual features were determined to be the most informative while performing sign language recognition (Jiang et al., 2021b).

Our final dataset, WLASL-Lex2001 (WLASL2000 + ASL-Lex 1.0), is composed of 10017 videos corresponding to 800 glosses, 3D skeletons (x, y, z from FrankMocap

and x, y and *score* from HRNet) labelled with their phonological properties. A characteristic of this dataset is that it follows a long-tailed distribution. Due to the nature of language, some phonological properties are more common than others, which means that some classes are more represented than others. On the one hand, the training setup for our models should take this factor into account, but on the other hand, the advantage of training over phonological classes instead of glosses is that different glosses can share phonological classes.

4.2.2 Pose estimation

In our previous work, we test HMMR (Kanazawa et al., 2019) to extract 3D coordinates from RGB videos about people speaking ASL. Among the many advantages, HMMR (Kanazawa et al., 2019) takes into account the temporal component at training time. In fact, at time t, the model predicts keypoints for the current frame and for the previous and next frame as well, at $t - \Delta t$ and $t + \Delta t$ respectively. However, one (if not the main) limitation is that it does not produce any information about the hands, as it is based on SMPL (Loper et al., 2015). This is clearly a major shortcoming for an application of sign language, where movements of the hands and the pose of the fingers are fundamental sources of information. To overcome this limitation, we test two different models from the literature (Rong et al., 2021; Wang et al., 2020).

SMPL-X (Pavlakos et al., 2019) is a detailed body model that represents the human figure in 3D. It is an extension of the original SMPL model (Loper et al., 2015). The "X" in SMPL-X stands for "eXpressive", which includes more realistic body shapes and poses, including hands and facial expressions, as illustrated in Figure 4.2. The SMPL-X model is constructed from a combination of body shape parameters and pose parameters. The body shape parameters define the overall size and proportions of the body, while the pose parameters define the position and orientation of each body joint. These parameters can be estimated from real-world data, such as 3D scans or motion capture data, allowing the model to closely match the appearance and motion of a real person.

FrankMocap (Rong et al., 2021) is a 3D pose estimation approach which aims to overcome a shortcoming of most pose estimation models: focusing exclusively on the body. By leveraging a modular design, it combines independent regression models for the face, hands and body. Figure 4.3 shows the FrankMocap framework. The face module uses RingNet (Sanyal et al., 2019) to extract facial expressions and poses. Since RingNet is based on the FLAME model (Faces Learned with an Articulated

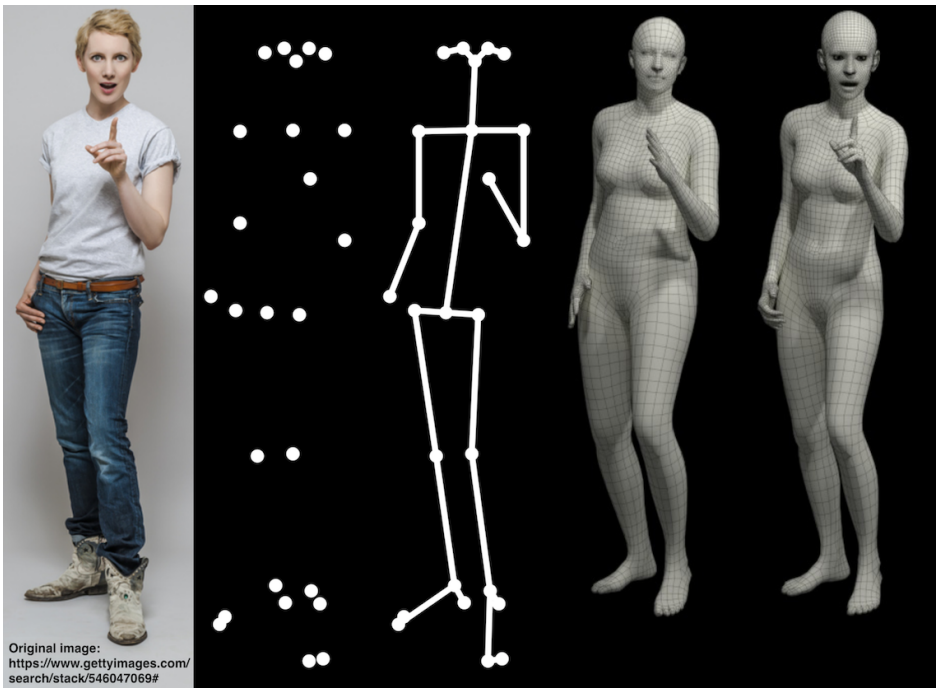


Figure 4.2: Comparison of SMPL and SMPL-X on an example image. Opposed to SMPL, SMPL-X includes hand poses and facial expressions (Reprinted from (Rong et al., 2021)).

Model and Expressions) (Li et al., 2017), its predictions are compatible with the face part of SMPL-X. The hand module is developed based on state-of-the-art approaches (Kanazawa et al., 2018, 2019; Kolotouros et al., 2019) and adapted to SMPL-X. The body module is derived from the SPIN (SMPL oPtimization IN the loop) model (Kolotouros et al., 2019) and finetuned on the Human3.6M dataset (Ionescu et al., 2014), as the originally proposed model was trained on SMPL. SPIN trains a deep network for 3D human pose and shape estimation combining a regression-based and an iterative optimization-based approach. The deep network initializes an iterative optimization routine that fits the body model to 2D joints within the training loop, and the fitted estimate is subsequently used to supervise the network. Finally, the integration module can use different approaches based on the trade-off between speed and accuracy, but the most complete is a neural network that predicts an approximation of the optimisation gradient to adjust the arm parameters. In the end, FrankMocap can provide as output both 3D keypoints and 3D joint rotations about the whole body and hands.

HRNet (Wang et al., 2020), or High-Resolution Network, is a type of convolutional neural network architecture that has been designed for tasks that require high-resolution inputs, such as image classification and object detection. HRNet is known

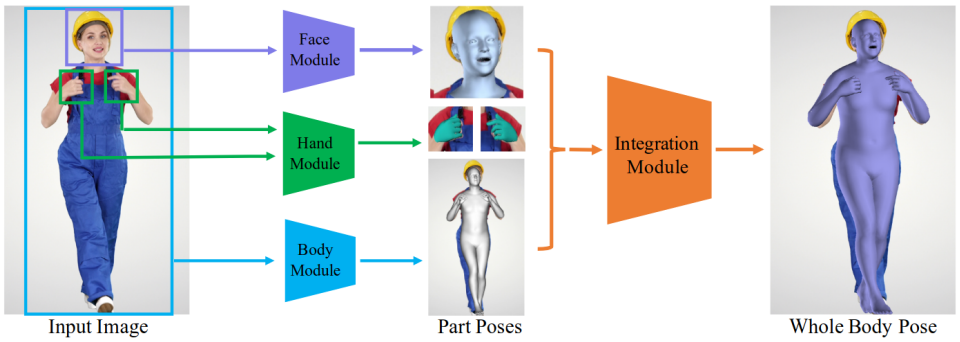


Figure 4.3: The FrankMocap framework. FrankMocap combines a face, a hand and a body module to extract a full body pose (Reprinted from (Rong et al., 2021)).

for its ability to maintain high-resolution representations throughout the network, which allows for improved performance on these types of tasks. HRNet has been shown to outperform other state-of-the-art models on a number of benchmarks and has been used in a variety of real-world applications, such as medical imaging and satellite imagery analysis. HRNet works by using a multi-scale processing approach to extract high-resolution representations from input images. This is accomplished through the use of parallel branches in the network architecture, each of which operates at a different resolution and spatial scale. The outputs from these branches are then combined and passed through several layers of convolutional and pooling operations, which extract and refine the high-resolution representations. These representations are then used to make predictions about the objects in the input image, such as their class labels and bounding box coordinates. Figure 4.4 provides a graphical representation of the HR-Net architecture, while Figure 4.5 an example of the output on a video frame. The network can also be used to estimate human poses. In fact, it can produce keypoints as output in the format (x, y, c) where x and y are the 2D coordinates of the keypoints (expressed in pixel coordinates) and c is the confidence score for the prediction.

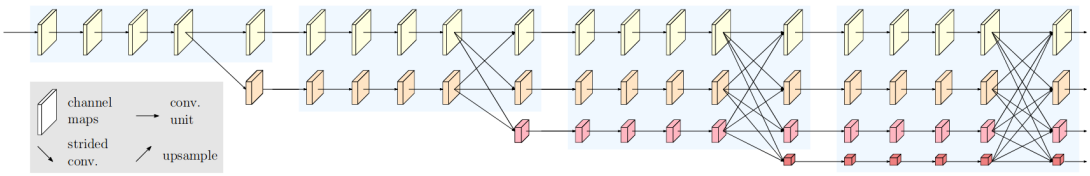


Figure 4.4: HRNet architecture (Reprinted from (Wang et al., 2020)). HRNet uses different channel maps in parallel and combines features extracted by the convolutional layer in order to improve the performance of the network.

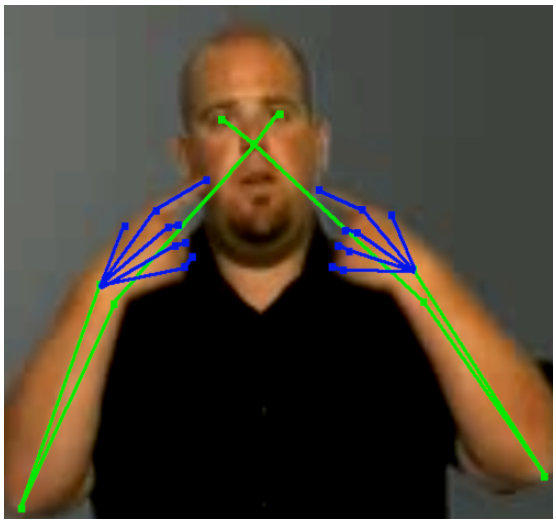


Figure 4.5: Example of the HRNet output applied to a sign language video from WLASL. HRNet extracts accurate predictions for both body and hands keypoints.

4.2.3 Classification algorithms

To estimate the complexity of the recognition task on the dataset, we use the majority-class baseline and the Multi-Layer Perceptron (MLP) as basic deep learning model. We further use Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRU) (Cho et al., 2014) as models capable of capturing the temporal component of videos. As state-of-the-art Sign Language Processing (SLP) architectures that have been used to perform SLR, we use

1. the I3D 3D Convolutional Neural Network (Carreira and Zisserman, 2017; Li et al., 2020a) able to learn from raw videos, and

2. the Spatio-Temporal Graph Convolutional Network (STGCN) (Jiang et al., 2021b) that captures both spatial and temporal components from the extracted keypoints.

Both have already been used to perform SLR on large datasets such as WLASL2000 (Li et al., 2020a) (Jiang et al., 2021b).

I3D, or Inflated 3D Convolutional Network (Carreira and Zisserman, 2017), is a type of deep learning architecture that is designed for tasks such as video classification and action recognition. It is based on the 2D convolutional networks used in image recognition but has been extended to operate on 3D data, such as videos, which are composed of multiple frames arranged in a temporal sequence. The I3D architecture (Figure 4.6) inflates the 2D filters used in conventional convolutional networks to 3D, allowing them to capture spatial and temporal information from the input data. The I3D

architecture has been shown to outperform other state-of-the-art models on a number of video understanding tasks.

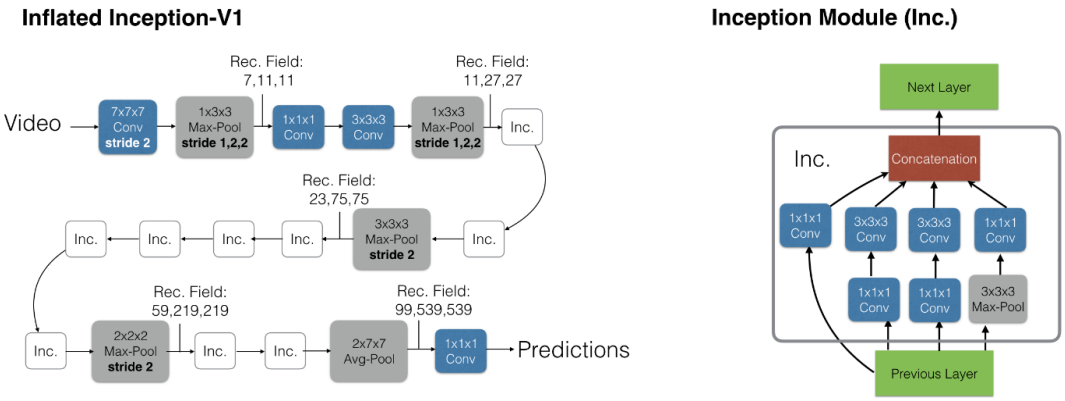


Figure 4.6: Architecture of the Inflated 3D Convolutional Network (Reprinted from (Carreira and Zisserman, 2017)). The architecture combines convolutional and max pooling layers, and the inception module which concatenates to output of different convolutions.

The Spatio-Temporal Graph Convolutional Network (STGCN) is a type of deep learning architecture that is designed for tasks such as action recognition and human pose estimation. It is based on the idea of using graph convolutional networks (GCNs), which are a type of neural network that operate on graph-structured data, to process sequential data such as videos, but taking as input skeletons provided by pose estimators like HRNet (Wang et al., 2020). The STGCN architecture (Figure 4.7) incorporates both spatial and temporal information in the graph structure, allowing it to capture the relationships between keypoints and their movements over time.

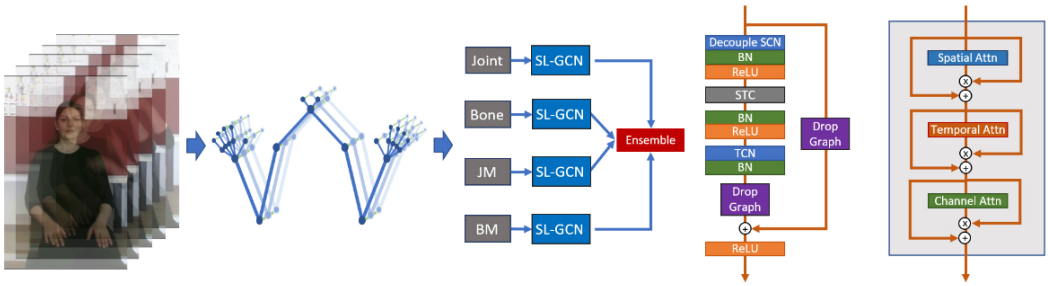


Figure 4.7: Architecture and components of the Spatio-Temporal Graph Convolutional Network (Reprinted from (Jiang et al., 2021b)). STGCN takes into account both spatial and temporal relationship of keypoints using attention.

In brief, we test the following classification algorithms:

- MLP,
- LSTM and GRU,
- 3D CNN, and
- ST-GCN.

4.3 Experiment

For each phonological property, we generate dataset splits and train dedicated models separately. While a multi-class multi-label approach could achieve higher scores, by relying on potential interdependencies of different properties, we chose to model the properties in isolation, to disentangle the factors that affect the learnability of each property. From now on, when we mention the *dataset*, we refer to an instance of the WLASL-Lex 2001 dataset, where labels are the values of a single phonological class.

We make this distinction because we produce six different train, validation and test splits (with a $70 : 15 : 15$ ratio) stratifying on the corresponding phonological property (*Phoneme*). By doing so, we make sure that all splits

- contain all possible labels for a classification target (i.e. phonological property), and
- follow the same distribution.

Since we source the videos from WLASL, we have multiple videos representing each gloss, therefore, randomly splitting our data will result in the fact that glosses in the test set might appear in the training set as well, signed by a different speaker. Thus, to investigate how well the models can predict properties on unseen glosses, we also produce label-stratified splits on gloss-level (*Gloss*), such that videos of glosses in the validation and test set do not appear in training data and vice versa. Thus, to summarise, experiments in the *Phoneme* setting aim to evaluate the capability to recognise phonological properties of signs that were already encountered in the training data but are performed by a different speaker in the test set. Conversely, experiments in the *Gloss* setting aim to evaluate the capability to recognise phonological properties of signs completely *unseen during training*.

We use the I3D model that has been pre-trained on Kinetics-400 (Carreira and Zisserman, 2017) and fine-tune it on raw videos from our datasets. The other models are

trained from scratch using 3D keypoints as input. We fix the length of all input to 150 frames, longer sequences are truncated while shorter sequences are looped to reach the fixed length. For the STGCN we use hyperparameters chosen by (Jiang et al., 2021a) because initial experiments on our data showed a difference of at most 2% accuracy, which is within the uncertainty estimate. To find the optimal hyperparameters for the other models, we perform Bayesian optimisation over a pre-defined set described in full detail in Table 4.1.

Model	Parameter	Value
MLP	number of layers	2, 4, 8
	hidden dimension	1024, 2048, 4096
	dropout	[0, 0.5]
	learning rate	$[10^{-4}, 10^{-1}]$
	scheduler step size	[10, 50]
	gamma	[0.1, 0.5]
	batch size	512, 1024
LSTM/GRU	number of temporal layers	1, 2, 3
	temporal dropout	[0, 0.5]
	bidirectional	True, False
	number of layers	0, 1, 2
	hidden dimension	128, 512, 1024
	dropout	[0, 0.5]
	learning rate	$[10^{-4}, 10^{-1}]$
	scheduler step size	[10, 50]
	gamma	[0.1, 0.5]
	batch size	64, 128
STGCN	learning rate	[0.01, 0.3]
	number of groups	8, 16, 32
	block size	[10, 25]
	window size	[50, 150]
	scheduler step size	3, 4
	warmup epochs	20
3D CNN	dropout	[0, 0.5]
	learning rate	$[10^{-4}, 10^{-1}]$
	gamma	[0.1, 0.9]
	scheduler step size	[10, 50]
	batch size	32

Table 4.1: Set of explored hyperparameters for each different model. All values enclosed between square brackets represent a range in which the value was sampled randomly.

We maximise Matthew's correlation coefficient (MCC) (Matthews, 1975) on the

validation sets of all six tasks

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{4.1}$$

with TP, TN, FP, FN being true/false positive/negative. We choose MCC as it provides a good trade-off between overall and class-level accuracy which is necessary due to the unbalance inherently present in our dataset. We select the best-performing model based on performance on the validation set, and for the final test set performance we train the models on both train and validation sets. We measure both accuracy to investigate how well models perform in general and class-balanced accuracy to take into account how well they are able to model different classes of phonological properties.

4.4 Results and discussion

Table 4.2 presents the results for the six dataset splits where glosses in test data could have appeared in training data as well in a stratified fashion. The poor performance of the simple MLP architecture suggests that the tasks are in fact challenging and do not exhibit easily exploitable regularities. Due to its simplicity, it is barely able to reach the baseline for some properties (34% vs. 35% and 44% vs. 50% for *movement* and *flexion* respectively). In particular, MLP classifying based on FrankMocap (MLP_F) output is often the worst-performing combination. This is an additional indication that we need to consider a model capable of taking in account the temporal component of our data, as opposed to the MLP. Conversely, STGCN using HRNet output (STGCN_H) outperforms other models on all six tasks. In some cases, for example, when predicting *movement* or *flexion*, it is the only model which significantly surpasses the majority class baseline. This superior performance is expected, as this specific combination of the STGCN operating over HRNet-extracted keypoints has been shown to be the largest contributor to the SLR performance on the WLASL2000 dataset (Jiang et al., 2021a). Clearly, a model that takes explicitly in account both the spatial and temporal peculiarity of the data leads to the highest performance. Models that operate over structured input often outperform the 3D CNN, demonstrating the utility of additional information provided by the skeleton features, like demonstrated with the STGCN. The results also suggest that models using the HRNet skeleton output outperform those who use FrankMocap, possibly due to the confidence scores produced by HRNet and associated with the coordinates. Providing an additional input to the

models that indicates how confident the pose estimation was in making that specific prediction can clearly ease the task of the classification model, which most likely ends up discarding inputs that have a low confidence score. This difference in performance suggests to conduct a more rigorous study to investigate the impact of different feature extraction methods as a possible future research direction.

	FLEXION		MAJLOCATION		MINLOCATION		MOVEMENT		FINGERS		SIGNTYPE	
	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$
Baseline	50.3	11.1	34.4	20.0	33.9	3.1	35.5	16.7	48.2	11.1	39.3	20
MLP_H	44.1 ± 2.5	11.1	70.3 ± 2.3	64.0	51.6 ± 2.5	28.2	34.5 ± 2.4	18.7	59.4 ± 2.5	25.0	73.9 ± 2.2	52.6
MLP_F	50.3 ± 2.5	11.1	57.8 ± 2.5	46.8	34.3 ± 2.4	9.1	34.3 ± 2.4	18.7	43.4 ± 2.5	12.9	67.0 ± 2.4	42.8
RNN_H	49.0 ± 2.5	30.0	75.8 ± 2.2	72.4	64.3 ± 2.4	46.0	35.1 ± 2.4	29.5	71.0 ± 2.3	46.5	78.7 ± 2.1	58.8
RNN_F	50.3 ± 2.5	11.1	64.6 ± 2.4	54.2	30.3 ± 2.3	4.0	35.4 ± 2.4	18.1	46.5 ± 2.5	12.4	70.9 ± 2.3	46.8
STGCN_H	$\mathbf{62.3 \pm 2.4}$	**45.0**	$\mathbf{83.2 \pm 1.9}$	**78.6**	$\mathbf{74.5 \pm 2.2}$	**63.5**	$\mathbf{63.6 \pm 2.4}$	**58.2**	$\mathbf{73.8 \pm 2.2}$	**56.0**	$\mathbf{84.5 \pm 1.8}$	**69.6**
STGCN_F	43.4 ± 2.5	20.8	70.5 ± 2.3	62.1	53.0 ± 2.5	40.0	45.7 ± 2.5	37.8	63.1 ± 2.4	32.8	73.0 ± 2.2	53.1
3DCNN	46.5 ± 2.5	13.2	64.3 ± 2.4	55.2	42.3 ± 2.5	18.6	32.9 ± 2.4	20.8	47.5 ± 2.5	14.5	69.5 ± 2.3	44.8

Table 4.2: Accuracy (A) and per-class averaged accuracy ($\overline{A}$) of various models on the test sets of the six tasks for the *Phoneme* split. For accuracy, we report the error margin as a confidence interval at $\alpha = 0.05$ using asymptotic normal approximation. We omit error margins for balanced accuracy as the low number of classes results in a small sample size. The dotted line indicates the division between models that use keypoints as input features and the one that uses videos.

Table 4.3 shows the performance of models to predict the phonological properties of unseen glosses. The performance of all tasks and all models deteriorates, suggesting that their success is partly derived from exploiting the similarities between glosses that appear in training and test data. However, the best model, STGCN_H, performs comparably to the *Phoneme*-split, with a drop of less than 10 accuracy points for five of the six tasks. Thus, when evaluating the performance of a model in recognising signs of properties, researchers should take into account the overlapping between samples in the training and test set. Moreover, we suggest that they should also consider (whenever possible) to perform two different evaluations like in our case: one in which the data is split according to the phoneme, and one in which the data is split according to the glosses.

Often, crowd-sourced (Polonio et al., 2018) or automatically constructed datasets such as ours, have a performance ceiling, possibly due to incorrectly assigned ground truth labels or low quality of input data (Chen et al., 2016). To investigate the former, we measure the agreement on videos that all models misclassify using Fleiss' κ. Intuitively, if models consistently agree on a label different from the ground truth, the ground truth label might be wrong. We find that averaged across the six tasks, the

	FLEXION		MAJLOCATION		MINLOCATION		MOVEMENT		FINGERS		SIGNTYPE	
	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$	A	$\overline{A}$
Baseline	**53.1**	11.1	35.7	20.0	42.0	5.0	35.2	16.7	47.4	12.5	38.3	20.0
MLP_H	44.6 ± 2.5	15.5	68.1 ± 2.3	56.6	47.3 ± 2.5	19.7	28.4 ± 2.2	19.8	56.2 ± 2.5	22.9	75.3 ± 2.2	50.7
MLP_F	52.8 ± 2.5	11.1	56.6 ± 2.5	42.9	38.3 ± 2.4	10.7	37.1 ± 2.4	21.7	39.3 ± 2.5	12.5	68.4 ± 2.4	41.2
RNN_H	39.6 ± 2.5	18.0	72.8 ± 2.2	67.3	49.3 ± 2.5	26.3	32.2 ± 2.3	24.9	60.7 ± 2.5	32.5	75.4 ± 2.2	53.5
RNN_F	53.0 ± 2.5	11.1	64.1 ± 2.4	52.6	44.4 ± 2.4	17.8	36.7 ± 2.4	20.1	27.3 ± 2.3	12.7	72.0 ± 2.3	46.9
STGCN_H	49.1 ± 2.5	**21.6**	$\mathbf{77.3 \pm 2.1}$	**70.0**	$\mathbf{55.1 \pm 2.4}$	**32.7**	$\mathbf{52.5 \pm 2.5}$	**46.5**	$\mathbf{65.7 \pm 2.4}$	**34.4**	$\mathbf{76.6 \pm 2.1}$	**54.4**
STGCN_F	39.0 ± 2.5	14.4	66.7 ± 2.3	60.1	45.1 ± 2.4	21.1	43.1 ± 2.5	34.9	60.0 ± 2.5	29.2	71.3 ± 2.3	47.5
3DCNN	46.0 ± 2.5	12.8	64.9 ± 2.4	52.0	10.8 ± 1.5	13.6	32.0 ± 2.3	19.3	45.9 ± 2.5	14.7	71.6 ± 2.3	46.3

Table 4.3: Accuracy (A) and per-class averaged accuracy ($\overline{A}$) of various models on the test sets of the six tasks for the *Gloss* split. For accuracy, we report the error margin as a confidence interval at $\alpha = 0.05$ using asymptotic normal approximation. We omit error margins for balanced accuracy as the low number of classes results in a small sample size. The dotted line indicates the division between models that use keypoints as input features and the one that uses videos.

agreement is negligible: 0.09 ± 0.06 and 0.11 ± 0.09 for *Phoneme* and *Gloss* split, respectively. Similarly, for the latter, if all models consistently fail to assign any correct label for a given video (e.g. all models err on a video appearing in the test sets of *movement* and *flexion*), this can hint at the low quality of the input, making it impossible to predict anything correctly. We find that this is not the case with WLASL-LEX2001, as videos appearing in test sets of different tasks tend to have a low mutual misclassification rate: 1% and 0.7% of videos appearing in test sets of two and three tasks were misclassified by all models for all associated tasks for the *Phoneme* split. For the *Gloss* split the numbers are 3 and 0% for two and three tasks, respectively. Together, these observations suggest that the models presented in this chapter are unlikely to reach the performance ceiling on WLASL-Lex2001 and more advanced approaches could obtain even higher accuracy scores. However, this cannot be corroborated by a quantitative analysis on the extracted features as there is no dataset which provides ground-truth for 3D hand keypoints for sign language. This hypothesis should be further investigated once such data becomes available. Table 4.4 provides an overview of the classification performance for the *Flexion* class (i.e. how the fingers are bent), using features extracted with HRNet and the STGCN to classify them. First of all, we can see by the short description we provide how the values share some similarities (e.g., curved open or closed, flat-open or curved open). Hence, it can be very difficult for a model to distinguish such features, especially when they involve fine-grained changes in the hands. We can also see how there is a strong imbalance in the representation of the classes, as the most represented has more than 5000 samples and the least represented has slightly more than 100. In fact, if we calculate the Pearson correlation coefficient between the

test set cardinality and the accuracy for the *Phoneme* split, we get a value of 0.994. Similarly, for the *Gloss split*, the Pearson correlation coefficient is 0.989. Thus, there seems to be a very strong correlation between the number of samples and the performance of the model, but it is worth noticing that the sample over which we calculate the correlation is very small (i.e., 9 samples). Such imbalance is due to the nature of the language, meaning that if the data is collected balancing examples of words rather than values of the different classes, such imbalance will always be present. Another observation we can draw from Table 4.4 is that training a model on the *Phoneme* split rather than the *Gloss* split not only gives a better overall performance (as previously observed from Table 4.2 and Table 4.3), but also achieves a higher score on every single value of the *Flexion* class. Moreover, precision and recall can gives us additional information about the errors that the model produces and they are more accurate descriptors of the accuracy, given the unbalance in the dataset. In particular, precision gives us an idea of how many values instances are correctly assigned a value over all the instances that were assigned that specific value (i.e., how many retrieved instances are relevant), while recall is an indicator of how many instances get a value assigned over all the instances that have such value (i.e., how many relevant instances are retrieved). We can see that for the *Gloss* split, except for the majority class, both precision and recall are always below 50%. Consequently, the classifier both produces more false than true positives (precision below 50%) and that it does not correctly classify the majority of instances of each value (recall below 50%). For the *Phoneme*, the precision both scores are higher than the *Gloss* split, but in most cases the performance is still not satisfactory.

