

PE2LGP 3.0: from European Portuguese to Portuguese Sign Language

Rui Ferreira
rui.aco@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2018

Abstract

Since Stokoe stated that Sign Languages are natural languages with rules, plenty of studies have been done for all sign languages. In the computational era that we live in, this means that a merge between computers and sign languages has been attempted, resulting in early stages of spoken languages text to 3D avatar animations' sign language translation. This thesis is focused on the translation from Portuguese to Portuguese Sign Language (LGP), creating an intermediate sign language representation and testing the avatar animation with JASigning's avatar. This project tokenizes the input and applies natural language processing techniques in order to prepare the input to follow the LGP grammar and syntax (the syntax is transformed, from Portuguese to LGP, using rules created with the help of an LGP specialist). This part is equivalent to the creation of *Sign Tokens*. Then, it changes the glosses for their equivalent in SiGML (an XML based transcription language). The result is tested in an existing avatar (JASigning) that directly reads SiGML. The system is evaluated in three different ways: the syntax correctness, the ease of its annotation and the overall correctness of the signs. This thesis presents an early work of an area that still has plenty to develop in the future.

Keywords: Sign Language, Automatic Translation, HamNoSys, SiGML, LGP.

1. Introduction

1.1. Motivation

According to the World Federation of the Deaf, about 70 million people use a sign language as their main source of communication ¹. Just this number alone should give the reader a feeling about the importance of studying this field. However, its study is not as straightforward as one might think: not only are sign languages not yet fully understood, there are also many sign languages available and they are all different. This contributes to the fact that, computationally speaking, a system that is able to process all of the sign languages in a generalized manner is not possible to achieve. In general, there are sign languages for each country (or region), ranging from American Sign Language (ASL), eventually to our own Portuguese sign language, Língua Gestual Portuguesa/Portuguese Sign Language (LGP). In addition to this, sign languages evolve with time, just like spoken natural languages (for instance English and Portuguese).

All of these sign languages share a common

problem: none have its official **writing system**, even though there is a strong attempt in Brazil to make *SignWriting* its official writing system. This means that if we somehow want to write a sentence using sign language and in an internationally (or nationally) accepted manner, we cannot. The advantages of having a writing system for sign languages are many, being one of them providing a way for the deaf to write and read sign language. One way to get the most out of a writing system of sign languages has been studied significantly in the last decades: automatic translation of an oral language to a sign language and, with the result, converting our text representation of the sign language into 3D animations, with the help of an avatar (i.e: write something in a spoken language and have an avatar translate it to signs of a certain sign language).

Sign languages not only require the use of both hands, but they also require the use of non manual elements, like facial expressions, including the use of the eyebrows, mouth and nose or body tilting. There is also plenty of recent related work, as the combination between sign languages and machine translation has been a hot topic over the recent past. However, a considerable amount of this re-

¹<https://wfdeaf.org/human-rights/crpd/sign-language/>

lated work is done for English, and not as much for Portuguese. In fact, despite some attempts, LGP's automatic processing did not have as much coverage as other sign languages within this area, yet making it a fresh field to start studying.

1.2. Goals

A great sum of the work that has been done in this area tried to do it all: from translating a text to a certain format that tries to hold the information needed from a certain sign language (henceforth called **Sign Language Intermediate Representation**), and using all of this data to somehow move an avatar's body, thus creating our translation via an avatar [18].

This dissertation does not focus on the creation of an avatar movement generation part, and develops an automatic translation system from European Portuguese to LGP, provided that the translated outcome (in the format of a Sign Language Intermediate Representation) is ready for future use by an avatar (Figure 1). Instead of creating an avatar from the beginning, this dissertation's project uses an already existing avatar in order to test its translations (this avatar is **JASigning**²). This means that instead of working on the whole process, this dissertation is focused on the part before having an avatar playing the translated sentences/signs in LGP. More specifically, it deals with the natural language processing of any input sentences in oral European Portuguese, translating it into a text format intermediate representation of what is being translated, in LGP. The avatar JASigning is used to test this Intermediate Representation.

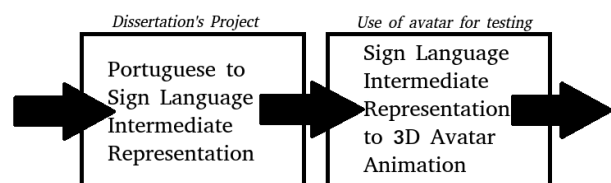


Figure 1: The first part represents the system that this dissertation implements. The second part is not dealt by this dissertation: instead, an existing avatar is used for testing.

Figure 1 shows us that the dissertation only implements the translation from Portuguese to LGP up to a text format (an intermediate representation). The process of this implementation includes changing the syntactic structure from oral Portuguese to LGP, since these are different languages and possess different syntax. Then, given the information we gather in all of the past work in this area, we use a *transcription language* (a language that holds information on signs, like its

²<http://vh.cmp.uea.ac.uk/index.php/JASigning>

movement, location, etc.) that fits best for our specific situation (this dissertation uses **Hamburg Notation System (HamNoSys)** [6]), taking into account the language we are dealing with and our needs, depending on the grammar rules, morphology, etc. **SiGML** is an Extensible Markup Language (XML) representation of HamNoSys, and it is used in order to leave the output readable by an avatar. Finally, when the output is in SiGML, it can be tested using JASigning.

