A rule triggering system for automatic text-to-sign translation

Michael Filhol, Mohamed Hadjadj, Benoît Testu

A rule triggering system for automatic Text-to-Sign translation

Michael Filhol, Mohamed N. Hadjadj, Benoît Testu
LIMSI–CNRS
Campus Paris sud, bât. 508, 91400 Orsay
michael.filhol@limsi.fr,hadjadj@limsi.fr,benoit.testu@u-psud.fr

September 20, 2019

Abstract

The topic of this paper is machine translation (MT) from French text to French Sign Language (LSF). After arguing in favour of a rule-based method, it presents the architecture of an original MT system, built on two distinct efforts: formalising LSF production rules and triggering them with text processing. The former is made without any concern for text or translation and involves corpus analysis to link LSF form features to linguistic functions. It produces a set of production rules which may constitute a full LSF production grammar. The latter is an information extraction task from text, broken down in as many subtasks as there are rules in the grammar. After discussing this architecture, comparing it to the traditional methods and presenting the methodology for each task, the paper present the set of production rules found to govern event precedence and duration in LSF, and gives a progress report on the implementation of the rule triggering system. With this proposal, it is also hoped to show how MT can benefit today from Sign Language processing.

Keywords: Sign Language modelling; Machine translation; Text-to-Sign translation.

1 Introduction

The Sign Language (SL) community interested in computer applications has spent significant time in the past decade working on language modelling and animation software implementation. Applications such as machine translation (MT) systems involving SLs would increase accessibility of information to the deaf public.

The problem when addressing MT often comes from the less-resourced status of SLs. The paper elaborates on this in the next section when presenting the different available approaches to the problem, and propose a few choices to tackle the issue. The MT system described is proposed to be based on a grammar of SL production rules, following a methodology and corpus presented in section 3. A new type of MT

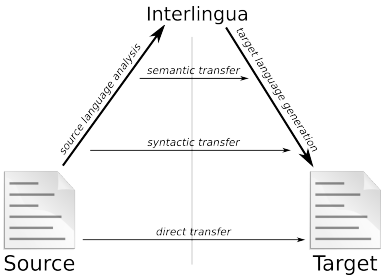

Figure 1: The Vauquois triangle

architecture is then presented, based on rule triggering from targeted text processing, which is discussed in section 4. Both sections report on the progress made, and a concluding section addresses a number of future prospects.

2 Machine translation

There are two methods used for MT in general: rule-based approaches and data-driven approaches. Presently, the latter are clearly the dominant ones when processing written or vocal languages.

2.1 Traditional and present methods

A rule-based approach tries to model linguistic knowledge to formalise rules allowing the processing of data from the input source through more abstract representations and over to the target language. As Vauquois illustrates it (Fig. 1), the more abstract the intermediate representation levels, the further one works from word-to-word translation, the easier the inter-language transfer step, hence probably the better the output [2]. Ideally, an "interlingua" could serve as a language-independent representation half-way through the process (top corner of the triangle), a goal argued to be unreachable.

Most of the MT literature considers text-to-text processes, and very little exists on sign languages. For SL, ViSiCAST is the latest and most significant rule-based achievement [10]. It presents a forward pipeline from English to a chosen SL, and Discourse Representation Theory as intermediate semantic representation. To the authors' knowledge, no full system was actually demonstrated but the full process was described and a semi-automatic prototype was built to cover a small domain. It included an HPSG grammar to support a syntactic level [12]. Prior efforts, including Zardoz [20] and Team [23] have used rule-based approaches, though the data-driven methods seemed to have been favoured since.

Data-driven methods are either example-based or statistical, sometimes hybrids. Both rely on word-aligned bitexts, i. e. parallel texts whose words in one language are identified in the other's word sequence.

Example-based methods employ an analogy principle comparing the input sentence against the bilingual corpus looking for candidate translations. Three steps are gener-

ally planned in a translation: finding the closest matches for the input on the source side of the parallel corpus, retrieving the aligned segments on the target side, and recombining target language segments to build the output translation.

The basic idea of the statistical methods is to maximise $p(T|S)$, the probability that sentence T be a target translation of source sentence S. Applying Bayes's theorem on this conditional probability, it is equivalent to maximising the product of $p(S)$, the probability that S can be found in the source language (a.k.a. language model), and $p(S|T)$, the probability that S is a translation of T (a. k. a. translation model). Both language and translation models are built by training machine learning algorithms on large sets of bitexts. To work properly, the statistical approach therefore necessitates:

(c1) that source and target productions be viewed as sequences of units arranged in segmented sentences;

(c2) large pre-aligned parallel corpora to train the system enough.

Here again, there is limited work to be found on SL data-driven approaches, compared to the large literature on text, though the newer attempts fall mostly in this category, see for example [22, 13, 16, 5, 18]. A hybrid approach was proposed, falling back on example search when pure statistical learning fails [21].

2.2 Our choices for SL

SL corpora are scarce, let alone looking for sufficient parallel data for statistical training. Even if there were enough video or mocap data, the question would remain of what to learn from them. A stream of pixels or body movements would need to be linearised and segmented, and then aligned before being usable for system training, as simultaneous articulators are constantly observable in the data stream. Moreover, bitext alignment is always a tedious task, difficult to afford. It is therefore believed that too many barriers stand in the way to satisfying conditions (c1) and (c2) of §2.1, hence it is prefered to take the path of a rule-based approach.

