



SIGNON

Sign Language Translation Mobile Application and Open Communications Framework

Deliverable 5.6: Final Sign language-specific lexicon and structure



Project Information
Project Number: 101017255
Project Title: SignON: Sign Language Translation Mobile Application and Open Communications Framework
Funding Scheme: H2020 ICT-57-2020
Project Start Date: January 1st 2021

Deliverable Information
Title: Final Sign Language-specific lexicon and structure
Work Package: WP5 - Target Message Synthesis
Lead beneficiary: UPF / GTI
Due Date: 31/12/2023
Revision Number: V0.1
Authors: Dimitar Shterionov, Irene Murtagh, Bram Vanroy, Josep Blat, Víctor Ubieta Nogales
Dissemination Level: Public
Deliverable Type: Other

Overview: In this deliverable we describe the final sign language (SL)-specific lexicon and structure together with the final pipeline for synthesis of SL avatars for the SignON project.

Revision History

Version #	Implemented by	Revision Date	Description of changes
V0.2	Dimitar Shterionov, Irene Murtagh, Bram Vanroy, Josep Blat, V́ctor Ubieto Nogales	19/12/2023	After comments from review are addressed
V0.1	Dimitar Shterionov, Irene Murtagh, Bram Vanroy, Josep Blat, V́ctor Ubieto Nogales	13/12/2023	First draft

The SignON project has received funding from the European Union’s Horizon 2020 Programme under Grant Agreement No. 101017255. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the SignON project or the European Commission. The European Commission is not liable for any use that may be made of the information contained therein.

The Members of the SignON Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the SignON Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Approval Procedure

Version #	Deliverable Name	Approved by	Institution	Approval Date
V0.1	D5.6	Aoife Brady	DCU	15/12/2023
V0.1	D5.6	Marco Giovanelli	FINCONS	19/12/2023
V0.1	D5.6	Vincent Vandeghinste	INT	15/12/2023
V0.1	D5.6	Adrián Núñez-Marcos	UPV/EHU	15/12/2023
V0.1	D5.6	John O’Flaherty	MAC	15/12/2023
V0.1	D5.6	Santiago Egea Gómez	UPF	15/12/2023
V0.1	D5.6	Irene Murtagh	TU Dublin	15/12/2023
V0.1	D5.6	Lorraine Leeson	TCD	16/12/2023
V0.1	D5.6	Jorn Rijckaert	VGTC	15/12/2023
V0.1	D5.6	Henk van den Heuvel	RU	15/12/2023
V0.1	D5.6	Catia Cucchiarini	TaalUnie (NTU)	15/12/2023
V0.1	D5.6	Myriam Vermeerbergen	KU Leuven	16/12/2023
Vx.x	D5.6	Rehana Omardeen	EUD	xx/xx/202x
V0.1	D5.6	Mirella De Sisto	TiU	18/12/2023

Acronyms

The following table provides definitions for acronyms and terms relevant to this document.

Acronym	Definition
AMR	Abstract Meaning Representation
AU	Action Units
BML	Behaviour Markup Language
ML / DL	Machine / Deep Learning
RRG	Role and Reference Grammar
RB	Rule-Based approach
MF	Manual Features
NMF	Non-Manual Features
ISL	Irish Sign Language
NGT	Sign Language of the Netherlands (Nederlandse Gebarentaal)
VGT	Flemish Sign Language (Vlaamse Gebarentaal)
PoS	Part Of Speech
SIGML	Signing Gesture Markup Language
MediaPipe	A tool for pose estimation

Table of Contents

1. Introduction	6
1.1 Keypoint frames	6
1.2 Moving forward	7
2. Generating glosses	8
2.1 Rule-based glossing: open vocabulary	8
2.2 AMR-based glossing into a closed vocabulary	8
2.3 Evaluation and conclusion of glossing methods	10
2.4 Gloss coverage	10
3. Animics	11
3.1 Blendshapes to Action Units	11
3.2 Sign Clips, SiGML and BML	14
3.2.1 Supporting generated SiGML repositories	15
3.2.2 Extending BML, Supporting SiGML, Realisation	15
3.2.3 Clips Available in Animics; SiGML-BML Correspondence	16
4. Conclusions and future work	39
5. References	40

1. Introduction

This deliverable describes the progress on the sign language lexicon and structure along with the processes that take place to use or produce these. This deliverable expands the work summarised in *D5.5 “Second Sign language-specific lexicon and structure”*. Since D5.5 we shifted focus from the Sign_A + RRG linguistically informed lexicon implementation towards a symbolic approach, which provided a more suitable fit for the goals of the project. The benefits to this approach, as outlined in D5.5, is that the lexicon semantic representations can be derived independently from a syntactic representation and the lexicon can be generated through an adaptation of the mBART model, which drives the InterL-E within the NLU pipeline of the project. As such in this deliverable we present the updates of the continuous work on the AMR-based synthesis approach and SignNets and stresses on the work on the Animics system which is the culmination of our (user-oriented) synthesis approach.

This deliverable builds on and relates to:

- D5.5 “Second Sign language-specific lexicon and structure”, as noted earlier, which presents an overview of the previous version of our SL lexicon and structure;
- D1.12 “User Generated Data” where the Animics system is described as a facilitator of user-generated content for SL synthesis;
- D4.1 “First symbolic intermediate representation - InterL-S” and D4.2 “Second symbolic intermediate representation - InterL-S” which cover the AMR approach.

1.1 Keypoint frames

In Section 2.2, Section 2.3 and Section 2.4 in deliverable D5.5 *“Second Sign language-specific lexicon and structure”* we presented work on the generation of lexical entries (signs) composed of keypoints. This work was a shift from the symbolic, linguistic approach of Sign_A and RRG to a data-driven approach. In particular, videos were converted into sequences of frames containing keypoints extracted using MediaPipe. Such entries form a vocabulary for building sequences of signs to then be used in avatar synthesis. Despite the potential of this system (discussed in D5.5), two issues arose: (i) the significant data size of the captured and saved structures (approximately 10x larger than the original videos)¹ and

¹ The .pose format, introduced in the sign-language-processing code repository (<https://github.com/sign-language-processing/>) and used within the WMT-shared task on SL translation, partially alleviates this issue.

(ii) the low accuracies in the ML pipeline of the avatar creation system² of converting the keypoint frames into avatar animations. However, (ii) has a bigger impact on the synthesis pipeline. As such, the work on providing synthesis support for a pipeline where landmarks would be obtained from recognition processes has not been pushed as much as the support for symbolic synthesis. The prioritisation of the latter was already discussed within D5.5, in relation to the foreseen output towards the pipeline of the recognition systems.