To conclude, there could be several factors contributing to the varying performance across classes:

- Class Imbalance: The distribution of instances among classes might not be balanced. Classes with larger cardinality may receive more training samples, leading to better performance. Conversely, classes with smaller cardinality might lack sufficient training data, resulting in lower performance.

- Data Complexity: The nature of the phonemes might differ among classes. Classes like ”Fully open” and ”Fully closed” could have distinct and easily recognisable characteristics, making them more distinguishable for the model. Conversely, classes like ”Bent (closed),” ”Flat-open,” and ”Flat-closed” might share similar features, making them more challenging to differentiate accurately.

Value	Short description	Cardinality	Phoneme				Gloss			
			Test set cardinality	*A*	*P*	*R*	Test set cardinality	*A*	*P*	*R*
1	Fully open	5037	756	41.5	68.7	82.5	787	39.9	60.9	75.2
2	Bent (closed)	693	104	2.3	44.2	32.7	98	1.4	24.7	21.4
3	Flat-open	909	136	3.1	42.7	34.6	144	1.7	30.5	17.4
4	Flat-closed	507	76	1.9	52.8	36.8	59	0.3	8.0	6.7
5	Curved open	1130	170	5.3	55.6	47.1	163	2.6	26.6	23.3
6	Curved closed	642	96	3.1	59.0	47.9	90	1.1	39.0	17.8
7	Fully closed	795	119	4.2	60.0	52.9	104	2.1	32.6	29.8
Stacked	Stacked	123	19	0.5	46.7	36.8	7	0.0	0.0	0.0
Crossed	Crossed	181	27	0.6	69.2	33.3	32	0.1	7.7	3.1

Table 4.4: Per-class accuracy (*A*), precision (*P*) and recall (*R*) for the STGCN model predicting *Flexion*, using HRNet as feature extractor. We provide the cardinality of each class in the whole dataset and in the test set, for both phoneme and gloss splits, as it can be a key factor in understanding model performance. The extended description for each value of the class can be found in Appendix C.

- Feature Extraction: The choice of HRNet as the feature extractor could impact the performance of the model. It is possible that the HRNet features are better suited for certain classes or that they struggle to capture the distinguishing characteristics of specific classes, leading to varied performance.

4.5 Conclusion

We discuss the task of Phonological Property Recognition (PPR). To do so, we automatically construct a dataset for the task featuring six phonological properties and analyse it extensively. Our findings show that there is potential for improvement over our presented data-driven baseline approaches. Researchers pursuing this direction can focus on developing better-performing models, for example by relying on jointly learning all properties, as labels for different properties can be mutually dependent. For example, if the *MajorLocation* property of a sign is "Head", the property *MinorLocation* can only have 8 out of otherwise 32 possible values. Most importantly, our findings confirm the need for a linguistically-informed approach to SLR as stated by Yin et al. (2021), as collecting data trying to balance the number of lemmas rather than the properties exhibited can lead to a sever imbalance in the representation of such properties. Another possibility is to investigate the feasibility of using PRR to perform *tokenisation* of continuous sign language speech, by decomposing it into multiple phonemes, which is one of the big challenges of SLP (Yin et al., 2021).

Chapter 5

Sign language imitation for fingerspelling

5.1 Introduction

Our experiments in Chapter 4 show how pose estimation models are generally able to extract motions that reflect the fine-grained movements executed while speaking sign language. More specifically, we explored the possibility of recognising phonological properties of sign language from keypoints extracted from videos. Such phonological properties reflect a finer level than detail about the movements the people perform when compared to lemmas. Thus, they are a more precise indicator of whether an algorithm can actually extract low-level information about the execution of the movements. Such data are of the utmost importance for a learning scenario based on imitation, as the imitated movement can only be as good as the data extracted from the demonstration. Consequently, we see the recognition of phonological properties of sign language as a preliminary step for approaching sign language acquisition, as it serves as an indicator about the quality of the data we will use in acquisition and which limitations we can expect to encounter when performing imitation with such data. Another reason why this is such important is because it removes a factor from the analysis of the results of an agent that learned to perform sign language based on demonstration. If we do not analyse in advance the data that we provide as an input, we cannot be sure whether the reason of poor performance is due to a problem with the imitation approach or the input data.

Given data extracted with pose estimation models, we propose an imitation learning approach to fingerspelled sign language acquisition (namely *HandMime*) based on

reinforcement learning, adapting (Peng et al., 2018a,c) approaches to a robotic hand. Our approach is illustrated in Figure 5.1. Firstly, we build a URDF (Unified Robotics Description Format, an XML specification used to model multibody systems such as robotic manipulator arms) model of a human hand to simulate an artificial hand using a physics engine (Coumans and Bai, 2016) and perform parameter tuning to estimate the controller parameters. Secondly, we extract a 3D mesh of the hand from videos – which contains 3D coordinates and rotations – by exploiting a pre-trained vision model. Finally, we use reinforcement learning to estimate a policy which, comparing random movements with the reference motion, enables the simulated hand to imitate the original sign.

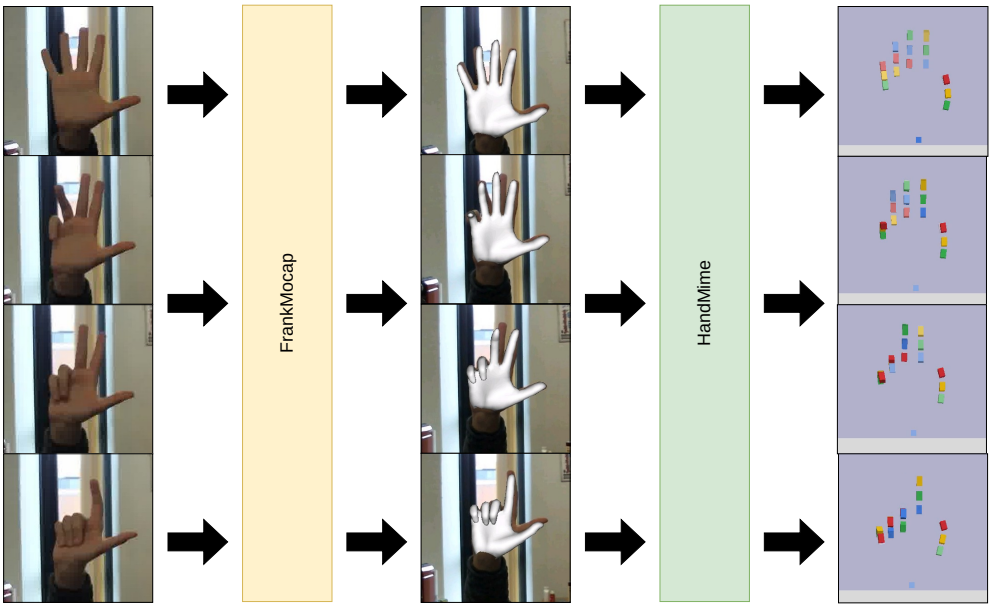


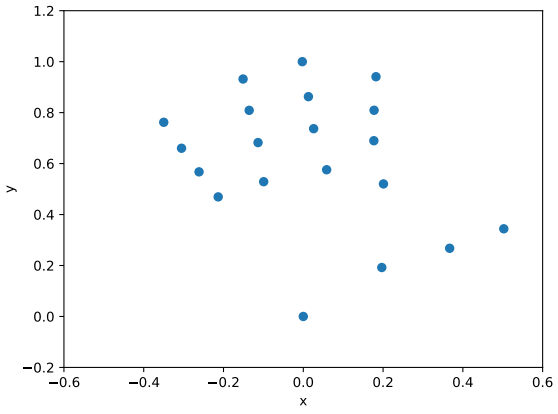
Figure 5.1: Our approach extracts 3D coordinates and rotations from RGB videos using deep learning models. It then trains a policy using reinforcement learning, in order to teach our robotic hand how to imitate the reference motion.

5.2 Methods

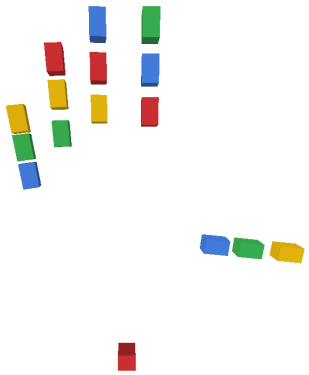
We start by describing how we build our hand model. We briefly explain how we use a deep learning model to extract information from RGB videos. Then, we describe how reinforcement learning can be used for motion imitation. Finally, we describe the problem we are tackling in terms of a Markov Decision Process.

5.2.1 Hand model

We build a model of a robotic hand based on the MANO model and the properties of a real human hand. To do so, we measure the position of the different joints of the hand and create a URDF model with the joints positioned accordingly. Figure 5.2a and Figure 5.2b show the reference keypoints (normalised based on maximum and minimum values for x and y) used to build the model and the respective URDF. Moreover, we impose realistic angular limitations to the joint of each finger.



(a) Normalised hand keypoints



(b) Hand URDF model

Figure 5.2: Construction of the hand model. By extracting keypoints from an image, we estimated the spatial relationship between joints. Then, we use this information to place the phalanges and build our simulated hand.

Our robotic hand is made up of 5 fingers, where each finger is made up of 3 joints. For our purposes, we do not consider the wrist as a mobile joint as we focus on the movements of the fingers. Moreover, we limit the movements of each finger joint to a single axis, as we believe it to be a reasonable initial starting point to approximate how a human hand works. Thus, in total, the model has 15 degrees of freedom (DoF). We impose a range limit for all the joints between $[0,2]$, where 0 radiants correspond to a fully-opened hand and 2 radiant (i.e., slightly more than $\pi/2$) to all the joints fully bent. We simulate the robotic hand using PyBullet (Coumans and Bai, 2016), an open-source physics simulator. For each joint motor, the simulated controller calculates the error as

$$\varepsilon = k_p \Delta P + k_v \Delta V \tag{5.1}$$

where k_p and k_v are, respectively, the position and velocity gains, and

$$\Delta P = P - \hat{P} \tag{5.2}$$

$$\Delta V = V - \hat{V} \tag{5.3}$$

are the position and velocity errors (i.e., the difference between the desired and the actual value).

Table 5.1 lists all the different physical parameters for our robotic hand. The only differences between our robotic hand and a human hand are the degrees of freedom and how the joints move. While in a finger, the joints are not independent (i.e., flexing one joint will cause other joints to flex as well), the joints in our robotic hand can move independently from each other.

Parameter	**Value**
Number of joints	15
Joint weight	0.05 kg
Joint DoF	1
Joint lower bound	0 rads
Joint upper bound	2 rads

Table 5.1: Parameters for the URDF of our robotic hand. Joints limits are slightly bigger than actual physical limits in order to avoid the controller getting stuck when reaching the maximum range.

5.2.2 Motion extraction

In order to extract motion from videos, we use the hand module from FrankMocap (Rong et al., 2021), a single-view 3D motion capture system. Taking an RBG image as input, it produces as output a 3D mesh based on the SMPL/SMPL-X (Loper et al., 2015) (Pavlakos et al., 2019) models. In addition, it also provides 3D keypoints and joint rotations for both body and hands. Hence, we take advantage of this pre-trained model to extract joint rotations for each finger. As mentioned above, the joints of our robotic hand are capable of rotating along a single axis. Consequently, as the output of FrankMocap is expressed in axis-angle format, we discard the 2 additional angles. Figure 5.3 illustrated the keypoints that can be extracted using FrankMocap. In addition, for each joint, except for the ones in the fingertips, FrankMocap extracts the corresponding 3D rotation.

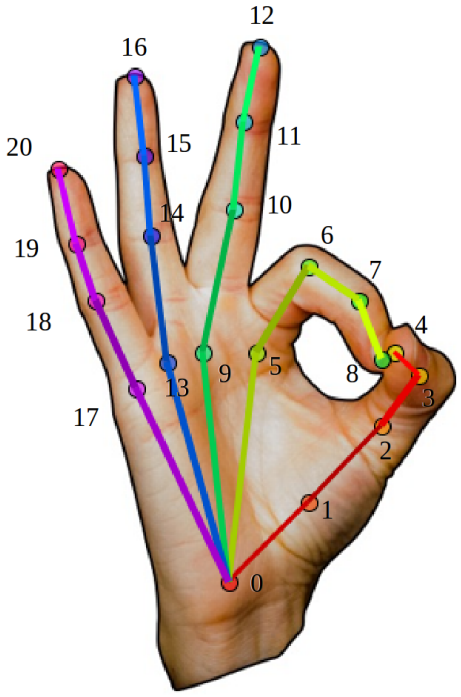


Figure 5.3: Hand keypoints extracted using FrankMocap (Reprinted from (Rong et al., 2021)).

5.2.3 Motion imitation

From a technical point of view, there are several ways to perform imitation. The most straightforward technique is motion retargeting, in which a correlation between the observed and reproduced motion is established a priori. However, this approach requires hand-crafting and usually works only on the specific chosen model. A better approach should involve a learning component between the two models.

Reinforcement learning (RL) (Sutton and Barto, 2018) has been widely applied to imitation learning and behavioural cloning. RL is usually formulated as a Markov Decision Process (MDP) (Bellman, 1957), which has four main components: a set of states S, a set of actions A, a reward function R, and a policy π. A learning agent observes the environment and its own state s, and based on these observations (i.e., state) takes an action a to transition to a new state s' based on a probability

$$P_a(s,s') = Pr(s_{t+1} = s' | s_t = s, a_t = a) \tag{5.4}$$

which yields a reward $R_a(s,s')$ (i.e., an indicator of how good/bad was the chosen

action). The final objective is to learn a policy – a mapping composed of actions chosen by the agent – which maximises the expected cumulative reward. Additionally, a policy can be parametric (π_θ). In this scenario, the policy has to find the optimal parameter θ^* that maximises the expected cumulative reward. There are several algorithms that can be used to find an optimal policy. The algorithm is usually chosen based on the type of action and/or state spaces (discrete or continuous) or can be on-policy or off-policy. For more details, we redirect the reader to (Sutton and Barto, 2018). For our scenario, we use the Proximal Policy Optimisation (PPO) (Schulman et al., 2017b) algorithm to estimate a policy for our problem.

PPO is an optimisation algorithm used when both action and state spaces are continuous. It is a policy gradient method, where the gradient of the expected cumulative reward is calculated using trajectories τ – i.e., sequences of (s,a,r) over a set of contiguous time steps – sampled by following the policy. Thus, given a parametric policy π_θ and T steps, the expected reward is

$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \tag{5.5}$$

where

$$p_\theta(\tau) = p(s_0) \prod_{t=0}^{T-1} p(s_{t+1}|s_t a_t) \pi_\theta(a_t|s_t) \tag{5.6}$$

is the distribution over all possible trajectories

$$\tau = (s_0, a_0, s_1, a_1, ..., a_{T-1}, s_T) \tag{5.7}$$

induced by the policy p_θ, $p(s_0)$ being the initial state distribution and $\gamma \in [0,1]$ is a discount factor used to ensure that the reward has an upper bound. The policy gradient can be estimated as

$$\nabla_\theta J(\theta) = \mathbb{E}_{s_t \sim d_\theta(s_t), a_t \sim \pi_\theta(a_t|s_t)} \left[\nabla_\theta log(\pi_\theta(a_t|s_t)) \mathcal{A}_t \right] \tag{5.8}$$

where is $d_\theta(s_t)$ is the distribution of states under the policy π_θ, while $\mathcal{A}_t = R_t - V(s_t)$ represents the advantage of taking an action a_t from a given state s_t. R_t is the reward by a particular trajectory starting from state s_t at time t, and

$$V(s_t) = \mathbb{E}\left[R_t | \pi_\theta, s_t \right] \tag{5.9}$$

is the value function that estimates the average reward for starting at s_t and following the policy for all subsequent steps.

5.2.4 Problem statement

We formulate the control problem of a robotic hand as an MDP. The action space is composed of different poses (i.e., the combination of joint positions) for the controller, while the state describes a configuration of the hand, position and linear/angular velocity for each component of the hand. Moreover, a phase component $\phi \in [0,1]$ is added to the state space to synchronise the target and reference motions. We represent the policy using a multilayer perceptron with 2 hidden layers, in which input and output dimensions are dictated by the space and action size, respectively. The number of hidden units for each layer is 1024, based on (Peng et al., 2018a). We estimate the pose error by calculating the scalar rotation of the quaternion difference between the simulated hand and the reference motion. At time t, given the desired position $p_{j,t}$ and the simulated position $\hat{p}_{j,t}$ for joint j, the pose error is

$$\varepsilon_t^p = \sum_j ||\hat{p}_{j,t} - p_{j,t}||^2 \tag{5.10}$$

but calculated in the quaternion space rather than the euclidian one, as in (Peng et al., 2018a).

We calculate the velocity error ε_t^v in the same manner, substituting the quaternions representing the pose with the velocities. The target velocity is computed from the data via finite difference

$$\frac{p_{j,t+\Delta t} - p_{j,t}}{\Delta t}. \tag{5.11}$$

Additionally, as done in (Peng et al., 2018a), we calculate an end effectors error ε_t^e and a root error ε_t^r, but in our scenario the end effectors are the fingertips and the root is the wrist. The former ensures that the 3D world position (in meters) of fingertips correspond, while the latter penalises deviations from root orientation when compared to the reference motion. Finally, the reward is calculated on the basis of the errors as follows

$$r_t = w^p r_t^p + w^v r_t^v + w^e r_t^e + w^r r_t^r \tag{5.12}$$

where w^x is a weight manually chosen (with the only condition of the different w^x

summing to 1) and r_t^x is the reward at time-step t for the component x, calculated as

$$r_t^x = e^{-k^x \varepsilon_t^p} \tag{5.13}$$

with k^x being a factor used to balance the reward based on the scale of the error. All the values for w and k are taken from (Peng et al., 2018a) and summarised in Table 5.2. As a proof-of-concept for our proposal, we train a policy for each different task. While it is not as efficient as training a single policy over different tasks, it enables us to easily evaluate the feasibility of our approach.

Measure	w	k
pose	0.65	2
velocity	0.1	0.1
end effectors	0.15	40
root	0.1	10

Table 5.2: Rewards weight and scaling factor from (Peng et al., 2018a).

5.3 Experiment

We carry out three major experiments: one to tune the controller, one for searching hyperparameters, and another for training on the actual data. We use Weights and Biases (Biewald, 2020) to carry out both the experiments for controller tuning and hyperparameters search. In particular, in order to reduce the amount of trials, we use a Bayesian approach instead of a grid search. Moreover, using an approach based on the Bayes rule makes it so the previous $N-1$ trials inform the selection of the parameters for the N^{th} trial. Figure 5.4 illustrates the workflow of our experimental setup. From the flowchart, it is evident how the bayesian approach takes the results of the current run and use them to inform the selection of the hyperparameters for the following attempt. Moreover, it highlights how unsatisfactory performance at each step of the experimental setup takes into account whether the error is introduced by the current step or by previous experiments. For example, if the imitation algorithm does not replicate the movements accurately, we perform an additional check on whether the error is cause by the imitation algorithm or if it is caused by a problem with the controller. Finally, we use Stable Baselines 3 (Raffin et al., 2021) implementation of PPO.

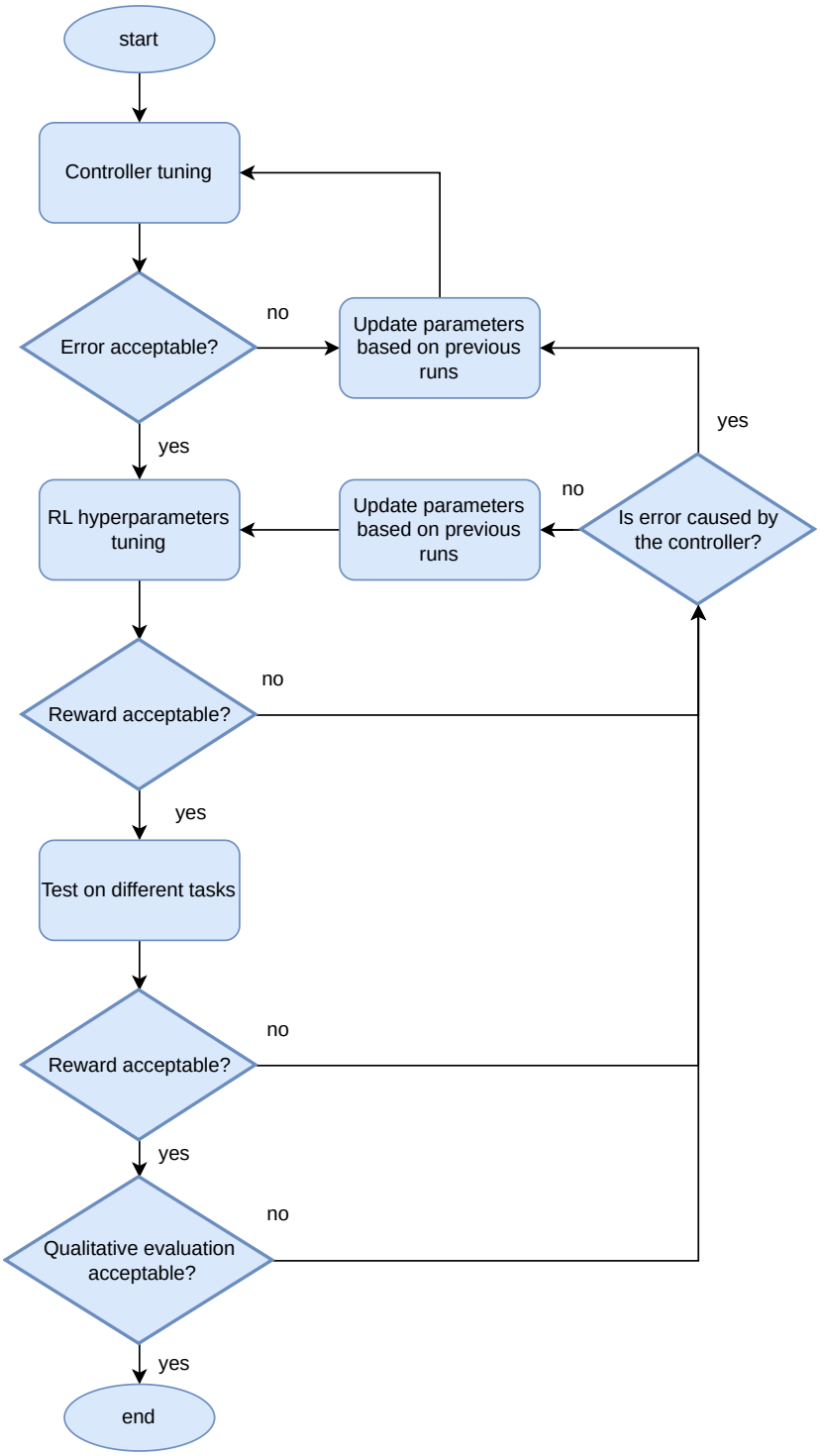


Figure 5.4: Flowchart for the experiments. We divide the whole experimental setup in three stages: tuning the controller, tuning the hyperparameters for the RL algorithm, and training/testing on the actual data.

5.3.1 Controller tuning

Our controller can be tuned using two variables, namely k_p and k_d. Firstly, we generate a random reference motion which we use as a baseline for the tuning. Secondly, we create a function to minimise, which purpose is to make the reference motion and the simulated hand motion as similar as possible. We define this function as the sum of the pose and velocity errors ε_t^p and ε_t^v

$$\varepsilon^{control} = \varepsilon_t^p + \varepsilon_t^v \tag{5.14}$$

where ε_t^p and ε_t^v are defined according to Equation (5.10). This is because we want the simulated motion to resemble as much as possible the reference one, both in terms of *how* (i.e., pose) and *when* (i.e., velocity) the action is executed. Finally, we use a Bayesian optimisation strategy to find values that minimise $\varepsilon^{control}$. To reduce the number of different simulations, we run 3 different swipes, characterised by different maximum values for the parameters k_p and k_d. The three different upper bounds are 100, 10 and 1 for both parameters.

5.3.2 Motion imitation

Model selection and hyperparameter tuning are two fundamental steps in deep and reinforcement learning. However, due to the amount of resources necessary to perform a single training instance (approximately between 8 and 14 hours with 1 NVIDIA RTX 2080Ti GPU and 8 cores, depending on the parameters), we opt for a hierarchical approach.

In the first instance, we explore a subset of hyperparameters in order to understand how they affect training speed and performance. We acquire a reference motion used for the sole purpose of tuning, which can be found in the Appendix D. Table 5.3 lists all the different values we tried during our hyperparameters tuning. Finally, we test the ability of our tuned model to generalise by training the model - with the set of best hyperparameters identified during the previous step - over 6 different reference motions (i.e., fingerspelled letters A, B, C, D, E, F)[1]. We repeat each training session 10 times using different random seeds, in order to ensure statistical significance. The choice of the first six letters of the alphabet is dictated by two factors. In the first instance, these letters can be reproduced despite the limitations we imposed on the model of the hand. Secondly, the choice of limiting the number of letters to six is dictated by a resource

[1] https://www.youtube.com/watch?v=tkMg8g8vVUo

constraint, as each training trail requires several ours and needs to be repeated over different seeds. Assuming 12 hours for a single train trial, repeating the process for 6 letters and 10 random six equals to a total of 720 hours, which corresponds to 30 days using a single GPU.

Parameter	**Values**
learning rate	(1, 3, 10, 30) x 10^{-6}
number of steps	512, 1024, 4096
weight decay rate	(1, 10) x 10^{-5}
batch size	128, 256, 512
orthogonal initialisation	true, false
discount factor	0.9, 0.95
log std dev	-3, -2, -1
number of epochs	3, 5, 10

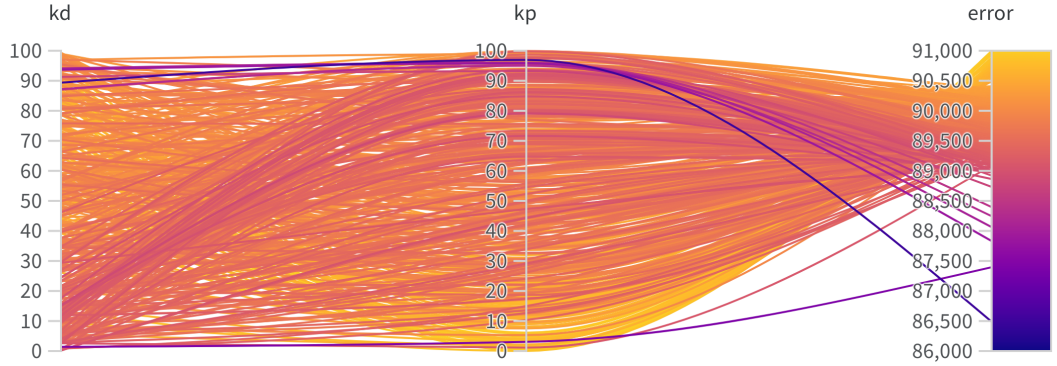
Table 5.3: Values of different hyperparameters explored during tuning.

5.4 Results and discussion

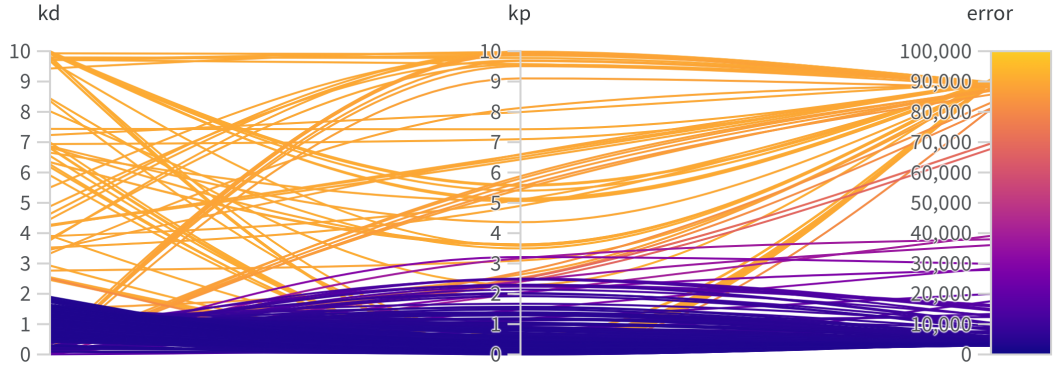
Here, we describe the graphs showing all the different ranges of values (providing Pearson correlation coefficient PCC (Freedman et al., 2007) between the error and the parameters). In addition, we analyse the results of both motion tracking and training different policies to imitate six fingerspelled letters.

Figure 5.5a illustrates the error value for values of k_p and k_d in the range [0, 100], Figure 5.5b for [0, 10] and Figure 5.5c for [0, 1]. Our first sweep is the one with the largest range: [0-100]. On one hand, there is no clear combination of values, which leads to a small error. However, it is noticeable that the runs leading to the lowest error are those in which $k_d < k_p$ (PCC 0.45 for k_d and -0.44 for k_p), even though the error is very high. On the other hand, the sweep with the range [0, 10] indicates that the values leading to the minimum error are the ones in the range [0, 2] (PCC 0.71 for both k_d and k_p). Additionally, we can notice how the lowest values in the error scale are much lower than the ones of the previous sweep (86K vs 10K). The best results are obtained when $k_d > k_p$ and with small values (PCC -0.49 for k_d and 0.51 for k_p).