Almost all known past and current SL description models impose a string of lexical signs in sequence, and whether or not they add syntactic constraints onto the sequence, the sign (gloss) is the basic unit for all production. Also, hands are generally the primary—if not the only—articulators of a Bébian/Stokoe parametric combination [19], which actually discards too much of SL's productive expressiveness. Quite contrarily, the authors have long argued that SLs should be studied as fundamentally multi-articulator languages, whose simultaneous movements and gestures need be synchronised, and this remains a strong requirement in this work. The only multi-linear formalisms, i. e. capable of representing simultaneous movements of many articulators, are P/C [11], a model able to partition the signing activity into simultaneous tracks, and AZee [6], which not only allows partitioning but also flexible generalisation of partition type patterns. This formalism will be used tin this work.

To account for LSF as a language of its own with as little bias as possible, it has been decided to model linguistic rules with no consideration for text in a first place. The following section explains the methodology to achieve this goal. Then, the resulting SL representation system (AZee) can be made the output format of a text translation

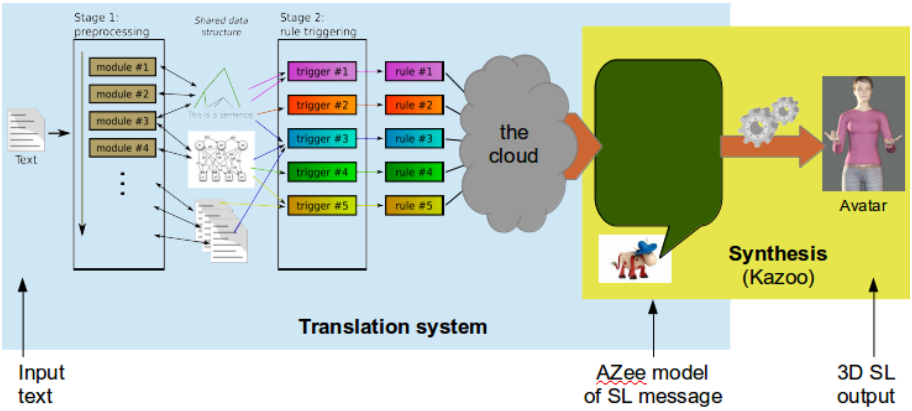

Figure 2: Global view of the translation system architecture

process, as illustrated in Fig. 2. The way the translation process is interfaced with the SL rule system is described later in this article. At this stage, the choice in design is comparable to the aforementioned concept of pivot representation, however this will be discussed in further detail as the comparison calls for some contrast as well.

3 SL rule creation

3.1 The AZee framework

AZee is a framework for describing parametrisable patterns linking signed forms to intentional linguistic functions, where:

- the forms are the visible features produced, i. e. the states, movements and synchronisation of the language's articulators, e. g. eyes closed, head nod preceding lower jaw drop by .1 seconds;
- the function is the interpreted purpose or meaning of the production, which can be of any nature (rhetoric, semantic, lexical, etc.), e. g. depict the path of a moving agent, indicate the place of an event, add intensity to an adverbial.

The AZee methodology can be summaraised as follows. Beginning with either a form or a function feature, a corpus for occurrences of the feature[1] is searched for, and the invariants in the counterpart functions or forms are identified, respectively. When a group of similar features is found, form and function are switched around, to look for occurrences of those features. This procedure is repeated back and forth to refine the features until an invariant in form can be found for all occurrences of an identified function. This raises a "production rule" parametrised by its variable contextual arguments. A production rule is identified by its function and formally

[1] This was done manually in the work reported here.

decribes the form to produce to express it, so it can unambiguously be generated by SL synthesis software with a virtual signer on the output end ("synthesis" box in Fig. 2).

Conversely, a definite function invariably detected for every occurrence of a form criterion raises an "interpretation rule", i. e. one to be triggered in SL recognition tasks. However, this is not relevant to the presented task, in which LSF is the target language, not the source. Note that no one-to-one mapping is required between form and function criteria.

The task addressed here is typically one where linguistics and computer processing overlap. In fact, the function-to-form linking effort may well fully be regarded as Sign linguistics. However, AZee requires neither prior identification nor systematic use of traditionally used language layers. Depending on its function, a rule can pertain to:

- the lexical level, if it describes a stabilised vocabulary element;
- syntax, if its description focuses on the order of the enclosed units;
- SL grammar in a wider (multi-linear) sense, when sequence alone is not sufficient and finer synchronisation rules of overlapping gestural parts are needed;
- discourse structure, if it defines an outline sequence for a full talk and emphasises section titles, adding the right pauses between them.

The working hypothesis of AZee is that a sufficient set of rules can be found to build a full SL production grammar, able to generate any LSF utterance—though at this early stage, it is impossible to really determine how many are needed.

3.2 Example search: durations and precedence of events

This section presents a study carried out recently following the methodology described above.