2. Background

2.1. Sign Languages

2.1.1 Sign Language Studies

Before the 60's, not much had been studied about sign languages. In fact, sign languages were not even considered natural languages at all. The first man to recognize sign languages as natural languages was William Stokoe [17]. In his work, contrary to what people believed before, Stokoe **said that just like oral languages, sign languages possessed minimal pairs**. He created the Stokoe Notation, being this the first attempt at textually representing the ASL³.

Meanwhile, **more has been studied and some conclusions are agreed upon everywhere. Phonology is related to the organization and construction of larger elements through smaller elements, and it studies the phonological relationships between these in a language** [9]. This *language* (when we talk about phonology) used to be considered only a spoken language, until Stokoe presented his studies, and introduced the world to **a new kind of phonology**, specific to sign languages. In fact, it was introduced as **cherology**, but **it was changed to phonology**, as a means to **intuitively demonstrate** the resemblance between the structure of both *modalities* of languages (oral and signed). In a spoken language we can define a phoneme as a single unit that is part of the construction of words. A good analogy lies in the use of Lego, where we use singular pieces to create unique combinations. Let these pieces be our phonemes. These pieces differ in shape and size, and if we change one of these specifications, it can entirely change the looks of our work of art. With signs, we also have **phonemes** (formerly known as **cheremes**), **which are separated into its configuration, localization, movement and orientation**. These are *sign*

³The first attempt to represent any sign language was made by L'Épée⁴ in the eighteenth century, to represent his *signes méthodiques*. [17]

⁴https://en.wikipedia.org/wiki/Charles-Michel_de_1'Épée

primes, and by changing one, we are changing the sign itself. Besides these, every sign language also possesses **non-manual elements**. The manual elements are the ones performed with our two hands. The non-manual elements, however, deal with other means of communicating other than with the hands, like using facial expressions, shoulders, neck and even gaze. These are just as important, since in LGP they can also entirely change the meaning of a sign.

2.1.2 About LGP

The first official study for Portuguese Sign Language was achieved in 1980 by Maria Isabel Prata and Raquel Delgado Martins [16], and it was considered our first dictionary for LGP. Eventually some studies were made on LGP [10, 12, 11, 2], but nothing compared to the likes of ASL, making LGP a fresh field for studying. It is of the utmost importance to try and understand the many characteristics of the Portuguese Sign Language. The difference between LGP and other signed languages lies in the rules that are applied in it, whether in its phonology, morphology or syntax.

The Portuguese syntax follows a Subject - Verb - Object (SVO) structure, while LGP's syntax generally follows an Object - Subject - Verb (OSV) structure. However, there is plenty of LGP grammar that we do not know of. Besides this, Portuguese is also made up of other syntactic functions, such as, for example, complements. LGP's grammar only states that it generally follows an OSV structure, and does not mention the other elements that can be found in a sentence.

In sign languages we also have **classifiers**. These are morphemic elements that are similar to signs, but in reality they are used to describe or qualify *people, animals, things* (like objects) and *actions* [2]. Classifiers exist in almost all sign languages, but they are/may be different in each sign language. Classifiers can be nominal or verbal [13]. Nominal classifiers are defined by a sign belonging to some category, such as feelings, places, shapes, textures, trajectories or locations. Verbal classifiers, which are more relevant in our thesis, represent specific situations where in order to connect a specific combination of verb with subject, or with an object, or to show the way it happens visually, or even the location, the sign itself is exceptionally changed. For instance, *eating a soup* or *eating an apple* will be represented by different classifiers, as the act of holding the soup/apple in order to eat them is different for each one.

In LGP, to sign names of people, like John, *gestural names* are used. This only applies, however, if a person has one associated with him/her. When

there is no gestural name associated with a person, his/her name is spelled letter by letter. For example, if a person named John does not have a gestural name, the letters J, O, H and N are spelled individually.

2.2. Computational Approach

Natural Language Processing includes a wide variety of techniques used to process '*naturally occurring texts*' [8]. This includes looking at things like morphology, syntax and semantics.

Both statistical and rule-based machine translation can be used for this project. In order to use a statistical approach to Machine Translation (MT) we need a large corpus. *Corpus* is a large collection of data that allows us to draw statistical models from it. Because we are dealing with a translation problem which starts with text in portuguese and ends in an sign language intermediate representation, if we were to take a statistical approach to our translation, our corpus would contain sets of text in Portuguese. In order to analyze the syntax of such corpus, we need to annotate it. This can be achieved with Part-of-Speech (POS) tagging, which tells us each word's morphosyntactic category.

Some sets of corpus that are annotated are available. These are the so called treebanks. A very complete library that allows us to work on Natural Language is the Natural Language Tool Kit (NLTK) ⁵. From it, we can access plenty of corpora, including the Portuguese treebank: Floresta Sintática ⁶. Another great library for Natural Language Processing is **Spacy**. Spacy is great since it also allows us to extract syntactic dependencies directly. Spacy is used in this thesis.

Corpora are very important in MT. Unfortunately, it is an extremely time-consuming job to build one, and because many times one is not available, it leads to projects taking a rule-based approach. This is the case of this thesis: our translation system adopts a rule-based approach.