1.2 Moving forward

Since D5.5, we focused on (i) the gloss-lookup approach, which is based on a lookup strategy but in the NGT SignBank (VGT also SignBank is also supported), rather than in local databases (as with the keypoint-frame-based repository) and can be linked to the SiGML representations and (ii) using the Animics system.

The Animics system, as discussed in Section 3 and in D1.12, taps into the idea of video-to-keypoint-frames along with a wide range of other functionalities which allow us to exploit the benefits of an ML pipeline and symbolic approaches. The Non-Manual Features (NMFs) integration is based on MediaPipe, as it happens with the previous work on Manual Features (MFs), which facilitates a more compact system.

The ML pipeline converts keypoint frames into avatar animations through a neural approach. This imposes the need for a frame-by-frame work, leading to the pipeline being less scalable.

On the other hand, this pipeline is fully integrated within Animics and with the Performs realiser (see Deliverable *D5.2 Final version of virtual character*).³ This way, both pipelines can be combined to produce Sign Language output by means of avatars, and the system is thus fully operational.

² The ML pipeline/approach (see also Section 3) is one of the approaches for generating avatar movements. It employs Mediapipe followed by our own deep neural network to convert input of video sequences to keypoint frames subsequently turned into skeleton joint rotations driving avatar movements/animations which can be retargeted. This has been outlined in deliverable *D5.3 “Interactive Co-creation Web-based Platform for Learning from User Input”*, where its use was limited to Manual Features (MFs). Since then, it has been extended to Non Manual Features (NMFs) We also would like to clarify that the ML approach is usable for synthesis of SL utterances as outputs and for converting into animated signs or animated signed utterances to be edited in the SignON avatar editor (currently renamed into Animics).

³ Animics and Performs are the new names of the SignON virtual character editor and the SignON realiser.

2. Generating glosses

The avatar is compatible with a wide range of input formats, one of which is glosses. This is discussed in more detail below. To generate glosses we built two systems that are capable of converting text into glosses. One is a linguistically motivated, rule-based pipeline, and the other is a complex pipeline based on AMR. Both of these have been discussed in more detail in D4.1 “*First symbolic intermediate representation - InterL-S*” and D4.2 “*Second symbolic intermediate representation - InterL-S*” (among others) respectively.

2.1 Rule-based glossing: open vocabulary

In brief, the rule-based approach (RB) requires a Dutch text as input which is lemmatised. The lemmatised forms are then re-ordered based on linguistic rules that have been uncovered in related research and through collaboration with sign linguists in the consortium (see D3.3 for details). Additional preprocessing steps are also included to, for instance, remove certain words and to replace specific word classes (like word nouns) with specific glosses. The result is therefore a pseudo-glossed text: the glosses are, in essence, a modified lemmatised form of the input but no assurance is given whether these glosses are actually recognised glosses used by linguists. Furthermore, there is no guarantee that these are available in lexical databases. In other words, such pseudo-glosses may or may not be part of the established lexicon. Furthermore, they stick very close to the input text (through lemmatisation), while meaning transfer through translation is typically preferred to be a more abstract and less literal process. Finally, this pipeline was built for Flemish Sign Language (VGT) and, while work was planned for Sign Language of the Netherlands (NGT), this work was not completed due to a change in staff. So while RB is intuitive and relatively straightforward to implement in a single pipeline, it requires Dutch input, its generated (pseudo-)glosses are not necessarily established, and its output may be a too literal translation of the input.

2.2 AMR-based glossing into a closed vocabulary

Due to the limitations in the RB system,, another approach was explored, especially with a focus on synthesis. Because the avatar supports a limited vocabulary of glosses, we looked into how we can generate glosses in a constrained lexicon. To do so we utilised existing SignBanks for NGT and VGT (giving support for both those languages), which contain an overview of the established annotation lexicon used by linguist experts. These SignBanks contain, crucially, glosses as well as potential word translations in

Dutch of those glosses. Therefore, we attempt to use the NGT and VGT SignBanks as a look-up dictionary, where a word can be linked to a gloss. This approach would ensure that we only generate glosses from a closed, established lexicon. Importantly, the SignBanks are not identical to the lexicon that the avatar supports, but we will come back to that.

Additionally, we also wanted to make this pipeline available for other input languages besides Dutch, and to move away from a literal translation in favour of a more meaning-focused transfer. We discussed this pipeline in full in D4.2 before and discussed our approach in D5.4 and D5.5; for completeness we summarise it here but refer the interested reader for more details to the aforementioned deliverables. In D5.5 we outline two challenges related to RRG + Sign_A: (i) the lack of automated approach to generate RRG logical structures for every language and (ii) the complexity of creating Sign_A representations. With respect to (ii) we presented in D5.5 an automated system to generate Sign_A entries (based on keypoints). However, the results of this system need to be manually corrected and edited which imposes additional overhead without the proper interface. Both of those challenges were tackled by using abstract meaning representation (AMR; Knight et al. 2020) , which is a semantic framework for representing (extracting) the meaning from an input sentence. This representation is a graph structure where key concepts (semantic frames) and relationships are given. To this end, we built multilingual and monolingual systems that can automatically parse given input in English, Spanish and Dutch to AMR. Irish (GA) was not included because, while our base model did not include that language, it is also the case that mouthing in ISL is based on English,⁴ although we do release synthetic text-to-AMR datasets for all languages, including Irish (GA), for research.⁵

While our systems can work with multiple input languages, it should be noted that AMR has its own output lexicon. Concepts are in fact Propbank frames (Palmer et al., 2004)⁶, which are oriented towards English. Therefore, after extracting concepts from the AMR structure, we are left with English concepts. To map English concepts to our NGT and VGT glosses in the SignBank, the SignBank was augmented with English translations so that every gloss is accompanied by its Dutch translations *as well as* potential English translations. This, in turn, allows us to use the AMR concepts, which are abstract and not literal

⁴ There are some Irish (GA) words like "Garda" "Taoiseach" etc. that are used widely in Hiberno-English that are also mouthed in ISL. See (Mohr, S., 2014) for more information on mouth actions in sign languages.

⁵ Data will be distributed via the Linguistic Data Consortium (LDC). Data has been submitted and a verbal agreement has been made. We are currently waiting for them to sign the distribution contract.

⁶ <https://github.com/propbank/propbank-frames/>

representations of the input, to retrieve glosses from a closed lexicon via a reverse look-up in the SignBanks. This is not sufficient, however, because the same English word may be linked with different glosses. Therefore, we disambiguate such ambiguous cases by measuring semantic similarity between the gloss candidates and the input sentence. The gloss which is closest in terms of semantics to the semantic representation of the input sentence is selected. Again, for a more elaborate description of this process, refer to D4.2 *“Second symbolic intermediate representation - InterL-S”*.