The corpus mostly used is one created late 2012 with professional translators[2], with the aim of building the first French-LSF parallel corpus [8]. It consists of a set of 40 short texts, each linked to 3 different LSF translations, totalling 120 videos (face and side views) and one hour of journalistic signing. The texts were real-life news, chosen to balance around 20 semantic criteria, hence contains substantial SL material on chronological precedence relations between open-domain events. It is filled with various (French) expressions meaning "A happened before B", "C happened long/shortly/10 mins after D", "E has lasted since F", with lettered elements standing for relative or absolute dates, events or states.

The fact that it is a parallel corpus is not relevant to this particular part of the work as the source texts are not used. However, it is important to note that the videos are not captures of on-the-fly interpretation. They are the result of a prepared translation process by native (deaf) signers. For this paper, translation can be considered as a way

[2] WebSourd® services contracted, `www.websourd.org`.

Figure 3: Still shot of the 2-view video corpus of news items

of elicitating selected aspects of language with non-constructed examples. For evaluation purposes, other SL resources have been used as well, in particular the narrative sections (vs. dialogue turn taking) of the DictaSign corpus[3].

Forms were systematically annotated, following the scheme summarised below, without looking at the text. For each observed articulator, the categories used for annotation are listed. Some categories are missing, e. g. lateral chin movements or eyelids open wide, though it was decided to leave the exceptional occurrences blank and come back and add categories afterwards if needed, to accumulate more video time and articulator annotation. For all articulators, a "standard" category is available, meaning it is in the same state as before even signing:

- eyelids (el): semi-closed, near-closed, closed;
- eyebrows (eb): mod-raised, mod-lowered ("mod" = not fully);
- head (hd) rotation (yes/no/maybe axes): nod-up, nod-down, rot-left, rot-right, tilt-left, tilt-right;
- head displacement (absolute orientation unchanged): fwd, back, left, right;
- eye gaze direction (eg): s-sp (signing space), left, right, up, and so forth;
- manual strokes (wh/sh = weak/strong hand).

Around 70 occurrences of chronological precedence and duration of events were found in the corpus and were subject to analysis. A few relevant forms in the searches conducted were:

- forward movement of a finger-spread hand, starting above the shoulder with a non-simultaneous trill on all fingers and eye gaze directed to the hand (see still in Fig. 4a);
- precedence of the event chronologically occurring first;
- use of the sign shown in Fig. 4b, inconsistantly glossed in dictionaries as "your/his turn", "immediately after" or "consequence".

(a)   (b)

Figure 4: Still shots of signs

Some relevant functions were:

- length of a time duration;
- whether an event occurs during the specified period, before or after;
- if a date is absolute or relative.

The following section presents the findings on event durations and precedence.

3.3 Results

The analysis produced a major finding: a consistent set of six nestable (recursive) production rules, consistent among all signers and whose functions are listed below (r1–6), and whose corresponding form specifications are illustrated in Fig. 5. In the figure:

- the boxes bound the time intervals during which the contained form occurs;
- the arrangement on the vertical axis is arbitrary—the diagrams are not to be read as annotation tiers where one would find one articulator per tier;
- italics in the boxes identify the rule arguments (i. e. boxes to be filled);
- quoted strings refer to frozen dictionary entries;
- *art* : *cat* is a form specification for articulator *art* using category *cat* (cf. §3.2).

The rules established are:

(r1) separation of two events or dates by a period under 10 days (arguments: the event happening first chronologically *pre*, the second event *post*, the optional duration *dur* separating the events);

(r2) chronological sequence (arguments: list of chronologically ordered events, dates or durations);

[3]EU-FP7 project (`www.dictasign.eu`) ended Feb. 2012, corpus available for viewing at `www.sign-lang.uni-hamburg.de/dicta-sign/portal`.

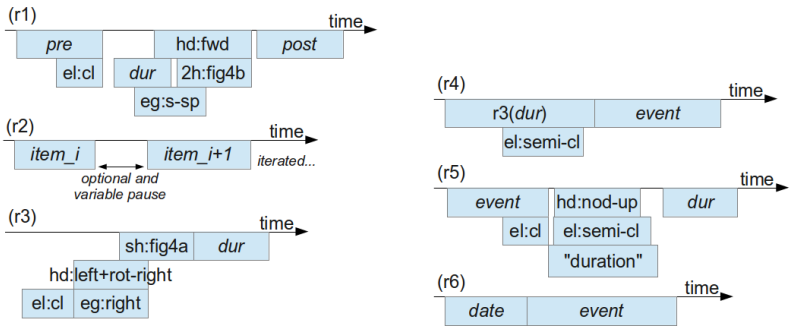

Figure 5: Form production for rules (r1–r6)

(r3) period of at least 10 days (argument: the duration *dur* of the period);

(r4) an event lasts for at least 10 days' time (arguments: the *event* and the duration *dur*);

(r5) an event lasts less than 10 days (arguments: the *event* and the optional duration *dur*);

(r6) dated/time-stamped event (arguments: *event* and *date*).

Here are a few corpus examples of LSF constructions using these rules, and possible English equivalents for each:

(e1) "E_1 three months after E_2": **r2**(E_2, **r3**(*dur*: "3 months"), E_1);

(e2) "E_3 one week before today's election": **r1**(*pre*: E_3, *post*: **r6**(*event*: "election", *date*: "today"), *dur*: "1 week");

(e3) "they were held as hostages for 3 days just after they left France": **r1**(*pre*: "left France", *post*: **r5**(*event*: "hostages", *dur*: "3 days")).

