A Combined Semantic and Motion Capture Database for Real-Time Sign Language Synthesis

Charly Awad, Nicolas Courty, Kyle Duarte, Thibaut Le Naour, Sylvie Gibet

▶ **To cite this version:**

Charly Awad, Nicolas Courty, Kyle Duarte, Thibaut Le Naour, Sylvie Gibet. A Combined Semantic and Motion Capture Database for Real-Time Sign Language Synthesis. 9th Int. Conference on Intelligent Virtual Agent (IVA 2009), Sep 2009, Amsterdam, Netherlands. pp.432-438. hal-00493426

HAL Id: hal-00493426
https://hal.science/hal-00493426v1

Submitted on 18 Jun 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Combined Semantic and Motion Capture Database for Real-Time Sign Language Synthesis

Charly Awad[1], Nicolas Courty[1], Kyle Duarte[1], Thibaut Le Naour[1] and Sylvie Gibet[1,2]

[1] Université de Bretagne Sud, Laboratoire VALORIA, Bâtiment Yves Coppens, F-56017 Vannes, FRANCE
[2] IRISA, Campus de Beaulieu, F-35042 Rennes, FRANCE

Abstract. Over the past decade, many fields of discovery have begun to use motion capture data, leading to an exponential growth in the size of motion databases. Querying, indexing and retrieving motion capture data has thus become a crucial problem for the accessibility and usability of such databases. Our aim is to make this approach feasible for virtual agents signing in French Sign Language, taking into account the semantic information implicitly contained in sign language data. We propose a new methodology for accessing our database, by simultaneously using both a semantic and a captured-motion database, with different ways of indexing the two database parts. This approach is used to effectively retrieve stored motions for the purposes of producing real-time sign language animations. The complete process and its in-use efficiency are described, from querying motion in the semantic database to computing transitory segments between signs, and producing animations of a realistic virtual character.

1 Introduction

Designing virtual humanoids that communicate in sign languages is a major challenge within the virtual agent animation community. The linguistic elements of sign languages (i.e., signs) differ sensibly from other non-linguistic gestures, as they are by essence multimodal. Each modality of a single sign, those being the gestures of two hands, and the signer's facial expressions and gaze direction, conveys meaningful information from the phonological level to the discourse level. Moreover, signs exhibit a highly spatial and temporal variability that can serve as syntactic modifiers of aspect, participants, etc.

In the past, data-driven computer animation methods have tried to handle signs' variability by using large databases of human motions to efficiently produce realistic real-time animations. These databases have been challenging to construct, as they must consider both semantic and motion-captured information as well as be flexible enough to adapt existing data to newly synthesized scenarios. Given sign languages' spatial and temporal morphemic and syntactic structures, grammatically inflecting signs may significantly modify the geometrical and timing aspects of the stored base motions. What's more, by effecting

these geometrical or timing modifications, the sign does not necessarily preserve significant sign phases, which may become deleted or collapsed. These considerations highlight the conflict between two modeling perspectives: the semantic perspective for linguistics purposes, and the signal perspective for animation purposes.

In this article, we will propose a new database-driven animation technique which takes advantage of both linguistic and signal modeling perspectives to synthesize novel sign sequences. We describe the structure of our database as containing both raw information in the form of motion files, and semantic information in the form of segmented and annotated data. In our database queries, we search both sets of information to arrive at signal selection and extraction through semantic query refinement. The selected motion is then passed to the animation engine to provide a real-time animation stream, taking into effect the captured sign data, as well as coarticulation concerns during transition phases.

The rest of this paper is organized as follows: in section 2, we briefly discuss related work, and then continue with the mainstay of this paper; in section 3, we describe our methods for modeling and annotating our French Sign Language (LSF) data; in section 4 we present our database architecture; section 5 describes the animation process; experiments and results are presented in section 6 for a subset of LSF sequences; and lastly we conclude and give future perspectives for this type of work.

2 Related Work

Previous work related to virtual intelligent agents is largely concerned with modeling communicative gestures and sign language signs, with high level specification languages for communicating agents, and with the data-driven animation of virtual characters.

Several gesture taxonomies have already been proposed in [1] and [2], some of which rely on the identification of specific phases that appear in co-verbal gestures and sign language signs [3]. Recent studies dedicated to expressive gesture rely on the segmentation and annotation of gestures to characterize the spatial structure of a sign sequence, and on transcribing and modeling gestures with the goal of later re-synthesis [4].