2.3 Evaluation and conclusion of glossing methods

In D4.5 *“A hybrid intermediate representation”* we benchmark both of these approaches on a subset of the NGT corpus (199 sentences). Note that while the AMR approach has built-in, specific support for NGT and VGT, the RB approach was only designed for VGT. Yet, because NGT and VGT glosses are very similar in surface form (they both “look like” a lemmatised form of Dutch), we were curious to make the comparison on NGT, in part also because the avatar currently only supports NGT glosses as input via an NGT SiGML lexicon. The results were telling: looking at the evaluation scores, RB outperforms the AMR approach across the board. The AMR pipeline has many components and is susceptible to data scarcity: it can only work based on the augmented SignBanks. If an English word is not present in the SignBank, no gloss will be found, and disambiguation may also yield the incorrect response. RB, on the other hand, is relatively straightforward and seems to work well but does require an input sentence in Dutch. Crucially, it is also not ensured that the model outputs recognised glosses - instead it will create pseudo-glosses based on the lemmatised input. For more details on this evaluation, see D4.5.

2.4 Gloss coverage

In the end, to make sure that we are constrained by a closed lexicon, the AMR implementation is currently implemented in the SignON framework. Arguments can also be made for the pseudo-glosses of the RB approach but we found it important that the glosses that we generate are at least recognised by sign linguists and not “potential” pseudo glosses. The methodology developed in our most recent AMR-based approach relies on the SignBanks to find established glosses in an attempt to improve compatibility with the avatar's supported closed lexicon. As alluded to before, to generate glosses we augmented the NGT and VGT SignBanks with English translations by building upon the Dutch translations that were already present. The NGT SignBank, then, has 2,381 entries (when we remove regional variations) and the NGT SiGML lexicon that is used in the avatar generation contains 2,203 signs. The

overlap between these two is unfortunately meagre, with only an overlap of 834 glosses. In practice that means that, while the AMR glosser outputs glosses from a closed vocabulary (the NGT Signbank), not all of these glosses are supported in the avatar's SiGML lexicon. Currently, the synthesis system ignores glosses that do not belong to the (NGT) lexicon supported, which, unfortunately, does not overlap fully with the NGT Signbank one. The motivation for ignoring the glosses is double. First of all, the synthesis system does not have a linguistic support allowing it to look for glosses in the supported system that might be useful in the context of communicating the meaning. Secondly, the coverage of the system is progressing fast at this stage (at the moment of writing this report an additional NGT lexicon has been received and is being added) and, furthermore, the priority would be to validate it. The coverage of the avatar is described in the next section.

3. Animics

The *Animics* system is the successor of the previously called *SignON editor* which allowed users to create avatars from videos. The SignON editor combined (i) a machine learning (ML) approach where users can input videos or capture themselves through a webcam that outputs body skeleton animations using a pose-estimation model and (ii) creating and editing symbolic representations, based on BML clips, to include rich NMFs and possible corrections of the MFs. Animics adds on the previous SignON editor a significantly enhanced pipeline for both the ML-driven and the symbolic approaches and provides a standardised way of building avatars.

In the next section, we discuss the different elements that underline the Animics system and are provided to the users to create their own avatars. In deliverable *D1.12 "User-generated Data"* we discuss how the Animics system is / can be used to develop user-generated avatars and expand the sign-clip repositories.

3.1 Blendshapes to Action Units

Animics provides the capability to generate both body and facial animations from a video or a webcam capture, recording both body landmarks (key points) and capturing the weights of blendshapes. Subsequently, the recorded positions are converted into bone rotations of a skeleton and then retargeted, enabling the animation to be applied to any avatar. For the acquired blendshapes scores to be aligned with the specific blendshapes of the character, instead of a direct mapping of the Mediapipe

blendshapes, we translate those into a standardised format referred to as Action Units (AU). Through this generalisation, the AU weights can be further applied into the blendshapes of different characters using the configuration file approach detailed in D5.2. This approach allows different characters to perform these animations if a suitable configuration file associated with the character is provided.

The AUs that our Animics system supports are:

MediaPipe blend shapes	Action Units
browDownLeft	AU4L: Brow_Lowerer_Left
browDownRight	AU4R: Brow_Lowerer_Right
browInnerUp	AU1: Inner_Brow_Raiser
browOuterUpLeft	AU33L: Outer_Brow_Raiser_Left
browOuterUpRight	AU33R: Outer_Brow_Raiser_Right
cheekPuff	AU33: Cheek_Blow_Left & Cheek_Blow_Right
cheekSquintLeft	AU6L: Cheek_Raiser_Left
cheekSquintRight	AU6R: Cheek_Raiser_Right
eyeBlinkLeft	AU46L: Wink_Left
eyeBlinkRight	AU46R: Wink_Right
eyeLookDownLeft	AU46L: Wink_Left
eyeLookDownRight	AU46R: Wink_Right
eyeLookInLeft	-
eyeLookInRight	-
eyeLookOutLeft	-
eyeLookOutRight	-
eyeLookUpLeft	AU5L: Upper_Lip_Raiser_Left
eyeLookUpRight	AU5R: Upper_Lid_Raiser_Right

eyeSquintLeft	AU44L: Squint_Left
eyeSquintRight	AU44R: Squint_Right
eyeWideLeft	AU5L: Upper_Lip_Raiser_Left
eyeWideRight	AU5R: Upper_Lid_Raiser_Right
jawForward	AU29: Jaw_Thrust
jawLeft	AU30L: Jaw_Sideways_Left
jawOpen	AU30I: Jaw_Drop
jawRight	AU30R: Jaw_Sideways_Right
mouthClose	-
mouthDimpleLeft	AU14L: Dimpler_Left
mouthDimpleRight	AU14R: Dimpler_Right
mouthFrownLeft	AU15L: Lip_Corner_Depressor_Left
mouthFrownRight	AU15R: Lip_Corner_Depressor_Right
mouthFunnel	AU22: Lip_Funneler
mouthLeft	AU18L: Lip_Puckerer_Left
mouthLowerDownLeft	AU16L: Lower_Lip_Depressor_Left
mouthLowerDownRight	AU16R: Lower_Lip_Depressor_Right
mouthPressLeft	AU24L: Lip_Pressor_Left
mouthPressRight	AU24R: Lip_Pressor_Right
mouthPucker	AU18: Lip_Puckerer_Left & Lip_Puckerer_Right
mouthRight	AU18R: Lip_Puckerer_Right
mouthRollLower	AU28D: Lip_Suck_Lower
mouthRollUpper	AU28U: Lip_Suck_Upper

mouthShrugLower	AU16: Lower_Lip_Depressor_Left & Lower_Lip_Depressor_Right
mouthShrugUpper	AU10: Upper_Lip_Raiser_Left & Upper_Lip_Raiser_Right
mouthSmileLeft	AU12L: Lip_Corner_Puller_Left
mouthSmileRight	AU12R: Lip_Corner_Puller_Right
mouthStretchLeft	AU20L: Lip_Stretcher_Left
mouthStretchRight	AU20R: Lip_Stretcher_Right
mouthUpperUpLeft	AU10L: Upper_Lip_Raiser_Left
mouthUpperUpRight	AU10R: Upper_Lip_Raiser_Right
noseSneerLeft	AU9L: Nose_Wrinkler_Left
noseSneerRight	AU9R: Nose_Wrinkler_Right
tongueOut	AU19: Tongue_Show

Table 1. Mapping from MediaPipe blendshapes to AU.