3.4 Discussion

In the videos, whether translated from French sentences using *avant* (before), *après* (after) or any other construction introducing precedence, it is found that times and events are generally signed in an order that is congruent to their chronological order (r1, r2). This confirms a result already present in the literature [4], namely the systematic preference for the chronological order when expressing time relations between events and dates.

Another expected observation is that when an event is dated, the date was always signed immediately before the event (r6). This holds in either case of an absolute or a relative date, and regardless of the chronological rank in event sequences. No use of the documented [4] transverse time axis for absolute dates (the horizontal axis normal to the sagittal plane) has been detected; all chronological arrangements were performed

along the sagittal axis. But the corpus containing only real news events, all texts had a link to the time of speech. Therefore it is not argued that absolute dating can take place on the transverse axis, though it does amend Cuxac's finding by suggesting that even though absolute, dates are rather signed along the sagittal axis if in the same sequence as a relative date. Besides, this excludes the time sequences referencing the future.

However, a more surprising result emerged. For all expressions of durations and whether or not there is an event taking place before, during or after it, there is a clear distinction in the signed form based on the duration length itself, drawing a line at about 10 days of absolute time. All expressions of durations exceeding this time lapse involve the form shown in Figure 4a (r3), whether it is a period of separation between events (r3 used in an r2 sequence) or a period during which an event is taking place (r4, which calls r3). The difference in form is in the semi-closed eyelids over 4a in the latter case. All durations shorter than 10 days are expressed differently. For these shorter periods, durations *of* events are expressed with a lexical sign meaning "duration" (r5), while durations *between* events make use of the sign shown in Figure 4b (r1).

The first consideration on this 10-day distinction was that it pertained less to the absolute duration than it did to some form of judgement on the duration. However the study carried out contradicted this hypothesis, and ruled out the use, say, of r1 in the case of month- or year- long durations even if judged short, as well as r3 for periods of a few days or hours, even when interpreted as long.

Also surprising is that precedence is ruled with two argument events in the case of short durations of separation (r1), though has no counterpart rule for the case of longer separations. Corpus observation has revealed no specific form structure for sequences of two events separated by a long period. Only the time sequence (r2) has surfaced as a useful production rule, itself capable of enclosing an r3-generated expression of a (long) duration. This shows how important it is to write SL rules with no preconception of a syntactic or semantic structure, as in all likelihood no semantic representation or graph system would have made such difference.

Of course, some form features fall out of this observation, in particular when related to the dynamics in the changes. All measurements here are still made manually by means of frame-by-frame viewing and video annotating, thus rather qualitative, and very little on dynamics can be properly described with such tools only. Motion capture recordings could achieve great precision in measuring movement velocities and accelerations, as well as lengths of pauses and holds, which all may well participate in a rule's form specification. A lot of additional information from technologies such as mocap is expected in the future.

Encoding these AZee rules is useful for Sign synthesis, however only represents half of the translation process from text, which the next section continues on to address.

4 An MT architecture for SL

The MT process is represented by the left-hand box in Figure 2, and consists in generating AZee output from input text. First, the main proposal is presented (§4.1), which is discussed afterwards (§4.2). Then, the various implementation choices are presented (§4.3), and an additional section (§4.4) reports on the progress thereon.

4.1 The proposal: rule triggering

Rule entries represent functions interpreted on the receiver's side. As a consequence, they are often semantically loaded and can offer quite an abstract level of representation of the discourse meaning and even rhetorics. However, unlike semantic graphs or first-order logic predicates, AZee rules are meant to remain language-specific. These two properties lead to the proposal that the set of rules established serve directly in place of a pivot in the presented rule-based MT system, following the idea that every function can be recognised from a detection process on the input text, thereby making the corresponding rule a good candidate for a correct translation.

This breaks the full problem down into as many text processing problems as there are rules available, each established rule's function creating an information extraction problem. The translation problem becomes one of detecting the established SL rules' functions in the input text. In other words, for each rule, one must look for forms in French that bear the function that defines the SL rule. The production of the SL output forms is a non-linguistic task as it is precisely what the invoked rule describes, and what made the function available in the first place.

To implement the framework, an NLP[4] module can be programmed for every production rule's header function, triggering its rule for the AZee output when detecting the right textual forms. Each triggering module will run separately, and may be more or less complex. To serve its purpose, it might call for any number of helper tools such as syntactic parsing, anaphor resolution, simple lexical or even trivial typographic matches, etc.

Once the rules are triggered separately, they still have to be arranged in a single utterance representation in a last stage. Comparably to example-based MT where retrieved segments must be combined to build a final translation proposal, rules triggered here will not fully be nested, different parts being isolated and still having to be combined. This last combining stage is represented by the multi-input and single-output cloud in Figure 2. This issue is put aside for now, and will be discussed in the prospects section.

4.2 Discussion

Definitely rule-based, the proposed framework is not trivially comparable to the pipeline Vauquois describes. Taking a step back on the proposed scheme, this is discussed in the present section, and will be referring to the diagram in Figure 6.