Studies on sign languages formed early description/transcription systems, such as those in [5] and [6]. More recently, at the intersection of linguistics and computation, gestures have been described with methods ranging from formalized scripts to a dedicated gestural language. In the JACK framework, the NPAR language has defined user commands for controlling virtual agents using natural-like language [7]. Further, the BEAT system [8], as one of the first systems to describe the desired behaviors of virtual agents, uses textual input to build linguistic features of gestures to be generated and then synchronized with speech. Gibet et al. [9] propose a gesture synthesis system based on a quantified description of the space around the signer; using the HamNoSys [6] sign language notation system as a base, the eSign project has further designed a

motion specification language called SigML [10]. Other *XML*-based description languages have been developed to describe various multimodal behaviors, some of these languages are dedicated to conversational agents behaviors, as for example MURML [11], or describe style variations in gesturing and speech [12], or expressive gestures [13]. More recently, a unified framework, containing several abstraction levels has been defined and has led to the XML-based language called BML [14], which interprets a planned multimodal behavior into a realized behavior, and may integrate different planning and control systems.

Passing from the specification of gestures to their generation has given rise to a few works. Largely, they desire to translate a gestural description, expressed in any of the above-mentioned formalisms, into a sequence of gestural commands that can be directly interpreted by a real-time animation engine. Most of these works concern pure synthesis methods, for instance by computing postures from specification of goals in the 3D-space, using inverse kinematics techniques, such as in [9], [15], [16]. Another approach uses annotated videos of human behaviors to synchronize speech and gestures and a statistical model to extract specific gestural profiles; from a textual input, a generation process then produces a gestural script which is interpreted by a motion simulation engine [17].

Alternatively, data-driven animation methods can be substituted for these pure synthesis methods. Most of the previous work on pure synthesis methods propose edition and composition techniques, with an emphasis of the re-use of motion chunks and adaptation of captured motion for creating new sequences of motion. Very few approaches deal with both motion-captured data and their implicit semantic content, and nearly nothing concerns communicative gestures.

Arikan et al. use, in [18], a semi-automatic annotation algorithm divided into two parts: the first part consisting of a manual annotation, and the second using their Support Vector Machine. In [19], Chao et al. present a different annotation approach for Tai Chi Chuan movements, which consists of building a Motion Index Table composed of Motion Clips and defining a Basic Motion Text as a set of sentences. Barbič et al. in [20] segment a motion sequence into distinct behaviors (such as walking, running, etc.) based only on the information available within this sequence. In [21], Morales stores motion captured data after converting it to an XML format. Chung et al. apply the same concept in [22] using a standard mark-up language, the Motion Capture Mark-up Language.

In this paper we propose a new database approach to generate a set of French Sign Language signs. In order to optimize sign-component access, we index our data set by signs' semantic structures, as well as by the spatial and kinematic features of the raw motion-captured data.

3 Sign Language Modeling

3.1 Spatio-temporal dimensions of sign languages

As sign languages are by nature spatial languages, forming sign strings requires a signer to understand a set of highly spatial and temporal grammatical rules and

inflection processes unique to a sign language. We can separate plain signs that do not use space semantically (like the American Sign Language sign HAVE which does not make any notable use of space other than which is necessary for any sign) from signs that incorporate depiction. This second group of signs includes the strongly iconic signs known as depicting verbs (or classifiers), which mimic spatial movements, as well as size-and-shape specifiers, which concern static spatial descriptions.

Moreover, indicating signs like indicating verbs and deictic expressions require the signer to interface with targets in the signing space by effecting pointing-like movements towards these targets. Indicating verbs include such signs as the LSF sign INVITER, in which the hand moves from the area around the invited party toward the entity who did the inviting [23]. Depending on the intended subject and object, the initial and final placements of the hand vary greatly within the signing space. Deixis, such as pronouns, locatives, and other indexical signs are often formed with a pointed index finger moving toward a specific referent, though other hand configurations have been reported in sign languages, such as American Sign Language [24].

3.2 Sign Segmentation

The signs of any sign language can be decomposed into the components of hand configuration, placement, orientation, facial expression, gaze, and others [25]. For the purposes of synthesis, we can consider each of these components to constitute a separate track (or grouping of tracks) that control the movement of a part of the body during a sign and its surrounding sign stream. Further, we can divide these components by timing units that are present within the sign and between signs. For the purposes of this paper, we consider the division between linguistic timing segments, i.e., those that constitute a sign from a sign language, and non-linguistic, transitional segments, i.e., those that occur between signs and offer no grammaticalized semantic information; we identify the linguistic segments as strokes, and the non-linguistic segments as transitions.