3.2 Sign Clips, SiGML and BML

Because of its alignment with research on conversational virtual characters and the expressive capabilities derived from this research, including precise timings, our symbolic descriptions are based on an (extended) BML.

Animics extends this symbol-driven approach of the previous SignON editor in several ways. A major challenging enhancement is to support MFs, leading to the creation of new clips, like hand configuration, hand placement, hand movement or hand orientation in the most recent tool version. A second major contribution in this line has been to provide support for existing SiGML repositories; this has run in parallel with the enhancement to support MFs through BML, and has led to a further extension of the BML in this period, with a much extended realiser. The following subsections discuss different aspects of this symbolic driven signing avatar synthesis.

3.2.1 Supporting generated SiGML repositories

SiGML (Kennaway 2004), stands for Signing Gesture Markup Language, and translates HamNoSys (Hamburg Notation System, Hanke 2004) symbols into XML tags. The SiGML representation derived from the HamNoSys notation of SL is more easily readable by computers and amenable to be rendered by 3D software. SiGML and its 3D rendering, including JASigning (Glauert and Elliott 2011) which facilitates its interactive use, were developed through a number of projects - ViSiCAST (2000-03), eSIGN (2002-04) and JASigning itself (2010-16). The representation finally provided explicit timing control, synchronisation between elementary motions and direction specification in various contexts. As designed for animating virtual signers, some timing attributes or very precise orientations can be specified in SiGML and not with HamNoSys. Because these descriptions are language-independent, they facilitate the automatic translation of phonetic representations into avatar animations, making them a very scalable strategy. Indeed, we have re-used a dataset for the Sign Language of the Netherlands (NGT), which contains the SiGML representation of 2,203 glosses. This has been used and tested for generating signing avatars for a transport use case (Van Gemert et al. 2022). It has also been used in healthcare contexts (Esselink et al. 2022), to turn Dutch glosses resulting from SignON MT into signing avatars with our own system, embedded within the SignON translation mobile app. Another dataset that we have used to test and expand the system includes the SL of the French-speaking part of Switzerland (LSF-CH); this dataset currently contains 2,031 sentences for the use case of conversations in a medical emergency context (Strasly et al. 2018, and David et al. 2022).

A final remark related to this topic is that both the end-to-end system, and the symbolic driven synthesis re-using SiGML are alternative unforeseen strategies to obtain actual SL output in the context of the challenges faced within SignON in the automatic population of Sign_A- or BML- based repositories.

3.2.2 Extending BML, Supporting SiGML, Realisation

The two repositories of symbolic sign language representations mentioned above are available within our system, and used to automatically generate animation clips based on extended BML. Indeed, as indicated earlier, the challenge to re-use these datasets has pushed us for a strong BML extension and the corresponding major realisation effort, based on Inverse Kinematics. On the positive side towards the future, the system could effortlessly support any SL dataset of similar characteristics (glosses or sentences in SiGML representation).

The integration of support for SiGML instructions has led to an extension of the BML specification, requiring the introduction of new actions and, consequently, the implementation of new clips. A *clip* is an instruction for applying a behaviour and each clip has an id, start time and duration. The realisation of each behaviour can be divided into phases, with each phase marked by a sync-point denoting the associated transition. Notably, these sync-point values are falling within the range [0, duration]. The time and the sync-point parameters are in seconds. Users have the flexibility to modify these parameters, along with other variable properties.

SiGML shares similarity with the BML structure we have just summarised. This has simplified the process of translating from SiGML to the Animics BML based representations, particularly in terms of attribute definition and clip creation. Making the parser has been a very manual and detailed process, and it has both pushed for the BML extension as it has validated the capability of this extension to represent and be able to render SLs. Once the parsing was carried out, the mapping in the editor was automatic.

This effort is comprehensively reflected in the next section, which contains detail of the clips available, outlining their BML and SiGML correspondence, and describing their specific attributes. It addresses the categories: *Face lexeme, Mouthing, Head movement, Gaze, Elbow raise, Shoulder movement, Arm location, Hand constellation, Directed motion, Circular motion, Palm orientation, Hand orientation, Wrist motion, Finger play, Hand shape, and Body movement.*

3.2.3 Clips Available in Animics; SiGML-BML Correspondence

Face Lexeme

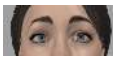

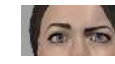

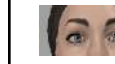
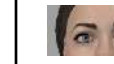






















Face lexemes are (partial) face expressions from a predefined lexicon. This behaviour offers a range of predefined expressions. Each lexeme is a convenience shorthand for combinations of more detailed low level face controls. The provided set of core lexemes allows one to perform face expressions using meaningful lexeme names.

Clip type	BML tag	SiGML tag
FaceLexeme	<faceLexeme>	<facialexpr_tier>

Attribute	Type	Use	Default	Description
id	string	required	<i>nameOfTheLexeme</i>	Identifier name
lexeme	string	required		Name of a face lexeme from the available list
intensity	float	required	1.0	A float value in the interval [0,1] to indicate the amount to which the expression should be shown on the face, 0 meaning 'not at all' and 1 meaning 'maximum, highly exaggerated'

Sync Attribute	Type	Description
start	float	Second at which the face expression begins
attackPeak	float	Second at which the maximum expression is achieved. In the interval [0, duration]
relax	float	Second at which the decay phase starts. In the interval [0, duration]
duration	float	Duration time, in seconds, of the face expression

- List of available face lexemes:

 <i>Arch</i>	 <i>Brow lowerer</i>	 <i>Brow lowerer left</i>	 <i>Brow lowerer right</i>	 <i>Brow raiser</i>	 <i>Brow raiser left</i>	 <i>Brow raiser right</i>
 <i>Inner brow raiser</i>	 <i>Outer brow raiser</i>	 <i>Squint</i>	 <i>Blink</i>	 <i>Eyes closed</i>	 <i>Upper lid raiser</i>	 <i>Upper lid raiser left</i>
 <i>Upper lid raiser right</i>	 <i>Cheek raiser</i>	 <i>Lid tightener</i>	 <i>Wink left</i>	 <i>Wink right</i>	 <i>Cheek suck</i>	 <i>Cheek suck left</i>
 <i>Cheek suck right</i>	 <i>Cheek blow</i>	 <i>Cheek blow left</i>	 <i>Cheek blow right</i>	 <i>Nose wrinkler</i>	 <i>Nostril dilator</i>	 <i>Nostril compressor</i>


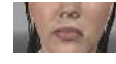

















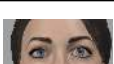






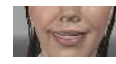




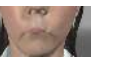




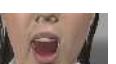


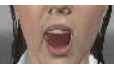
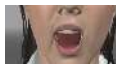
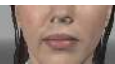

 <i>Lip corner depressor</i>	 <i>Lip corner depressor left</i>	 <i>Lip corner depressor right</i>	 <i>Lip corner puller</i>	 <i>Lip corner puller left</i>	 <i>Lip corner puller right</i>	 <i>Lip stretcher</i>
 <i>Lip funneler</i>	 <i>Lip tightener</i>	 <i>Lip puckerer</i>	 <i>Lip puckerer left</i>	 <i>Lip puckerer right</i>	 <i>Lip pressor</i>	 <i>Lips part</i>
 <i>Lip suck</i>	 <i>Lip suck upper</i>	 <i>Lip suck lower</i>	 <i>Lower lip depressor</i>	 <i>Lower lip depressor left</i>	 <i>Lower lip depressor right</i>	 <i>Upper lip raiser</i>
 <i>Upper lip raiser left</i>	 <i>Upper lip raiser right</i>	 <i>Chin raiser</i>	 <i>Dimpler</i>	 <i>Dimpler left</i>	 <i>Dimpler right</i>	 <i>Lip bite</i>
 <i>Smile teeth</i>	 <i>Smile teeth wide</i>	 <i>Smile closed</i>	 <i>Round open</i>	 <i>Round closed</i>	 <i>Mouth stretch</i>	 <i>Close tight</i>
 <i>Jaw drop</i>	 <i>Jaw thrust</i>	 <i>Jaw sideways left</i>	 <i>Jaw sideways right</i>	 <i>Tongue bulge left</i>	 <i>Tongue bulge right</i>	 <i>Tongue up</i>
 <i>Tongue show</i>	 <i>Tongue wide</i>	 <i>Lip wipe</i>	 <i>Neck tightener</i>			

Table 2. List of available face lexemes.

Mouthing

Generation of lip movements from a text in ARPABET 1-letter notation. In which, '!' and ' ' symbols are also supported.

Clip type	BML tag	SiGML tag
Mouthing	<speech>	<mouthing_picture>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
text	string	required		String of the text which is wanted to be simulated in the form of ARPABET 1-letter
speed	float	optional	1.0	Phonemes per second of the whole string. Humans speak at 8 phonemes per second (lower boundary). It can be specified individually for each phoneme
intensity	float	optional	1.0	Visual exaggeration of phonemes. In the interval [0,1]. It can be specified individually for each phoneme

Sync Attribute	Type	Description
start	float	Second at which the mouthing begins
duration	float	Duration time, in seconds, of the utterance. Automatically computed using the content properties

Head movement

Head movement retrieved from a gesticon by requesting the associated lexeme.

Clip type	BML tag	SiGML tag
HeadMovement	<head>	<head_movement>

Attribute	Type	Use	Default	Description
id	string	required	<i>nameOfTheLexeme</i>	Identifier name
lexeme	string	required		Name of the particular head behaviour from the available list
repetitions	integer	required	1	Number of times the basic head motion is repeated
intensity	float	required	1.0	A float value between 0..1 to indicate the amount of the movement. 0 means immeasurable small; 0.5 means "moderate"; 1 means maximally large

Sync Attribute	Type	Description
start	float	Second at which the preparation phase of the movement starts
ready	float	Second at which the preparation phase ends. In the interval [0, duration]
strokeStart	float	Second at which starts the stroke. In the interval [0, duration]
stroke	float	Second at which the head motion strokes. In the interval [0, duration]
strokeEnd	float	Second at which the stroke ends. In the interval [0, duration]
relax	float	Second at which the decay phase starts. In the interval [0, duration]
duration	float	Duration time, in seconds, of the head motion

- List of head lexemes: *Nod, Shake, Tilt, Tilt left, Tilt right, Tilt forward, Tilt backward, Forward, Backward*

Gaze

Temporarily directs the gaze of the character towards a target.

Clip type	BML tag	SiGML tag
Gaze	<gaze>	<eye_gaze> <head_movement>

Attribute	Type	Use	Default	Description
id	string	required	<i>gazeInfluence</i>	Identifier name
influence	string	required		Determines what parts of the body to move to affect the gaze direction: <i>Eyes, Head, Neck</i>
target	string	required		A reference towards a target instance that represents the target direction of the gaze
shift (base gaze)	boolean	optional	false	Permanently changes the gaze direction of the character towards the target
offsetAngle	float	optional	0.0	Adds an angle degrees offset to gaze direction relative to the target in the direction specified in the <i>Offset direction</i> . In degrees
offsetDirection	string	optional		Direction of the <i>Offset angle</i> . Same directions as target property

- List of available targets: *Front, Left, Right, Up, Down, Up left, Up right, Down left, Down right*.

Sync Attribute	Type	Description
start	float	Second at which the gaze starts to move to new target
ready	float	Second at which the gaze target is acquired. In the interval [0, duration]
relax	float	Second at which the gaze starts returning to default direction. In the interval [0, duration]
duration	float	Duration time, in seconds, of gaze returned to default direction

Elbow raise

Raises the elbow (added to the elbow raise automatically computed while moving the arm).

Clip type	BML tag	SiGML tag
ElbowRaise	Extended <gesture>	-

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
side	string	required	<i>dominantHand</i>	Which arm is being moved: <i>Left, Right, Both</i>
Amplitude (intensity)	float	required	1	A float value in the interval [0,1] to indicate the amount of the movement. 0 means immeasurable small; 0.5 means "moderate"; 1 means maximally large
shift (base position)	boolean	optional	false	Permanently changes the position of the elbow

Sync Attribute	Type	Description
start	float	Second at which the elbow starts to raise

attackPeak	float	Second at which the elbow raise is acquired. In the interval [0, duration]
relax	float	Second at which the elbow starts returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of elbow returned to default position

Shoulder movement

Move the shoulder up or forward.