The most abstract elements in this proposal are the functional entries described by the AZee rules combined in the production, which makes AZee the top-most corner of a Vauquois-type illustration. However, they are built exclusively from SL corpus analysis, regardless of text or indeed of any other language or translation purpose. Once the AZee representation is reached, the composition is therefore already "fully Sign", so it is placed on the right-hand (SL) side of the diagram.

It has been seen that on the SL side, descriptions are not arranged on a stack of levels. The AZee animation system will synthesise the sign form described by any rule combination, usually falling through recursive rules all the way down to the articulator

[4] "Natural language processing", used here in its traditional sense of "text processing".

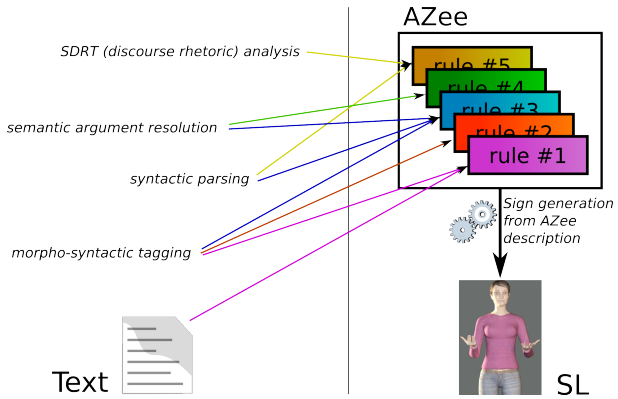

Figure 6: The AZee translation diagram

movements (cf. phonetics). This explains why no stages are represented in the synthesis process on the right-hand side of the diagram. It is to be considered as a whole, with high-level entries chosen as input, and low-level articulation/animation as output.

On the text side, linguistic layers can be identified, in particular by the tools analysing the source, though they do not have to be taken in the same order or up to the same level for every rule. Also, rules may need to process data on more than one identified level. This is represented on the diagram by the set of arrows pointing rightwards across the vertical language separation line. Theoretically, the starting point of these arrows could be located anywhere on the vertical axis, even higher than the AZee target level itself, but no reliable text processing tool really allows that yet. The result is a set of up&rightward paths from many (rather lower) layers of French to high-level SL-specific rules. In this sense, the approach presented in this paper is compared to a transfer approach, and more specifically to a "multi-level ascending transfer" [2]. There is no language-independant representation or interlingua; rather, a multi-rule and multi-layer transfer is implemented, producing a set of fully-SL rules triggered and to be combined.

With a traditional rule-based approach, more and more abstract representations are produced as the text is processed, to be rephrased in SL in a second step. On the contrary, the proposed framework searches in the text for patterns, clues, forms, etc. whose functions it already knows the SL forms for. In other words, instead of a forward pipeline approach taking steps towards the right, upwards first and then downwards, this proposal is built backwards from existing candidate SL rules to text processing tasks. The two-fold benefit is:

- no distinctions or categories present in the source text are carried "rightwards" if they are not relevant on the SL side;
- no ambiguity can be transfered over if the SL grammar makes a distinction between two (or more) cases.

This is an original target-driven approach to rule-based machine translation, which perfectly recalls that professional translators unanimously prefer translations performed

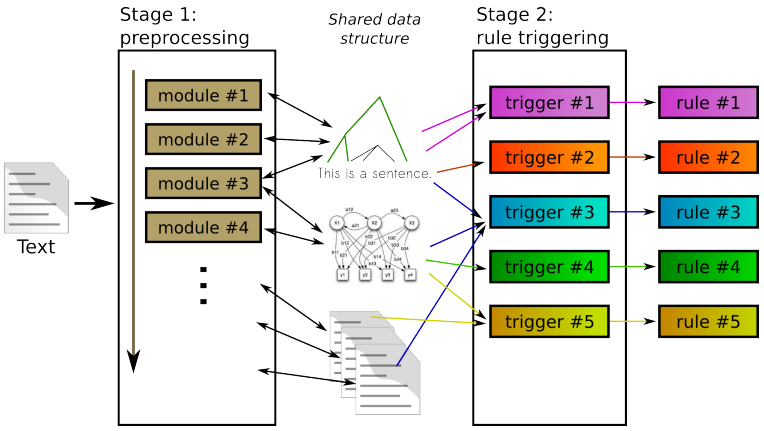

Figure 7: The text processing software architecture

into a native language. One performs better translation if taking foreign input and producing output with a target-language mind set. Similarly, the system runs with AZee rules in memory, and recognises in the input text the forms it is prepared to produce in a "perfectly Sign" way.

4.3 Implementation

This section presents the work carried out with regards to the text processing, i. e. rule triggering, side of the system. It is the part of the system whose implementation was initiated most recently, as before producing anything from the text, the point was to have a linguistically informed SL animation system with no concern for—hence no bias from—written language lexicon, clause order or linear structure.

The architecture consists of a list of rule triggering processes, one for every available target SL rule. Of course, factorisation of common processing tasks eventually becomes useful. For example, morpho-syntactic types (POS tags) are obviously useful for many purposes, so a POS-tagging tool can be run in an earlier preprocessing stage, whose output can serve different modules in the triggering stage. To be able to add new triggering (and preprocessing) abilities easily as time goes, a modular plugin system as illustrated in Fig. 7 was implemented.