3. Related Work

3.1. Transcription Languages

Since the 60's, many were the attempts at representing sign languages in text. Many of these are represented through icons that try to simulate and show what the sign actually looks like, and all of its elements. These are quite more readable to the human eye, but in terms of computer processing, they are not as straightforward to process (a great

⁵<http://www.nltk.org/>

⁶<http://www.linguateca.pt/Floresta/principal.html>

example is HamNoSys [6]). Other attempts are the so called *glosses*, which are represented by the letters of our own alphabet. These are easier to process, but harder to understand (an example is the Stokoe Notation [17]).

The first attempt at a notation system for a transcription language was the Stokoe notation, created by William Stokoe himself [17]. Later in 1974, SignWriting was created [3]. It is made up of created symbols to represent sign language, and unlike the Stokoe Notation, it allows for the representation of non-manual elements. Each *gloss* written using the Stokoe Notation is made up of three characters: one for location, another for handshape and yet another for movement. However, SignWriting uses only one symbol per sign. Each symbol tries to visually emulate a sign entirely: both manual and non-manual elements (as seen in Figure 2).

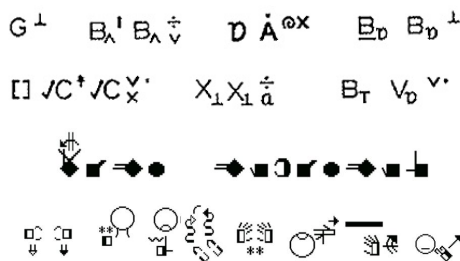
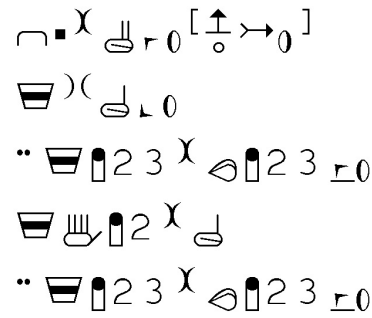


Figure 2: From top to bottom, an example of the Stokoe Notation and one of SignWriting, respectively.

XML is a meta-language that serves as a data holder for any data that we want to store ⁷. Due to its flexibility, XML is usually thought of as a practical solution for the problem of storing an icon-based transcription language. It was certainly the answer to SignWriting’s problem, as SignWriting Markup Language (SWML) was created. SWML can be used to represent each sign into a set of its features, according to SignWriting.

Another very famous attempt at sign language representation is the HamNoSys. It combines both the Stokoe Notation, in the sense that each sign is represented by handshape, orientation, location and actions, and the SignWriting, in the sense that these are represented by icons. Contrary to the Stokoe Notation and SignWriting, it was not made for the common reading and usage from deaf people: its creation was for academic purposes, and it is a widely popular choice. Just like SignWriting, computationally, HamNoSys can be represented by XML based languages. HamNoSys was used in this project and the used markup-language was SiGML. An example of both can be seen in Figure 3.

⁷<https://www.w3.org/XML/>



```
<hns_sign gloss="SPROD:A">
  <hamnosys_nonmanual>
</hamnosys_nonmanual>
  <hamnosys_manual>
    <hampinch12/>
    <hamindexfinger/>
    <hammiddlefinger/>
    <hamfingerstraightmod/>
    <hamextfingeru/>
    <hampalml/>
    <hamchest/>
    <hamclose/>
  </hamnosys_manual>
</hns_sign>
```

Figure 3: At the top, HamNoSys, and at the bottom, SiGML.

Finally, a more recent transcription language called AZee [5] becomes more appealing than the others as is it **multi-modal**. The other transcription languages associated non-manual elements to signs and not in time: AZee however, being multi-modal, treats non-manual elements anywhere in time, leading to more realistic animations in a final avatar.

3.2. Implemented Systems

3.2.1 International Sign Languages’ Systems

TEAM (Translation from English to ASL by Machine) [18] was the first attempt at fully translating english into an intermediate language and then using it to create 3D animations to represent it. Its intermediate language used *glosses* alongside some other information that came embedded to it, like some non-manual elements. The project was made for ASL, but it also allows for the processing of other languages. The movement of the avatar works by **keypoints**. In this work, there are two kinds of keypoints: Goal keypoints and Via keypoints. The first define a target location for the hand, and the latter, locations where the movement must go through to reach certain Goal keypoints. For the translation itself, a Lexicalized Tree-Adjoining Grammar [7] is used. Rules that are known to apply for ASL are used to change the syntax from english to ASL, given that typically ASL uses the SVO order (Subject - Verb - Object), which facilitates things, since it is the same as in English.

On more recent studies, a project whose aim is

to translate vietnamese to Vietnamese Sign Language (VSL) [14] tries to engage in the problems that we also face in this thesis: a language's syntax, phonology and morphology. Apart from their final translation approach, it is crucial to verify that the first problem they tackle is the syntax: the order of the tokens, negations, numerals or question sentences. For all of these things, they analyse examples in Vietnamese and examples in VSL. Then, with all this information, they use rule-based syntax trees to change the syntax from Vietnamese to VSL. They take an interesting approach on the translation itself, where first they label all the words with their syntactic functions, and if they cannot label a word, they try to find a synonym and label the initial word with the synonym's syntax role. When all is labeled, they find an already existing sentence with the same structure as the one they are transforming, and change the syntax accordingly. This is a very promising approach, as it scored 97.5% of success in its evaluation, using BLEU [15], on 200 test sentences.