Clip type	BML tag	SiGML tag
ShoulderMovement	Extended <gesture>	<shoulder_movement>

Attribute	Type	Use	Default	Description
id	string	required	<i>movementType</i>	Identifier name
movement	string	required		Name of the particular shoulder movement. <i>Raise or Hunch</i>
side	string	required	<i>dominantHand</i>	Which arm is being moved: <i>Left, Right, Both</i>
Amplitude (intensity)	float	required	1	A float value in the interval [0,1] to indicate the amount of the movement. 0 means immeasurable small; 0.5 means "moderate"; 1 means maximally large
shift (base position)	boolean	optional	false	Permanently changes the position of the shoulder

Sync Attribute	Type	Description
start	float	Second at which the shoulder starts the movement
attackPeak	float	Second at which the shoulder position is acquired. In the interval [0, duration]

relax	float	Second at which the shoulder starts returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of the shoulder returned to default position

Arm location

Moves the hand to the specified location of the body.

Clip type	BML tag	SiGML tag
ArmLocation	Extended <gesture>	<location_bodyarm>

Attribute	Type	Use	Default	Description
id	string	required	<i>movementType</i>	Identifier name
arm location	string	required		A body reference towards a target instance that represents the target arm location
arm	string	required	<i>dominantHand</i>	Which arm is being moved: <i>Left, Right, Both</i>
shift (base position)	boolean	optional	false	Permanently changes the position of the arm
side	string	optional		Generic horizontal offset of the chosen point: <i>Left, Right, Slightly Left, Slightly Right</i>
second location	string	optional		Same as <i>Arm location</i> . If this property is set, the final point corresponds to the middle point between both. Same locations as <i>Arm location</i> property
second side	string	optional		Generic horizontal offset of the chosen point of the second location

distance	float	optional		How far from the body the wrist has to be located. 0 means immeasurable close; 1 means maximally arm extended
displace	float	optional		Offset direction of the target location
displace distance	float	optional		How far the hand is displaced of the offset direction in metres
location	string	optional	<i>Wrist</i>	Part of the hand that will try to reach the target
side	string	optional		Specific hand side that will try to reach the target. Not applied if the location value is <i>Tip</i>
finger	string	optional		Finger that will try to reach the target: <i>Thumb, Index, Middle, Ring, Pinky</i> . Only applied if the location is <i>Tip, Pad, Mid, Base</i>

- List of available arm locations: *Head, Head top, Forehead, Nose, Below nose, Chin, Under chin, Mouth, Earlobe, Earlobe Right, Earlobe Left, Ear, Ear Right, Ear Left, Cheek, Cheek Right, Cheek Left, Eye, Eye Right, Eye Left, Eyebrow, Eyebrow Left, Eyebrow Right, Mouth, Neck, Chest, Shoulder Line, Shoulder, Shoulder Right, Shoulder Left, Stomach, Below stomach, Neutral*.
- List of available displace directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right*.

Sync Attribute	Type	Description
start	float	Second at which the arm starts the movement
attackPeak	float	Second at which the hand position is acquired. In the interval [0, duration]
relax	float	Second at which the hand starts returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of the arm returned to default position

Hand constellation

Moves the hands relative to each other. The motion is stopped if an *Arm location* clip is executed afterwards.

Clip type	BML tag	SiGML tag
HandConstellation	Extended <gesture>	<handconstellation>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Main hand that will be moved: <i>Left, Right, Both, Dominant, Non dominant</i> . If <i>Both</i> is set, the both hands positions will be moved
location	string	required	Pad	Part of the hand that will try to reach the other hand. The secondary hand can have values that include both hand and arm locations
side	string	required		Specific hand side that will try to reach the other hand. Not applied if the location value is <i>Tip</i>
finger	string	required		Finger that will try to reach the other hand. Only applied if the location is <i>Tip, Pad, Mid, Base</i>
second location	string	required		The secondary hand can have values that include both hand and arm locations
second side	string	required		Generic horizontal offset of the chosen point of the second location
second finger	string	required		Finger that will try to reach the other hand. Only applied if the location is <i>Tip, Pad, Mid, Base</i>
distance	float	optional		Distance between hand targets. 0 is touching; 1 is the arm size

distance direction	string	optional		Direction in which to apply the distance. If not provided, defaults to horizontal outwards direction
shift (base position)	boolean	optional	false	Permanently changes the position of the arm

- List of available hand parts: *Tip, Pad, Mid, Base, Thumb ball, Hand, Wrist.*
- List of available arm parts: *Forearm, Elbow, Upper arm.*
- List of available hand sides: *Left, Right, Ulnar, Radial, Front, Back, Palmar.*
- List of available directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right.*

Sync Attribute	Type	Description
start	float	Second at which the hand/s start/s the movement
attackPeak	float	Second at which the hand/s position is/are acquired. In the interval [0, duration]
relax	float	Second at which the hand/s start/s returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of hand/s returned to default position

Directed motion

Moves the arm (wrist) along a linear path. The motion is stopped if an Arm location clip is executed afterwards.

Clip type	BML tag	SiGML tag
DirectedMotion	Extended <gesture>	<directedmotion>

Attribute	Type	Use	Default	Description
-----------	------	-----	---------	-------------

id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Which hand will be moved: <i>Left, Right, Both</i>
direction	string	required		Direction of the movement of the available list
second direction	string	optional		If this property is set, the final wrist position corresponds to the middle point between both directions
distance	float	optional	0.2	Meters of the displacement
curve direction	string	optional		Defines the curve direction to reach the final position of the wrist. If not specified, the wrist movement follows a straight path
second curve direction	string	optional		If this property is set, the final curve motion corresponds to the middle point between both directions
amplitude	float	optional		Amplitude of the curve motion. In metres
zig-zag direction	string	optional		Defines the zig-zag direction to reach the final position of the wrist. If not specified, the wrist movement follows a straight path. List of available directions
zig-zag amplitude	float	optional		Amplitude of the zig-zag motion. In metres
zig-zag speed	float	optional		Oscillations per second

- List of available directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right.*
- List of available curve directions: *Up, Down, Left, Right, Up Left, Up Right, Down Left, Down Right.*

Sync Attribute	Type	Description
start	float	Second at which the arms the movement
attackPeak	float	Second at which the wrist motion is acquired. In the interval [0, duration]
relax	float	Second at which the arm starts returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of the arm returned to default position.

Circular motion

Moves the arm (wrist) along a circular path. The motion is stopped if an Arm location clip is executed afterwards.