When executed, the program first runs all available preprocessing (stage 1) modules. These save useful output in a shared and easily explorable data space (in the middle in the figure), either saving text files or extending the pre-existing multi-tag&tree structure. Initially, this structure contains the list of the lexical units of the input text, which tag-oriented modules can label with attributes and values and which tree-building modules can use as leaf nodes. Any sort of additional data file can be stored as well, for the purpose of specific rules.

Stage 2 comes next, where all available rule triggering modules are run. Each stage 2 module searches through the shared data space or through the text itself for clues that its rule is to be triggered, i.e. considered for use in the output translation. A

rule can of course be triggered several times for a single text, or none at all, and can populate some of its arguments if it is able. Every trigger is associated with a reason for the trigger and a reliability mark.

Though preprocessing is run forward from the text before knowing what rules are to be triggered afterwards, it is used for cases where the preprocessing is needed by several triggers, to avoid running duplicate code when running the triggering stage. In a sense, it can really be regarded as mere code factoring; no preprocessing exists if not to serve at least a rule trigger. Thus the whole architecture remains driven by the AZee rule set, which is a SL model exclusively, from the input text to the SL animation.

4.4 Progress

All this has recently been implemented, and is bound to undergo further development, however the architectural diagram drawn in Figure 7 is ready and modules can already be plugged in or out of the system for both stages. Today, the system is composed of 5 stage 1 modules, and of most triggers for the rules described in §3.3, plus one for the "open list" function. This rule was added first, because it was already available and well established. It is renamed from the earlier published "enumeration" rule [7], as strictly speaking enumerations may have different types, and the authors believe it only covers lists to which more items can be appended.

The five pre-processing modules implemented are: "tree tagger", "XIP", "enum", "Wmatch-timer" and "time seq graph". The first one, "tree tagger", performs an external call to the tool it is named after, which provides with the POS tags for the sentence tokens [17]. XIP, the Xerox® Incremental Parser, is another external tool that builds syntactic trees for the input sentences [1]. As these tasks are very often needed, their place is in stage 1. The last three are explained later, where useful. The following describe the trigger modules implemented in the software, the intention being to cover the largest number of production rules rather than to compete with the state of the art in every NLP sub-task addressed.

The "open list" trigger mostly relies on typographical and lexical cues, and includes some syntactic analysis as well. Lists in French are often strings of comma-separated elements of the same syntactic type (nouns, verb phrases...), followed by the last item introduced by the conjunction "et" or "ou". Typography also helps to detect parenthesised or colon-initiated lists, and ellipses. Lexical checks are useful for including patterns such as example (e4) below, and for excluding example (e5). Case (e6) can be ruled out with a semantic hyponym check.

(e4) *W. Pickett, qui avait signé des titres comme "In the midnight hour" ou "Mustang Sally" est décédé jeudi.* – W. P., author of songs like ... or ..., died on Thursday.

(e5) *Le gouvernement a annoncé ses trois priorités pour l'année : la recherche, l'éducation et la culture.* – The government announced its three priorities for the year (to come): public research, education and culture.

(e6) *Deux personnes, un Français et un Britannique, ont été arrêtés hier soir.* – Two people, one French and one British, were arrested last night.

There is no enumeration pattern in (e4) per se, though the "like ... or ..." construction has the function of an open list, which should trigger the same LSF production rule. On the contrary, (e5) and (e6) have an enumeration pattern, but should not be included. The introductory clause in (e5) contains the lexical unit for "three", giving away a fully-exhaustive list after it, and while (e6) is theoretically ambiguous, the hypernym status of the first item related to all others combined with a lexical count as in (e5) favours a closed list over an open one.

Except for this last (e6) type of criterion, everything here is implemented. Preparing for the other types of lists (e. g. lists of mutually exclusive options, revealed in the DictaSign project under the name "alternative constructions"), textual detection of enumerations was moved into a pre-processing module named "enum", whose output is a file with all detected lists of comma-separated items, tagged with a few markers such as the conjuntion used before the last enumerated item. This way, the trigger to come for the "option list" rule will share part of its processing with the "open list" one. Each trigger module simply filters out the enumerations it is not concerned with, and performs any additional search needed to find textual patterns that are not enumerations, e. g. like in (e4). Similarly, modules were written for most rules of section 3.3. Rules (r1)–(r5) need duration detection, and rules (r1) and (r2) are about sequences, which calls for two preprocessing modules, respectively "Wmatch-timer" and "time seq graph", the latter still under development.

Wmatch is a semantic parser engine for text, developed at LIMSI [15]. It allows custom grammars, recognises sequences of lexical units or subtrees and builds partial semantic/syntactic trees over the input. It was used to build the preprocessing module "Wmatch-timer", which detects occurrences of durations, as well as binary syntactic connectors like "trois jours avant" (three days before). Using its output and more syntactic analysis, another pre-processing module "time seq graph" detects sequences of times and events, and each item's duration when possible. As the output form is still under implementation, details about it will not be discussed in the context of this paper. The idea is to help rules (r1) and (r2), both requiring sequence detection.

Here are the textual clues that each trigger module uses for the time being:

(r1) Look in output of "time seq graph" for two consecutive nodes separated by a short duration.

(r2) Build the sequence from the output of "time seq graph", triggering (r4) to include long durations of separation and (r1) when two consecutive nodes are separated by a shorter longer duration.