Another recent project is called Dicta-Sign [4]. Dicta-Sign appears in 2010 and provides a dictionary that allows to search through signs, a translation between two signed languages and a Wiki for sign language in general. It is available for British Sign Language (BSL), Langue des Signes Française/French Sign Language (LSF), Greek Sign Language (GSL) and Deutsche Gebärdensprache/German Sign Language (DGS).

3.2.2 LGP System

The next project focuses on the translation of Portuguese European to LGP [1], and the used architecture can be seen in Figure 4.

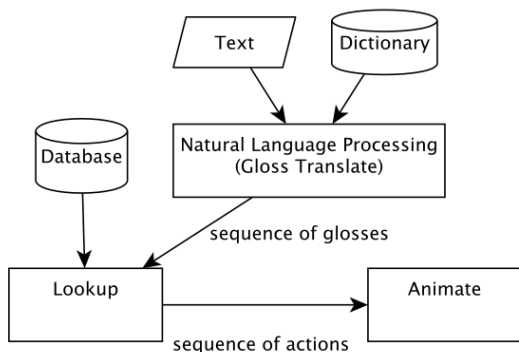


Figure 4: Architecture used in [1].

As observed in Figure 4, a gloss system was used. A rule-based system was made and used with custom-made rules. A *stemmer* was used in the processing of the portuguese text, as it allows us to see what a word's stem and afixes are. The

stem, in portuguese, is associated with the word's radical. The afixes (prefixes and suffixes) give us the option of looking at a word's attributes, for instance, its gender.

To sign the name of a person, if that person has a known gestural name, it is used. If not, it is spelled, letter by letter. Through a bilingual dictionary, the words are exchanged for their glosses and the syntax changed from Portuguese to LGP. Then, for each gloss, a correspondent *action* is looked up from a database, implemented with JSON⁸. An action may be made of one or more signs, as long as it fully represents its gloss. For these problems on Natural Language processing, NLTK was used. 57 basic signs were processed to use in the actions. This project, however, only considers sentences with a structure of a noun, a verb and then another noun because, like mentioned in Section 2.1.2, as far as it is known, LGP has a structure of Object-Subject-Verb (OSV). This is a limitation. Another downfall lies in the fact that there were no facial animations. Since plenty of important grammar aspects of LGP require non-manual elements, this is something else that needs to be done in order to get better results.

4. PE2LGP 3.0

4.1. Architecture

The order of the system created in this thesis is the following: text is received as input (in Portuguese), and the output is the *sign version* ready to be processed by a 3D animated avatar. In between, there are **two** main modules: the **Sign Tokenization Module** (Section 4.1.2) and the **SiGML Generation Module** (Section 4.1.3). Each serves its separate purpose. The full model can be visualized in Figure 5.



Figure 5: Visualization of the project's architecture, from input to output.

4.1.1 Used Transcription Language

AZee [5] is the best option when it comes to functionality, because it lets us reproduce non-manual elements whenever in time, regardless of the other manual elements. Unfortunately, after having contacted Filhol, we received the answer that AZee is only yet a theoretical concept. On the bright side, as of now, a 3D movement translation using AZee

⁸<https://www.json.org/>

is being worked on by Filhol, DFKI in Saarbrücken⁹ and DePaul University in Chicago¹⁰. Until then, unfortunately, using AZee for this purpose does not seem possible.

Despite not being completely *multimodal*, SiGML using HamNoSys is still a viable solution. It is an XML variation, and as such, multi-modality can be easily added in the future. Unfortunately, with the amount of work that has been put into this thesis, *multimodality* will have to be future work (in Section 3.1 the concept of *multimodality* is explained). **eSIGN** is a software that allows for the creation of signs in SiGML, from HamNoSys (Figure 6). It is practical for the automatic generation of XML-like signs. It allows for the fingerspelling of words and typing numerals. However, its *search by gloss* feature is only available for english words. But with it, we can still manually annotate words for the dictionary used by our system. The fact that we can annotate words (signs) in **eSign** was a key factor in the choice of using HamNoSys for our system.

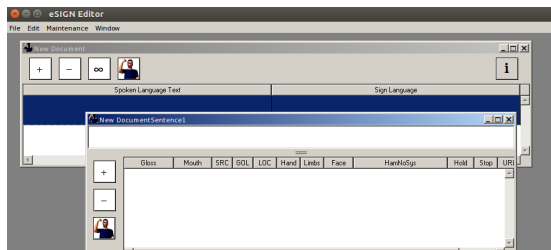


Figure 6: eSIGN's window.

SiGML is an XML variation. As such, it is not only really easy to use, but also to change. This is handful if we want to add a timestamp to it in the future. For all of these reasons, SiGML (using *HamNoSys*) is the used transcription language in this thesis.

4.1.2 Sign Tokenization Module

Overview: Figure 7 shows us how the input passed to the system (the sentence(s) to be translated) is processed from start to finish of this part of the thesis. In order to change the syntax, there is a morphology extraction step first, which includes POS Tagging, and a dependency parsing step in which we gather the syntactic dependencies of the input. With this information, the syntax is transformed into LGP's. At the end, the annotated classifiers and non-manual expressions are also included into our *Sign Tokens*.

How this module works will be explained with the following example: “*O João come a sopa.*” (in english: *John eats the soup.*).