Naturally, when constructing sign strings from stored data, transitions must be created dynamically, as they rely heavily on the phonology of both the preceding and following sign for their path and timing values, among others. Though these segments are non-linguistic and heavily dependent on their linguistic counterparts, this is not to say that they hold a lesser significance in the sign stream. To the contrary, transitions in sign languages are active parts of the discourse, differing greatly from the easily hidden silence that spoken language discourse uses as its transition. As sign language transitions are just as visible as the signs they connect, they must importantly be fluid movements that allow the sign stream to continue intelligibly.

Therein we find the difficulty of our present study - that only a very small part of the transition between two captured signs, if any of it, can ever be used in later synthesis sessions, since each sign will be taken out of its recorded context, and the context-dependent transition will therefore be rendered useless. For example, in the situation that we replace a single sign produced in front of the signer's

chest with another sign produced near the signer's head, the existing transition towards the chest area of the signer cannot be used, and must be dynamically replaced by a new transition that moves towards the head.

In desiring to separate the linguistic parts of our captured data from the non-linguistic ones, we follow the phase representation of Kendon, and the model of Kita, and define a phrase as a sequence of signs, each of which contains a preparation, a stroke and a retraction. This can be represented as

$$Unity = Phrase\{Phrase\} \tag{1}$$

$$Phrase = [Preparation]\{Stroke\}[Retraction] \tag{2}$$

where {} is an operator of repetition, with zero or more than one occurrence, and [ ] is an optional element.

3.3 Annotation Tools

The annotation process includes identifying building-block motion units following the above phrase structure. Within these motion segments, it is possible to describe the motion of the whole body as a single unit, or separate the motion into multiple channels, each describing a specific part of the body (i.e., the upper arm, wrist, hand, etc.), as has been done in both the linguistics [26] and computer science realms [27]. In any case, the segmentation process divides pertinent phases with time stamps, either manually or automatically.

In our work, we have manually annotated two LSF video files with corresponding motion capture data. Following the above description of Kendon and Kita's analyses of motion parts, we have separated each from its respective preparation and retraction strokes within the data. In this case, our work was carried out on an XML annotation system developed for the linguist, ELAN[28], though an animation perspective-driven program, ANVIL[29], also works for this task.

4 Database Architecture

An overview of our system, shown in Figure 1, illustrates the main methodology of the approach. In order to accelerate the motion retrieval process, we propose a two-level indexation process of the data: the first one uses the semantic structure of the data, and the second one is based on the different ways of accessing the raw motion data for synthesis purposes. Once motion is retrieved and selected, it goes into an animation process which is run in real time.

4.1 Indexation of the data

In this section, we detail the representation of motion data in the databases. The use of two different databases (a raw database and a semantic database) is justified by the fact that the two types of data (non-segmented raw data and segmented semantic data) are handled differently.

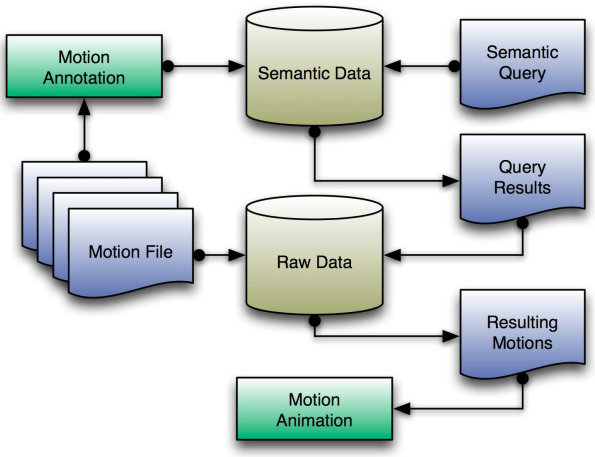


Fig. 1. Overview of the architecture

Semantic Indexation After semantically annotating our video and motion-capture files, their annotation values are stored in a semantic database in one of two ways: to optimize document load time, annotation files can be stored sequentially in the database without modifications, while storing each file as an individual node in the database will render a faster query time. The selection of storage methods is dependent on the goals of the test, but given the small size of our semantic database, we chose the first method.