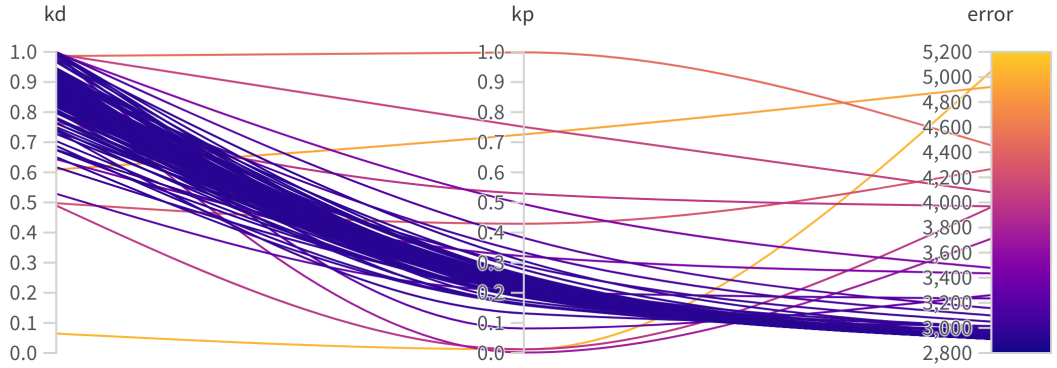
Finally, Figure 5.6 shows a comparison of the position and velocity of the reference and simulated motions of the last phalanx of the index finger, using the best values $k_d = 0.87$ and $k_p = 0.22$. Ideally, we want both position and velocity error to be zero. However, during our experiments we found out that excessive parameter



(a) Max value 100



(b) Max value 10



(c) Max value 1

Figure 5.5: Exploration of different values of k_p and k_d over 3 different optimisation iterations using different ranges of values (100, 10 and 1). The first two columns indicate the combination of k_d and k_p, while the third indicates the error achieve by replicating the motion using those value.

tweaking to reduce one error can lead to a much greater error on the other side. For example, reducing the responsivness of the controller by tweaking k_d can actually increase the pose error, as the controller can be become too responsive and overshoot, or not sufficiently responsive and delay the tracking.

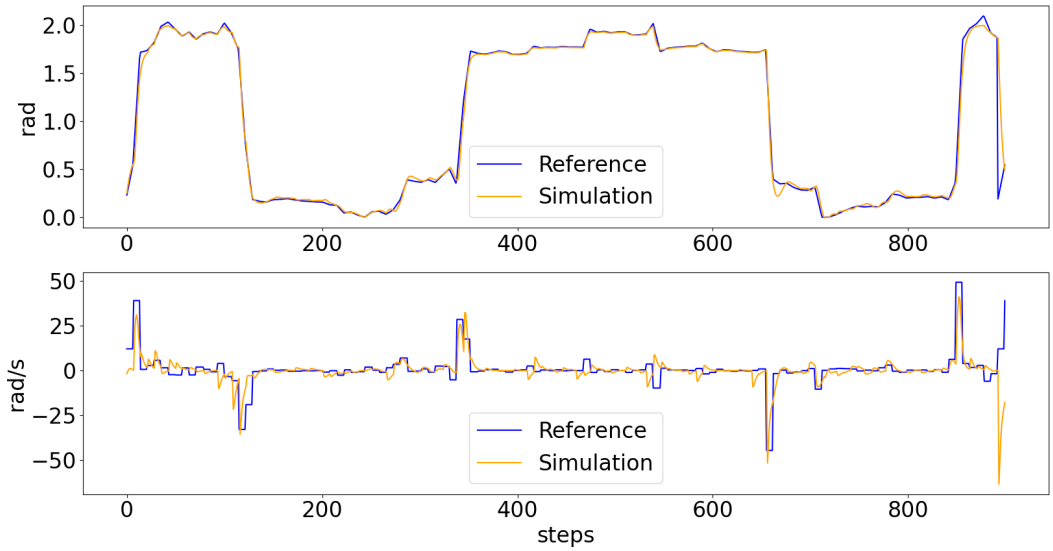


Figure 5.6: Comparison between the reference and simulated position (top) and velocity (bottom) for the last phalanx of the index finger using the best couple of parameters. Forcing the controller to be more responsive (i.e., smaller velocity error) can lead to a higher pose error, as it might overshoot.

During our experiments, we found that no particular hyperparameter stood out for having a specific correlation, which yields a high reward. However, from Figure 5.7, it is clear that there is a huge difference between good and bad combinations of hyperparameters. We found out that the worst run (which leads to a mean reward of 854) uses the following values: $batch\_size = 128$, $gamma = 0.9$, $learning\_rate = 10^{-5}$, $log\_std\_init = -3$, $n\_epochs = 10$, $n\_steps = 1024$, $ortho\_init = False$ and $weight\_decay = 10^{-5}$. On the opposite, the best run (mean reward equal to 1624) uses $log\_std\_init = -2$ but all the other parameters are the same. Nevertheless, when we calculate the PCC between this parameter and the reward, we obtain a value of 0.081, indicating no clear linear correlation between the two values.

For a more in-depth comparison of the relationship between single hyperparameters and the final reward, Figure 5.8 illustrates the correlation matrix between hyperparameters and the final reward, which confirms our finding that no singular hyperparameter significantly contributes to increase the reward, as there is no clear correlation between the reward and any parameter.

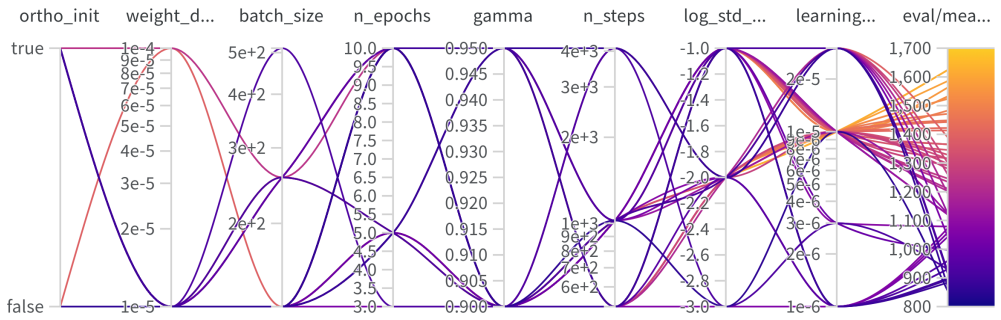


Figure 5.7: Results of hyperparameter tuning across 50 different simulations. For each different combination of hyperparameters, we report the resulting reward.

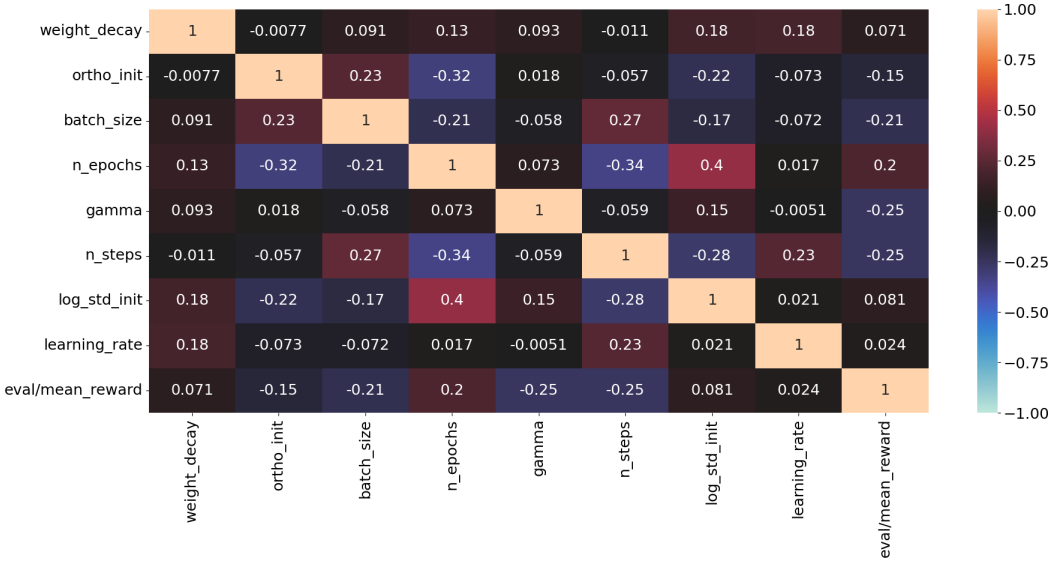


Figure 5.8: Correlation matrix between hyperparameters and reward. Based on the matrix, there is no particular hyperparameter who alone contributes to significantly increase the reward.

Last but not least, we train our policies to imitate six different fingerspelled letters using the previously identified hyperparameters. Figure 5.9 illustrates the results of our training over 10 different seeds. All policies have a minimum value of 0.4 due to the normalised centre-of-mass reward being always 1, given that the hand as a whole is not moving from its initial position. Additionally, all the policies converge at most after 50 million steps. The normalised values for the reward vary between 0.8 and 0.95. Hence, we conclude that our model is able to learn different motions using the same model and hyperparameters. As additional evidence for this statement, we provide frames from videos of the final results in the Appendix D, which can be used for a qualitative evaluation. Finally, we point out that the two dips in the reward curve for

the letter ”F” in Figure 5.9 are due to two separate runs that present a single huge spike around 40 million and 50 million time steps respectively.

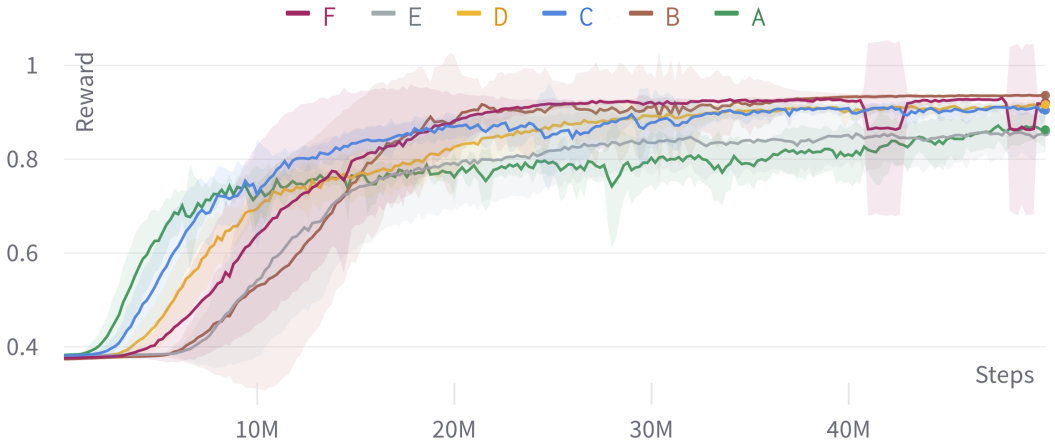
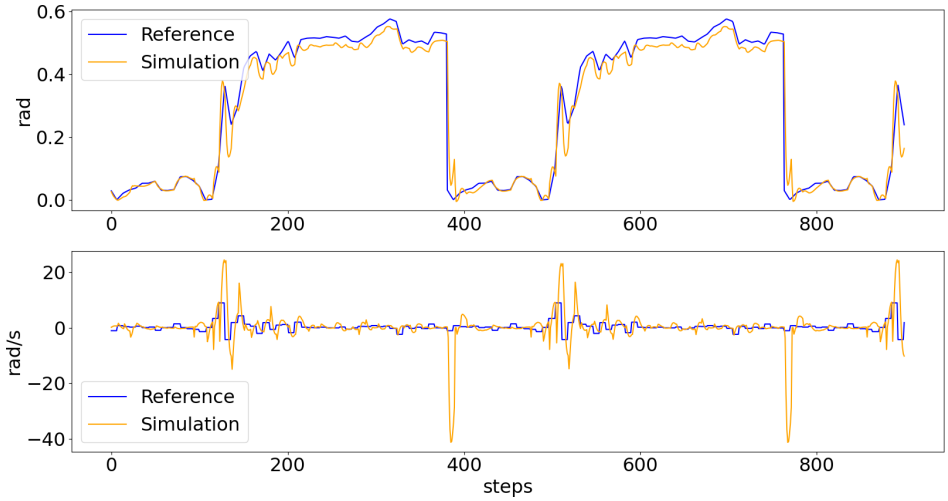


Figure 5.9: Average and standard deviation for the reward of each different letter, calculated over 10 different seeds.

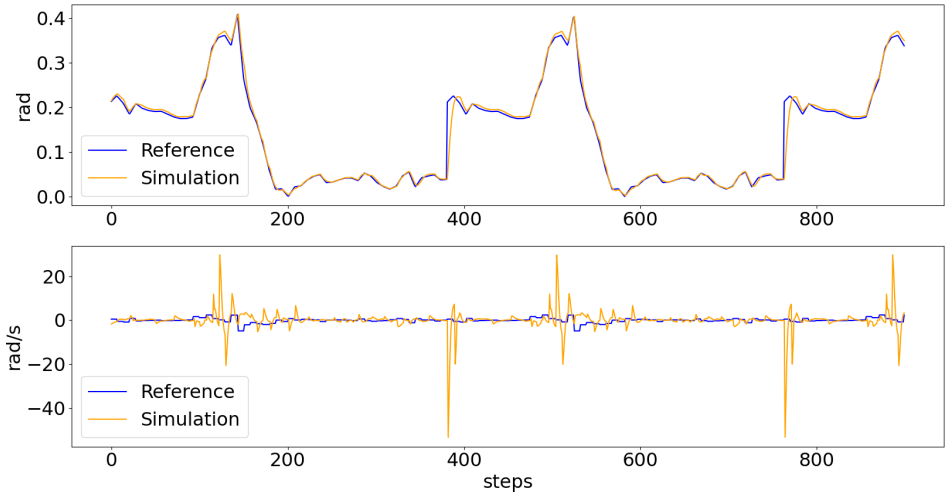
Figure 5.10 illustrates the information regarding the index finger when trying to perform the letter “F”. We can see that, for each joint of the finger, the reference and simulated positions are very close. Hence, we concluded that the non-similarity between the imitated and reference letter “F” is due to limitations in the tracking algorithm.

5.5 Conclusion

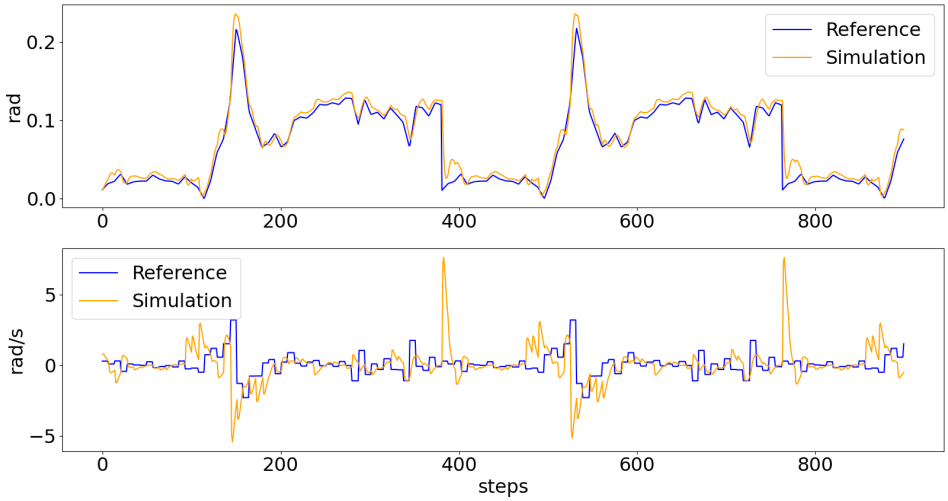
Our research focuses on the acquisition of sign language fingerspelling through imitation learning from RGB videos. This is a challenging task, as it requires the imitation of fine-grained movements. We developed a URDF model of a robotic hand and identified the parameters for the hand PD controller using a Bayesian approach and the hyperparameters for the imitation algorithm. In the end, we achieve imitation over 6 different fingerspelled letters. As future steps, we envision a new simulated robotic hand with additional degrees of freedom, capable of imitating more complex motions. In addition, we plan to expand the evaluation to cover the entire fingerspelled alphabet and explore more efficient methodologies, such as mixture-of-experts (Won et al., 2020) or motion priors (Peng et al., 2021). Such model could not only be used for scenarios in robotics (e.g., generation of grasping poses) but also for generating realistic animations for simulated characters. Ultimately, our goal is to achieve a fully functional model that can be tested and deployed on a physical robotic hand.



(a) Metacarpophalangeal joint



(b) Proximal interphalangeal joint



(c) Distal interphalangeal joint

Figure 5.10: Comparison between the reference and simulated position (top) and velocity (bottom) of the index finger joints for the motion representing the letter “F”.

Chapter 6

Sign language imitation for lemmas

6.1 Introduction

In the previous chapter, we describe how reinforcement learning can be used to teach a simulated robotic hand to learn how to perform fingerspelling based on imitation from RGB videos. Additionally, in the literature, we find examples of how this approach has been used for learning skills which do not involve hands (Peng et al., 2018a,c). Thus, one naturally wonders whether it is possible to combine these approaches and what challenges they might face in trying to do so.

We describe our approach to learning whole-body sign language based on imitation. Figure 6.1 illustrates the different stages of our algorithm. Similarly to our previous approach, we exploit pre-trained deep models to extract 3D information from RGB videos. Then, we feed this data to a learning algorithm based on RL so that our simulated character can learn to replicate the movements following the demonstration. Moreover, we explore the role of hyperparameters and different rewards in the learning process.

6.2 Methods

We begin by discussing how we construct our whole body model. We then provide a brief overview of how we utilise a deep learning model to analyse RGB videos and extract data necessary for learning to imitate. Next, we describe we use reinforcement learning to imitate sign language.

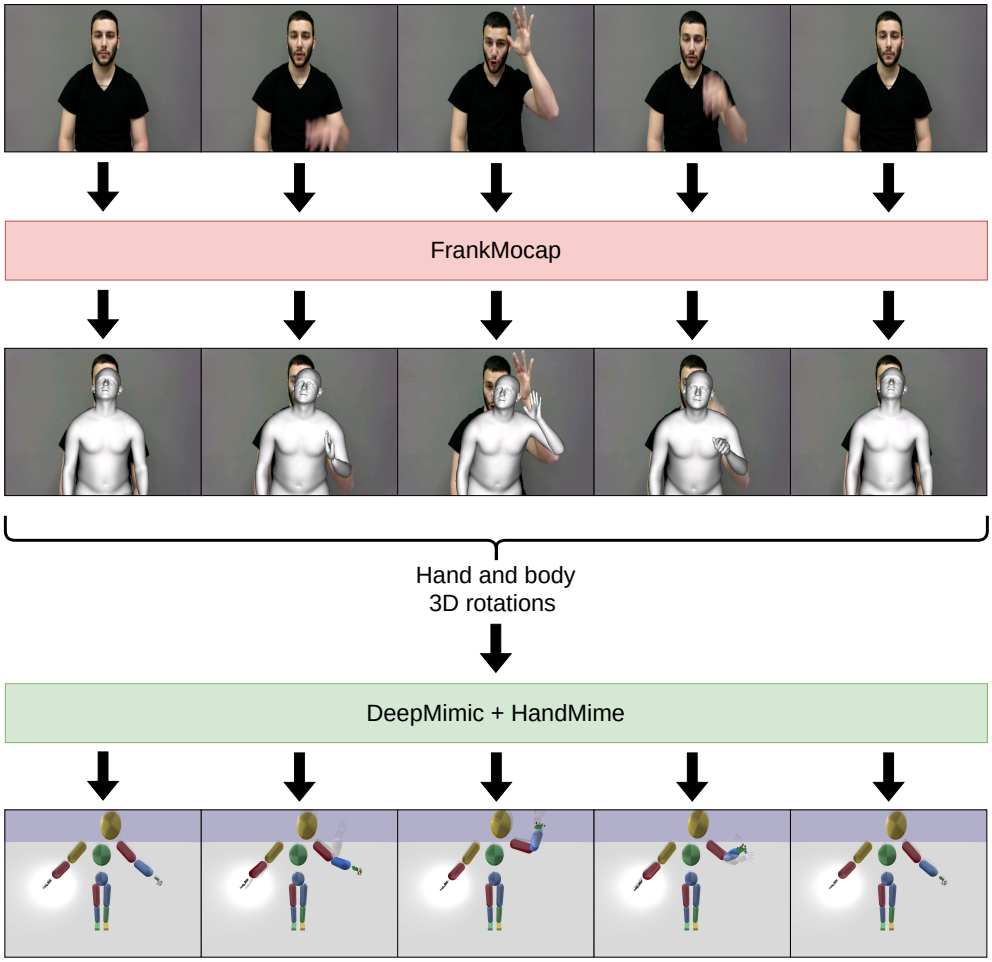
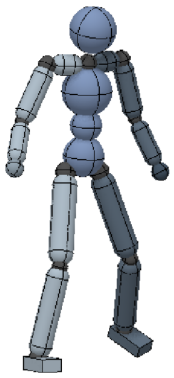


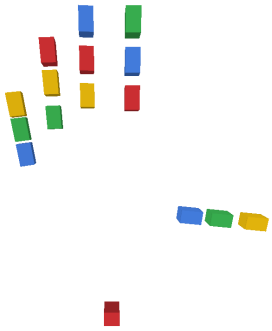
Figure 6.1: Framework for whole body imitation. We extract poses from videos using FrankMocap (Rong et al., 2021) and use them as a reference for our imitation approach, based on DeepMimic (Peng et al., 2018a) and our approach for fingerspelling.

6.2.1 Whole body model

We re-adapt a humanoid model available in the literature to our scenario. In particular, Figure 6.2a illustrates the URDF model from (Peng et al., 2018a). It is composed of 13 joints (2 shoulder/elbows/hips/knees/ankles and 1 neck, chest and root). 9 out of 13 joints have 3 degrees of freedom, while the remaining 4 (knees and elbows) have only 1 DoF, for a total of 31 DoFs. It is 1.62cm tall and weighs 45 kg. As for the hands, we reuse the model we describe in Chapter 5 (re-illustrated for convenience in Figure 6.2b), but we duplicate the model and mirror it to reflect the orientation of the right and left hands. Moreover, we change the shape of the wrist from a small cube to a parallelepiped in order to make it resemble the palm of a hand. Figure 6.3 shows our final model from a different point of view.



(a) DeepMimic body model (Reprinted from (Peng et al., 2018a))

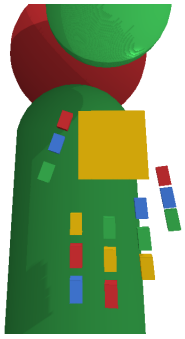


(b) Our hand model

Figure 6.2: Body and hand models (proportions not respected for illustration purposes).

(a) Front view

(b) Side view

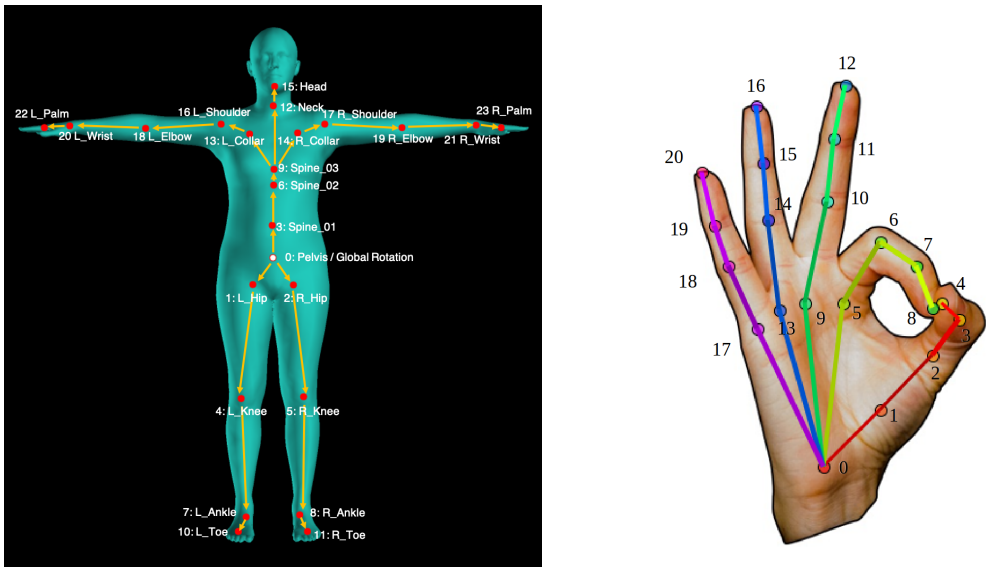


(c) Close-up of the hand

Figure 6.3: Whole body model. We integrated previously avaiable body models with our hand model, replicated and mirrored to obtain both left and right hands.

6.2.2 Motion extraction and imitation

Peng et al. (2018c) uses HMR (Kanazawa et al., 2018) combined with a motion reconstruction approach to extract rotations from videos and feed them to DeepMimic (Peng et al., 2018a). However, their use case involved peculiar motions (e.g., backflips), which are usually not contained in datasets used to train such models. In our scenario, there are no particularly odd motions. Thus, we use FrankMocap (Rong et al., 2021) to extract 3D rotations and keypoints about both the upper part of the body and the hands. Figure 6.4 illustrates the keypoints (and rotations) that we extract for the body and the hands. However, since we are interested in sign language, we disregard information regarding the lower part of the body (i.e., legs).



(a) Body joints

(b) Hand joints

Figure 6.4: Body and hand keypoints extracted using FrankMocap. The body is composed of 24 keypoints, while each hand has 21 keypoints. (Both reprinted from (Rong et al., 2021))

In our previous experiments, we formulate the fingerspelling sign language acquisition problem as a control problem. We model it as an MDP and use PPO (Schulman et al., 2017a) to learn a policy for each different motion. However, as we added a full body model, it was not clear whether it could be possible to reuse Equation (5.12). One option would be to recalibrate the weights of each different sub-reward, but that would imply additional experiments to find the optimal balance between different rewards. In a resource eager scenario like when using reinforcement learning, additional experiments are a major drawback. Hence, we take inspiration from (Won et al., 2020) and define our reward as a *multiplicative* reward (rather than a additive) as follows

$$r_t = r_t^p \cdot r_t^v \cdot r_t^e \cdot r_t^r \tag{6.1}$$

where r_t^x is the reward at time t for quantity x, as previously described in Equation (5.13). In addition, we redefine r_t^p and r_t^v as

$$r_t^p = r_t^{p,b} \cdot r_t^{p,h} \tag{6.2}$$

$$r_t^v = r_t^{v,b} \cdot r_t^{v,h} \tag{6.3}$$

with b indicating the body and h the hands. We choose to divide the rewards (and

the errors) for body and hands as we believe this give us a more clear indication of what the algorithm can (not) learn. As for the end-effectors reward r_t^e, we consider just the position of the wrists, as opposed to our previous approach in which we considered the position of the fingertips.

6.2.3 Problem statement

We define the whole body sign language acquisition approach as an imitation learning problem. As done for the virtual hands, we treat it as an MDP. However, we need to implement some additions to our previous approach, given that our state and action spaces are different. Table 6.1 summarises the differences between the three approaches. Complementary, scaling factors are the same from (Peng et al., 2018a) and described in Table 5.2, for both body and hands.

	DeepMimic	HandMime	Whole upper body
Number of joints	13	16	45
Number of DoFs	34	15	50
State space size	197	210	509
Action space size	36	15	57

Table 6.1: Comparison between DeepMimic, our approach for fingerspelling and our whole body approach.

6.3 Experiment

6.3.1 Controller tuning

As for the controllers tuning, we do not need to go through the whole procedure described for the simulated hand, as we can reuse the value of k_p and k_d from (Peng et al., 2018a) for the body and from Chapter 5 for the hands. However, we need to refine the values for the arms, as the fact that we attached end effectors to them means that we also changed the dynamics of the whole arms. As opposed to our previous approach to tuning the controller parameters, we do not deploy an automated approach, but rather an empirical one. Considering we aim to adapt only the values for 2 joints (i.e., shoulder and elbow), we find it easier to manually adapt the values balancing the

body and hand error rather than develop a sophisticated function to find an acceptable equilibrium between the two errors.