⁹<https://www.dfki.de/web/kontakt/dfki-saarbruecken>

¹⁰<https://www.depaul.edu/Pages/default.aspx>

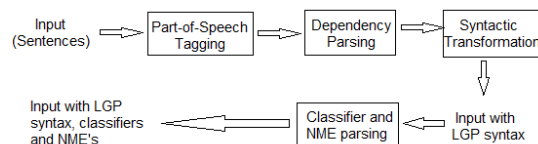


Figure 7: Overview of this part of the thesis. In the end we have the input in LGP syntax order, classifiers and Non-Manual Expressions, all of which together correspond to *Sign Tokens*.

The input, given as a STRING, can contain one or more sentences. The input is tokenized into *Sentence Tokens* (each *Sentence Token* is a sentence), and the *Sentence Tokens* are tokenized into regular *tokens* (elements of the sentence: words, named entities and punctuation). To perform this we use NLTK. To exemplify this step, let us suppose that our system receives as input the following STRING: “*O João come a sopa. A Maria come a maçã.*” (in english: *John eats the soup. Maria eats the apple.*). In order to process each sentence on its own, we first **separate** them into *O João come a sopa.* and *A Maria come a maçã.* Then we tokenize each of the sentences.

Two main things are done for each sentence from the input: **gathering necessary morphology information** and **exchanging the syntax from European Portuguese's to LGP's**. The tool being used for these steps is Spacy¹¹, since it offers trained dependency models. Spacy allows us, for example, to infer that the *Sopa* (in english: *Soup*) is Singular (not Plural). Another important example is getting the verb tenses (and other information related to them). *Come* (from the example sentence “*O João come a sopa.*”) presents itself on the *Presente do Indicativo, 3ª Pessoa do Singular* (Singular Third-Person, Present Simple, Indicative mood).

Changing the syntax is done with the help of an annotated *csv* document, in which one of the columns contains possible syntactic structures in Portuguese, and in the other column, their equivalent syntax in LGP. Each pair is represented as a **syntax rule**. The rule that is applied to our example (*O João come a sopa.*) can be found in Listing 1.

nsubj PREDP-I obj . -> nsubj obj PREDP-I

Listing 1: Syntax rule that is applied to *O João come a sopa.*'s syntax structure (more precisely, *João come sopa.*: this is because some tokens - articles - are removed). A delimiter (→) is used, dividing the input from the output.

The rule shown in Listing 1, related to *O João come a sopa.*, does not account for the articles *O* and *a*. As an example, instead of having a rule for *art nsubj PREDP-I art obj*, we simply remove

¹¹<https://spacy.io/>

the articles and directly apply the sentence to follow the rule in Listing 1. This is because there is an exclusion of words that are not used in LGP. The articles in the earlier example are not signed in its translation, and the rule is applied for the sub-sentence *João come sopa*. Its output becomes *João sopa come*.

Listing 1 says that when a sentence has a subject (*nsubj*), a predicate in the present tense of the Indicative mood (*PREDP-I*) and a direct object (*obj*), all of these in this specific order, the syntax is changed to subject, object and then the verb. The punctuation, in this specific rule, was also dropped: this happens because in a declarative sentence in LGP there is no need to sign a period. It appears in the input because we parse all of the syntactic elements of a sentence (including punctuation), and in the output, in this case, it disappears because it is not needed. As such, in the left-hand side of Listing 1 there is a period at the end of the syntax structure, while at the right-hand side there is not. With the help of such rules, the syntax is exchanged.

At the end, the *Sign Tokens* are returned in the form of a *.txt* file, in between brackets. A representation of the sign tokens for *O João come a sopa*. can be seen in Figure 8.

```
{
  token = João
  gender = M
  type = PROP
  number = S
  syntax = nsubj
}
{
  token = sopa
  gender = F
  type = N
  number = S
  syntax = obj
}
{
  token = come|
  number = S
  syntax = PREDP
  person = 3
  mode = IND
  time = PR
  type = V
}
```

Figure 8: Representation of *Sign Tokens*. They appear in *LGP*'s syntax's order, contain morphological information about them and can be later easily read in the SiGML Generation Module.

Syntax Rule Creation: We created 100 sentences in portuguese in order to accomodate 100 different syntactic phrase structures. Our goal was to hand these sentences to an LGP professional and have that person translate its syntaxes to the

LGP equivalents. Not only was the syntax exchange annotated, but classifiers and non-manual expressions that are required to sign the words were included.

From these annotations **manual rules** were created for each syntactic phrase structure, where for a received phrase structure, the output would be the resulting respective annotation. These annotations were *normalized* in order to make the input and the output have the same notation and simply rearranging its order (for instance, $A B C \rightarrow A C B$). The used notation was Spacy's dependency tags directly, alongside a predicate variation added by us (Spacy does not return predicate verb dependency tags that contain all of the verb's information in its tag name). For the sentence *O João come a sopa.*, it would translate to *nsubj PREDP-I obj .*, and then, the output would be (according to our annotations) *nsubj obj PREDP-I*. The rule is applied and the sentence becomes *João sopa come*.

nsubj PREDP-I obj .	nsubj obj PREDP-I
nsubj PREDP-I obj ?	nsubj obj PREDP-I ?
nsubj PREDPP-I obj .	nsubj obj PREDPP-I
nsubj PREDF-I obj .	nsubj obj PREDF-I
nsubj aux INF obj ?	nsubj obj PREDF-I ?
nsubj INF obj ?	obj INF nsubj ?

Figure 9: This Figure shows a few of the used manual rules for the syntactic transformation.