Clip type	BML tag	SiGML tag
CircularMotion	Extended <gesture>	<circularmotion>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Which hand will be moved: <i>Left, Right, Both</i>
direction	string	required		Direction of the axis rotation of the available list
second direction	string	optional		If this property is set, the final wrist position corresponds to the middle point between both directions
distance	float	optional		Radius of the circular displacement in metres.
start angle	float	optional	0.0	Defines the starting angle for the motion within the circle. 0.0 means up; 180 means down. In degrees

end angle	float	optional	360	Defines the finishing angle for the motion within the circle. 0.0 means up; 180 means down. In degrees
ellipse axis ratio	float	optional	1	Ratio of the minor/major radius in an ellipse. 1 means circle with a radius of distance.
ellipse axis direction	string	optional		Direction of the major radius of the ellipse. Only applied if the <i>Ellipse axis ratio</i> value is different to 1. List of available ellipse directions
zig-zag direction	string	optional		Defines the zig-zag direction to reach the final position of the wrist. If not specified, the wrist movement follows a straight path. List of available directions
zig-zag amplitude	float	optional		Amplitude of the zig-zag motion. In metres
zig-zag speed	float	optional		Oscillations per second

- List of available directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right.*
- List of available ellipse directions: *Up, Down, Left, Right, Up Left, Up Right, Down Left, Down Right.*

Sync Attribute	Type	Description
start	float	Second at which the arms the movement
attackPeak	float	Second at which the wrist motion is acquired. In the interval [0, duration]
relax	float	Second at which the arm starts returning to default position. In the interval [0, duration]
duration	float	Duration time, in seconds, of the arm returned to default positio.

Palm orientation

Orientation of the hand palm rolling the wrist joint.

Clip type	BML tag	SiGML tag
PalmOrientation	Extended <gesture>	<handconfig>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Which hand will be moved: <i>Left, Right, Both</i>
direction	string	required		Direction relative to the arm. List of available directions
shift (base orientation)	boolean	optional		Permanently changes the palm's rotation
second direction	string	optional		If this property is set, the final orientation corresponds to the middle point between both directions

- List of available directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right.*

Sync Attribute	Type	Description
start	float	Second at which the wrist starts the rotation.
attackPeak	float	Second at which the palm orientation target is acquired. In the interval [0, duration]
relax	float	Second at which the wrist starts returning to default rotation. In the interval [0, duration]

duration	float	Duration time, in seconds, of wrist returned to default rotation
-----------------	-------	--

Hand orientation

Orientation of the hand involving both yawing and pitching of the wrist joint.

Clip type	BML tag	SIGML tag
HandOrientation	Extended <gesture>	<handconfig>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Which hand will be moved: <i>Left, Right, Both</i>
direction	string	required		Direction relative to the arm. List of available directions
shift (base orientation)	boolean	optional		Permanently changes the hand's rotation
second direction	string	optional		If this property is set, the final orientation corresponds to the middle point between both directions

- List of available directions: *Up, Down, Left, Right, In, Out, Up Left, Up Right, Up In, Up Out, Left In, Left Out, Right In, Right Out, Down Left, Down Right, Down In, Down Out, Up Out Left, Up Out Right, Down Out Left, Down Out Right.*

Sync Attribute	Type	Description
start	float	Second at which the wrist starts the rotation
attackPeak	float	Second at which the hand orientation target is acquired. In the interval [0, duration]

relax	float	Second at which the wrist starts returning to default rotation. In the interval [0, duration]
duration	float	Duration time, in seconds, of wrist returned to default rotation

Wrist motion

Repetitive swinging, nodding, and twisting motions of the wrist, akin to a wiggle for the wrist.

Clip type	BML tag	SiGML tag
WristMotion	Extended <gesture>	<wristmotion>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	Which hand will be moved: <i>Left, Right, Both</i>
motion	string	required		Which movement the wrist will perform of the available list
speed	float	optional		Oscillations (repetitions) per second of the movement
intensity (amount)	float	optional		Amplitude of the movement

- List of available motions: *Nod, Swing, Twist, Stir CW, Stir CCW, All*.

Sync Attribute	Type	Description
start	float	Second at which the wrist starts the rotation

attackPeak	float	Second at which the motion peak is acquired. In the interval [0, duration]
relax	float	Second at which the wrist starts returning to default rotation. In the interval [0, duration]
duration	float	Duration time, in seconds, of wrist returned to default rotation

Finger play

Move the fingers of the hand in a wiggling motion.

Clip type	BML tag	SiGML tag
FingerPlay	Extended <gesture>	<fingerplay>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand	string	required	<i>dominantHand</i>	From which hand the fingers are being moved: <i>Left, Right, Both</i>
fingers	string	optional	<i>All</i>	Which fingers are being moved: <i>Thumb, Index, Middle, Ring, Pinky.</i>
speed	float	optional		Oscillations (repetitions) per second of the movement
intensity (amount)	float	optional		Amplitude of the movement

Sync Attribute	Type	Description
start	float	Second at which the fingers start the motion

attackPeak	float	Second at which the motion peak is acquired. In the interval [0, duration]
relax	float	Second at which the fingers start returning to default rotation. In the interval [0, duration]
duration	float	Duration time, in seconds, of fingers returned to default rotation

Hand shape

Specifies the posture of the fingers of the hand.

Clip type	BML tag	SiGML tag
Handshape	Extended <gesture>	<handconfig>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
hand shape	string	required	<i>dominantHand</i>	From which hand the fingers are being configured: <i>Left, Right, Both</i>
shape	string	required		Configuration of the hand from the available list. Fingers are represented as numbers, 1 means <i>thumb</i> ; 5 means <i>pinky</i>
shift (base shape)	string	optional		Permanently changes the hand shape
thumb shape	string	optional		Specific posture of the thumb finger. It overwrites the hand shape configuration for this finger. If not specified, the predefined hand shape will be used

thumb combination opening	float	optional		Distance between the thumb and the other fingers. Expressed on a scale of 0 to 1, where 0 signifies close proximity; 1 indicates maximum separation
main splay	float	optional		Distance between fingers. Expressed on a scale of 0 to 1, where 0 signifies close proximity; 1 indicates maximum separation. By default is 0.5
main blend	string	optional		Flexion applied to chosen fingers from the default hand shape. Basic hand shapes and thumb combination shapes have distinct behaviour. List of bend states
second hand shape	string	optional		Second configuration of the hand from the available list
second thumb shape	string	optional		Second posture of the thumb finger
second combination opening	float	optional		Second distance between the thumb and the other fingers
second main splay	float	optional		Second flexion applied to chosen fingers from the default hand shape
specific fingers	string	optional	<i>All</i>	Specific fingers to apply the configuration.