6.3.2 Motion imitation

We exploit a motion file from (Rong et al., 2021) for the purpose of hyperparameter tuning. However, preliminary results immediately showed that the scaling factors for the hands (that were the same as for (Peng et al., 2018a)) did not yield to satisfactory results. In fact, as the reward is a product of sub-rewards, if one of the sub-rewards is 0, then the total reward will be 0 as well. Thus, we run additional searches for the parameter for hand pose and velocity. We use a mixture of automated and manual approaches, alternating hyperparameters and scaling factors explorations. We test the permutation of the following values for pose and velocity: [2, 1, 0.5, 0.2] and [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}] respectively. Intuitively, given that the reward is expressed as

$$r = e^{-kx} = 1/e^{kx}, \tag{6.4}$$

tuning the scaling factor k makes the reward more (or less) lenient towards the error x. In practice, we want a reward that is permissive enough to enable our algorithm to learn, but not too much so that it promotes very high errors. Figure 6.5 illustrates this property. We can see how, given the same value of x, functions that have a smaller k result in a higher value. Thus, keeping in mind that some quantities might be prone to higher errors than others, it becomes trivial to understand how choosing a proper value for k is fundamental.

We perform a hyperparameter search based on Bayesian optimisation to find the ideal hyperparameters. When performing such a search, we train our algorithm for 25 million steps, which is 50% of how long we usually train to achieve imitation. The set over which we perform the search is summarised in Table 6.2. Finally, we test the generalisability of such hyperparameters over five different signs, repeating each attempt 10 times using different seeds and training for 50 million steps. We choose the different signs via a qualitative evaluation of the output of FrankMocap. As explained in Section 5.3.2, we choose a small amount of different motions to imitate due to extensive time required to train and statistically validate multiple policies. In addition, when choosing the signs we also take into account the limitations of our model (e.g., only 1 DoF for the elbow, wrist or phalanges of each finger). The lemmas corresponding to the five signs are `above`, `snow`, `father`, `mother` and `yes`. Their

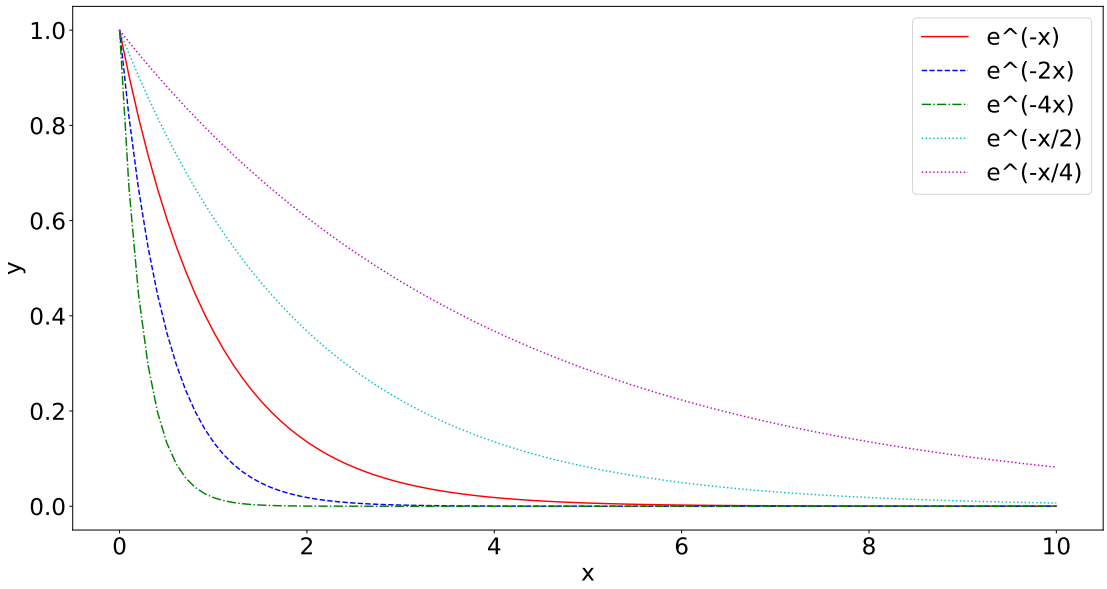


Figure 6.5: Example of how the function $y = e^{-kx}$ changes with different values of k. As k decreases, the curve becomes less steep.

respective identifiers from WLASL (Caselli et al., 2017) are `00433`, `52861`, `69318`, `69402` and `69546`, which we use as identifiers of the lemmas in our results.

Parameter	**Values**
learning rate	(1, 3, 10, 30, 100) x 10^{-6}
number of steps	512, 1024, 4096
batch size	128, 256, 512
log std dev	-5, -3, -2, -1
discount factor	0.9, 0.95
number of epochs	3, 5, 10

Table 6.2: Values of different hyperparameters for the PPO algorithm we explored during tuning.

6.4 Results and discussion

Our fist steps aim at finding the ideal scaling factors $k^{p,h}$ and $k^{v,h}$ for the rewards $r_t^{p,h}$ and $r_t^{v,h}$. We explore the permutations of $k^{p,h}$ = [2, 1, 0.5, 0.2] and $k^{v,h}$ = [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}] using a Bayesian approach. Figure 6.6 illustrates the results of our experiments. There is a combination of parameters which is clearly the best, the one with $k^{p,h} = 0.2$ and $k^{v,h} = 10^{-4}$. This means that, in order to enable our algorithm to

be able to learn, our reward needs to be more lenient with the errors regarding the pose and velocity of the hands.

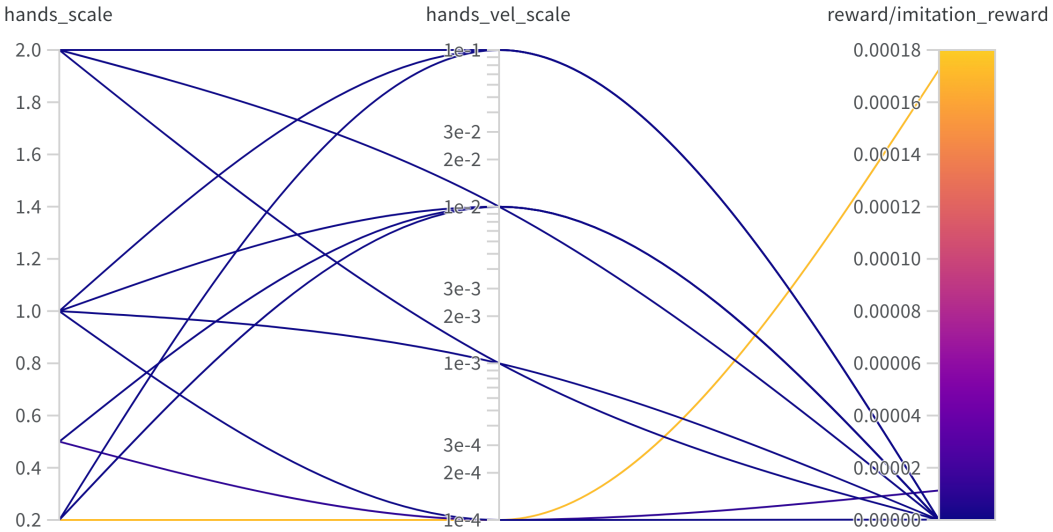
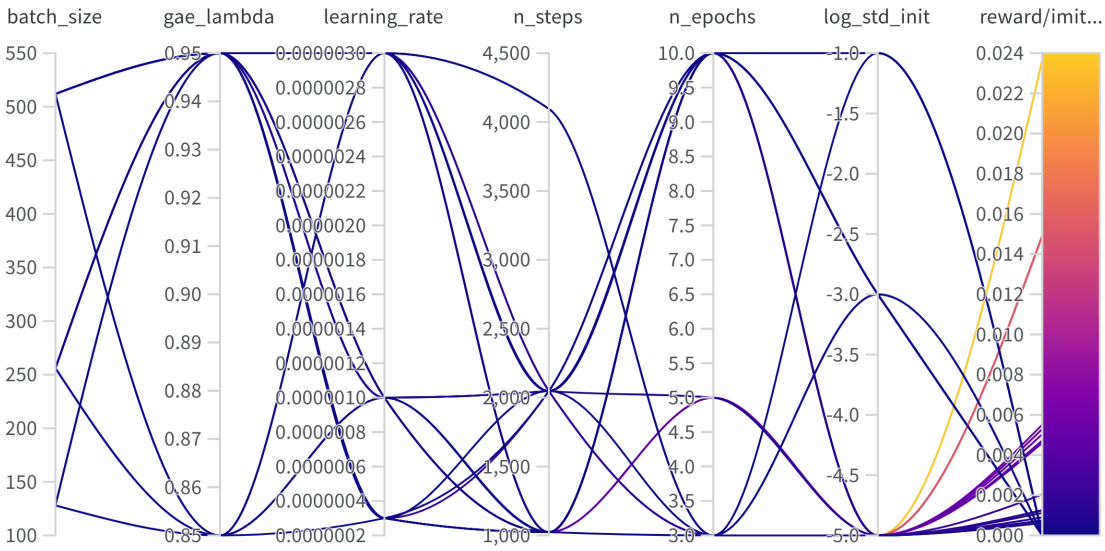


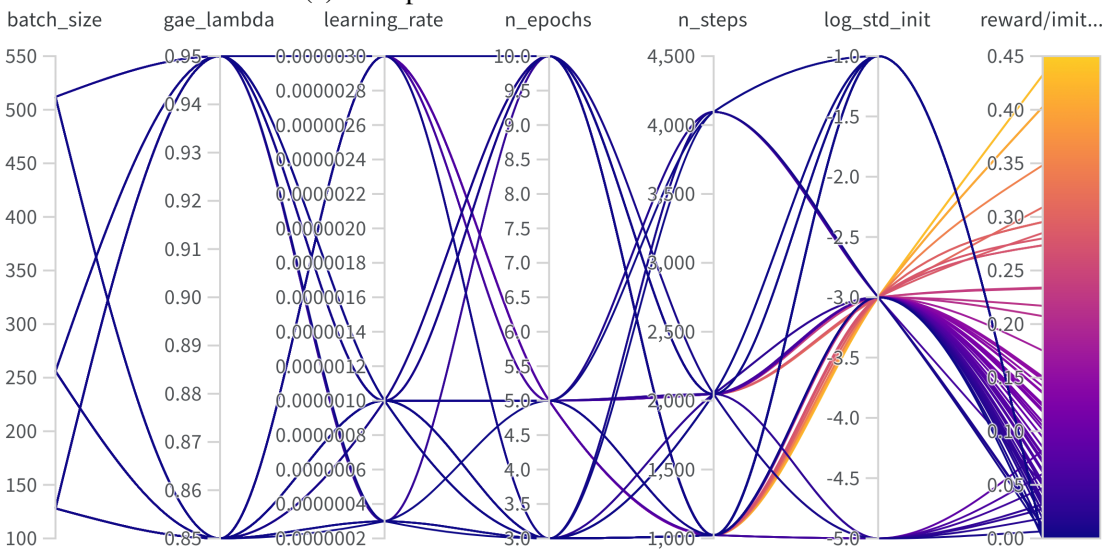
Figure 6.6: Exploration of different values for $k^{p,h}$ and $k^{v,h}$. We provide the reward for each different combination of parameters.

Consequently, we perform two hyperparameter searches to test different combinations of values $k^{p,h}$ and $k^{v,h}$. In particular, Figure 6.7a illustrates the sweep for $k^{p,h} = 0.5$ and $k^{v,h} = 5 \cdot 10^{-4}$, while Figure 6.7b shows the sweep for $k^{p,h} = 0.2$ and $k^{v,h} = 10^{-4}$. Clearly, the former does not provide any satisfactory result (i.e., max reward after 25 million steps around 0.02), while the latter provides some interesting candidates, as a couple of runs have a reward higher than 0.4 at 50% of the training. Thus, we confirm that the ideal parameters are $k^{p,h} = 0.2$ and $k^{v,h} = 10^{-4}$.

We test these parameters on the sign `above`. We analyse the results by checking the error of the imitation results – i.e., the difference between the reference motion and the simulated one. By doing so, we observe a delay between the replicated and reference position of elbows (Figure 6.8a). Thus, we modify the velocity gain k_d for the shoulder and elbow, in order to improve the responsiveness of our model. By changing the value from 40 to 8 for the shoulder and 30 to 6, we notice a significant improvement (Figure 6.8a vs Figure 6.8b) without significantly affecting the rest of the connected joints, like the wrist (Figure 6.8c vs Figure 6.8d). In addition, as we changed one of the parameters of the controller and slightly increased the velocity error, we decreased the scaling factor via some manual tuning. We change $k^{b,v}$ from 10^{-1} to $5 \cdot 10^{-3}$ so that the reward related to the body velocity error is higher. In the end, we select the configuration summarised in Table 6.3.



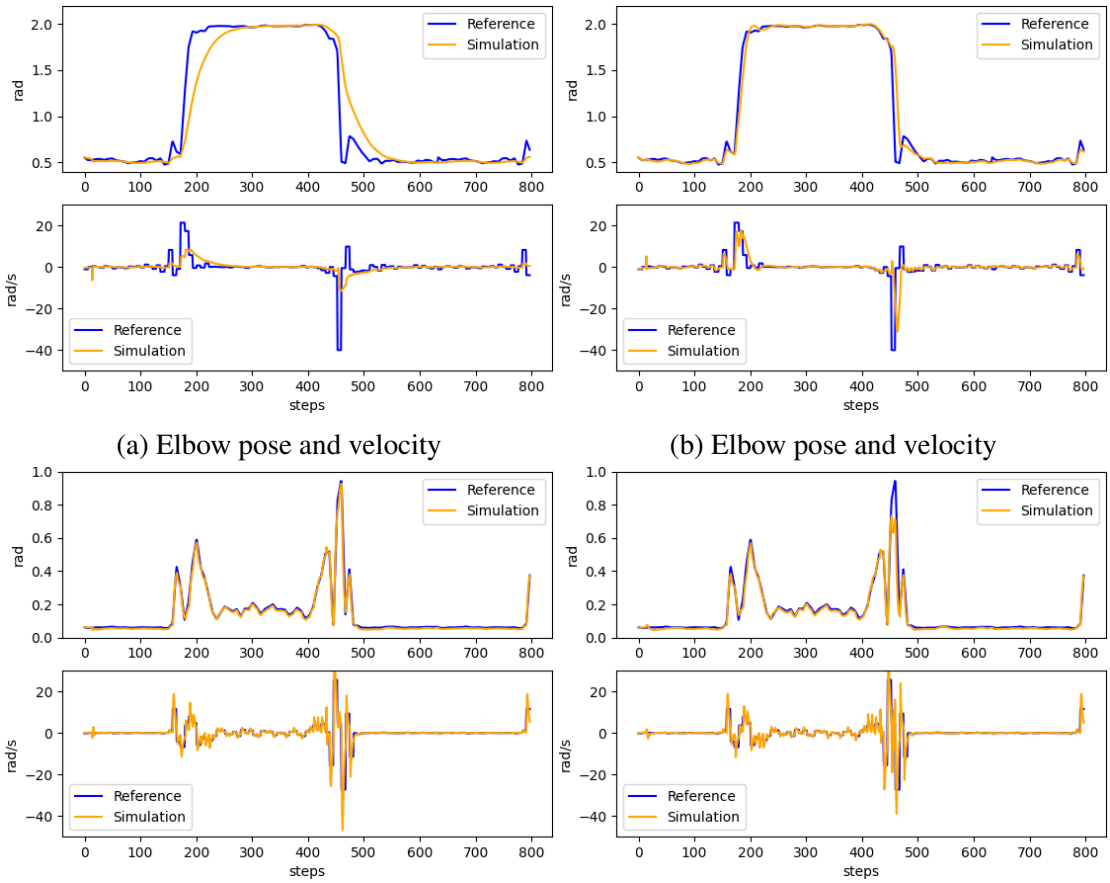
(a) Sweep for $k^{p,h} = 0.5$ and $k^{v,h} = 5 \cdot 10^{-4}$



(b) Sweep for $k^{p,h} = 0.2$ and $k^{v,h} = 10^{-4}$

Figure 6.7: Hyperparameter search for PPO trained on the tuning motion using different values of $k^{p,h}$ and $k^{v,h}$

.



(a) Elbow pose and velocity

(b) Elbow pose and velocity

(c) Wrist pose and velocity

(d) Wrist pose and velocity

Figure 6.8: $k_d = 40$ for elbow and $k_d = 30$ for wrist (left side) vs $k_d = 8$ for elbow and $k_d = 6$ for wrist (right side).

hidden layers size	learning rate	N steps	batch size	log std dev	discount	N epochs
256, 512, 256	$3 \cdot 10^{-6}$	1024	128	-3	0.95	10

Table 6.3: Final set of hyperparameters selected using the tuning motion.

Table 6.4 summarises the different scaling factors we explore. In addition, we provide an approximation of the component $r^p \cdot r^v$ of the reward, based on the error measured while replicating the original movement and Equation (6.4). We can see how changing $k^{b,v}$ can bring a (hypothetical) increment of approximately 10%.

Given that we identified reward and hyperparameters, we proceed to test the generalisability of such values over different motions. Figure 6.9 shows the results for the five different signs. Each sign is calculated as mean (and standard deviation) over 10 different seeds. We can see that, out of five different signs, only three achieve a reward

	Default	Run 1	Run 2	Run 3
$k^{p,b}$	2	2	2	**2**
$k^{p,h}$	2	$5 \cdot 10^{-1}$	$2 \cdot 10^{-1}$	$2 \cdot 10^{-1}$
$k^{v,b}$	10^{-1}	10^{-1}	10^{-1}	$5 \cdot 10^{-3}$
$k^{v,h}$	10^{-1}	$5 \cdot 10^{-4}$	10^{-4}	10^{-4}
$r^p \cdot r^v$	0	0.297	0.694	**0.790**

Table 6.4: Estimated sub-rewards for different scaling values, calculate using the tuning motion.

of around 0.8. The two lemmas for which the algorithm does not converge to acceptable results are `father` and `mother`. The signs are similar, as both involve placing the hand fully open, with the former attaching the thumb forehead and the latter to the chin.

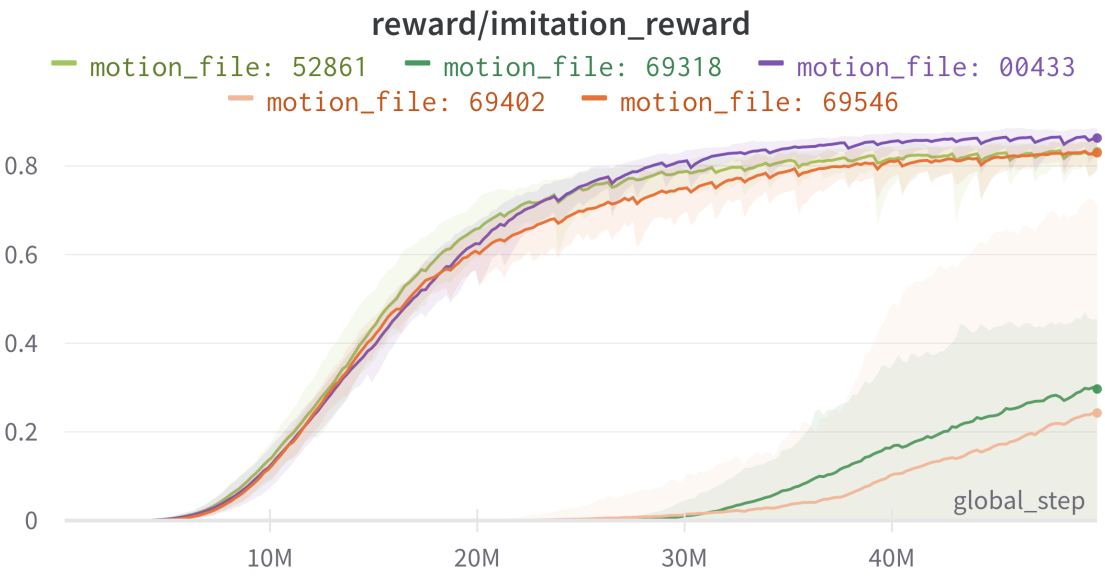
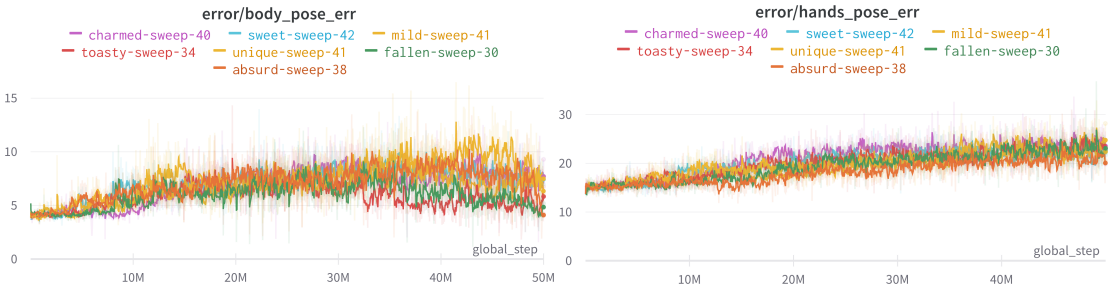


Figure 6.9: First run of 5 signs, the average reward is calculated over 10 different seeds

By looking at the single runs, we notice that the error never converges over 7 different runs with different seeds, as illustrated in Figure 6.10.

Considering we change the reference motion, we roll back to test two previous configurations for the scaling factors. In this way, we can double-check whether $k_{p,h}$ and $k_{v,b}$ actually contribute to the learning process. We carry out a hyperparameters search over the following configurations, summarised by the first two columns of Table 6.5.

The first one aims to reduce the relevance of the reward regarding of the pose of the hands, while the second one targets the body and hand velocity. However, as we can



(a) Body pose

(b) Hands pose

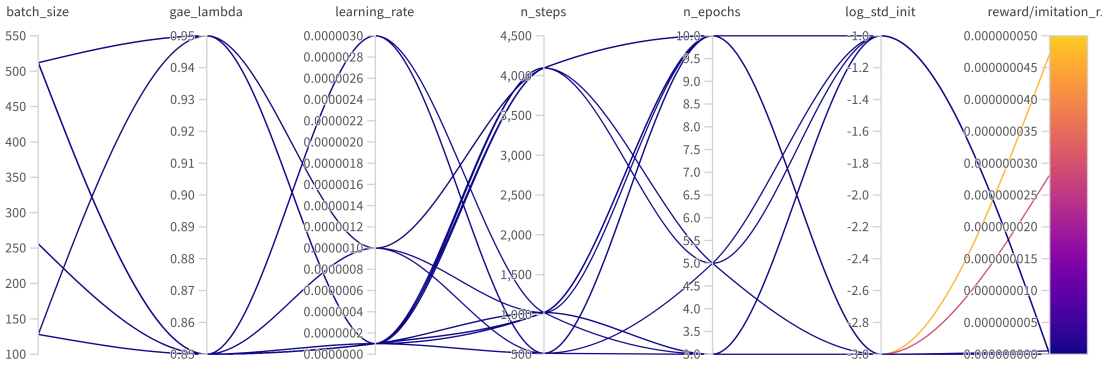
Figure 6.10: Moving average of the error which does not converge for 7 different seeds.

	Run 1	**Run 2**	**Run 3**	**Run 4**
$k^{p,b}$	2	2	1	$5 \cdot 10^{-1}$
$k^{p,h}$	2	$2 \cdot 10^{-1}$	$2 \cdot 10^{-1}$	$2 \cdot 10^{-1}$
$k^{v,b}$	$5 \cdot 10^{-3}$	10^{-1}	10^{-3}	$5 \cdot 10^{-3}$
$k^{v,h}$	10^{-4}	$5 \cdot 10^{-4}$	10^{-4}	10^{-4}
$r^p \cdot r^v$	0.656	0.060	**0.785**	0.715

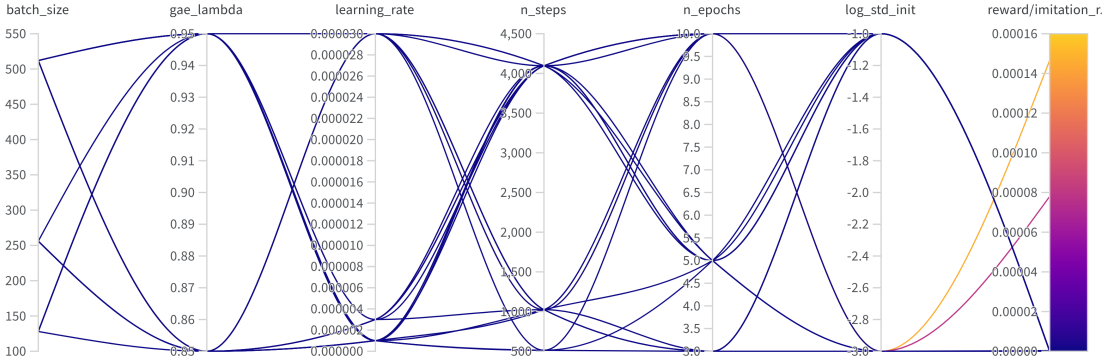
Table 6.5: Estimated sub-rewards for different scaling values, calculate using the `father` motion.

see in Figure 6.11a and Figure 6.11b, neither of the sweeps produces a single run in which the different error components converge, and thus the reward is approximately 0. Hence, we try to modify the scaling factors according to the last 2 columns of Table 6.5. In the first run (Figure 6.11c), we change all the factors except for the one associated with the pose of the hands. We can see a significant improvement over the previous sweep. However, we notice that the reward for the body pose oscillates considerably, while the reward for the body velocity has a high value but with very few variations. Thus, we try to address these issues with our second sweep. We reduce the scaling factor associated with the body pose and increase the one associated with the body velocity. From this final sweep, we select a set of hyperparameters to test over the other signs. Table 6.6 summarises the best set of hyperparameters we discover in our extensive exploration. Overall, the only difference is the parameters regarding the number of epochs and the number of steps, which changed from 10 to 5 and 1024 to 512 respectively.

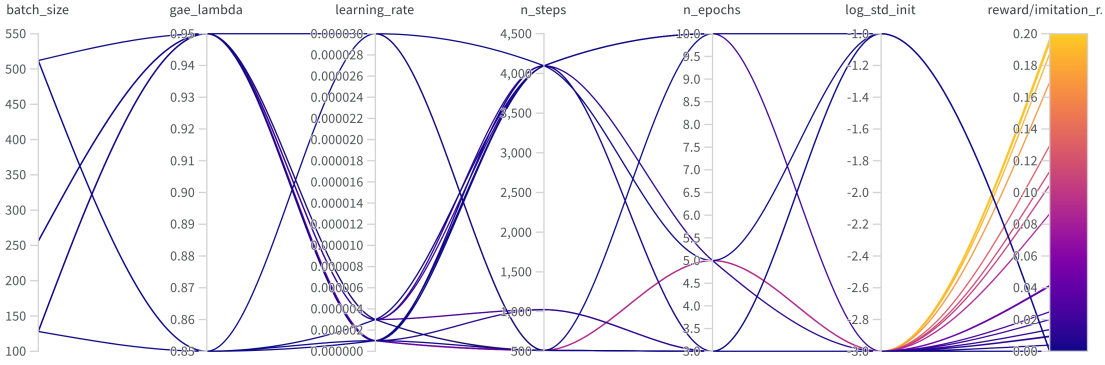
Finally, we train our model with the selected hyperparameters over the different signs. Figure 6.12 shows the learning curves as average (and standard deviation) over 10 different seeds. We can see how, when compared to Figure 6.9, the learning curve is



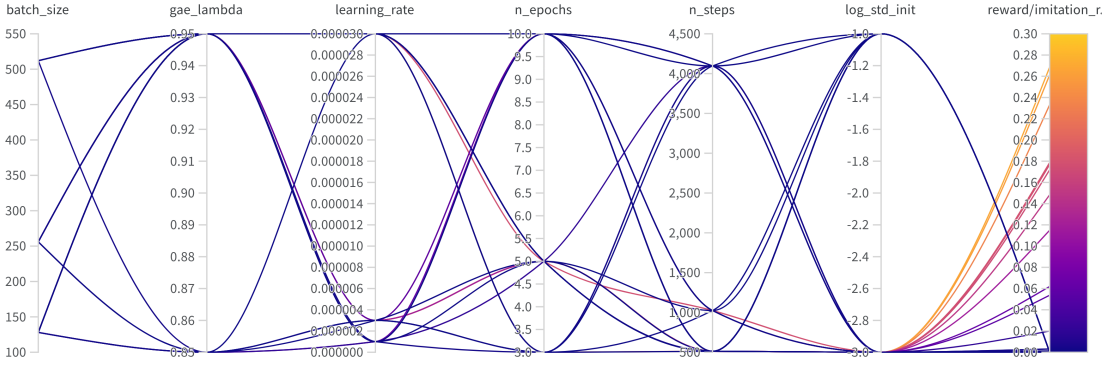
(a) Sweep for $k^{p,b} = 2$, $k^{p,h} = 2 \cdot 10^{-1}$, $k^{v,b} = 5 \cdot 10^{-3}$, $k^{v,h} = 10^{-4}$



(b) Sweep for $k^{p,b} = 2$, $k^{p,h} = 2$, $k^{v,b} = 10^{-1}$, $k^{v,h} = 5 \cdot 10^{-4}$



(c) Sweep for $k^{p,b} = 1$, $k^{p,h} = 2 \cdot 10^{3}$, $k^{v,b} = 10^{-3}$, $k^{v,h} = 10^{-4}$



(d) Sweep for $k^{p,b} = 5 \cdot 10^{-1}$, $k^{p,h} = 2 \cdot 10^{-1}$, $k^{v,b} = 5 \cdot 10^{-3}$, $k^{v,h} = 10^{-4}$

Figure 6.11: Sweeps on the motion file for the sign `father`.

hidden layers size	learning rate	N steps	batch size	log std dev	discount	N epochs
256, 512, 256	$3 \cdot 10^{-6}$	512	128	-3	0.95	5

Table 6.6: Selected hyperparameters using the motion representing the sign `father`. These hyperparameters are selected following quantitative (i.e., reward) and qualitative (i.e., visually analysing the training curve) evaluations.

less steep. However, all five signs achieve a reward above 0.7, as opposed to our previous attempt in which two of them did not surpass 0.5. Table 6.7 provides a convenient comparison of the results of the two different runs.