About a third of the rules were drop for having inconsistent outputs: instead of simple word alignment, some syntax translations became impossible without adding a few words to the output that were not in the input, and as such, generic syntax rules were not feasible for those cases. **Exception sentences** were added as a means to accomodate some of these sentences: greetings (example: *boa tarde*, which in english translates to *good afternoon*).

A notation was also created for classifiers and Non-Manual Expressions and it can be defined by:

[] square brackets are used for classifiers (example: *nsubj obj PREDP-I[obj]*). By following this rule, we are not just adding a verb to that position: we are adding the verbal classifier between the verb and the object of the sentence. For *João sopa come*, the result would be *João sopa come[sopa]*, and in the next module, when fetching the sign for *come[sopa]*, the system will know we are dealing with a verbal classifier);

{ } curly brackets are used for non-manual expressions (example: *nsubj obj PREDP-I[obj]{?}*). The character ? adds an interrogative non-manual expression, resultant from the input being a question);

4.1.3 SiGML Generation Module

In Figure 10, we can take a look at how the second and final part of this thesis runs from the parsing of the *Sign Tokens* (generated from the first part of the thesis) to outputting the SiGML file required to play our translated animations.

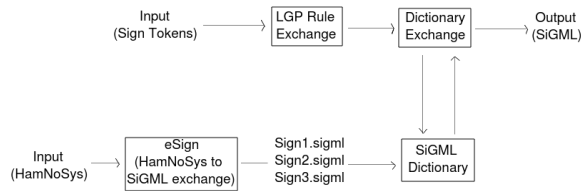


Figure 10: Overview of the second part of the thesis. The system receives the *Sign Tokens* and outputs a file with the SiGML necessary for the 3D animations.

As a means of demonstration, let us follow the example of the sentence: *O João come a sopa.* The input file is read, and the *Sign Tokens* are stored in a list of tuples where the first element is the token and the second element a dictionary of all of its attributes. Afterwards, with the help of these attributes, a number of rules is applied to our example within the LGP Rule Exchange Submodule. The token *João* is exchanged for the tokens *J*, *O*, *A* and *O*, because it is a name of a person. *Sopa* and *come* are merged into a single token in order to form a classifier: *sopa_come*. Besides these rules, glosses such as *FEMALE* are added.

The next step is changing each token for their SiGML equivalent with the help of our dictionary. The words have been annotated using *eSign*, a program that transcribes HamNoSys into SiGML. This means that in order to annotate words ourselves, we need to know HamNoSys. Each tokens' SiGML is fetched and put into a single SiGML file. This process can be seen in Figure 11.

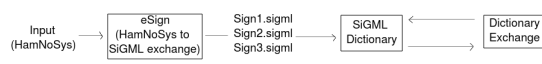


Figure 11: Overview of the part of this module that deals with the creation of our SiGML dictionary.

SiGML is the XML version of HamNoSys. This means that the content of a SiGML file is made up of HamNoSys characters, but in a *avatar-readable* format of XML. Our dictionary contains SiGML files of a limited set of words. An example of a file can be seen in Figure 12.

The final *.sigml* file will contain our full SiGML, ready to be reproduced by a 3D avatar.

```
<sigml>
  <hns_sign gloss="$PROD:J">
    <hamnosys_nonmanual>
    </hamnosys_nonmanual>
    <hamnosys_manual>
      <hamflathand/>
      <hamthumbopenmod/>
      <hamextfingeru/>
      <hampalm/>
      <hamchest/>
      <hamclose/>
    </hamnosys_manual>
  </hns_sign>
</sigml>
```

Figure 12: An example of a SiGML file. It has the opener and closer tags on the first and last line, and the letter *J*'s equivalent in HamNoSys (in order to give a simple example).

5. Evaluation

5.1. Syntax Evaluation

5.1.1 Definition

The first evaluation that we perform is the Syntax Evaluation. It consists on evaluating the probability of matching a sentence on the input with one of the syntax translations, given a new set of random sentences. We created two sets of sentences: one set purposely created in accordance with the specific rules of the system, and another set created at random (not having in mind the rules of the system). Both sets are made up of 50 sentences. The main goal of the first set is to verify why a syntax translation could go wrong, even if the rule for it exists.

5.1.2 Results

The results are shown in Fig. 13.

```
Number of >RANDOM< sentences that had a syntax rule applied to them:
( 9 / 50 )
Number of >NOT RANDOM< sentences that had a syntax rule applied to them:
( 30 / 50 )
```

Figure 13: Number of sentences, for each set of sentences, that had a matching syntax rule applied to them.

Before analyzing the random set of sentences, we will analyze the set of sentences that was created having in mind the existing rules. As we can see in Fig. 13, 30 out of 50 found a matching rule, while the other 20 did not. The biggest problem in these sentences is the dependency tag that is assigned to each token by Spacy, when gathering a sentence's syntax. Spacy uses a probabilistic model, which is not perfect, to assign the morphosyntactic category to a word: as such, in some sentences, even if we correctly apply the same syntactic structure, Spacy may still assign it differently. This is something that we have to live with. An example lies in the sentence *Vais vender*

a casa?, to which Spacy assigns the following syntax: *nsubj INF obl ?* (a subject, a verb in the infinitive form and an oblique nominal). The rule that we were trying to trigger with this test sentence was the one of: *nsubj INF obj ?*, and its original annotated sentence was *Vais comer a sopa?*. Notice how we want *casa* in the example sentence to be identified as a direct object of the predicate (identified with the tag *obj*), but instead, it is identified with a different tag by spacy (it is assigned the tag *obl*). As it is a Spacy problem, the most that we can do in the future is to cover these cases and create rules for them as well.

