Signing Avatar: Say hello to Elsi!

Michael Filhol, Annelies Braffort and Laurence Bolot
LIMSI/CNRS
Campus d’Orsay, bât. 508
F-91403 Orsay cedex - France
first.lastname@limsi.fr

Abstract. This work introduces the Elsi project, a signing avatar whose purpose is to make announcements in French Sign Language (LSF) to deaf people. For its animation to be acceptable, we first need to build an internal model of the language, on which the whole system will be based. We describe the first steps of the model design and the way we plan to evaluate them.

1. Introduction

Limsi recently initiated a signing avatar project called Elsi: with the purpose of generating French Sign Language (LSF). There already exists European projects capable of producing sequences of signs [8], but we would like to take them a step further, as we explain further down.

This paper describes the organisation of our work on sign production. Section 2 goes through the main characteristics of LSF and addresses the consequent problems posed for generation and animation. Section 3 presents the Elsi software platform as well as the methodology we chose for its development and ongoing evaluation. Section 4 states our progress and discusses some of the prospects of the Elsi platform.

2. SL discourse generation: what to tackle?

The number of body features involved in LSF communication allows for a lot of information at once: Sign Languages (SLs) do not only use hands, but also shoulders, eye gaze, facial expression, head movements (fig. 1a); linguistic studies of LSF show a heavy and consistent use of the "signing space", i.e. the portion of space in which the signs are performed (fig. 1cd); iconicity is also a relevant feature of both its lexicon (fig. 1b) and its grammar [3, 9]...

It is proven by linguists that often neglected details such as eye gaze or tension all contribute to the construction of meaning in a more than significant way [3]. Therefore, while modelling the signs of the language, i.e. at the lexical level, all the possible context-driven variations must be considered as they are fully part of the

2 ELSI: Elsi is Limsi’s SIGner

Fig. 1. Examples of LSF signs: (a) [WHAT?], (b) [BIRTH], (c) “here”, (d) [GROUP].

3. SL generation & animation software on the way...

3.1. Structure of the software environment

An avatar animation platform is presently under development at Limsi. The lexical signs involved in a sign utterance are first ordered, then performed according to context and output to a video. Diagram 2 sketches out the structure of the Elsi sign production software.

Sentence generation (M1 in diagram), described in section 3.2, is based on a model of the signing space (K2) and uses spatio-temporal structures (K1), a common feature not discussed in this paper [1, 2]. The sign generation module (M2) uses descriptions of the wanted signs (K3) – see 3.3. The animation engine module (M3) is described in section 3.4. Given a formal XML input representing the sentence production, its role is to animate Elsi in a separate window. It uses a hierarchical description of the avatar (K4) and a bank of animated signs performed by our graphics expert.

3.2. Sentence generation (M1)

Producing an LSF clause consists in producing a sequence of LSF signs that suits the LSF grammar. We use a simple model of the signing space (module K2 in fig. 2) for now. It contains the signed units and their respective locations, orientations and sizes. It allows production of isolated gap clauses with predefined format and use of both manual and non-manual signs. Production is carried out by signing the units
back to back. We chose an incremental approach so that each module be tested independently and every step forward guaranteed the reliability of its basis:
1. design predefined units and check the fluidity of a single animation;
2. try sequences of those units and check the fluidity of the interpolated transitions;
3. replace a predefined unit by a lexical description processed by M2;
4. add context values to the input of M2.

![Fig. 2. Structure of the ELSI animation platform.](image)

### 3.3. Sign generation (M2): the lexical model

In this section we suggest a lexical model (for part K3 of diagram 2), flexible enough to account for all these context-dependent variations of the signs and allows a description – i.e. a sign – to be used in different clauses even if it should not be performed the same way. Using a straightforward and "hard-wired" approach, we would have to build a new description for each use of the sign, in other words create a new animation for each use.

Our model uses geometric constraints [5, 6]. Signs are no more regarded as tuples of universal parameters like in Stokoe-based approaches [11], but rather as dynamic spatial geometric figures. A description may build any useful set of geometric objects like planes or points in space, and constrains body segments to describe positions and gestures. For instance, when hands are symmetric, we rather build the symmetry and not repeat parts of the description for both hands.

Now provided a geometric description, we are still missing an engine to animate our virtual character. The geometric description must be compiled to articulatory commands acting on the rotation angles of the character’s virtual bones.
4. Current progress and prospects

The first step of the evaluation process given in 3.2 is close to being achieved. An example of a predefined animation can be found on our website. In the second step, we work on gap-production "welcome to Laurence's home page, who works on avatars" (signs are stored K5). This test was put together in the prospect of an automatic welcome banner on our group's website.

For the next two steps, the production we are concerned with is: "the TGV train number J234 will be stopping in Laval and in Brest". A consistent layout in space is needed to locate the towns in the utterance.

Once the three modules will have been tested enough to be considered functional, we will start integrating bank K1 and generating the corresponding sign sequences. This of course is to keep increasing the automation and capabilities of the software.

5. References