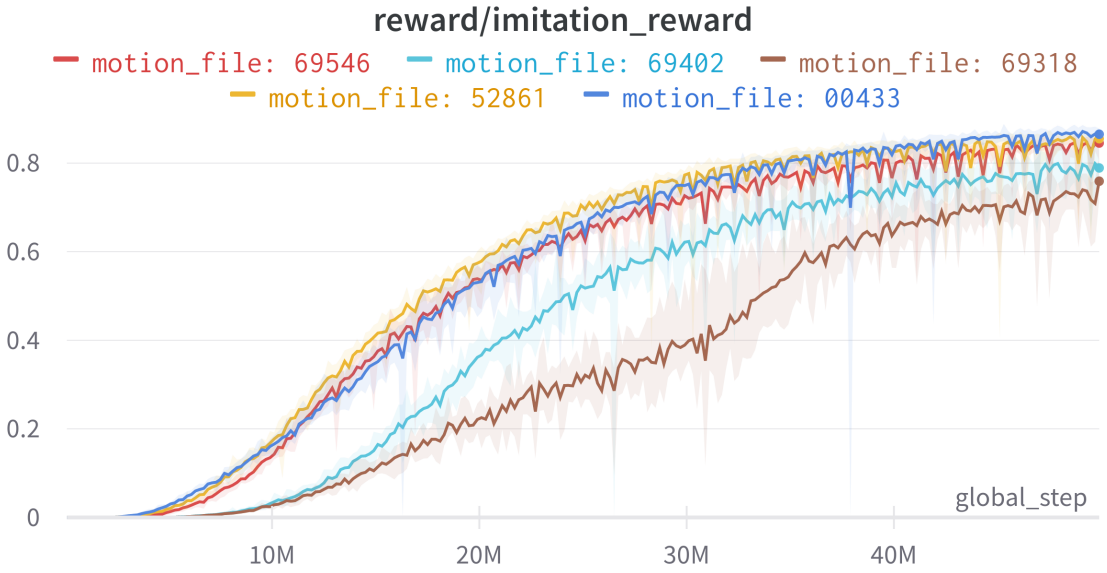


Figure 6.12: Final run, the average reward is calculated over 10 different seeds

Motion	Run 1	Run 2
00433	0.827 ± 0.020	$\mathbf{0.868 \pm 0.007}$
52861	0.832 ± 0.022	$\mathbf{0.853 \pm 0.017}$
69318	0.297 ± 0.166	$\mathbf{0.745 \pm 0.018}$
69402	0.243 ± 0.304	$\mathbf{0.791 \pm 0.032}$
69546	0.830 ± 0.020	$\mathbf{0.842 \pm 0.018}$

Table 6.7: Comparison between the rewards over the two different sets of hyperparameters

6.5 Conclusion

In this chapter, we address the problem of sign language acquisition from RGB video for lemmas. We create a URDF model of a simulated character, which to the best of our knowledge is the first to enable imitation of the whole body and both hands. We describe how available pre-trained pose estimation models can be used to extract information about 3D rotations from videos and how we can model rewards to enable imitation. Then, we run extensive experiments regarding parameters for rewards and hyperparameters for training models. As often happens with reinforcement learning, we iterate through different steps of experiments and analysis to identify the ideal parameters. In the end, we identify a reward and a set of hyperparameters that enable our approach to learn how to imitate 5 different signs.

Chapter 7

Conclusions

7.1 Overview

Artificial agents like robots have the potential to revolutionise many fields, like industrial settings, hospitality and healthcare. Such changes could impact the lives of billions of people, with many social consequences. However, in order to reach that point, they face many challenges ahead. Currently, there are several limitations for these agents, as they

1. are not as mobile as humans, meaning that they cannot properly navigate complex environments and adapt to unexpected obstacles,
2. have difficulty recognising and interpreting their surroundings, which makes it hard for them to interact with the environment in a human-like way,
3. are not nearly as good as humans at grasping and manipulating objects, especially when it comes to delicate or irregularly-shaped objects,
4. have difficulty understanding and responding to human emotions, gestures, and social cues, which makes it hard for them to interact with humans in a natural way, and
5. are expensive, which makes it hard for companies to develop, maintain or generally invest in them.

On the one hand, one cause of such limits is hardware. Currently, very few machines are built with the hardware necessary to perform complex and diverse tasks. For example, most humanoid robots are not equipped with dexterous hands. Additionally,

there is no company specialising in the production of components on a large scale. On the other hand, some of these limitations can be attributed to software. For a long time, motion and manipulation have been approached as control problems. However, while this has led to good results in specific scenarios, it does not generalise well, as opposed to humans, who are capable to learn and generalise to new scenarios.

In our work, we investigate the problem of acquiring sign language in artificial agents as a problem of learning dexterous skills from demonstrations. In particular, we teach a simulated agent how to speak sign language by imitating videos of people speaking different words. However, given the lack of 3D pose annotations for sign language scenarios, we exploit phonological classes to assess the quality of pose estimators based on deep models. We demonstrated how a couple of these properties (i.e., where the sign is executed and how many hands it involves) can be recognised without multiple examples of the same signs by fairly simple algorithms, such as SVM or MLP. Moreover, while creating a large-scale dataset of signs and associated phonological properties, we show how a more advanced model like STGCN can recognise several phonological classes, regardless of whether the same sign is executed by different individuals is both in the training and the test set. Thus, we can conclude that – in some measure – pose estimators are capable of extracting from videos the fine-grained movements described by phonological classes. Finally, we demonstrate how the data extracted can be used to make a simulated agent acquire sign language. In the first instance, we provide evidence that an approach based on reinforcement learning (and in particular PPO) is a feasible solution for fingerspelling acquisition. Then, we describe how such an approach can be adapted to a scenario in which the avatar has to replicate signs using its whole upper body.

In conclusion, we demonstrate how pose estimation models combined with reinforcement learning are viable options for imitation learning applied to sign language. The experiments we conducted demonstrated how the 3D information extracted from videos can be used to recognise phonological properties. Complementary, we showed how recognition algorithms on specific fine-grained movements can be used to compare pose estimation models. In doing so, we also create a dataset of videos of people speaking ASL and, for each word, their respective phonological classes. Finally, we show how methods used in computer animation can be adapted to teach a simulated humanoid fingerspelling and, more generally, sign language.

7.2 Summary of contribution to knowledge

This section revisits the research questions and expected contributions formulated in Chapter 1 in light of what we discussed throughout the previous chapters. The key goal of the work presented in this thesis was to address the problem of learning based on imitation applied to sign language. More specifically, we wanted to teach a simulated character sign language based on information that can be extracted from video demonstrations.

Here we reiterate the research questions introduced in Chapter 1 and provide an answer to each of them based on the work discussed throughout this thesis:

RQ1 *Given the lack of 3D annotations for sign language, can a pose estimation model extract data which can be recognised in an automated fashion as high-level properties of sign language?*

In our experiments, we trained several different machine learning models to recognise sequences of 3D keypoints. Such sequences were extracted using a pre-trained pose estimation algorithm based on deep learning. Based on the results we obtained, we can affirm that our approach can indeed extract data which are good enough to represent high-level properties of ASL, like where the sign is executed and the number of hands that are being used. Tavella et al. (2022a) refers to this contribution.

RQ2 *Can classification algorithms generalise to a larger set of signs and properties? If so, can they also recognise the same properties over unseen signs?*

We generated a dataset with thousands of different lemmas (represented as sequences of keypoints) and associated phonological classes. The former is obtained by two different pose estimators, while the latter represents different fine-grained movements and pose configurations during each sign. Once again, we trained different models to recognise each different property. As these algorithms perform above the expected baseline, we can conclude that the data extracted contains the specific patterns identified by the phonological classes. However, we notice how the ability of our algorithms is influenced by the fact that examples of the same signs, although shown by different signers, are both in training and test set. Hence, while it is still possible to recognise properties

over unseen signs, our findings show that doing so causes a drop in accuracy by at least 10 points. Tavella et al. (2022b) refers to this contribution.

RQ3 *Does information extracted by pose estimation model constitute a good source to learn fingerspelling on a robotic hand based on imitation?*

Our approach extracts 3D hand poses from videos exploiting a pre-trained pose estimation algorithm. Then, we train a reinforcement learning algorithm to replicate the sequence of poses, using the extracted ones as a reference. Our experiments demonstrate how it is possible to perform a hyperparameter search on a reference motion and transfer the same hyperparameters to train the same network on 6 different motions representing fingerspelled letters in ASL. Hence, we conclude that it is in fact possible to use pose estimation models as an input source to learn fingerspelling based on imitation. Tavella et al. (2023) refers to this contribution.

RQ4 *Can the approach adopted to learn fingerspelling generalise to a more complex task like learning signs involving the whole upper body?*

We replicate the procedure we defined for the fingerspelling approach. However, in this scenario, running a hyperparameter search on a reference clip and transferring hyperparameters to new examples does not work for all the different motions. This result shows the importance of a proper reward for reinforcement learning settings. Hence, we update the reward and repeat the hyperparameter tuning based on the clip returning the lowest reward. This time, all 5 networks can learn how to replicate the signs represented by the respective 3D motion files. Hence, we conclude that it is possible to transfer our previous approach to a full-body scenario. However, it is worth noticing that the quality of the results is heavily dependent on the pose estimator, so any error produced by it will be replicated by the imitation approach.

To summarise, our overall achieved contributions are the following:

1. a small dataset (i.e, less than 800 samples) of videos of people performing signs, 3D keypoints and the associated phonological properties,
2. a large-scale dataset (i.e., more than 10000 samples) of videos of people performing signs, 2D and 3D keypoints and the associated phonological properties,

3. a benchmark of different classification algorithms over our large-scale dataset, defining a baseline on which other researchers can confront their approaches,
4. motion files of 3D rotations for 6 different letters of the fingerspelled ASL alphabet,
5. a simulated hand model, its respective controller and an environment to train reinforcement learning algorithms to replicate fingerspelling based on examples,
6. motion files of 3D rotations for 5 different ASL lemmas, and
7. a simulated humanoid model with both hands, its respective controller and an environment to train reinforcement learning algorithms to replicate signs based on examples.

7.3 Limitations

This section provides an overview of some of the limitations of the work presented in this thesis and discusses some possible solutions. During the discussion of the results in the various chapters, we already described some limitations. However, we report them for the convenience of the reader and to further discuss them.

- Probably, the most obvious limitation of our work on the properties of sign language is that our study focuses on American Sign Language. Thus, it is not clear whether they could transfer to other signed languages. In the end, other signed languages use the same modality to communicate (i.e., organised movements of arms, hands and facial expressions), so we can expect some of the characteristics to translate to other languages.
- An additional limit of our approach is that we base our methodology on a supervised approach. As such, our algorithms are only able to distinguish classes that are in the training set. Thus, if any of the samples present characteristics that were not already seen (i.e., an outlier), it is not recognisable by the algorithm. Alternatively, an approach that distinguishes samples solely based on their characteristics rather than matching features to labels could potentially group samples that share such features.
- Let's recall that the flexion property is defined as the aperture of the selected fingers of the dominant hand at sign onset. Our experiments showed how, in the

case where the same property is not in both the training and test set, none of our algorithms can recognise such properties. This needs further investigation, but the most plausible hypothesis is that the pose estimation algorithm is not able to distinguish different hand poses, and thus the extracted keypoints and rotations are not recognisable. Moreover, the different values for the flexion can be quite similar. For example, one class is defined as ”base and non-base joints flexed with contact”, while another as ”base and non-base joints fully flexed”. Interestingly,

- When we created our hand model, we limited the degrees of freedom for the sake of simplicity. While this led to satisfactory results so far, it will need to be addressed in the future. Doing so will enable not only a more accurate imitation of sign language but also a model able to perform other tasks (e.g., grasping).

- Our approach is not as efficient as we would like it to be. First and foremost, reinforcement learning is heavily resource-consuming. Training a single policy can take up to one day. One of the reasons for this inefficiency is that every time we want to learn a new policy, we start training from scratch. This is inefficient, as the agent is learning everything from scratch every single time. Secondly, another inefficiency is that we are learning every single policy separately. Ideally, we would want to train a single network to learn multiple different policies as it is more efficient.

- We limited our experiments to 6 fingerspelled letters and 5 signs, due to physical limitations of the models and resource constraints. Obviously, this is a starting point for sign language acquisition rather than the final destination.

- Finally, we must point out how our approach to imitation is just as good as the data extracted from the pose estimation module. Without additional knowledge, the imitation algorithm will just try to replicate as closely as possible what the pose estimation module provided. Hence, if the data contain noise or errors, this will be transferred to the agent during the learning phase.

7.4 Future work

The work described in this thesis opens up several opportunities for future investigations. Some of these opportunities aim at addressing the limitations we describe in the

previous chapter, while others build on top of our work.

7.4.1 Sign language processing

One of the first steps we envision is to replace ASL-Lex (Caselli et al., 2017) with ASL-Lex 2.0 (Sehyr et al., 2021). The new version of ASL-Lex introduces more than 2700 lemmas (as opposed to less than 1000 in the previous version), which corresponds to a threefold improvement. In addition, they provide new and more detailed phonological classes. Once a new model is trained on these phonological classes, it could be used to expand ASL-Lex by automatically tagging new lemmas which are currently not included in the dataset. Secondly, as phonological properties can be used to identify signs, we would like to use these trained algorithms to perform sign language recognition. This approach would provide a more interpretable SLR model (as opposed to an end-to-end approach), given that an erroneously classified phonological property can provide a specific indication about why a sign is misclassified. Finally, as new pose estimation models are created almost every month, we think it would be very interesting to replace our pose estimation module with a more effective one, possibly pre-trained on a sign language dataset.

An algorithm that can recognise properties and signs has many potential applications, such as

- Translation: the algorithm can be used to translate sign language into written or spoken language, enabling communication between individuals who are deaf or hard of hearing and those who do not understand sign language. This can facilitate better accessibility and inclusion in various settings such as education, healthcare, customer service, and public interactions

- Assistive Technology: The algorithm can be integrated into devices such as smartphones, tablets, or wearables to assist individuals with hearing impairments in their daily lives. It can interpret sign language gestures and provide corresponding actions or responses. For example, it can help in controlling smart home devices, accessing digital content, or making phone calls

- Education and Training: Sign language recognition algorithms can be utilised in educational settings to enhance the learning experience for individuals learning sign language. It can provide real-time feedback on the accuracy of signing, assist in language instruction, and offer interactive learning applications or games

- Accessibility in Public Spaces: By integrating sign language recognition algorithms into public spaces, such as airports, train stations, or government offices, individuals with hearing impairments can have improved accessibility. The algorithm can be used to provide visual displays or notifications in sign language, ensuring important information is effectively communicated.

7.4.2 Anatomical modelling

As previously mentioned, one of the main limitations of our approach is that we limited the degrees of freedom of the hand. Thus, the first step in improving our approach is to increase the degrees of freedom for each joint from 1 to 2. As trivial as it might sound, this change creates another issue that needs to be addressed, which is collision avoidance between fingers. Similarly, we believe that the forearm needs to be redesigned. Currently, the forearm is a single cylinder connected to one joint on one end (the elbow) and one joint on the other hand (the wrist). However, it is well known that the forearm is composed of two bones (radius and ulna), which enable it to rotate so that the hand can rotate from the palm facing the ground to the palm facing the sky. This is another improvement which is necessary for sign language acquisition, as currently, it is not possible to rotate the hand as we just described.

A physically accurate model of the human body can not only benefit computer graphics applications, like videogames, but also research on simulated human-robot interaction, by providing an accurate representation of how a human would move, and interact with objects and robots.

7.4.3 Sign language acquisition

Humans do not learn to move from scratch every single time they need to acquire a new skill. Consequently, our possible future direction is finding a way of reusing previous experience to speed up the learning process for new signs. Some interesting results in this direction are offered by motion priors (Peng et al., 2021), which can be used to learn new skills based on previous acquisitions, and thus learn new signs based on previous ones. An additional novel way of improve the scalability of sign language acquisition would be learning to acquire phonological properties instead of signs, and then learning to combine them to construct signs. Moreover, to increase furthermore the efficiency, instead of training one network for each policy, one could try to train the same network over multiple clips. Another interesting idea would be to

include human feedback in the learning process (i.e., human-in-the-loop) to improve the qualitative results. For example, if the hand for a specific sign is expected in front of the chest, but during training the results show that it is in front of the shoulder, finding a mechanism for a person to correct the algorithm by saying *"move closer to the chest"* could potentially lead to more realistic results. Finally, the biggest step to deploy our work would be transferring the policies from a simulated environment to a real one, including a real robot able to perform signs and a human which would have to recognise the signs performed by the robot. One possibility is to exploit sim-to-real (Tan et al., 2018; Yu et al., 2019; Siekmann et al., 2021; Peng et al., 2018b), the process of transferring a learned model or policy from a simulated environment to the real world. Sim-to-real techniques aim to bridge this gap by making the learned models or policies more robust and adaptable to real-world conditions. This involves various strategies such as domain adaptation, transfer learning, and system identification.

To conclude, a robot that can speak sign language would be extremely beneficial in many different scenarios, such as

- Mediation: The robot can act as a communication bridge between individuals who use sign language and those who do not understand sign language. It can interpret spoken language into sign language gestures and vice versa, facilitating communication and interaction in various settings such as healthcare, education, customer service, and public spaces.

- Education and Training: The robot can be utilised as a teaching tool in sign language education. It can provide interactive lessons, practice sessions, and real-time feedback to individuals learning sign language, enhancing their learning experience and proficiency.

- Assistive Technology: The robot can serve as an assistive device for individuals who are deaf or hard of hearing. It can understand sign language gestures and provide spoken language output, enabling access to information, services, and communication in environments where sign language is not commonly understood.

- Social Companion: The robot can function as a social companion for individuals who use sign language. It can engage in conversations, provide companionship, and assist with daily activities using both sign language and spoken language, helping to reduce social isolation and enhance overall well-being.

- Accessibility in Public Spaces: Robots capable of sign language can be deployed in public spaces such as airports, train stations, or government offices to provide assistance and communication support to individuals who are deaf or hard of hearing. They can provide information, directions, or perform tasks in sign language, ensuring inclusivity and accessibility.

- Human-Robot Interaction Research: Robots that can speak sign language can be used in research studies to explore human-robot interaction and develop more effective communication methods between humans and robots. This research can lead to improvements in robotic systems, language understanding, and gesture recognition technologies.

Bibliography

Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. BSL-1K: Scaling Up Co-articulated Sign Language Recognition Using Mouthing Cues. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 35–53, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58621-8. doi: 10.1007/978-3-030-58621-8_3.

Maryam Asadi-Aghbolaghi, Albert Clapés, Marco Bellantonio, Hugo Jair Escalante, Víctor Ponce-López, Xavier Baró, Isabelle Guyon, Shohreh Kasaei, and Sergio Escalera. A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 476–483, May 2017. doi: 10.1109/FG.2017.150.

Atlas robot. Atlas boston dynamics. https://www.bostondynamics.com/atlas, 2013. Accessed: 2022-11-04.

Robbin Battison. *Lexical Borrowing in American Sign Language*. Linstok Press, 1978. ISBN 978-0-932130-02-0.

Richard Bellman. A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957. ISSN 0095-9057.

Ursula Bellugi and Susan Fischer. A comparison of sign language and spoken language. *Cognition*, 1(2):173–200, January 1972. ISSN 0010-0277. doi: 10.1016/0010-0277(72)90018-2.

Darrin C. Bentivegna, Christopher G. Atkeson, and Gordon Cheng. Learning tasks from observation and practice. *Robotics and Autonomous Systems*, 47(2):163–169, June 2004. ISSN 0921-8890. doi: 10.1016/j.robot.2004.03.010.

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. DReCon: Data-driven responsive control of physics-based characters. *ACM Transactions on Graphics*, 38(6):206:1–206:11, November 2019. ISSN 0730-0301. doi: 10.1145/3355089.3356536.

Lukas Biewald. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. Robot Programming by Demonstration. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 1371–1394. Springer, Berlin, Heidelberg, 2008. ISBN 978-3-540-30301-5. doi: 10.1007/978-3-540-30301-5_60.

Andrea Bonarini. Communication in Human-Robot Interaction. *Current Robotics Reports*, 1(4):279–285, December 2020. ISSN 2662-4087. doi: 10.1007/s43154-020-00026-1.

Mark Borg and Kenneth P. Camilleri. Phonologically-Meaningful Subunits for Deep Learning-Based Sign Language Recognition. In Adrien Bartoli and Andrea Fusiello, editors, *Computer Vision – ECCV 2020 Workshops*, Lecture Notes in Computer Science, pages 199–217, Cham, 2020. Springer International Publishing. ISBN 978-3-030-66096-3. doi: 10.1007/978-3-030-66096-3_15.

C. Breazeal, C.D. Kidd, A.L. Thomaz, G. Hoffman, and M. Berlin. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 708–713, August 2005. doi: 10.1109/IROS.2005.1545011.

Diane Brentari. *A Prosodic Model of Sign Language Phonology*. MIT press, February 1999. ISBN 978-0-262-02445-7.

Diane Brentari, Jordan Fenlon, and Kearsy Cormier. Sign Language Phonology. https://oxfordre.com/linguistics/view/10.1093/acrefore/9780199384655.001.0001/acrefore-9780199384655-e-117, July 2018.

Richard W. Byrne and Anne E. Russon. Learning by imitation: A hierarchical approach. *Behavioral and Brain Sciences*, 21(5):667–684, October 1998. ISSN 1469-1825, 0140-525X. doi: 10.1017/S0140525X98001745.

Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural Sign Language Translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7784–7793, 2018.

Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign Language Transformers: Joint End-to-end Sign Language Recognition and Translation, March 2020.

Joao Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.502.

Naomi K. Caselli, Zed Sevcikova Sehyr, Ariel M. Cohen-Goldberg, and Karen Emmorey. ASL-LEX: A lexical database of American Sign Language. *Behavior Research Methods*, 49(2):784–801, April 2017. ISSN 1554-3528. doi: 10.3758/s13428-016-0742-0.

Danqi Chen, Jason Bolton, and Christopher D. Manning. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1223.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179.

Ching-Hua Chuan, Eric Regina, and Caroline Guardino. American Sign Language Recognition Using Leap Motion Sensor. In *2014 13th International Conference on Machine Learning and Applications*, pages 541–544, December 2014. doi: 10.1109/ICMLA.2014.110.

Helen Cooper, Brian Holt, and Richard Bowden. Sign Language Recognition. In

Thomas B. Moeslund, Adrian Hilton, Volker Krüger, and Leonid Sigal, editors, *Visual Analysis of Humans: Looking at People*, pages 539–562. Springer, London, 2011. ISBN 978-0-85729-997-0. doi: 10.1007/978-0-85729-997-0_27.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995. ISSN 1573-0565. doi: 10.1007/BF00994018.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning, 2016. URL http://pybullet.org.

Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3(4):362–369, December 2019. ISSN 2366-598X. doi: 10.1007/s41315-019-00103-5.

Pedro M. Ferreira, Jaime S. Cardoso, and Ana Rebelo. On the role of multimodal learning in the recognition of sign language. *Multimedia Tools and Applications*, 78 (8):10035–10056, April 2019. ISSN 1573-7721. doi: 10.1007/s11042-018-6565-5.

Fingerspelling with Machine Learning. Fingerspelling.xyz. https://fingerspelling.xyz.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-Shot Visual Imitation Learning via Meta-Learning. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 357–368. PMLR, October 2017.

David F. Fouhey, Wei-cheng Kuo, Alexei A. Efros, and Jitendra Malik. From Lifestyle Vlogs to Everyday Interactions. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4991–5000, June 2018. doi: 10.1109/CVPR.2018.00524.

David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

Jennifer J. Gago, Valentina Vasco, Bartek Łukawski, Ugo Pattacini, Vadim Tikhanoff, Juan G. Victores, and Carlos Balaguer. Sequence-to-Sequence Natural Language to Humanoid Robot Sign Language. In *EUROSIM 2019 Abstract Volume*, 2019a. doi: 10.11128/arep.58.

Jennifer J. Gago, Juan G. Victores, and Carlos Balaguer. Sign Language Representation by TEO Humanoid Robot: End-User Interest, Comprehension and Satisfaction. *Electronics*, 8(1):57, January 2019b. ISSN 2079-9292. doi: 10.3390/electronics8010057.

K. Grobel and M. Assan. Isolated sign language recognition using hidden Markov models. In *Computational Cybernetics and Simulation 1997 IEEE International Conference on Systems, Man, and Cybernetics*, volume 1, pages 162–167 vol.1, October 1997. doi: 10.1109/ICSMC.1997.625742.

Leonard Hasenclever, Fabio Pardo, Raia Hadsell, Nicolas Heess, and Josh Merel. CoMic: Complementary Task Learning & Mimicry for Reusable Skills. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4105–4115. PMLR, November 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

Jonathan Ho and Stefano Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 36(4):42:1–42:13, July 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073663.

Seok-ju Hong, Nurul Arif Setiawan, and Chil-woo Lee. Real-Time Vision Based Gesture Recognition for Human-Robot Interaction. In Bruno Apolloni, Robert J. Howlett, and Lakhmi Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems*, Lecture Notes in Computer Science, pages 493–500, Berlin, Heidelberg, 2007. Springer. ISBN 978-3-540-74819-9. doi: 10.1007/978-3-540-74819-9_61.

Seyed Ramezan Hosseini, Alireza Taheri, Ali Meghdari, and Minoo Alemi. Teaching Persian Sign Language to a Social Robot via the Learning from Demonstrations Approach. In Miguel A. Salichs, Shuzhi Sam Ge, Emilia Ivanova Barakova, John-John Cabibihan, Alan R. Wagner, Álvaro Castro-González, and Hongsheng He, editors, *Social Robotics*, Lecture Notes in Computer Science, pages 655–665, Cham, 2019. Springer International Publishing. ISBN 978-3-030-35888-4. doi: 10.1007/978-3-030-35888-4_61.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation Learning: A Survey of Learning Methods. *ACM Computing Surveys*, 50(2):21:1–21:35, April 2017. ISSN 0360-0300. doi: 10.1145/3054912.

Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, July 2014. ISSN 1939-3539. doi: 10.1109/TPAMI.2013.248.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. Sign Language Recognition via Skeleton-Aware Multi-Model Ensemble, October 2021a.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. Skeleton Aware Multi-modal Sign Language Recognition. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3408–3418, Nashville, TN, USA, June 2021b. IEEE. ISBN 978-1-66544-899-4. doi: 10.1109/CVPRW53098.2021.00380.

Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-End Recovery of Human Shape and Pose. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7122–7131, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00744.

Angjoo Kanazawa, Jason Y. Zhang, Panna Felsen, and Jitendra Malik. Learning 3D Human Dynamics from Video, September 2019.

Byeongkeun Kang, Subarna Tripathi, and Truong Q. Nguyen. Real-time sign language fingerspelling recognition using convolutional neural networks from depth map. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 136–140, November 2015. doi: 10.1109/ACPR.2015.7486481.

Taehwan Kim, Greg Shakhnarovich, and Karen Livescu. Fingerspelling Recognition with Semi-Markov Conditional Random Fields. In *2013 IEEE International Conference on Computer Vision*, pages 1521–1528, December 2013. doi: 10.1109/ICCV.2013.192.

Taehwan Kim, Jonathan Keane, Weiran Wang, Hao Tang, Jason Riggle, Gregory Shakhnarovich, Diane Brentari, and Karen Livescu. Lexicon-Free Fingerspelling Recognition from Video: Data, Models, and Signer Adaptation, September 2016.

Vadim Kimmelman, Alfarabi Imashev, Medet Mukushev, and Anara Sandygulova. Eyebrow position in grammatical and emotional expressions in Kazakh-Russian Sign Language: A quantitative study. *PLOS ONE*, 15(6):e0233731, June 2020. ISSN 1932-6203. doi: 10.1371/journal.pone.0233731.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017.

Oscar Koller. Quantitative Survey of the State of the Art in Sign Language Recognition, August 2020.

Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, 141:108–125, December 2015.