Raw Data Indexation Motions are traditionally stored on the hard drive with various formats (.bvh, .fbx, .asf/amc, etc.), and interpreting these files amounts to building an internal representation of the motion in CPU memory. In our system, this internal representation is a collection of poses, with each pose being composed of a root position and an ordered vector of quaternions (joint rotations). This representation is globally consistent, provided that all the poses share a common hierarchical structure which is commonly named "bind pose". The time needed to read a motion file into this internal representation depends on the complexity of the parser and the amount of geometrical computation (for instance, cumulation of local transformations, switching from Euler angles to quaternions, etc.). This time is usually far from being negligible, and prevents dynamic loads for interactive applications. Traditional databases function with a set of paired-value data: one key, preferably unique, is associated to the useful data, i.e., the motion. The simplest way to then proceed is to associate the whole motion file with a unique key which can be chosen as the name of the original data file. The whole sequence would then be handled by the database manager, and stored on the hard drive. This approach assumes that when retrieving the motion, all the data will be reconstructed in the CPU memory. However, in the

context of a real-time animation controller, where small pieces of the motion are dynamically combined to achieve a desired goal, this approach is no longer efficient. In our system, such files are loaded and interpreted a single time, and stored as a sequence of bits in our database, which is designed to handle several access modes to the data. The following modes determine particular fragmentation approaches that are suited to particular types of queries:

- *FullAccess* - one single motion is associated with one unique key. The first part of the value is the bind pose, followed by a sequence of poses formed by a root position and one quaternion per joint.
- *ByFrame:p* - the whole sequence of frames is divided into packets of p frames; one motion will produce $\left[\frac{n}{p}\right]$ entries in the database if n is the total number of frames in the motion.
- *ByJoint* - the whole motion is divided into m packets, each one corresponding to a given joint of the bind pose. The first joint usually also contains the root position, though this may change if several joints also have some translational degrees of freedom.
- *ByJointandFrame:p* - the whole motion is decomposed into packets that account for the motion of one joint over a sequence of p frames. One motion will produce $m * \left[\frac{n}{p}\right]$ entries in the database if n is the total number of frames and m the number of joints in the motion.

Fragmenting motions in the database is interesting because only a small portion of the motion, that containing the query results, is reconstructed in the memory. However, this operation has a cost because it can multiply the number of entries in the database, thus increasing the search time and the index size. In our framework, we allow how data is stored in the database to be modified online, effecting the best access mode for a given application. Hence, if the application is likely to access only specific subparts of the skeleton (like the hand for instance) over longer motions, the *ByJoint* access mode is optimal, and would be utilized.

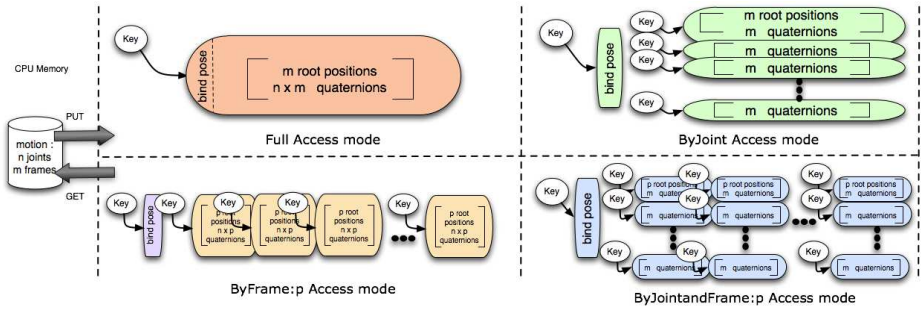


Fig. 2. Different access modes to the database

4.2 Motion Retrieval from Semantic Query and Animation

The process of retrieving data from the database is divided into two parts. The first part of the process consists of querying the semantic database, allowing us to extract information contained in the segmented files. Retrieving data from the semantic database is achieved by specifying one-condition or multiple-condition queries, called *PhaseQuery*. The query results are expressed as sequences of segments, which each segment a name as well as starting and ending time stamps. A simple one-condition query often returns a large number of motion candidates, whereas a multiple-condition query tends to return fewer candidates. In the second part of the data-retrieval process, the query results are interpreted so that each segment triggers access to the raw database, producing the segment's corresponding motion frames.