Let us now look at the random set of sentences. It is no surprise that only 9 had a matching syntax. There is a total of 63 rules that can be matched with the input. Obviously, there are plenty more ordered syntax combinations that can be passed in the input, and as such, the probability that a sentence will fall under a rule that currently resides in our system is very low. Random sentences show us the biggest liability in a rule-system: we have to be able to dynamically cover more and more rules in order to accommodate more and more combinations of syntactic structures passed in the input. The more rules we add, the more this random set's results should improve. Rules can always be added in the future, which means that this system works, and can evolve through time.

5.2. SiGML Annotation Evaluation

5.2.1 Definition

For this evaluation, the main goal is to understand how easy and practical it is to add new entries (new words) to our SiGML dictionary through eSign, for people who have never experienced HamNoSys and with little instruction on how to use it. To do this, we used *Google Forms* and delivered a form with a personal introduction page (for the testee to fill in with some details such as age and affiliation), an introduction to sign language, an introduction to HamNoSys and eSign, an Annotation phase and a final questionnaire on the annotation.

On the Questionnaire step it was asked to the testee how easy it was to learn the little taught theory on Sign Language, HamNoSys and *eSign*, how easy it was to use the HamNoSys keyboard in *eSign*, it was asked of them to evaluate themselves comparing their annotations with the videos in SpreadTheSign and it was also asked of them if it took them a long time to annotate each sign. Some questions had a range of 1 – 5 in order to be answered, and others were of open answer. The questions that are asked in the last step (Questionnaire) to the testee are shown in the next list: first the question is shown, and then the format answer

in which each question is evaluated. This test was divided into 2 batches of 5 tests each, in which the first batch the SiGML was not gathered and the in the second one the SiGML was gathered as well. The first batch served the purpose of seeing if this test was feasible, and its answers can still be used.

5.2.2 Results

After analyzing each person's SiGML for each word (the words were *RAPAZ*, *RAPARIGA*, *CARNE* and *BOLO*), it was interesting to see that only two (out of the 5 from the second batch of people) answered 4 to the question that asked the testees to compare their annotations to SpreadTheSign's. One of the 5 people also delivered invalid SiGML, so we will discard his test. Of the remaining 4, the worst SiGML was from one of the testees who answered 4 to this question. The rest were very similar, and close to the end result, even if the testees thought they had a poor performance on it. By analyzing the SiGML further, it is easily concluded that there was one main difficulty while annotating and testing the signs for the first time: the orientation. By looking at the SiGMLs and playing them on JaSigning, the hand configurations and movements all look alike, and are correct. However, most hand or palm orientations are slightly off. This may be due to the fact that when a HamNoSys symbol points right, for instance, it is pointing to the right side of the signing avatar (in the signing avatar's perspective, and not in our perspective). When learning HamNoSys, choosing the right orientations got confusing.

As for the time it took to annotate each word, the results show us that the hardest annotation was the first one. From then, there is a decline in the time taken to annotate each word. The only exception is the second and third word, which took about the same time. Still, a **learning curve** can be seen: as the annotations progress, the easier it gets, and the more accustomed the testee is with annotating.

From all of these results, it is possible to say that if these results were gathered with people who have never had contact with sign languages, much better results should come from someone who is a specialist. A feature of this system is that **anyone can annotate it**. For now, there is not much vocabulary inserted in the system, but if someone wants to add more, little instruction should suffice. Besides, even if a sign is wrongly annotated into the system, it can be reported by someone who notices and re-annotated.

5.3. Final Signing Evaluation

5.3.1 Definition

In this final evaluation, we have specialists looking at a set of annotated signs and evaluate them. In order to do this, we show them these being signed by an avatar: *JaSigning*. *JaSigning* is an avatar that reads SiGML and plays the signs in it. It is not meant as a final avatar software for this system in particular, but for testing this part of the system.

In order to perform this test, just like in the previous evaluation (Section 5.2), *Google Forms* was used. In order to complete the test, the testees are first given a *Personal Introduction* form to fill, like the one in Section 5.2.1. Afterwards, a quick tutorial of how the test will work is presented to the testee, with an example. Then, 10 pages of *forms* are presented to the testee one after the other. Each page contains either one word or one sentence, shown only in the form of avatar movements with *JaSigning*. For each of the words/sentences, the testees are asked to write what they were able to perceive from the avatar movements (i.e: what was the word/sentence that the avatar was signing). Finally, they evaluate these on readability (related to the avatar movements) and precision (linguistic correctness of the word/sentence itself), and if it was not readable or precise, they write what was wrong with the avatar's sign. These two concepts are explicitly explained to the testees.

5.3.2 Results

There were 3 testees doing this test, all of them experts in the LGP area, but only one of them being a native speaker. For all testee's and words/sentences, there were two main problems. First, *JaSigning*'s speed of signing was too fast, specially for dactilology. The signing speed, however, can be modified within the avatar's interface. The second problem was that the words were annotated with the help of *SpreadTheSign*, and some words were not fully perceivable to the experts, as *SpreadTheSign* does not contain all possible variations of a sign. This makes it so that having annotations being done by non-experts just with the help of *SpreadTheSign* may not be enough. Fortunately, experts can annotate the system through the method evaluated in Section 5.2.