Nikos Kolotouros, Georgios Pavlakos, Michael Black, and Kostas Daniilidis. Learning to Reconstruct 3D Human Pose and Shape via Model-Fitting in the Loop. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2252–2261, Seoul, Korea (South), October 2019. IEEE. ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00234.

Hatice Kose, Rabia Yorganci, and Itauma I. Itauma. Humanoid robot assisted interactive sign language tutoring game. In *2011 IEEE International Conference on Robotics and Biomimetics*, pages 2247–2248, December 2011. doi: 10.1109/ROBIO.2011.6181630.

Hatice Kose, Rabia Yorganci, Esra H. Algan, and Dag S. Syrdal. Evaluation of the Robot Assisted Sign Language Tutoring Using Video-Based Studies. *International Journal of Social Robotics*, 4(3):273–283, August 2012. ISSN 1875-4805. doi: 10.1007/s12369-012-0142-2.

Hatice Köse, Pınar Uluer, Neziha Akalın, Rabia Yorgancı, Ahmet Özkul, and Gökhan Ince. The Effect of Embodiment in Sign Language Tutoring with Assistive Humanoid Robots. *International Journal of Social Robotics*, 7(4):537–548, August 2015. ISSN 1875-4805. doi: 10.1007/s12369-015-0311-1.

Anna Kuznetsova. Using Computer Vision to Analyze Non-manual Marking of Questions in KRSL. page 11, 2021.

Kam Lai, Janusz Konrad, and Prakash Ishwar. A gesture-driven computer interface using Kinect. In *2012 IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 185–188, April 2012. doi: 10.1109/SSIAI.2012.6202484.

Kyungho Lee, Seyoung Lee, and Jehee Lee. Interactive character animation by learning multi-objective control. *ACM Transactions on Graphics*, 37(6):180:1–180:10, December 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275071.

Seunghwan Lee, Moonseok Park, Kyoungmin Lee, and Jehee Lee. Scalable muscle-actuated human simulation and control. *ACM Transactions on Graphics*, 38(4): 73:1–73:13, July 2019. ISSN 0730-0301. doi: 10.1145/3306346.3322972.

Matthew K. Leonard, Ben Lucas, Shane Blau, David P. Corina, and Edward F. Chang. Cortical encoding of manual articulatory and linguistic features in American Sign Language. *Current biology : CB*, 30(22):4342–4351.e3, November 2020. ISSN 0960-9822. doi: 10.1016/j.cub.2020.08.048.

Dongxu Li, Cristian Rodriguez Opazo, Xin Yu, and Hongdong Li. Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison, January 2020a.

Dongxu Li, Xin Yu, Chenchen Xu, Lars Petersson, and Hongdong Li. Transferring Cross-Domain Knowledge for Video Sign Language Recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6204–6213, June 2020b. doi: 10.1109/CVPR42600.2020.00624.

Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics*, 36(6):1–17, November 2017. ISSN 0730-0301, 1557-7368. doi: 10.1145/3130800.3130813.

Yuwei Liang, Weijie Li, Yue Wang, Rong Xiong, Yichao Mao, and Jiafan Zhang. Dynamic Movement Primitive based Motion Retargeting for Dual-Arm Sign Language Motions. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8195–8201, May 2021. doi: 10.1109/ICRA48506.2021.9561120.

Scott K. Liddell and Robert E. Johnson. American Sign Language: The Phonological Base. *Sign Language Studies*, (64):195–278, 1989. ISSN 0302-1475.

Kian Ming Lim, Alan Wee Chiat Tan, Chin Poo Lee, and Shing Chiang Tan. Isolated sign language recognition using Convolutional Neural Network hand modelling and Hand Energy Image. *Multimedia Tools and Applications*, 78(14):19917–19944, July 2019. ISSN 1573-7721. doi: 10.1007/s11042-019-7263-7.

Sheng-Yen Lo and Han-Pang Huang. Realization of sign language motion using a dual-arm/hand humanoid robot. *Intelligent Service Robotics*, 9(4):333–345, October 2016. ISSN 1861-2784. doi: 10.1007/s11370-016-0203-8.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6):248:1–248:16, November 2015. ISSN 0730-0301. doi: 10.1145/2816795.2818013.

Flaminia L. Luccio and Diego Gaspari. Learning Sign Language from a Sanbot Robot. In *Proceedings of the 6th EAI International Conference on Smart Objects and Technologies for Social Good*, GoodTechs '20, pages 138–143, New York, NY, USA, September 2020. Association for Computing Machinery. ISBN 978-1-4503-7559-7. doi: 10.1145/3411170.3411252.

Yongsen Ma, Gang Zhou, Shuangquan Wang, Hongyang Zhao, and Woosub Jung. SignFi: Sign Language Recognition Using WiFi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(1):23:1–23:21, March 2018. doi: 10.1145/3191755.

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2): 442–451, October 1975. ISSN 0005-2795. doi: 10.1016/0005-2795(75)90109-9.

Nikolaos Mavridis. A review of verbal and non-verbal human–robot interactive communication. *Robotics and Autonomous Systems*, 63:22–35, January 2015. ISSN 0921-8890. doi: 10.1016/j.robot.2014.09.031.

Rachel Mayberry and Bonita Squires. Sign Language: Acquisition. In *Encyclopedia of Language & Linguistics*, pages 291–296. December 2006. ISBN 978-0-08-044854-1. doi: 10.1016/B0-08-044854-2/00854-3.

Rachel I. Mayberry, Matthew L. Hall, and Meghan Zvaigzne. Subjective frequency ratings for 432 ASL signs. *Behavior Research Methods*, 46(2):526–539, June 2014. ISSN 1554-3528. doi: 10.3758/s13428-013-0370-x.

Ali Meghdari, Minoo Alemi, Mohammad Zakipour, and Seyed Amir Kashanian. Design and Realization of a Sign Language Educational Humanoid Robot. *Journal of Intelligent & Robotic Systems*, 95(1):3–17, July 2019. ISSN 1573-0409. doi: 10.1007/s10846-018-0860-2.

Richard P. Meier. Why different, why the same? Explaining effects and non-effects of modality upon linguistic structure in sign and speech. In David Quinto-Pozos, Kearsy Cormier, and Richard P. Meier, editors, *Modality and Structure in Signed and Spoken Languages*, pages 1–26. Cambridge University Press, Cambridge, 2002. ISBN 978-0-521-80385-4. doi: 10.1017/CBO9780511486777.001.

Josh Merel, Arun Ahuja, Vu Pham, Saran Tunyasuvunakool, Siqi Liu, Dhruva Tirumala, Nicolas Heess, and Greg Wayne. Hierarchical visuomotor control of humanoids, January 2019a.

Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control, January 2019b.

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & Carry: Reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics*, 39(4):39:39:1–39:39:12, August 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392474.

Dimitris Metaxas, Mark Dilsizian, and Carol Neidle. Scalable ASL sign recognition using model-based machine learning and linguistically annotated corpora. May 2018. ISSN 0010-4817.

Mohamed Mohandes, Mohamed Abdelouaheb DERICHE, and Salihu Oladimeji ALIYU. Arabic sign language recognition using multi-sensor data fusion, June 2017.

Jill P. Morford and James Macfarlane. Frequency Characteristics of American Sign Language. *Sign Language Studies*, 3(2):213–225, 2003. ISSN 0302-1475.

Amit Moryossef, Ioannis Tsochantaridis, Roee Aharoni, Sarah Ebling, and Srini Narayanan. Real-Time Sign Language Detection Using Human Pose Estimation. In Adrien Bartoli and Andrea Fusiello, editors, *Computer Vision – ECCV 2020 Workshops*, Lecture Notes in Computer Science, pages 237–248, Cham, 2020. Springer International Publishing. ISBN 978-3-030-66096-3. doi: 10.1007/978-3-030-66096-3_17.

Robert Östling, Carl Börstell, and Servane Courtaux. Visual Iconicity Across Sign Languages: Large-Scale Automated Video Analysis of Iconic Articulators and Locations. *Frontiers in Psychology*, 9, 2018. ISSN 1664-1078.

Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive Body Capture: 3D Hands, Face, and Body From a Single Image. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10967–10977, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.01123.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, and David Cournapeau. Scikit-learn: Machine Learning in Python. *MACHINE LEARNING IN PYTHON*, page 6, 2011.

Luigi Penco, Nicola Scianca, Valerio Modugno, Leonardo Lanari, Giuseppe Oriolo, and Serena Ivaldi. A Multimode Teleoperation Framework for Humanoid Loco-Manipulation: An Application for the iCub Robot. *IEEE Robotics & Automation Magazine*, 26(4):73–82, December 2019. ISSN 1558-223X. doi: 10.1109/MRA.2019.2941245.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics*, 37(4):143:1–143:14, July 2018a. ISSN 0730-0301. doi: 10.1145/3197517.3201311.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, 2018b. doi: 10.1109/ICRA.2018.8460528.

Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. SFV: Reinforcement learning of physical skills from videos. *ACM Transactions on Graphics*, 37(6):178:1–178:14, December 2018c. ISSN 0730-0301. doi: 10.1145/3272127.3275014.

Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning Agile Robotic Locomotion Skills by Imitating Animals, July 2020.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. AMP: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics*, 40(4):144:1–144:20, July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459670.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. ASE: Large-Scale Reusable Adversarial Skill Embeddings for Physically Simulated Characters. *ACM Transactions on Graphics*, 41(4):1–17, July 2022. ISSN 0730-0301, 1557-7368. doi: 10.1145/3528223.3530110.

Jan Peters, Daniel D. Lee, Jens Kober, Duy Nguyen-Tuong, J. Andrew Bagnell, and Stefan Schaal. Robot Learning. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, Springer Handbooks, pages 357–398. Springer International Publishing, Cham, 2016. ISBN 978-3-319-32552-1. doi: 10.1007/978-3-319-32552-1_15.

Laura Ann Petitto and Paula F. Marentette. Babbling in the Manual Mode: Evidence for the Ontogeny of Language. *Science*, 251(5000):1493–1496, March 1991. doi: 10.1126/science.2006424.

Davide Polonio, Federico Tavella, Marco Zanella, and Armir Bujari. GHio-Ca: An Android Application for Automatic Image Classification. In Barbara Guidi, Laura Ricci, Carlos Calafate, Ombretta Gaggi, and Johann Marquez-Barja, editors, *Smart Objects and Technologies for Social Good*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pages

248–257, Cham, 2018. Springer International Publishing. ISBN 978-3-319-76111-4. doi: 10.1007/978-3-319-76111-4_25.

Junfu Pu, Wengang Zhou, Hezhen Hu, and Houqiang Li. Boosting Continuous Sign Language Recognition via Cross Modality Augmentation. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, pages 1497–1505, New York, NY, USA, October 2020. Association for Computing Machinery. ISBN 978-1-4503-7988-5. doi: 10.1145/3394171.3413931.

Nicolas Pugeault and Richard Bowden. Spelling it out: Real-time ASL fingerspelling recognition. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1114–1119, November 2011. doi: 10.1109/ICCVW.2011.6130290.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. ISSN 1533-7928.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations, June 2018.

Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. Sign Language Recognition: A Deep Survey. *Expert Systems with Applications*, 164:113794, February 2021. ISSN 0957-4174. doi: 10.1016/j.eswa.2020.113794.

Nathan Ratliff, J. Andrew Bagnell, and Siddhartha S. Srinivasa. Imitation learning for locomotion and manipulation. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 392–397, November 2007. doi: 10.1109/ICHR.2007.4813899.

Yu Rong, Takaaki Shiratori, and Hanbyul Joo. FrankMocap: A Monocular 3D Whole-Body Pose Estimation System via Regression and Integration, August 2021.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

Wendy Sandler. The Phonological Organization of Sign Languages. *Language and linguistics compass*, 6(3):162–182, March 2012. ISSN 1749-818X. doi: 10.1002/lnc3.326.

Wendy Sandler. The Challenge of Sign Language Phonology. *Annual Review of Linguistics*, 3(1):43–63, 2017. doi: 10.1146/annurev-linguistics-011516-034122.

Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael J. Black. Learning to Regress 3D Face Shape and Expression From an Image Without 3D Supervision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7755–7764, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00795.

Shane Saunderson and Goldie Nejat. How Robots Influence Humans: A Survey of Nonverbal Communication in Social Human–Robot Interaction. *International Journal of Social Robotics*, 11(4):575–608, August 2019. ISSN 1875-4805. doi: 10.1007/s12369-019-00523-0.

Brian Scassellati, Jake Brawer, Katherine Tsui, Setareh Nasihati Gilani, Melissa Malzkuhn, Barbara Manini, Adam Stone, Geo Kartheiser, Arcangelo Merla, Ari Shapiro, David Traum, and Laura-Ann Petitto. Teaching Language to Deaf Infants with a Robot and a Virtual Human. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–13, New York, NY, USA, April 2018. Association for Computing Machinery. ISBN 978-1-4503-5620-6. doi: 10.1145/3173574.3174127.

Stefan Schaal. Learning from Demonstration. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996.

Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, June 1999. ISSN 1364-6613. doi: 10.1016/S1364-6613(99)01327-3.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017a.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017b.

Zed Sevcikova Sehyr, Naomi Caselli, Ariel M Cohen-Goldberg, and Karen Emmorey. The ASL-LEX 2.0 Project: A Database of Lexical and Phonological Properties for 2,723 Signs in American Sign Language. *The Journal of Deaf Studies and Deaf Education*, 26(2):263–277, April 2021. ISSN 1081-4159. doi: 10.1093/deafed/enaa038.

Bowen Shi, Aurora Martinez Del Rio, Jonathan Keane, Jonathan Michaux, Diane Brentari, Greg Shakhnarovich, and Karen Livescu. American Sign Language fingerspelling recognition in the wild, February 2019.

Jonah Siekmann, Yesh Godse, Alan Fern, and Jonathan Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7309–7315, 2021. doi: 10.1109/ICRA48506.2021.9561814.

T. Starner and A. Pentland. Real-time American Sign Language recognition from video using hidden Markov models. In *Proceedings of International Symposium on Computer Vision - ISCV*, pages 265–270, November 1995. doi: 10.1109/ISCV.1995.477012.

T. Starner, J. Weaver, and A. Pentland. Real-time American sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375, December 1998. ISSN 1939-3539. doi: 10.1109/34.735811.

William C. Stokoe, Jr. Sign Language Structure: An Outline of the Visual Communication Systems of the American Deaf. *The Journal of Deaf Studies and Deaf Education*, 10(1):3–37, January 2005. ISSN 1081-4159. doi: 10.1093/deafed/eni001.

Stephanie Stoll, Necati Cihan Camgoz, Simon Hadfield, and Richard Bowden. Text2Sign: Towards Sign Language Production Using Neural Machine Translation and Generative Adversarial Networks. *International Journal of Computer Vision*, 128(4):891–908, April 2020. ISSN 1573-1405. doi: 10.1007/s11263-019-01281-2.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning, Second Edition: An Introduction*. MIT Press, November 2018. ISBN 978-0-262-35270-3.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *ArXiv*, abs/1804.10332, 2018.

Federico Tavella, Aphrodite Galata, and Angelo Cangelosi. Phonology Recognition in American Sign Language. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8452–8456, May 2022a. doi: 10.1109/ICASSP43922.2022.9747212.

Federico Tavella, Viktor Schlegel, Marta Romeo, Aphrodite Galata, and Angelo Cangelosi. WLASL-LEX: A Dataset for Recognising Phonological Properties in American Sign Language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 453–463, Dublin, Ireland, May 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.49.

Federico Tavella, Aphrodite Galata, and Angelo Cangelosi. Signs of language: Embodied sign language fingerspelling acquisition from demonstrations for human-robot interaction, 2023.

Jitendra V. Tembhurne and Tausif Diwan. Sentiment analysis in textual, visual and multimodal inputs using recurrent neural networks. *Multimedia Tools and Applications*, 80(5):6871–6910, February 2021. ISSN 1573-7721. doi: 10.1007/s11042-020-10037-x.

Michael Tomasello, Ann Cale Kruger, and Hilary Horn Ratner. Cultural learning. *Behavioral and Brain Sciences*, 16(3):495–511, September 1993. ISSN 1469-1825, 0140-525X. doi: 10.1017/S0140525X0003123X.

Sandrine Tornay, Marzieh Razavi, Necati Cihan Camgoz, Richard Bowden, and Mathew Magimai.-Doss. HMM-based Approaches to Model Multichannel Information in Sign Language Inspired from Articulatory Features-based Speech Processing. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2817–2821, May 2019. doi: 10.1109/ICASSP.2019.8683167.

Sandrine Tornay, Marzieh Razavi, and Mathew Magimai.-Doss. Towards Multilingual Sign Language Recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6309–6313, May 2020. doi: 10.1109/ICASSP40776.2020.9054631.

Nam Nguyen Tu, Shinji Sako, and Bogdan Kwolek. Fingerspelling recognition using synthetic images and deep transfer learning. In *Thirteenth International Conference*

on Machine Vision, volume 11605, pages 528–535. SPIE, January 2021. doi: 10.1117/12.2587592.

John C. Tuthill and Eiman Azim. Proprioception. *Current Biology*, 28(5):R194–R203, March 2018. ISSN 0960-9822. doi: 10.1016/j.cub.2018.01.064.

Pınar Uluer, Neziha Akalın, and Hatice Köse. A New Robotic Platform for Sign Language Tutoring. *International Journal of Social Robotics*, 7(5):571–585, November 2015. ISSN 1875-4805. doi: 10.1007/s12369-015-0307-x.

UN. International day of sign language - un. https://www.un.org/en/observances/sign-languages-day, 2022. Accessed: 2022-08-23.

Clayton Valli and Ceil Lucas. *Linguistics of American Sign Language: An Introduction*. Gallaudet University Press, 2000. ISBN 978-1-56368-097-7.

C. Vogler and D. Metaxas. Parallel hidden Markov models for American sign language recognition. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 116–122 vol.1, September 1999. doi: 10.1109/ICCV.1999.791206.

Ulrich von Agris, Jörg Zieren, Ulrich Canzler, Britta Bauer, and Karl-Friedrich Kraiss. Recent developments in visual sign language recognition. *Universal Access in the Information Society*, 6(4):323–362, February 2008. ISSN 1615-5297. doi: 10.1007/s10209-007-0104-x.

Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep High-Resolution Representation Learning for Visual Recognition, March 2020.

WHO. Deafness and hearing loss - who. https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss, 2021. Accessed: 2022-08-23.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Transactions on Graphics*, 39(4):33:33:1–33:33:12, August 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392381.

Jungdam Won, Deepak Gopinath, and Jessica Hodgins. Control strategies for physically simulated characters performing two-player competitive sports. *ACM Transactions on Graphics*, 40(4):146:1–146:11, July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459761.

Yezhou Yang, Yi Li, Cornelia Fermuller, and Yiannis Aloimonos. Robot Learning Manipulation Action Plans by "Watching" Unconstrained Videos from the World Wide Web. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), March 2015. ISSN 2374-3468. doi: 10.1609/aaai.v29i1.9671.

Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. Including Signed Languages in Natural Language Processing, July 2021.

Wenhao Yu, Visak CV Kumar, Greg Turk, and C. Karen Liu. Sim-to-real transfer for biped locomotion. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3503–3510, 2019. doi: 10.1109/IROS40897.2019.8968053.

Zahoor Zafrulla, Helene Brashear, Thad Starner, Harley Hamilton, and Peter Presti. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 279–286, New York, NY, USA, November 2011. Association for Computing Machinery. ISBN 978-1-4503-0641-6. doi: 10.1145/2070481.2070532.

Mohammad Zakipour, Ali Meghdari, and Minoo Alemi. RASA: A Low-Cost Upper-Torso Social Robot Acting as a Sign Language Teaching Assistant. In Arvin Agah, John-John Cabibihan, Ayanna M. Howard, Miguel A. Salichs, and Hongsheng He, editors, *Social Robotics*, Lecture Notes in Computer Science, pages 630–639, Cham, 2016. Springer International Publishing. ISBN 978-3-319-47437-3. doi: 10.1007/978-3-319-47437-3_62.

Haodong Zhang, Weijie Li, Jiangpin Liu, Zexi Chen, Yuxiang Cui, Yue Wang, and Rong Xiong. Kinematic Motion Retargeting via Neural Latent Optimization for Learning Sign Language. *IEEE Robotics and Automation Letters*, 7(2):4582–4589, April 2022. ISSN 2377-3766. doi: 10.1109/LRA.2022.3151433.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation. In *2018 IEEE International Conference on Robotics*

and Automation (ICRA), pages 5628–5635, May 2018. doi: 10.1109/ICRA.2018.8461249.

Weiyu Zhang, Menglong Zhu, and Konstantinos G. Derpanis. From Actemes to Action: A Strongly-Supervised Representation for Detailed Action Understanding. In *2013 IEEE International Conference on Computer Vision*, pages 2248–2255, December 2013. doi: 10.1109/ICCV.2013.280.

Xu Zhang, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(6):1064–1076, November 2011. ISSN 1558-2426. doi: 10.1109/TSMCA.2011.2116004.

Lihong Zheng, Bin Liang, and Ailian Jiang. Recent Advances of Deep Learning for Sign Language Recognition. In *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–7, November 2017. doi: 10.1109/DICTA.2017.8227483.

Da Zhi, Thiago E. Alves de Oliveira, Vinicius Prado da Fonseca, and Emil M. Petriu. Teaching a Robot Sign Language using Vision-Based Hand Gesture Recognition. In *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 1–6, June 2018. doi: 10.1109/CIVEMSA.2018.8439952.

Jörg Zieren and Karl-Friedrich Kraiss. Non-intrusive sign language recognition for human-computer interaction. *9th IFAC/IFIP/IFORS/IEA Symposium Analysis, Design, and Evaluation of Human-Machine Systems*, January 2004.

Appendix A

Implementation details

A.1 Hardware

For developing, we used a workstation with 32GB of RAM, a CPU Intel(R) Core(R) i7-9700K CPU @ 3.60GHz, and a GPU NVIDIA GeForce RTX 2080 Ti with 12GB of VRAM. For training the models, we used a server with 256GB of RAM, a CPU Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz, and 8 GPUs NVIDIA GeForce RTX 2080 Ti with 12GB of VRAM each.

A.2 Classification

All the material necessary to replicate the results of Chapter 3 and Chapter 4 can be found on GitHub at `github.com/tfederico/ASLPhonologicalClassification`. Here, we describe the details about the data we provide and the software we used to create our algorithms.

A.2.1 Data

For each of the dataset splits (based on gloss or phoneme), we provide the following files:

- `file2gloss.json`, a dictionary containing the map between the video files and the corresponding gloss
- `file_list.txt`, a list of all the identifiers of the videos

- `gloss2file.json`, the reverse map of `file2gloss.json`
- `label2id.json`, a map which converts the string indicating a phonological property to an integer
- `split.json`, a file in which all the identifiers in `file_list.txt` are divided between training, validation and test set
- `S_data_joint_P.npy`, a matrix of shape $(size, dim, duration, keypoints, 1)$, where `S` can be either, `train`, `val`, `train+val` and `test`, and `P` can either be `frank` or `hrt` depending on the pose estimation algorithm used to extract the keypoints, and where
  - *size* is the number of samples in the split S
  - *dim* is the number of dimensions, (x, y, z) for FrankMocap and (x, y, c) for HRNet
  - *duration* is the number of frames
  - *keypoints* is the number of different keypoints

A.2.2 Software and Libraries

All the software is written for Python 3.7 and uses the following libraries, which can be installed using either `pip` or `conda`:

- numpy 1.21.6
- pandas 1.3.5
- Pillow 8.4.0
- pyyaml 6.0
- scikit-learn 1.0.2
- scipy 1.7.3
- seaborn 0.12.1
- tensorboard 2.7.0
- torch 1.10.0

- torchvision 0.11.1
- tqdm 4.64.1
- wandb 0.13.5

Moreover, we readapted the following GitHub repositories according to the needs of our experiments:

- HMMR from `github.com/akanazawa/human_dynamics`
- HRNet from `github.com/HRNet/HigherHRNet-Human-Pose-Estimation`
- FrankMocap from `github.com/facebookresearch/frankmocap`
- 3D CNN from `github.com/dxli94/WLASL`
- ST-GCN from `github.com/jackyjsy/CVPR21Chal-SLR`

A.3 Imitation

All the material necessary to replicate the results of Chapter 5 and Chapter 6 can be found on GitHub at `github.com/tfederico/ASLMimic`, once all the work contained in this thesis will be published. Here, we describe the details about the software we used to create our algorithms.

A.3.1 Software and Libraries

All the software is written for Python 3.9 and Ubuntu 18.04 and uses the following libraries, which can be installed using either `pip` or `conda`:

- gym 0.21.0
- matplotlib 3.5.2
- mpi4py
- numpy 1.23.1
- opencv-python 4.6.0.66
- pandas 1.4.3

- pybullet 3.2.5
- pytorch3d 0.6.2
- scipy 1.8.1
- stable-baselines3 1.6.0
- torch 1.11.0
- torchvision 0.12.0
- tqdm 4.64.0
- wandb 0.13.1

The basic code structure for the PyBullet environment was adapted from the example provided in github.com/bulletphysics/bullet3.

Appendix B

Appendix to Chapter 3

Tables B.1 to B.6 describe in detail the meaning of values for all the phonological classes according to ASL-Lex Caselli et al. (2017).

The cardinality is calculated on WLASL-Lex, which is why some classes that are in ASL-Lex are not represented (i.e., cardinality equal to 0).

Value	Definition	Cardinality
imrp	index, middle, ring, pinky finger	4824
imr	index, middle, ring finger	95
mrp	middle, ring, pinky finger	28
im	index, middle finger	1296
ip	index, pinky finger	51
mr	middle, ring finger	0
mp	middle, pinky finger	0
rp	ring, pinky finger	0
i	index finger	2547
m	middle finger	259
r	ring finger	0
p	pinky	407
thumb	thumb	510

Table B.1: Values and relative definitions for selected fingers

Value	Definition	Cardinality
Head	Sign is produced on or near the head	3137
Arm	Sign is produced on or near the arm	219
Body	Sign is produced on or near the trunk	1019
Hand	Sign is produced on or near the non-dominant hand	2194
Neutral	Sign is not produced in another location on the body	3448
Other	Sign is produced in another unspecified location on the body	0

Table B.2: Values and relative definitions for major location

Value	Definition	Cardinality
1	Fully open: no joints of selected fingers are flexed	5037
2	Bent (closed): non-base joints are flexed	693
3	Flat-open: base joints flexed less than 90 degrees	909
4	Flat-closed: base joints flexed equal to or more that 90 degrees	507
5	Curved open: base and non-base joints flexed without contact	1130
6	Curved closed: base and non-base joints flexed with contact	642
7	Fully closed: base and non-base joints fully flexed	795
Stacked	Stacked: Flexion of selected fingers differs	123
Crossed	Crossed	181

Table B.3: Values and relative definitions for flexion

Value	Definition	Cardinality
HeadTop	Sign is produced on top of the head	20
Forehead	Sign is produced at the forehead	246
Eye	Sign is produced near the eye	616
CheekNose	Sign is produced on the cheek or nose	511
UpperLip	Sign is produced on the upper lip	53
Mouth	Sign is produced on the mouth	431
Chin	Sign is produced on the chin	717
UnderChin	Sign is produced under the chin	74
UpperArm	Sign is produced on the upper arm	39
ElbowFront	Sign is produced in the crook of the elbow	0
ElbowBack	Sign is produced on the outside of the elbow	13
ForearmBack	Sign is produced on the outside of the forearm	32
ForearmFront	Sign is produced on the inside of the forearm	10
ForearmUlnar	Sign is produced on the ulnar side of the forearm	56
WristBack	Sign is produced on the back of the wriset	23
WristFront	Sign is produced on the front of the wrist	0
Neck	Sign is produced on the neck	68
Shoulder	Sign is produced on the shoulder	101
Clavicle	Sign is produced on the clavicle	419
TorsoTop	Sign is produced in the upper third of the torso	0
TorsoMid	Sign is produced in the middle third of the torso	0
TorsoBottom	Sign is produced in the bottom third of the torso	19
Waist	Sign is produced at the waist	34
Hips	Sign is produced on the hips	59
Palm	Sign is produced on the plam of the non-dominant hand	925
FingerFront	Sign is produced on the front of the fingers of the non-dominant hand	99
PalmBack	Sign is produced on the back of the palm of the non-dominant hand	218
FingerBack	Sign is produced on the back of the fingers of the non-dominant hand	186
FingerRadial	Sign is produced on the radial side of the non-dominant hand	410
FingerUlnar	Sign is produced on the ulnar side of the non-dominant hand	40
FingerTip	Sign is produced on the tip of the fingers of the non-dominant hand	158
Heel	Sign is produced on the heel of the non-dominant hand	88
Other	Sign is produced in an unspecified location on the body	707
Neutral	Sign is not produced on or near the body	3390

Table B.4: Values and relative definitions for minor location

Value	Definition	Cardinality
One Handed	Sign only recruits one hand	3939
Symmetrical Or Alternating	Sign recruits both hands Phonological specifications for both hands are identical Movement of both hands is either symmetrical or alternating	3358
Asymmetrical Same Handshape	Sign recruits both hands Only the dominant hand moves The location and orientation of the hands may differ, but the other specifications of handshape are the same Non-Dominant hand must be an unmarked handshape (B A S 1 C O 5)	938
Asymmetrical Different Handshape	Sign recruits both hands Only the dominant hand moves The location and orientation of the hands may differ, and the other specifications of handshape are not the same Non-Dominant hand must be an unmarked handshape (B A S 1 C O 5)	1639
Other	Sign violates Battison's Symmetry and Dominance Conditions	143

Table B.5: Values and relative definitions for sign type

Value	Definition	Cardinality
Straight	Straight movement of the dominant hand through xyz space	1938
Curved	Single arc movement of the dominant hand through xyz space Hands may or may not make contact with multiple locations	1255
BackAndForth	Sequence of more than one straight or curved movements	3549
Circular	Circular movement of the dominant hand through space Rotation alone does not constitute a circular movement	1129
None	Entire sign (or first free morpheme) does not have a path movement	1748
Other	Sign has another unspecified path movement	398

Table B.6: Values and relative definitions for movement

Appendix C

Appendix to Chapter 4

C.1 Seed dependency

Table C.1 illustrates the performance on the test set for each model with respect to chance as measured by training 5 models from different random seeds. The performance difference is negligible suggesting that model training is largely stable with regard to chance.