Concerning the time access to the whole database, we expect that there is a compromise between the time-processing of the semantic data and the time-processing of the raw data, with the complexity of the request being directly linked to the number of potential results. For example, in our real-time motion synthesis context, our goal is to concatenate signs that are each related to a lexical unit. As the signs are composed of a sequence of segments as presented above (*Preparation*, *Stroke*, *Retraction*) we attempt to preserve the meaningful segments (*Stroke*) and to create the transitions between these strokes by interpolation.

During the selection process, isolated signs are retrieved from the whole database by XML queries. The selection of the sign from among several candidates is expedited thanks to pre-processed information which is stored in the database. Largely, the pre-processing extracts temporal and spatial information from the transitory segments (i.e., Cartesian positions and directions at the beginning and end of the different segments, as well as their duration and kinematics profiles), allowing us to consider the coarticulation effects of a natural sign stream. In addition, statistical information about the mean values and distribution of these transitions is stored in the database for the purposes of timing invariant aspects of the transition segments, such as their mean duration.

For each pair of selected signs, a concatenation algorithm utilizes the *Retraction* phase R_i of the first sign $Sign_i$ and the *Preparation* phase P_j of the second sign $Sign_j$. This algorithm (see below) extracts the first m_i frames of the phase R_i and the last m_j frames of the phase P_j, so that the transition segment can be interpolated between the p last frames of the first sign and the p first frames of the second sign. The different parameters of the algorithm are chosen according to the pre-processed information discussed above, i.e., statistical and local data of the segments.

5 Experiments and Results

5.1 Data Acquisition and Analysis

The motion data with which we conducted this study consists of 44 minutes of motion-captured LSF signs produced by a single deaf signer. Based on a unique

Algorithm 1 Concatenation Algorithm between $Sign_i$ and $Sign_j$

1: $(P_i, S_i, R_i) \leftarrow ProcessQuery(Sign_i)$
2: $(P_j, S_j, R_j) \leftarrow ProcessQuery(Sign_j)$
3: $R'_i[F_1 : F_{m_i}] \leftarrow ExtractMotion(R_i, m_i)$
4: $P'_j[F_{n-m_j} : F_n] \leftarrow ExtractMotion(P_j, m_j)$
5: $FirstMotion \leftarrow InsertMotion(P_i, S_i)$
6: $TransitoryMotion \leftarrow Interpolate(R'_i[F_{m_i-p+1} : m_i], P'_j[F_1 : F_p])$
7: $SecondMotion \leftarrow InsertMotion(S_j, R_j)$

motion-capture configuration, two synchronized data streams were computed from the raw captured data: a video stream as well as a three-dimensional animated skeleton stored in the BVH motion format. The video stream was used for manual annotation of the signing sequences, while the three-dimensional BVH data served as a base for numerical computation and comparison of the motions.

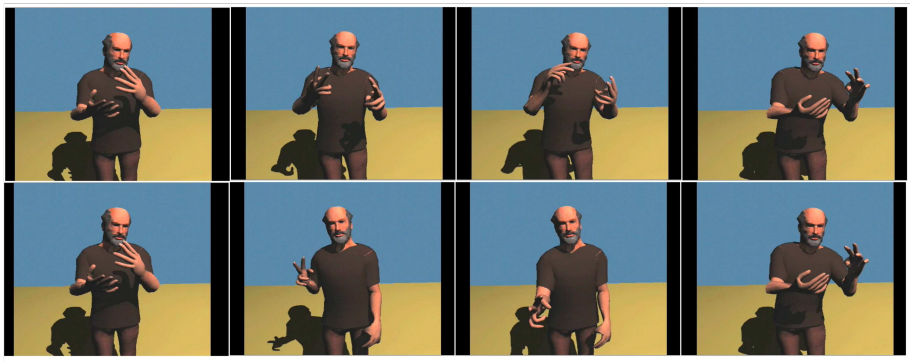


Fig. 3. Frames captures of the original and the first replacement scenario. Upper row: the end of the sign LE-TEMPS-PREVU, the beginning and the end of the sign BRETAGNE, and the beginning of the sign MATIN. Lower row: the end of the sign LE-TEMPS-PREVU, the beginning and the end of the sign VANNES, and the beginning of the sign MATIN.