(r3) Nothing triggers this rule alone; it is indirectly triggered by (r2) and (r4) modules when needed.

(r4–5) These rules use the output of "Wmatch-timer", additional syntactic clues and a duration check.

(r6) Not implemented yet. The plan is to use mostly syntactic clues to start, and an already available Wmatch grammar recognising date patterns.

Precision and recall values for a few of the implemented modules are already being measured. However, evaluating each subtask independently will not assess the proposed architecture as a whole, and the authors have no intention of challenging the NLP state of the art for the individual modules (they would rather collaborate with text NLP experts and associate them to the SL translation field). Fully aware that it will become a crucial question soon enough, this is listed as future work and will be discussed in the last section.

4.5 Example

Let us assume two additional AZee rules, one named "lex-school" whose form description is that of the dictionary sign meaning "school" (*école* in French), and a similarly lexical rule "lex-education". Both corresponding trigger modules are assumed to detect any occurrence of *éducation* or *école*, as given the frequent metonymic use in French for one another, each SL form is a candidate translation for either term. This section describes what happens when sentence (e5) is input to the system, implemented as proposed.

First, stage 1 runs its modules, namely:

1. "tree tagger", which populates the data structure with POS tags on the sentence (*gouvernement*/noun, *pour*/preposition, etc.);
2. "XIP", which tags syntactic groups and dependencies (e. g. *la recherche* = noun phrase);
3. "enum", which detects the final colon-initialised and comma-separated sequence of chunks of same syntactic nature and marks it for future look-up;
4. "Wmatch-timer", which has no output here as no expression of duration can be found;
5. "time seq graph", which has no output here either as no unit meaning or implying "before" or "after" exists in the input.

Then, stage 2 triggering modules all run independently. Here are different types of behaviour that will occur in this stage:

- true positive for "lex-education": the simple criterion for this rule (presence of the word *éduction*) works well here as its form is the one to choose for translation;
- false positive for "lex-school": for the same reason as "lex-education", this rule will trigger whereas it is not needed for translation in SL;
- true negative—two example cases of correct non-triggers:
  - (r1): nothing found in the output of "time seq graph", hence nothing triggered;

– "open list" rule: looking at the shared data structure, this module will locate the text enumeration and check for non-exhaustiveness of the list, which fails;

The final not yet implemented step left to be designed is the combination of all triggered rules into one consolidated output tree structure representing the full SL utterance. Positional information in the source text is essential. Indeed, an inclusion criterion will manage to instantiate the arguments of the three-item (closed) list detectable in (e5): the text enumeration and item boundaries correspond well to those detected by the lexical rules respectively triggered by *recherche*, *éducation* and *culture*. However, this is only a first hint on how to approach rule combination, as it remains clear that a lot more work will be needed to achieve this goal. For example, several triggers may exist for an identical text input (e.g. *éducation* above), each potentially representing a different translation strategy, or simply wrong in the given context (e.g. "lex-school"). If no human translator is in the loop, automatic selection remains a challenge.

5 Conclusion and future work

After showing limitations to data-driven methods for SL translation, i. e. lack of segmented data and linearity, an original rule-based MT architecture has been suggested and the initial stages of its implementation have beed presented. It is based on a SL model allowing multi-linear formalisation of SL-specific structures without influence from text structure, after thorough corpus analysis. These rules drive the translation, in the sense that it is the rules' functions that tell the system what to look for in the text. We have started with open lists and a subset of rules found to be governing event, date and period precedence as well as event durations. Incidentally, the authors believe this architecture could be explored for other language pairs, even in the well-addressed field of text-to-text translation.

After implementing enough rules, it is necessary to think of a way to evaluate such a system. Currently, the way to evaluate an output translation is either to measure similarity between the output and a set of human-provided references with metrics such as BLEU [14], or to ask for human assessment of the output, either with judgements on criteria including adequacy and fluency, by ranking translations or by modifying the output with as few steps as possible to reach a satisfactory translation (HTER). However these techniques are based on lexical segmentation and sequence, and statistical approaches were discarded for the same reason. Moreover, full output translations are not produced, rather SL rule triggering from text analysis, while all the methods above assume fully generated sentences to operate on. This in fact raises a new challenge, pertaining to a full research field: MT evaluation. Hence, the proposed evaluation protocol will need time and work to be established, and the question may even be premature before progress on the missing final stage of the system (triggered rule combination).

Before concluding, it would be useful to address the problem of rule combination through an interesting prospect. As things stand, one ends up with a set of triggered rules, unsure of how they are to be nested under one another. While full automatic translation, i. e. comparable to human production, is not foreseen any time soon, an

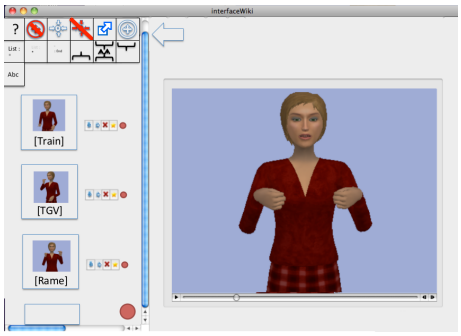

Figure 8: Sign wiki interface

application of this work could be to assist human translation, leaving the final arrangement task to the translator, for example through an interface inspired by DictaSign's Sign wiki prototype [9].