Model	**Accuracy**
MLP	74.39 ± 0.35
RNN	79.12 ± 0.46
STGCN	84.12 ± 0.29
3D CNN	69.23 ± 0.93

Table C.1: Mean and standard deviation of accuracy of all architectures trained with the HRNet output, measured on the SIGNTYPE test set and averaged over 5 different random seeds.

C.2 Additional results

Table C.2 illustrates additional results for several different metrics. In particular, we report micro- and macro precision/recall and Matthews correlation coefficient. These metrics help to give a better understanding of the classification results, as they are affected more by data imbalance when compared to accuracy.

		Flexion					MajLocation					MinLocation					Movement					Fingers					Signtype				
		P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC	P_μ	P_M	R_μ	R_M	MCC
Phoneme	Baseline	50.3	5.59	50.3	11.11	0.0	34.4	6.88	34.4	20.0	0.0	33.87	1.06	33.87	3.12	0.0	35.46	5.91	35.46	16.67	0.0	48.17	5.35	48.17	11.11	0.0	39.32	7.86	39.32	20.0	0.0
	MLP_H	44.1	24.5	44.1	20.7	14.6	70.3	65.8	70.3	64.0	58.9	51.6	37.3	51.6	28.2	41.6	34.5	28.0	34.5	26.9	13.9	59.5	29.6	59.5	25.0	37.7	73.9	54.1	73.9	52.6	62.5
	MLP_F	50.3	5.6	50.3	11.1	0.9	57.8	52.3	57.8	46.8	41.2	34.3	13.9	34.3	9.1	17.9	34.3	13.1	34.3	18.7	5.7	43.4	17.5	43.4	12.9	4.6	67.1	38.1	67.1	42.8	52.7
	RNN_H	49.0	32.1	49.0	30.0	25.4	75.8	75.2	75.8	72.4	66.4	64.3	54.3	64.3	46.0	57.4	35.1	30.1	35.1	29.5	15.9	71.0	53.3	71.0	46.5	56.6	78.7	61.2	78.7	58.8	69.4
	RNN_F	50.3	5.6	50.3	11.1	0.0	63.9	56.7	63.9	52.2	50.1	30.3	4.7	30.3	4.0	4.9	35.4	21.4	35.4	18.1	5.2	46.5	9.2	46.5	12.4	8.5	70.9	60.6	70.9	46.8	58.3
	GTN_H	62.4	55.4	62.4	45.0	43.9	83.2	80.6	83.2	78.6	76.8	74.5	66.7	74.5	63.5	69.8	63.6	62.1	63.6	58.2	52.7	73.8	71.7	73.8	56.0	61.1	84.5	74.9	84.5	69.6	77.7
	GTN_F	43.4	23.6	43.4	20.8	15.3	70.5	66.4	70.5	62.1	58.9	53.0	43.9	53.0	40.0	43.8	45.7	40.8	45.7	37.8	28.6	63.1	39.0	63.1	32.8	44.3	73.0	56.8	73.0	53.1	61.1
	3DCNN	46.5	17.8	46.5	13.2	5.4	64.3	57.2	64.3	55.2	50.3	42.3	22.8	42.3	18.6	29.1	32.9	23.4	32.9	20.8	7.5	47.5	17.8	47.5	14.5	14.6	69.5	44.9	69.5	44.8	55.6
Gloss	Baseline	53.03	5.89	53.03	11.11	0.0	35.69	7.14	35.69	20.0	0.0	42.03	2.1	42.03	5.0	0.0	35.21	5.87	35.21	16.67	0.0	47.38	5.92	47.38	12.5	0.0	38.28	7.66	38.28	20.0	0.0
	MLP_H	44.6	18.6	44.6	15.5	8.3	68.1	62.0	68.1	56.6	55.5	47.3	16.8	47.3	13.1	32.5	28.4	20.4	28.4	19.8	4.9	56.2	21.4	56.2	20.3	32.0	75.3	50.6	75.3	50.7	64.3
	MLP_F	52.8	5.9	52.8	11.1	−2.1	56.7	46.2	56.7	42.9	39.5	38.3	11.7	38.3	7.9	18.1	37.1	15.9	37.1	21.7	12.5	39.3	10.4	39.3	11.1	0.4	68.4	37.7	68.4	41.2	54.3
	RNN_H	39.6	19.8	39.6	18.0	10.9	72.8	68.0	72.8	67.3	62.4	49.3	19.6	49.3	17.5	36.7	32.2	25.7	32.2	24.9	11.3	60.7	36.9	60.7	32.5	40.3	75.4	55.0	75.4	53.5	64.6
	RNN_F	53.0	5.9	53.0	11.1	0.0	64.1	57.3	64.1	52.6	50.5	44.4	15.1	44.4	12.3	27.9	36.7	11.2	36.7	20.1	10.0	27.3	10.6	27.3	12.7	3.0	72.0	41.3	72.0	46.9	60.4
	GTN_H	49.1	25.6	49.1	21.6	18.9	77.3	72.1	77.3	70.0	68.6	55.1	25.1	55.1	23.3	43.4	52.5	49.4	52.5	46.5	38.0	65.7	37.2	65.7	30.6	47.8	76.6	54.9	76.6	54.4	66.2
	GTN_F	39.0	15.1	39.0	14.4	4.7	66.7	63.2	66.7	60.1	53.9	45.1	15.7	45.1	13.2	31.1	43.1	36.0	43.1	34.9	25.8	60.0	32.5	60.0	29.2	39.4	71.3	47.6	71.3	47.5	58.5
	3DCNN	46.0	12.0	46.0	12.8	4.5	65.0	57.5	65.0	52.0	51.8	10.8	12.0	10.8	9.7	9.5	32.0	18.7	32.0	19.3	6.0	45.9	15.1	45.9	14.7	10.7	71.6	46.3	71.6	46.3	58.7

Table C.2: Micro-averaged (μ), macro-averaged (M) precision (P) and recall (R) and Matthews correlation coefficient (MCC) of various models on the test sets of the six tasks. We omit error margins as the low number of classes results in a small sample size.

Appendix D

Appendix to Chapter 5

D.1 Reinforcement learning tuning

Table D.1 provides an alternative version of the results illustrated in Figure 5.7. The table is sorted in ascending order based on the reward. Complementarily, Figure 5.8 provides the correlation to the reward (i.e., last column and/or row) for each parameter.

batch_size	gamma	learning_rate	log_std_init	n_epochs	n_steps	ortho_init	weight_decay	eval/mean_reward
128	0.9	0.00001	-2	10	1024	false	0.00001	1624.15
128	0.9	0.00001	-2	10	512	false	0.00001	1585.71
128	0.9	0.00001	-1	10	1024	false	0.00001	1531.93
128	0.9	0.00001	-2	10	512	true	0.00001	1523.36
128	0.9	0.00001	-1	10	512	false	0.00001	1482.61
128	0.9	0.00001	-2	5	512	true	0.00001	1471.42
128	0.9	0.00001	-2	5	512	false	0.00001	1448.38
128	0.9	0.00001	-2	10	512	true	0.00001	1443.02
128	0.9	0.00001	-2	5	512	true	0.00001	1422.17
256	0.9	0.00001	-2	5	1024	true	0.00001	1413.56
128	0.9	0.00003	-1	10	1024	false	0.0001	1389.84
128	0.95	0.00001	-1	10	512	true	0.00001	1365.09
128	0.9	0.00001	-1	5	1024	false	0.00001	1365.08
128	0.95	0.00001	-1	10	512	false	0.00001	1345.54
128	0.9	0.00001	-2	5	1024	true	0.00001	1328.81
256	0.95	0.00001	-1	10	1024	false	0.00001	1316.15
128	0.95	0.00001	-2	5	512	true	0.00001	1314.80
128	0.9	0.00003	-1	10	1024	false	0.00001	1300.67
128	0.95	0.00001	-2	10	512	true	0.00001	1297.27
256	0.9	0.00001	-1	10	512	true	0.00001	1291.71
128	0.9	0.00001	-2	10	4096	false	0.00001	1280.08
128	0.9	0.00003	-2	10	1024	true	0.00001	1260.49
256	0.9	0.00001	-1	5	1024	true	0.00001	1246.09
256	0.95	0.00001	-1	10	1024	true	0.0001	1235.01
128	0.95	0.00001	-1	10	1024	true	0.00001	1210.89
256	0.9	0.00001	-1	10	1024	true	0.00001	1202.36
256	0.95	0.00003	-2	10	512	true	0.00001	1194.46
128	0.95	0.00003	-1	10	1024	false	0.00001	1124.79
128	0.95	0.00001	-2	10	1024	false	0.00001	1123.54
128	0.9	0.00001	-1	10	1024	true	0.00001	1120.74
256	0.9	0.000003	-1	10	1024	false	0.00001	1079.17
128	0.9	0.000001	-2	10	1024	true	0.00001	1072.54
128	0.9	0.000001	-3	5	512	true	0.00001	1057.06
128	0.9	0.00003	-2	3	4096	true	0.00001	1044.13
128	0.95	0.000001	-2	5	1024	true	0.00001	1042.69
128	0.9	0.000003	-1	5	1024	false	0.00001	997.65
256	0.9	0.000001	-2	10	512	false	0.00001	958.96
128	0.95	0.000001	-1	10	1024	false	0.00001	930.24
512	0.9	0.000003	-3	3	4096	true	0.00001	929.63
128	0.9	0.00003	-1	10	512	false	0.00001	922.81
256	0.9	0.00003	-2	5	1024	true	0.00001	915.33
128	0.95	0.000001	-1	10	512	false	0.00001	898.50
128	0.9	0.000001	-1	10	512	true	0.00001	893.27
128	0.95	0.000001	-1	5	512	true	0.00001	883.68
128	0.95	0.00003	-2	5	4096	false	0.00001	870.71
128	0.9	0.00001	-3	10	1024	false	0.00001	854.40

Table D.1: Hyperparameters combinations and relative results for the tuning based on the reference motion

D.2 Qualitative results

Here we present the findings of a qualitative analysis of our imitation approach. The data was collected to complement the quantitative analysis (i.e., error and reward) that we carried out to test our system. The findings are presented in the form of multiple images corresponding to the reference video, the pose of the hand extracted from the video using FrankMocap (Rong et al., 2021) and the results of our imitation algorithm.

Figure D.1: Qualitative results for the tuning reference motion

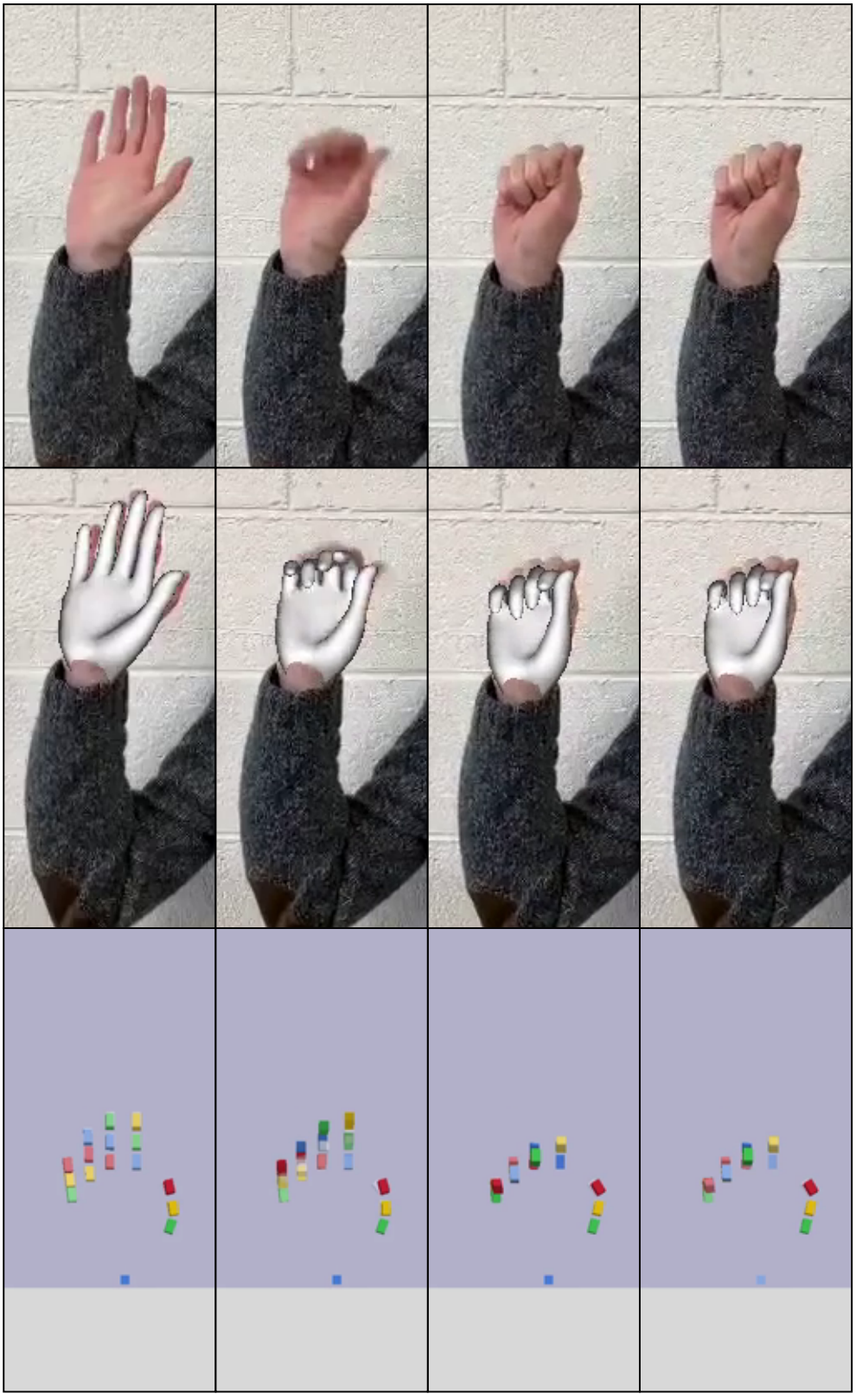


Figure D.2: Qualitative results for the reference motion representing the letter A

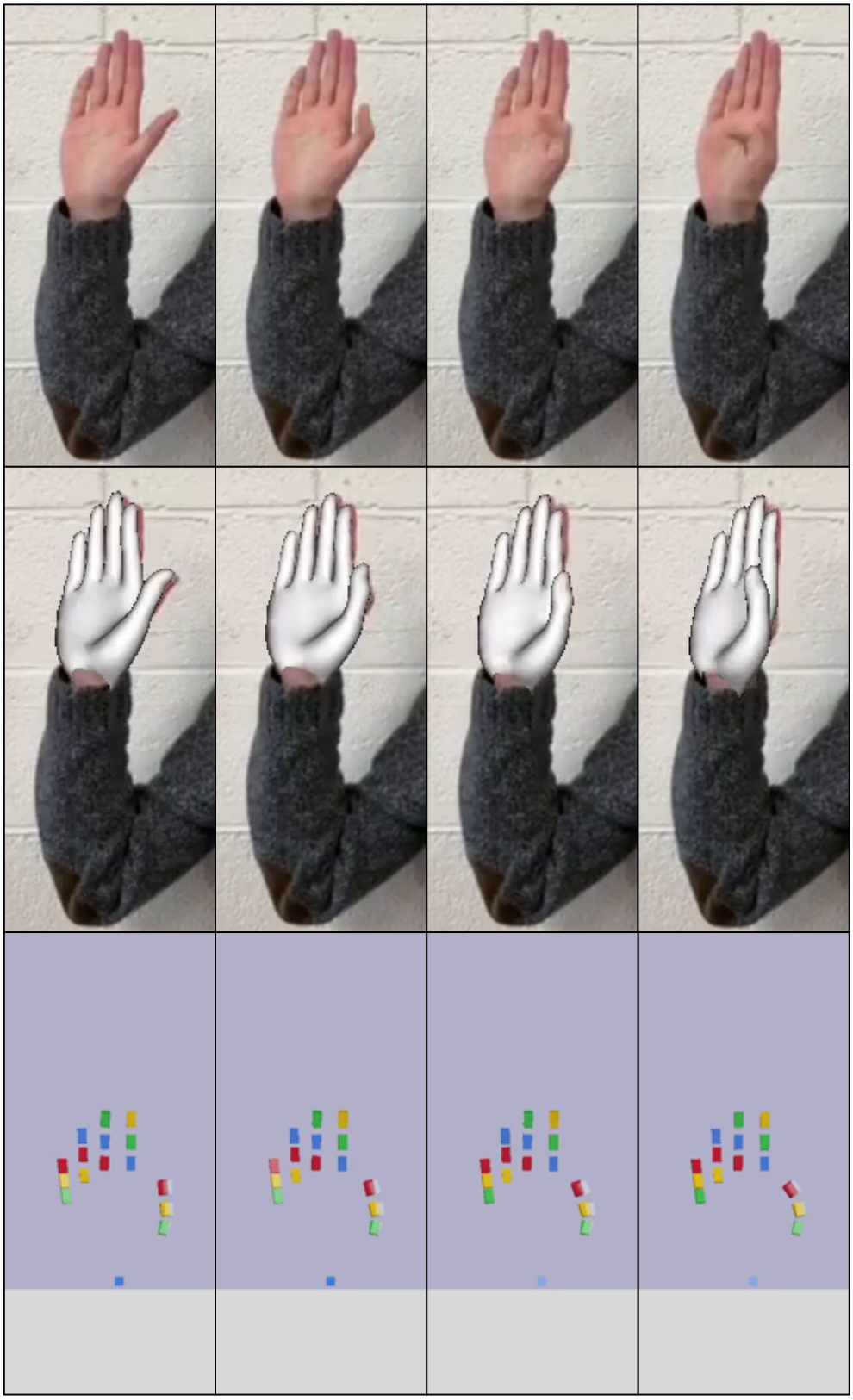


Figure D.3: Qualitative results for the reference motion representing the letter B

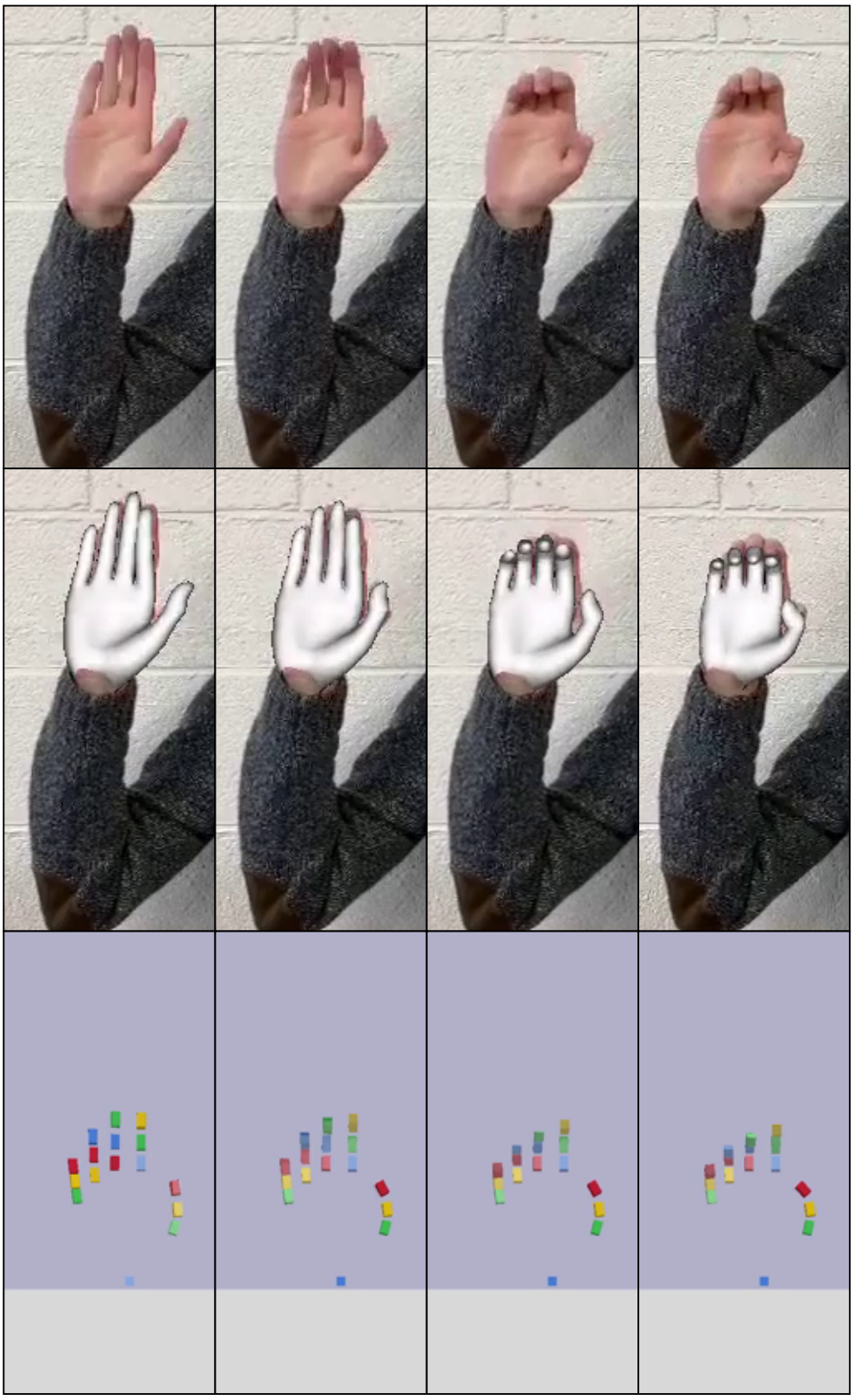


Figure D.4: Qualitative results for the reference motion representing the letter C

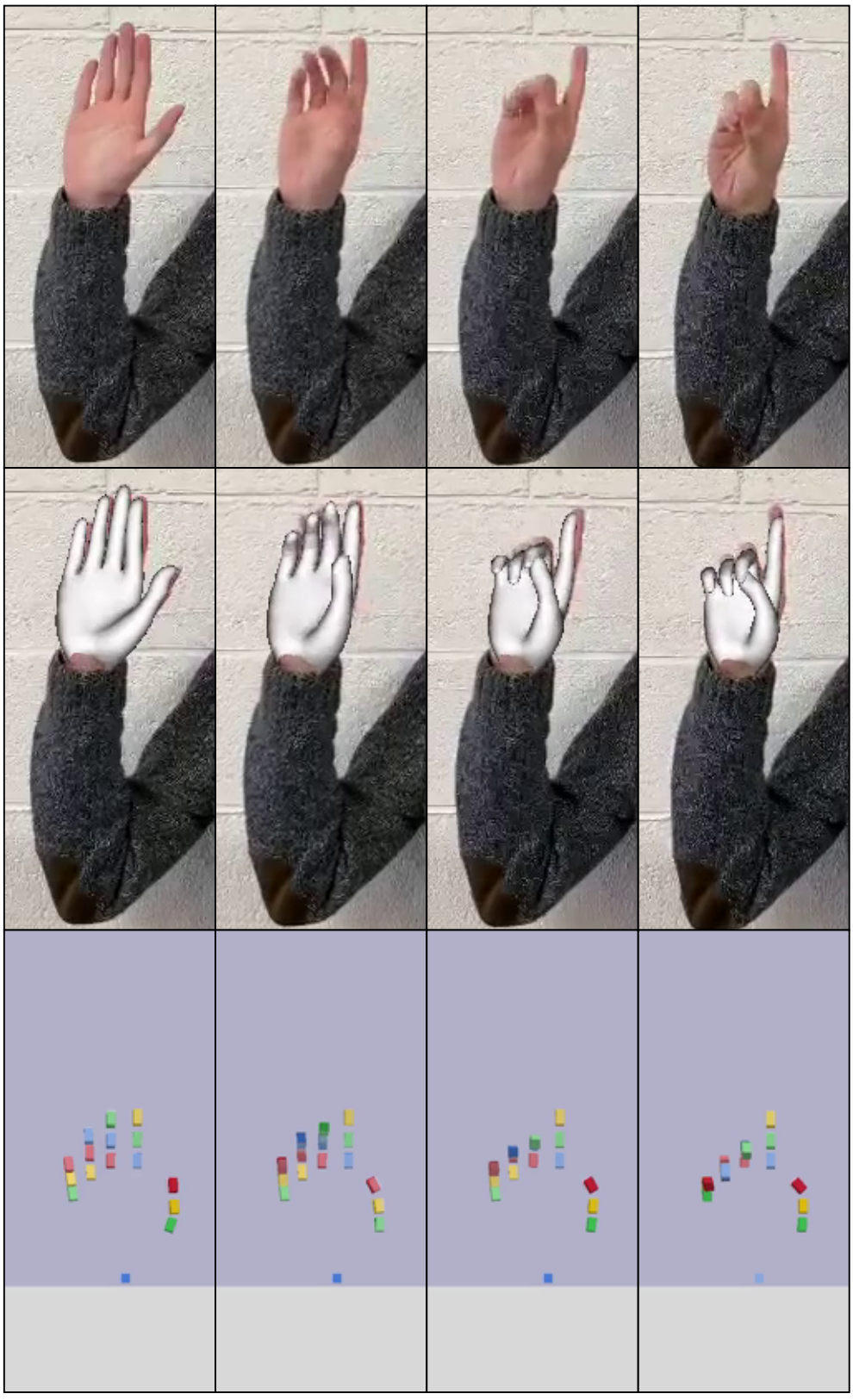


Figure D.5: Qualitative results for the reference motion representing the letter D

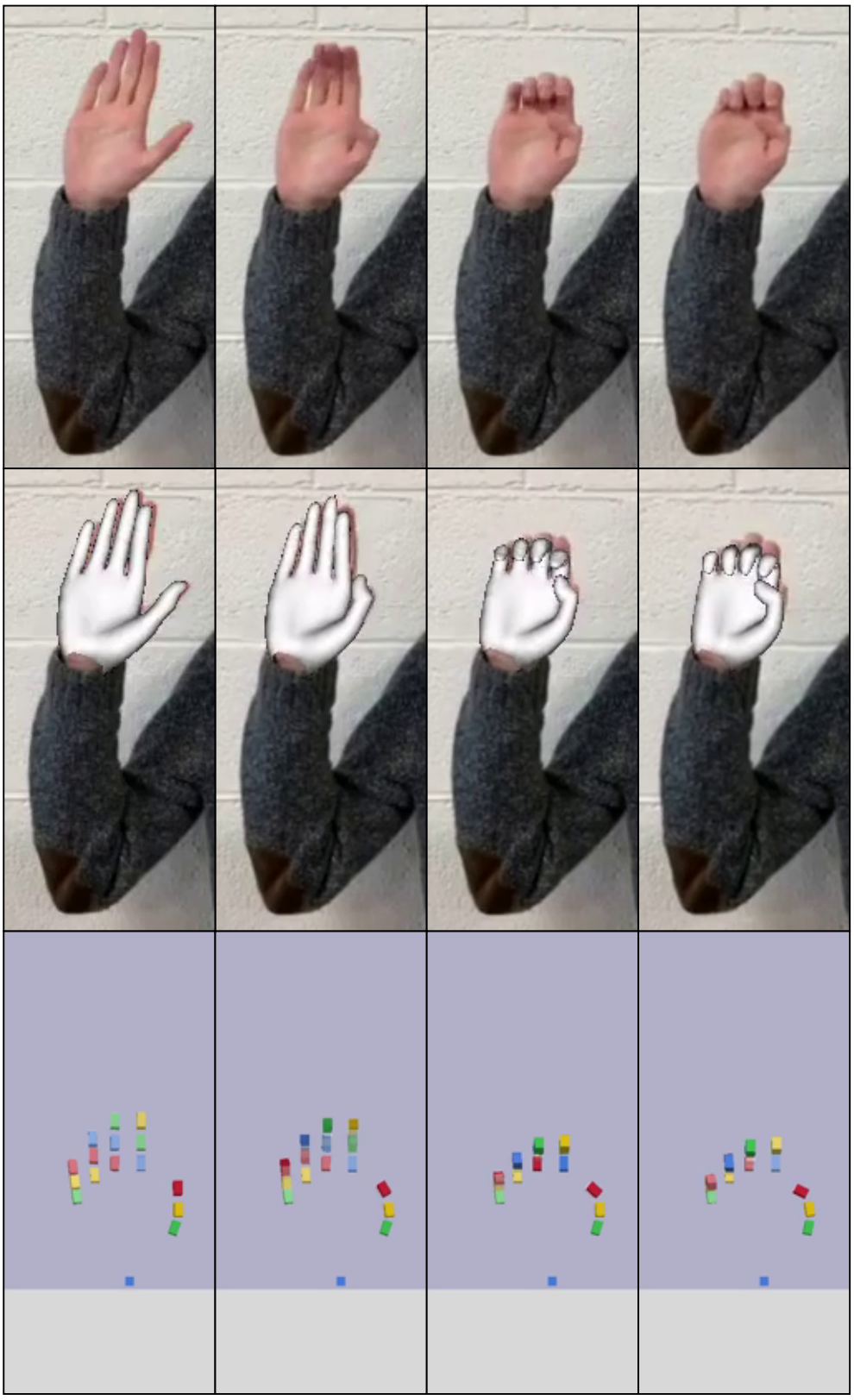


Figure D.6: Qualitative results for the reference motion representing the letter E