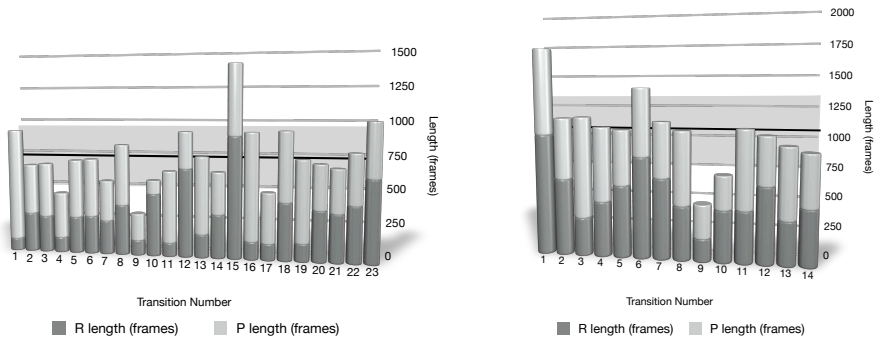
Lexically, the sequences are mainly composed of a succession of linguistic units, read LSF signs. These sequences depict weather forecast presentations and a recitation of different cities in France. An example gloss sequence is give below:

.../ATTENTION/AUJOURD'HUI/SIX JUILLET/LE-TEMPS-PREVU/ "BRETAGNE"/MATIN/NUAGES/APRES-MIDI/PLUIE/"DEMAIN"/ SOLEIL-BRILLER/CHAUD-SEC-FERA/NAGER/CE-SERA-LE-MOMENT/ "VENDREDI"/NUAGES/NUIT/EN-MER/"MATIN"/BROUILLARD/...

Two scenarios for processing were developed for the purposes of this study. The first replaces the quoted words above with the city signs VANNES, MAR-

SEILLE, RENNES, and LE MANS, respectively. These replacements were chosen especially to be challenging, as they require the right and left hands to perform very different actions before and after the replaced signs. Our second scenario switches weather predictions around within the original production, so that the prediction for "today" becomes the prediction for "Friday" and vice versa. Again, this manipulation requires that the two hands take different positions than in the original monologue, such as going from rest to action.

As part of our attempt to make novel transitions between recalled signs appear fluid and natural, we performed a statistical analysis of the captured transitions to develop standards for novel transition animation. We analyzed the duration, displacement, and average velocity of the signer's hands over adjacent retraction and preparation segments, as well as over the combined transition. Interestingly, the length of each transition (in frames or milliseconds) remained rather constant across our data capture. The graphs below show the length of each transition in our two data sequences, broken into the length of its respective retraction and preparation segments.



(a) (b)

Fig. 4. (a) Météo sequence transition lengths, (b) Villes sequence transition lengths.

In the background of each graph, the heavy black line indicates the mean value of the length of the transition, and the gray band represents one standard deviation on either side of the mean. Comparing the two graphs to each other, we find similar standard deviations (224 for the Météo sequence versus 289 for the Villes sequence) at differing means (735 for the Météo sequence versus 1064 for the Villes sequence). We can conjecture that the Villes sequence had longer transitions because it was less like a natural sign sequence than the Météo sequence, which had a more realistic syntactic structure; the Villes sequence was formed solely of town names produced in sequence. We have thus assumed that the lengths of transitions in the course of natural conversations would follow the structure of transitions found in our Météo data, having a mean length of

approximately 735 frames distributed naturally around a standard deviation of 224 frames.

5.2 Data Annotation

Annotating our video-captured data was performed with the help of the ELAN annotation tool [28]. Using a nested tier structure, ELAN allows annotations to be entered as independent of one another, or as dependent on annotations in a parent tier. For our work, it was most important to ensure that named signs were divided into three associated segments that had a combined duration equivalent to that of their associated sign, and that were aligned exactly with that sign. Thus, a parent-child tier relationship was chosen, with sign names being represented on the parent tier, and Preparation, Stroke, and Retraction phases represented for each sign on a child tier.

The screen shot below shows the ELAN interface during the annotation process. At the top left, a video of an avatar recreation of the signer is used for visual reference of the signs' Preparation, Stroke, and Retraction segments. At the top right, positional data for the hand on three axes (X, Y, and Z) is shown synched with the video data and the annotations, which are entered on the tiers along the bottom on the image. The highlighted portion of the annotation tier and the positional data represents the Stroke phase of the sign VANNES.