With such wiki interface, the user could edit/create SL animations by manipulating items taken from a set of basic linguistic functions (top left-hand corner icons in fig. 8), all linked to a production rule in this paper's terms. He could insert or remove signs, and arrange the hierarchy of rules in the left-hand pane to ultimately build a full AZee-based representation of the desired output, which could be generated in one click and animated in the right-hand pane. The application prospect here is to propose, after processing an input text, an initial set of production rules whose functions (and arguments when possible) are identified in the text, more or less still to be combined and filled with missing contents, and relieved from false positives.

This basically merges the final arrangement task with the professional human post-editing process that would be necessary anyway. Working on text, professionals already use the help of translation memory managers to easily recall previously translated equivalents for repeated text patterns. Comparably, the presented proposal can assist human translators with pre-formatted pieces of output from a text processing stage. Such project would have to include the participation of translators, for example to poll them about the type of interface they would find helpful, or about their preference for recall- vs. precision-oriented rule triggers. Indeed, it is not obvious whether throwing out superfluous suggestions is more comfortable or time-saving than trusting most of the output, even if it means looking for the missing pieces by hand. This is a very exciting prospect for everyone involved in the work presented in this paper, as it will be the first to explore the potential benefit of SL MT in the translation service industry.

References

[1] Aït-Mokhtar, S., J. P. Chanod, and C. Roux. 2002. Robustness beyond shallowness: Incremental deep parsing. *Natural Language Engineering* 8: 121–144.

[2] Boitet, C. 2003. Automated translation. *Revue française de linguistique appliquée (in English)* 8: 99–121.

[3] Braffort, A., M. Filhol, L. Bolot, M. Delorme, C. Verrecchia, and A. Choisier. 2013. KAZOO: A sign language generation platform based on production rules. In *Proceedings of the sign language translation and avatar technology workshop (SLTAT)*.

[4] Cuxac, C. 2000. *Langue des signes française, les voies de l'iconicité*, Vol. 15–16. Ophrys.

[5] Dandapat, S., S. Morrissey, A. Way, and M. L. Forcada. 2011. Using example-based MT to support statistical MT when translating homogeneous data in a resource-poor setting. In *15th annual meeting of the european association for machine translation*.

[6] Filhol, M. 2012. Combining two synchronisation methods in a linguistic model to describe sign language. *Gesture and Sign Language in Human-Computer Interaction and Embodied Communication, Springer LNCS/LNAI* 7206.

[7] Filhol, M., and A. Braffot. 2010. DictaSign deliverable D4.2: Report on the linguistic structures modelled for the Sign wiki. DictaSign project, EU-FP7.

[8] Filhol, M., and X. Tannier. 2014. Construction of a French–LSF corpus. In *Language resources and evaluation conference, building and using comparable corpora*. Iceland.

[9] Glauert, J., et al. 2012. DictaSign deliverable D7.3: Sign wiki prototype.

[10] Hanke, T., et al. 2002. ViSiCAST deliverable D5-1: interface definitions.

[11] Huenerfauth, M. 2006. Generating american sign language classifier predicates for english-to-ASL machine translation. PhD diss, University of Pennsylvania.

[12] Marshall, I., and É. Sáfár. 2004. Sign language generation in an ALE HPSG. In *Proceedings of HPSG04*.

[13] Morrissey, S. 2008. Data-driven machine translation for sign languages. PhD diss, Dublin City University.

[14] Papineni, K., S. Roukos, T. Ward, and W. J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *40th annual meeting of the ACL proceedings*.

[15] Rosset, S., O. Galibert, G. Bernard, E. Bilinski, and G. Adda. 2008. The LIMSI participation to the QAst track. In *Working notes of CLEF workshop*.

[16] San-Segundo, R., R. Barra, R. Córdoba, L. F. D'Haro, F. Fernández, J. Ferreiros, J. M. Lucas, J. Macías-Guarasa, J. M. Montero, and J. M. Pardo. 2008. Speech to sign language translation system for spanish. *Speech Communication* 50/11-12.

[17] Schmid, H. TreeTagger: http://www.cis.uni-muenchen.de/˜schmid/tools/TreeTagger.

[18] Stein, D., C. Schmidt, and H. Ney. 2012. Analysis, preparation, and optimization of statistical sign language machine translation. *Machine Translation* 26:4: 325–357.

[19] Stokoe, W. 1960. Sign language structure: an outline of the visual communication system of the american deaf. *Studies in linguistics, occasional papers* 8.

[20] Veale, T., A. Conway, and B. Collins. 1998. The challenges of cross-modal translation: English to sign language translation in the ZARDOZ system. *Machine Translation* 13.

[21] Vertan, C., and D. Karagiozov. 2012. Deliverable D6.1: Machine translation in Atlas.

[22] Wu, C. H., H. Y. Su, Y. H. Chiu, and C. H. Linal. 2007. Transfer-based statistical translation of taiwanese sign language using PCFG. In *ACM transactions on asian language information processing (TALIP)*.

[23] Zhao, L., K. Kipper, W. Schuler, C. Vogler, N. Badler, and M. Palmer. 2000. A machine translation system from english to american sign language. *Association for Machine Translation in the Americas*.