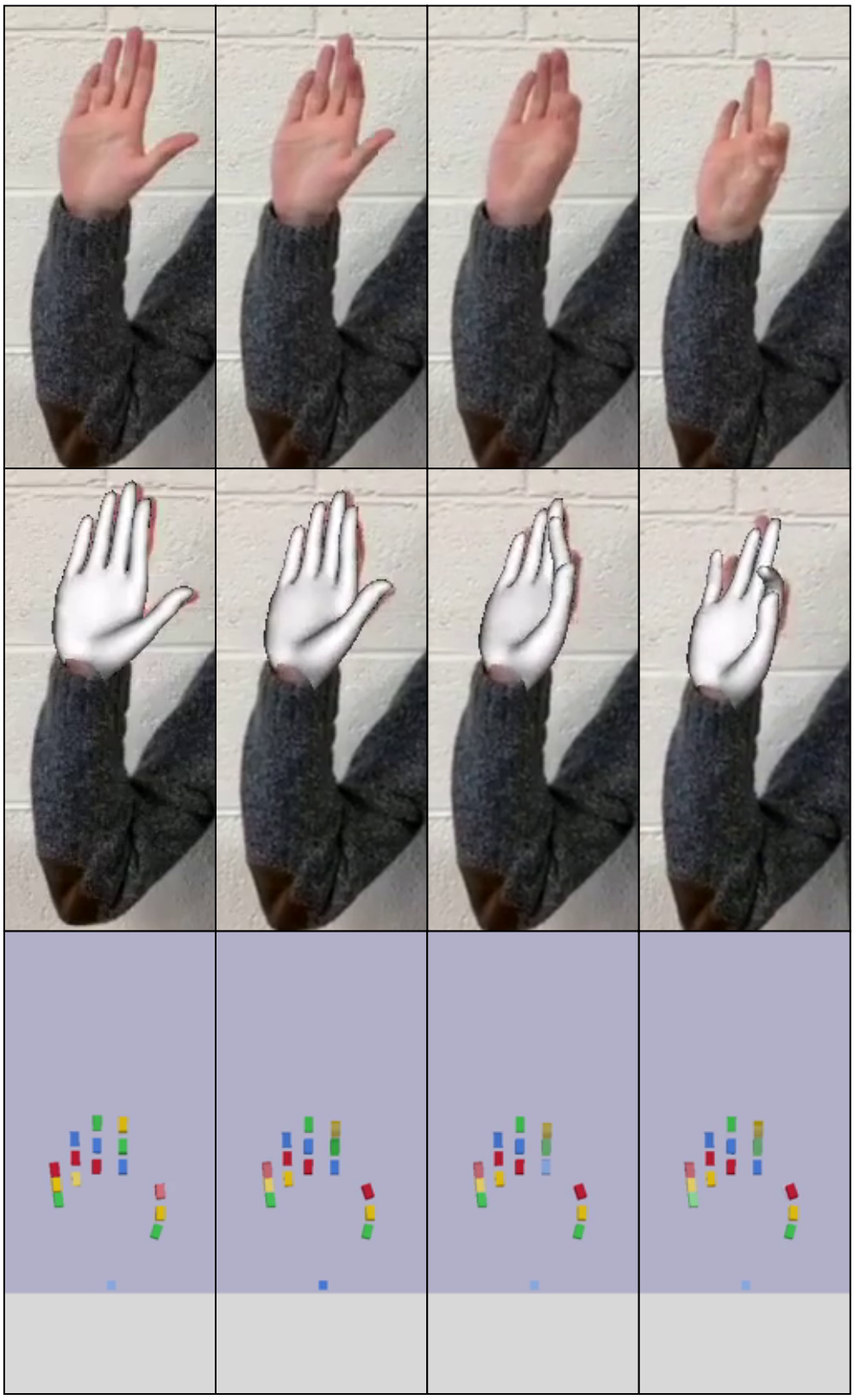


Figure D.7: Qualitative results for the reference motion representing the letter F

Appendix E

Appendix to Chapter 6

E.1 Qualitative results

Here we present the findings of a qualitative analysis of our imitation approach. The data was collected to complement the quantitative analysis (i.e., error and reward) that we carried out to test our system. The findings are presented in the form of multiple images corresponding to the reference video, the pose extracted from the video using FrankMocap (Rong et al., 2021) and the results of our imitation algorithm.

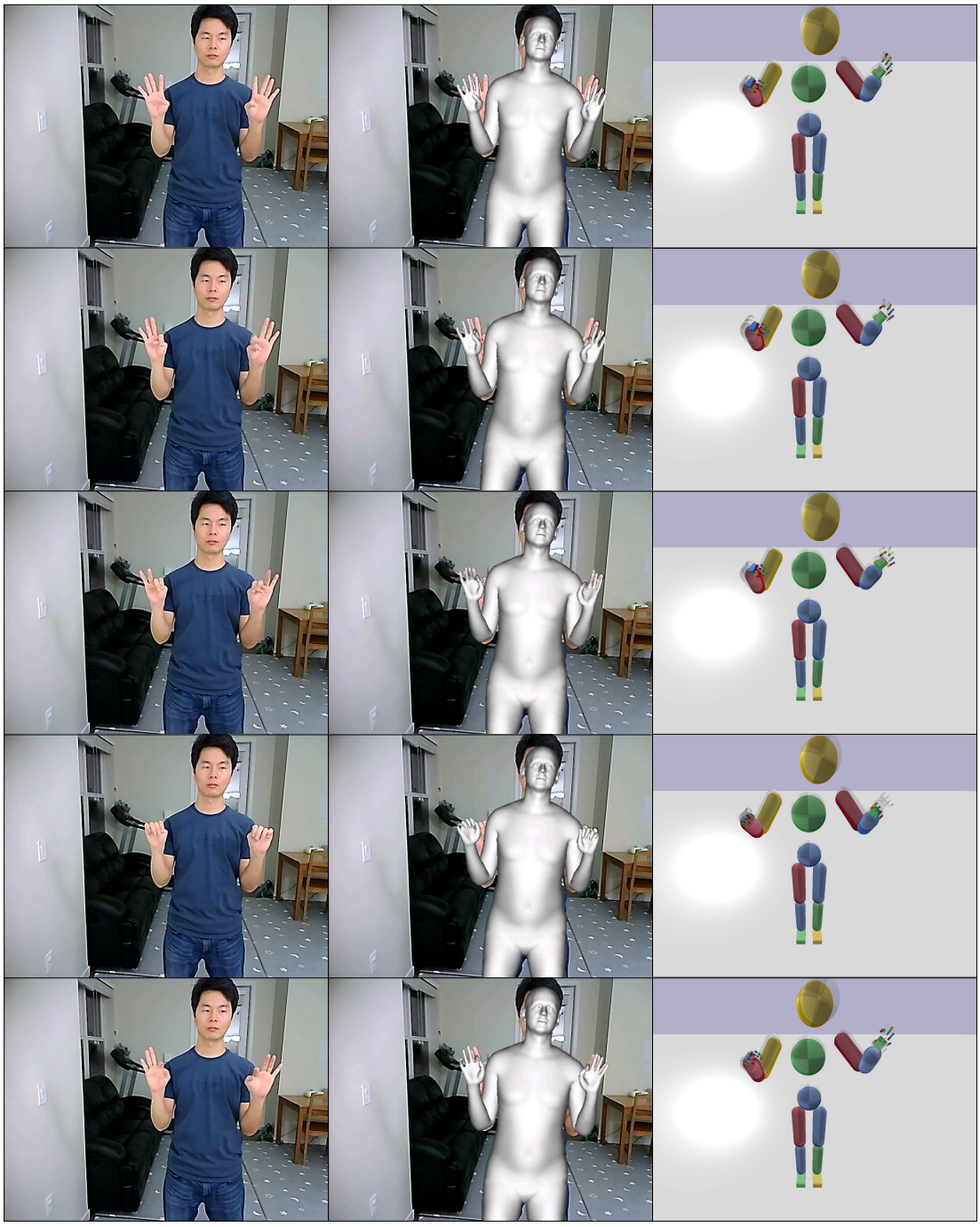


Figure E.1: Qualitative results for the tuning reference motion

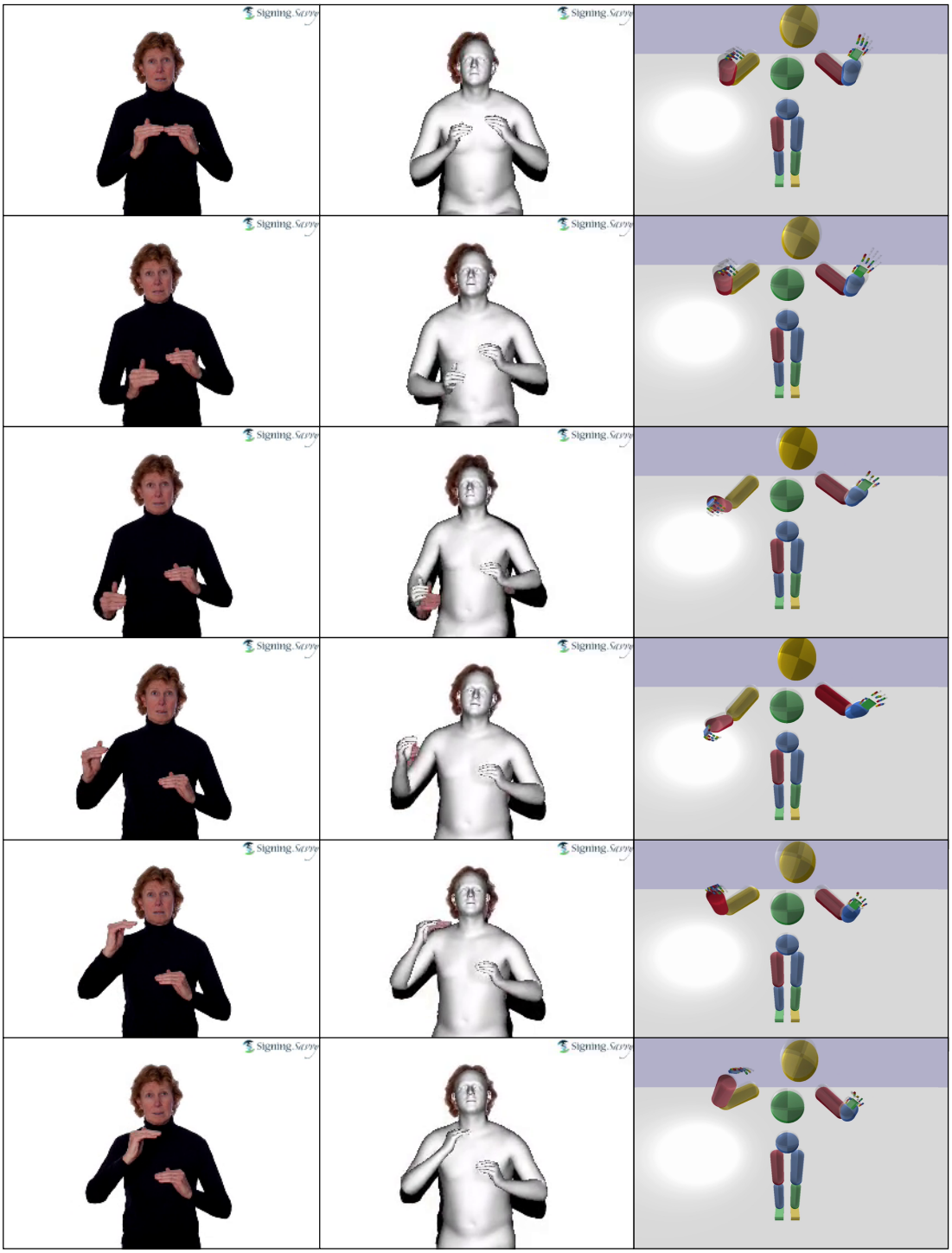


Figure E.2: Qualitative results for the reference motion representing the lemma `above` (WLASL id 00433)

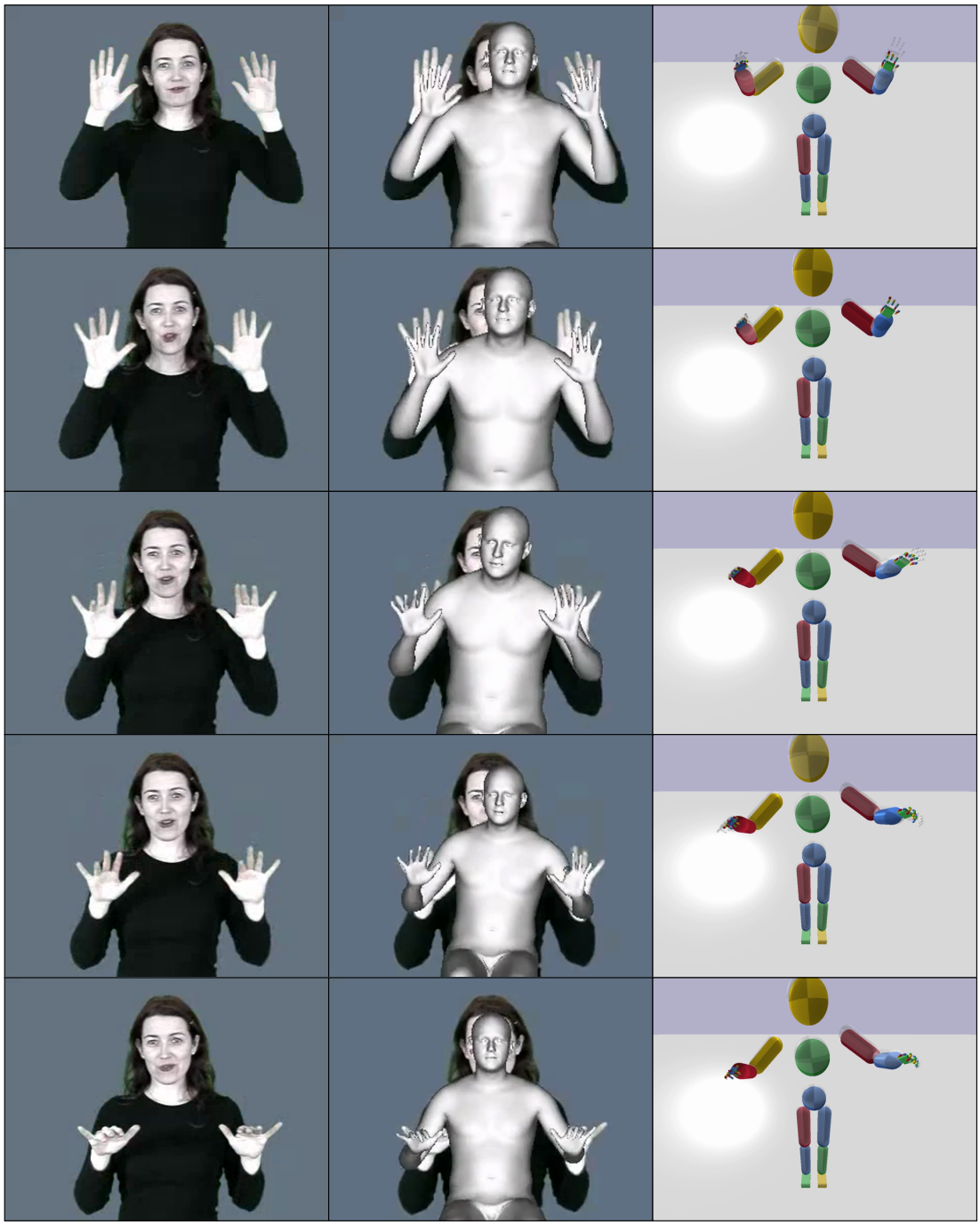


Figure E.3: Qualitative results for the reference motion representing the lemma `snow` (WLASL id 52861)

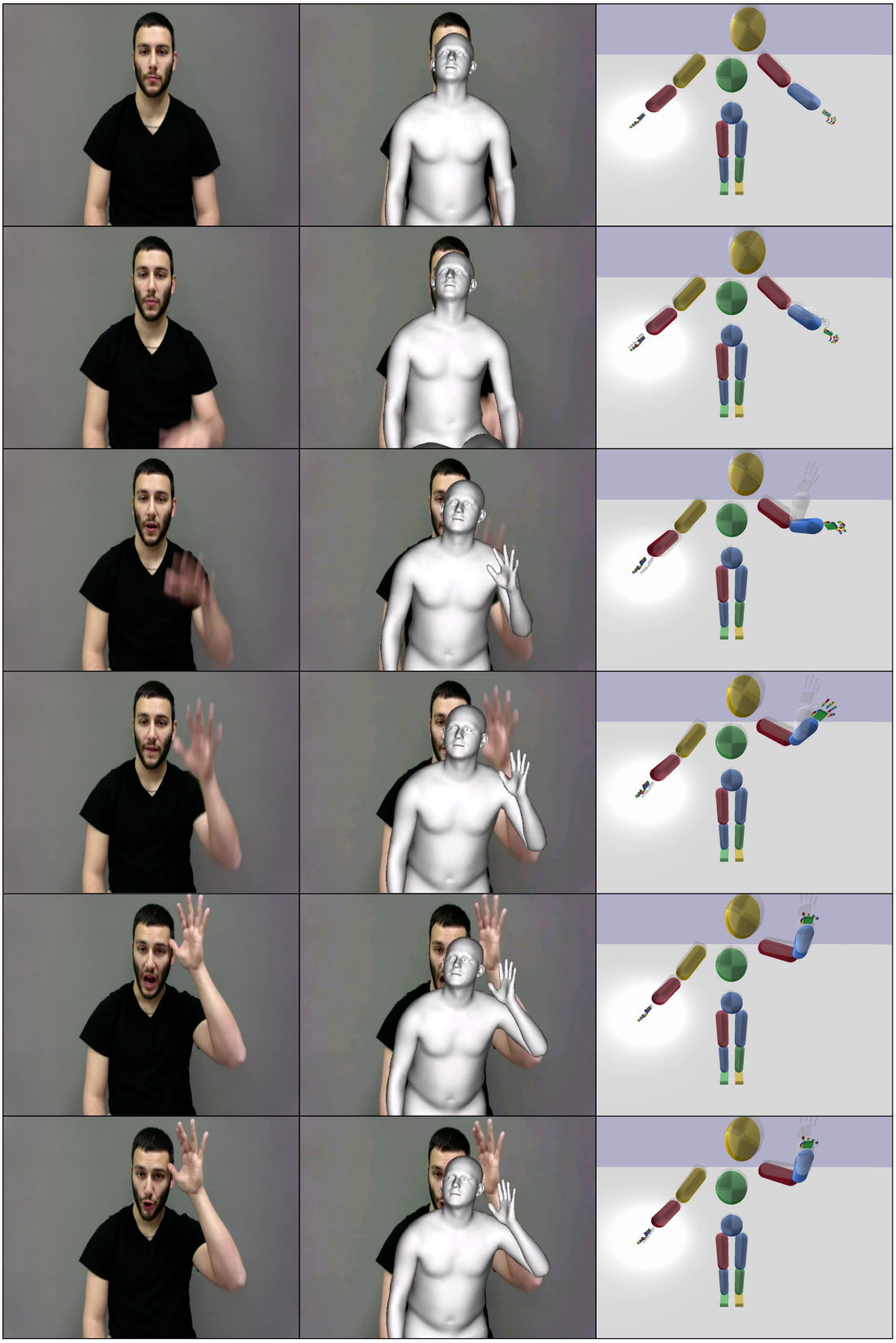


Figure E.4: Qualitative results for the reference motion representing the lemma `father` (WLASL id 69318)

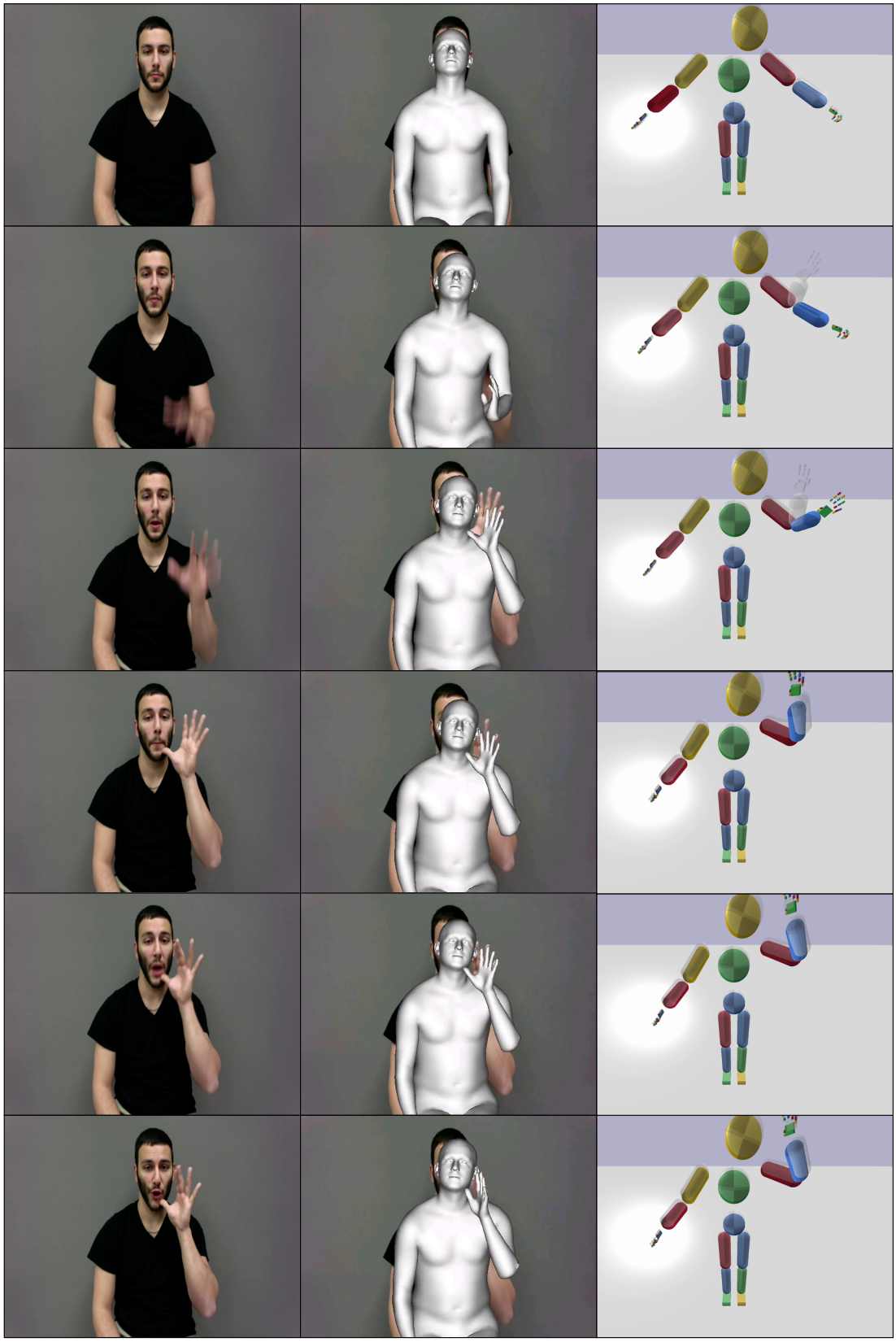


Figure E.5: Qualitative results for the reference motion representing the lemma `mother` (WLASL id 69402)

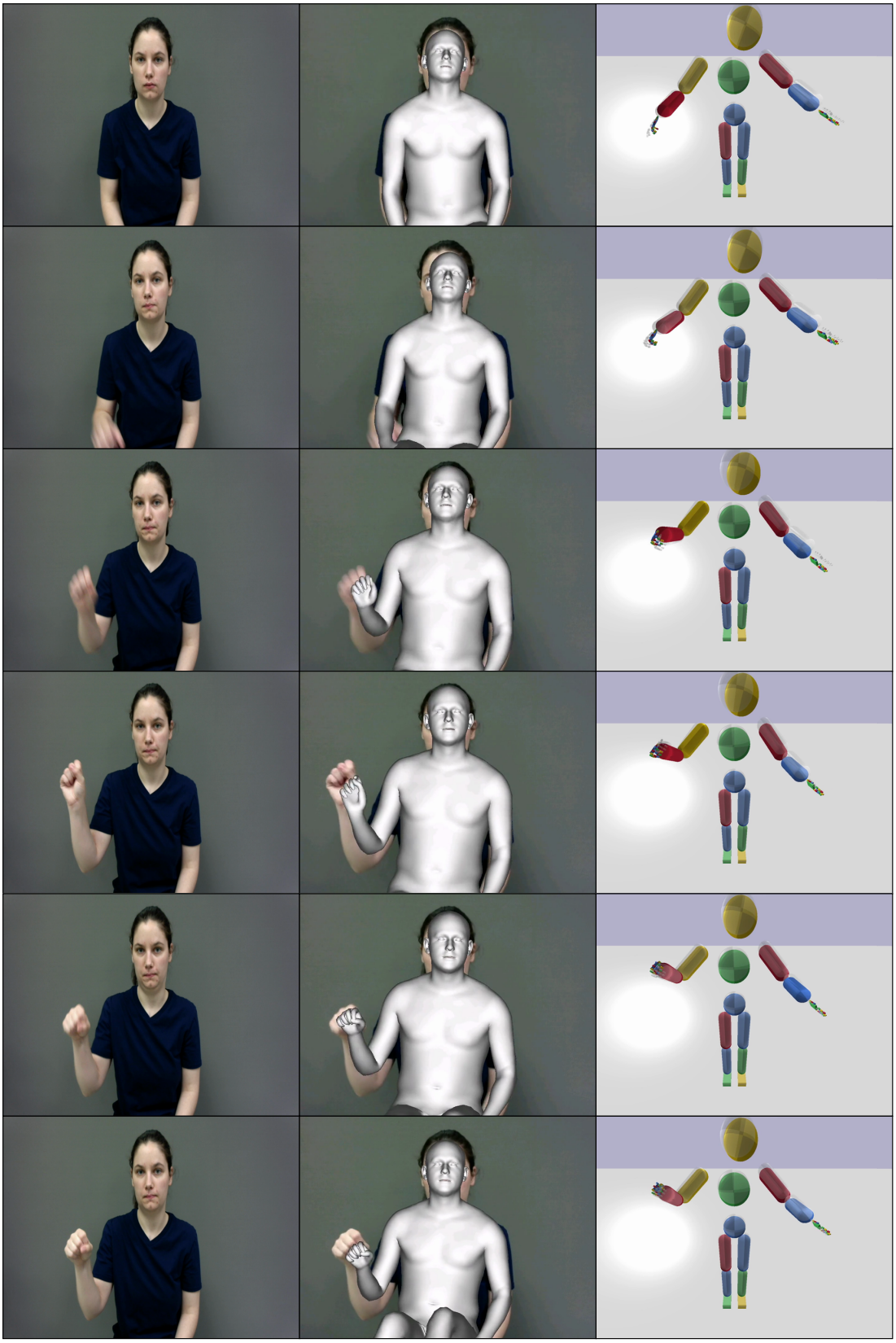


Figure E.6: Qualitative results for the reference motion representing the lemma `yes` (WLASL id 69546)