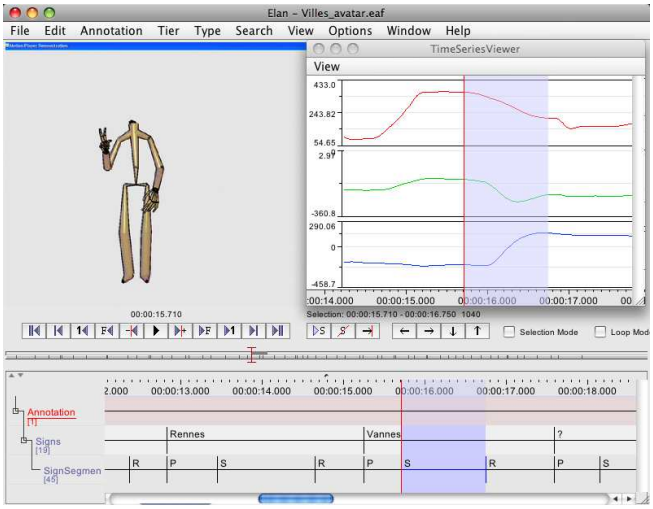


Fig. 5. A screen shot of the ELAN annotation tool. This annotation file contains three tiers of annotations (bottom) and references a video file for viewing captured signs (top left), as well as motion data which is synchronized with the semantic data using an ELAN sub-tool for later retrieval from the database.

5.3 Motion retrieval and Animation

We measured the computational time needed to retrieve and animate motions in order to analyze the efficiency of the proposed architecture; specifically, we were interested in the time required to query the semantic database, the time required to retrieve motions from the raw database, the time required to concatenate the retrieved motions, and the overall time of access. We tested our architecture on the two scenarios explained in section 5.1 and computed the time needed to animate these scenarios. The results are detailed in Table 1, which shows that the time needed to retrieve motions from the semantic database and the raw database is much smaller than the time needed to concatenate the retrieved motions. It is important to note that the concatenation process consists of aligning two motions chunks (in position and orientation), concatenating these chunks, and finally generating a new bigger chunk. Thus, the concatenation time is higher than the retrieval time and depends directly on the number of chunks to be concatenated, as well as each's size in frames; in our tests, *Scenario1* required 8 concatenations and *Scenario2* required 3.

Table 1. Querying the Databases

	Semantic Query(ms)	Number of Frames	Motion Retrieval(ms)	Concatenation Time(ms)	Total Time(ms)
Scenario 1	557	7050	1786	13856	20492
Scenario 2	144	6094	1418	5337	8733

The experiments were run on a Macbook Pro dual-core 2.4 GHz running Mac OSX 10.5.6 and equipped with 4 GB of memory. As for the choice of the databases, we tested two different APIs: Oracle Berkeley DB and Tokyo Cabinet[3]; both engines led to approximately similar results. In all our tests a hash map was used as an index structure to recover the data.

6 Conclusion

The work described here stems from the problem confronted when generating sign language signs by retrieving motions in real-time from motion capture databases. Our proposed new methodology, simultaneously using a semantic and a raw-motion database, takes into account the way motion data is indexed in these two databases.

We have proposed an overall architecture that combines different access methods for the databases, from querying the semantic and raw databases, to the animation process. Our XML description of the sign language utterances relies on an annotation process performed on a video stream. Then, sign language signs were synthesized by retrieving motions from semantic queries of the

[3] http://tokyocabinet.sourceforge.net

databases. The retrieved motions were then concatenated at the intersection of the transition segments, thus preserving the meaningful segments of the signs and providing a co-articulation processing of the sequence of signs. The experimental results were quantitatively evaluated for replacement scenarios which require that the transitions between signs be able to adapt to high spatial variations; they show that our proposed architecture is suitable for real-time applications. Moreover, we show that the efficiency of the synthesis can be increased if data is pre-processed.

In future work, we intend to generalize this approach to larger databases of French Sign Language signs and gestures containing both semantic and raw data. We also want to investigate other indexing techniques, and to implement different motion retrieval techniques. This will allow us to produce more complex real-time animations within different contexts and story-telling scenarios, and to evaluate these animations from a quantitative as well as from a qualitative point of view.