6. Conclusions

The topic over which this thesis picks up on is, to our surprise, very popular among studies. Its importance is backed up by plenty of research around the world, and that plays a big role in captivating

researchers into dwelling onto this topic. Moreover, there is still so much to study for each Sign Language that each country should become busy over them in the next decades: the motivation for Sign Language translation projects is very strong. These systems offer, just like common Spoken Language to Spoken Language translators, a helping hand in short utterances, and not perfect translations over full unpredictable texts and books. This is important for people understanding the pros of researching these systems, and also for evaluating these systems. This system could be practical for this simple usage: the fact that everyone has the ability to annotate words into the dictionary if they want to, without a big learning curve, really makes it a system for everyone to use on all ends. Syntax rules and new vocabulary can be added or deleted at anytime.

Every work done about this topic seems to capture the essence of a step in the right direction, and hopefully, with this thesis, the translation of Portuguese to LGP is now a step closer to its goal than before.

7. Future Work

This project uses an existing avatar (*JaSigning*) for the reproduction of the signs. As such, some possible future work could dwell with the creation of a new and better avatar that can sign the output of this project, with smooth movements and taking a bigger approach on computer graphics.

Besides adding another module to the project, more should be added to the current one. Morphologically speaking, adding a compatibility to all types of subjects will be a good addition to the system. Analyzing more Spacy tags and studying how they can improve our translation is also something that can be done. Since everyone is able to annotate and add syntactic rules to the system, another interesting work that can be done is adding an interface that allows users to easily input new rules into the system, or even to update their system's rules according to someone's rules, creating different rules presets - this could be used for different user's preferences or even different Sign Languages.

Acronyms

ASL	American Sign Language
LGP	Língua Gestual Portuguesa/Portuguese Sign Language
VSL	Vietnamese Sign Language

BSL	British Sign Language
LSF	Langue des Signes Française/French Sign Language
GSL	Greek Sign Language
DGS	Deutsche Gebärdensprache/German Sign Language
XML	Extensible Markup Language
SWML	SignWriting Markup Language
HamNoSys	Hamburg Notation System
MT	Machine Translation
POS	Part-of-Speech
NLTK	Natural Language Tool Kit
NLP	Natural Language Processing

References

- [1] I. Almeida, L. Coheur, and S. Candeias. Coupling natural language processing and animation synthesis in portuguese sign language translation. pages 94–103, 01 2015.
- [2] H. Carmo, M. Moita, and A. Mineiro. Os classificadores da língua gestual portuguesa (lgp): estudo piloto. *Revista Leitura*, 1(58), 2017.
- [3] A. Da Rocha Costa and G. Pereira Dimuro. Signwriting and swml—paving the way to sign language processing. *Atelier Traitement Automatique des Langues des Signes, TALN 2003*, 12 2003.
- [4] E. Efthimiou, S. Fontinea, T. Hanke, J. Glauert, R. Bowden, A. Braffort, C. Collet, P. Maragos, and F. Goudenove. Dicta-sign—sign language recognition, generation and modelling: a research effort with applications in deaf communication. In *Proceedings of the 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies*, pages 80–83, 2010.
- [5] M. Filhol. A combination of two synchronisation methods to formalise sign language animation. In *Proceedings of the 9th international Gesture Workshop*, 2011.
- [6] T. Hanke. Hamnosys – representing sign language data in language resources and language processing contexts. In *LREC*, volume 4, 2004.
- [7] A. K. Joshi, Y. Schabes, L. Lab, A. K. Joshi, Y. Schabes, A. K. Joshi, and Y. Schabes. An introduction to tree adjoining grammars. In *Mathematics of Language. John Benjamins*, 1987.
- [8] E. D. Liddy. Natural language processing. In *Encyclopedia of Library and Information Science*. Marcel Decker, Inc., 2 edition, 2001.
- [9] A. Mineiro and D. Colaço. *Introdução à Fonética e Fonologia na LGP e na Língua Portuguesa*. 2010.
- [10] M. Moita, P. Carmo, H. Carmo, J. P. Ferreira, and A. Mineiro. Estudos preliminares para a modelização de um avatar para a lgp: os descritores fonológicos. *Cadernos de Saúde*, 4 (2).
- [11] M. Moita, P. Carmo, J. P. Ferreira, and A. Mineiro. Phonological description of portuguese sign language for computational modelling purposes. (Eds.) Pawel Rutkowski. *Different Faces of Sign Language Research*.
- [12] M. Moita, P. Moita, P. Carmo, and A. Mineiro. The handshapes in portuguese sign language: phonological study. *TISLR*, 11.
- [13] S. Nascimento and M. Correia. *Um olhar sobre a Morfologia dos Gestos*. 2011.
- [14] T. B. D. Nguyen and T.-N. Phung. Some issues on syntax transformation in Vietnamese sign language translation. *IJCSNS*, 17(5):292, 2017.
- [15] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, pages 311–318, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.
- [16] M. I. Prata. *Mãos que falam*. Lisboa: D.G.E.B., 1980.
- [17] W. 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, 2005.
- [18] L. Zhao, K. Kipper, W. Schuler, C. Vogler, N. Badler, and M. Palmer. A machine translation system from english to american sign language. pages 191–193, 10 2000.