- List of hand shapes: *Fist, Finger 2, Finger 23, Finger 23 spread, Finger 2345, Flat, Pinch 12, Pinch 12 open, Pinch all, Cee all, Cee 12, Cee 12 open.*
- List of thumb shapes: *Default, Out, Opposed, Across, Touch.*
- List of bend states: *Straight, Half bent, Bent, Round, Hooked, Double bent, Double hooked.*

Sync Attribute	Type	Description
start	float	Second at which the fingers start the motion

attackPeak	float	Second at which the hand shape is acquired. In the interval [0, duration]
relax	float	Second at which the fingers start returning to default configuration. In the interval [0, duration]
duration	float	Duration time, in seconds, of fingers returned to default hand configuration

Body movement

Moves the position of the body (trunk) by tilting forward-backward, tilting left-right, and rotating left-right. New gestures can be incorporated alongside the existing ones, without replacing them.

Clip type	BML tag	SiGML tag
BodyMovement	Extended <gesture>	<body_movement>

Attribute	Type	Use	Default	Description
id	string	required	<i>clipType</i>	Identifier name
movement	string	required		Which movement the body performs from the available list
intensity (amount)	string	required		Amplitude of the movement

- List of movements: *Tilt forward, Tilt backward, Tilt left, Tilt right, Rotate left, Rotate right.*

Sync Attribute	Type	Description
start	float	Second at which starts the motion
attackPeak	float	Second at which the hand shape is acquired. In the interval [0, duration]
relax	float	Second at which the body starts returning to default posture. In the interval [0, duration].

duration	float	Duration time, in seconds, of the body returned to default posture
-----------------	-------	--

4. Conclusions and future work

The research conducted along this line (sign repositories and sign structures for the scope of synthesis) has proven to be quite challenging given that the manual work required to create Sign_A + RRG repositories was highly demanding while little other data for synthesis was available. We explored several approaches with two contrasting ideas - one, completely data-driven and based on a machine learning pipeline for synthesis (sign keypoint-frames representation to sequences of such sign representations to avatar) and another based on symbolic, gloss-based (gloss-to-SiGML-to-BML-to-avatar). Through the progress on AMR and the rule-based systems and the development of the Animics system, we bring the best of these worlds together. Within the Animics system, the available sign repositories can be modified while new ones can be added through video, key points or SiGML inputs.

We acknowledge that the gloss-based approach is not ideal. However, the wide research we conducted within SignON indicates that the lack of data (to facilitate better ML and deep learning methods) as well as the excessive human labour needed for representations such as Sign_A require that (i) a middle ground is found at this stage and (ii) a framework is developed that will allow the collection and processing of data on sign- and utterance- levels for the specific purpose of synthesis, i.e. Animics.

When it comes to future work we foresee and encourage the following research directions:

- Use the Animics system to expand the lexicons of symbolic sign representations (e.g. SiGML) adding new signs and new languages. We acknowledge that human efforts will be needed, but nonetheless, such repositories is what most of current SLS systems rely on. Through the ML pipeline, followed by a post-editing step, even more such representations can be created.
- Advance the Animics system and turn it into a standard in SLS. To do so, new functionalities should be implemented, e.g. generating Sign_A representations; extending the parser to cover input in variants of SiGML not yet considered (currently the support is for Gestural SiGML); improving the ML system we presented in D5.5 which has already been extended to NMFs, thus allowing more flexible input with results of better quality, which can be integrated with symbolically generated ones through Performs, getting the best from diverse inputs; improved ML models - add new generic and language-specific models.

- Advance the AMR implementation to support more languages and potentially associate the AMR predicates with sign representations other than glosses; potentially develop an LLM capable to generate AMR graphs from sentences (spoken languages) or, through an alignment with SLR embeddings, from signed utterances. Develop a sign language specific post-processing to map AMR graphs to SL utterances with fewer intermediate steps. Advances in SignNets can be found in D3.3 “Linguistic description for ISL, BSL, VGT, NGT and LSE”.
- Advance the work on SignNets and develop a machine translation pipeline that utilises SignNets.

5. References

David, Bastien; Mutal, Jonathan David; Strasly, Irene; Bouillon, Pierrette; Spechbach, Hervé: BabelDr, un système de traduction du discours médical vers l’animation virtuelle signée. In: Handicap 2022 - 12e conférence de l’IFRATH sur les technologies d’assistance. Paris. Paris : IFRATH, 2022. p. 46–51

Esselink, L., Roelofsen, F., Dotlacil, J., Mende-Gillings, S., De Meulder, M., Sijm, N. and Smeijers, A. Exploring automatic text-to-sign translation in a healthcare setting. Unpublished., 2022.

John Glauert and Ralph Elliott. Extending the sigma notation; a progress report. In Proceedings of the Second International Workshop on Sign Language Translation and Avatar Technology, 2011.

Hanke, T. HamNoSys -- Representing sign language data in language resources and language processing contexts. Fourth International Conference on Language Resources and Evaluation (LREC 2004). Representation and Processing of Sign Languages Workshop (pp. 1-6). Paris: European Language Resources Association. 2004

Knight, K., Badarau, B., Baranescu, L., Bonial, C., Griffitt, K., Hermjakob, U., Marcu, D., O’Gorman, T., Palmer, M., Schneider, N., & Bardocz, M. (2020). Abstract Meaning Representation (AMR) Annotation Release 3.0 (p. 153936 KB) [Data set]. Linguistic Data Consortium. <https://doi.org/10.35111/44CY-BP51>

Leeson, L., Morissey, S., Stein, D., Sterionov, D., Van den Heuvel, H. & A. Way (forthcoming) How it Started and How it’s Going: Sign-Language Machine Translation and Engagement with Deaf Communities over the past 25 years. <https://link.springer.com/book/9783031473616>

Mohr, S. (2014). *Mouth actions in sign languages: An empirical study of Irish Sign Language* (Vol. 3). Walter de Gruyter GmbH & Co KG.

Palmer, Martha, Kingsbury, Paul, Babko-Malaya, Olga, Cotton, Scott, & Snyder, Benjamin. (2004). Proposition Bank I [dataset]. Linguistic Data Consortium. <https://doi.org/10.35111/9R29-GT54>

Strasly, I., Sebäi, T., Rigot, E., Marti, V., Gonzalez, J., M., Gerlach, J., Spechbach, H. and Bouillon, P. Le projet babeldr : rendre les informations médicales accessibles en langue des signes de suisse romande (Isf-sr). 2018.

Van Gemert, B., Cokart, R., Esselink, L., De Meulder, M., Sijm, N., and Roelofsen, F. First steps towards a signing avatar for railway travel announcements in the Netherlands. In Proceedings of the 7th International Workshop on Sign Language Translation and Avatar Technology: The Junction of the Visual and the Textual: Challenges and Perspectives, pages 109–116, Marseille, France, jun 2022. European Language Resources Association.