References

1. A. Kendon, *Tools, Language and Cognition*, chapter Human gesture, pp. 43–62, Cambridge University Press, 1993.
2. D. McNeill, *Hand and Mind - What Gestures Reveal about Thought*, The University of Chicago Press, Chicago, IL, 1992.
3. S. Kita, I. van Gijn, and H. van der Hulst, Movement Phase in Signs and Co-Speech Gestures, and Their Transcriptions by Human Coders, in *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pp. 23–35, London, UK, 1998, Springer-Verlag.
4. M. Kipp, M. Neff, K. H. Kipp, and I. Albrecht4, Toward natural gesture synthesis: Evaluating gesture units in a data-driven approach, in *Intelligent Virtual Agents (IVA'07)*, pp. 15–28, 2007.
5. W. C. Stokoe, *Semiotics and Human Sign Language*, Walter de Gruyter Inc., 1972.
6. S. Prillwitz, R. Leven, H. Zienert, T. Hanke, and J. Henning, *Hamburg Notation System for Sign Languages - An Introductory Guide*, University of Hamburg Press, 1989.
7. N. Badler, R. Bindiganavale, J. Bourne, M. Palmer, J. Shi, and W. Schuler, A parameterized action representation for virtual human agents, in *Embodied Conversational Agents*, pp. 256–284, MIT Press, 2000.
8. J. Cassell, J. Sullivan, S. Prevost, and E. F. Churchill, *Embodied Conversational Agents*, The MIT Press, 2000.
9. S. Gibet, T. Lebourque, and P. Marteau, *Journal of Visual Languages and Computing* **12**, 657 (2001).
10. R. Elliott, J. Glauert, V. Jennings, and J. Kennaway, An Overview of the SiGML Notation and SiGML Signing Software System, in *Workshop on the Representation and Processing of Signed Languages, 4th Int'l Conf. on Language Resources and Evaluation*, 2004.
11. A. Kranstedt, S. Kopp, and I. Wachsmuth, MURML: A Multimodal Utterance Representation Markup Language for Conversational Agents, in *Proceedings of the AAMAS02 Workshop on Embodied Conversational Agents - let's specify and evaluate them*, Bologna, Italy, 2002.

12. H. Noot and Z. Ruttkay, *Int. J. Hum.-Comput. Stud.* **62**, 211 (2005).
13. B. Hartmann, M. Mancini, and C. Pelachaud2, *Lecture Notes in Computer Science : Gesture in Human-Computer Interaction and Simulation* **3881/2006**, 188 (2006).
14. H. Vilhalmsson, N. Cantelmo, J. Cassell, N. Chafai, M. Kipp, S. Kopp, M. Mancini, S. Marsella, A. Marshall, C. Pelachaud, Z. Ruttkay, K. Thorisson, H. van Welbergen, and R. van der Werf, The Behavior Markup Language: Recent Developments and Challenges, in *IVA 2007*, 2007.
15. D. Tolani, A. Goswami, and N. I. Badler, *Graphical Models* **62**, 353 (2000).
16. S. Kopp and I. Wachsmuth, *Journal Computer Animation and Virtual Worlds* **15(1)**, 39 (2004).
17. M. Neff, M. Kipp, and I. Albrecht, *ACM Transactions on Graphics* **27(1), article 5**, 233 (2008).
18. O. Arikan, D. A. Forsyth, and J. F. O'brien, *ACM Transactions on Graphics* **22**, 402 (2003).
19. S.-P. Chao, C.-Y. Chiu, S.-N. Yang, and T.-G. Lin, *Comput. Animat. Virtual Worlds* **15**, 259 (2004).
20. J. Barbič, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard, Segmenting motion capture data into distinct behaviors, in *GI '04: Proceedings of Graphics Interface 2004*, pp. 185–94, Ontario, Canada, 2004, Canadian Human-Computer Communications Society.
21. C. R. Morales, Development of an XML Web Based Motion Capture Data Warehousing and Translation System for Collaborative Animation Projects, in *WSCG '01: International Conference in Central Europe on Computer Graphics and Visualization*, pp. 168–173, 2001.
22. H.-S. Chung and Y. Lee, *Computer Standards and Interfaces* **26**, 113 (2004).
23. *La Langue des Signes*, IVT.
24. V. Smith, Unpublished paper on deictic expressions in ASL, 2008.
25. S. K. Liddell and R. E. Johnson, *Sign Language Studies* **64**, 195 (1989).
26. S. K. Liddell and R. E. Johnson, Sign Language Phonetics: Archiecture and Description, Forthcoming, a.
27. A. Heloir and S. Gibet, A Qualitative and Quantitative Characterization of Style in Sign Language Gestures, in *Proc. of GW 2007*, LNCS, 2007.
28. ELAN Linguistic Annotator, http://www.lat-mpi.eu/tools/elan/.
29. Anvil, the Video Annotation Research Tool, http://www.anvil-software.de/.