Requirements for a Gesture Specification Language

A comparison of two representation formalisms

Alexis Heloir and Michael Kipp

DFKI, Embodied Agents Research Group
Campus D3 2, 66123 Saarbrücken, Germany
`firstname.surname@dfki.de`

Abstract. We present a comparative study of two gesture specification languages. Our aim is to derive requirements for a new, optimal specification language that can be used to extend the emerging BML standard. We compare MURML, which has been designed to specify coverbal gestures, and a language we call LV, originally designed to describe French Sign Language utterances. As a first step toward a new gesture specification language we created EMBRScript, a low-level animation language capable of describing multi-channel animations, that can be used as a foundation for future BML extensions.

Key words: embodied conversational agents, gesture description language, comparative study

1 Introduction

Describing human movement is a challenging task, given the many degrees of freedom of the human body. When using embodied agents in a human-computer interface context, a formal description of gestures is needed to ensure a faithful rendering by the underlying character animation engine. The design of a gesture description language is determined by three factors: the producer (human author or generation module) of the language wants it to be expressive and easy to use, the consumer (animation module) requires it to be complete, precise and convenient to interpret and, finally, there is usually an underlying theory that directs the language design. The behavior markup language (BML) [1, 2] offers such a specification. However, the current version of BML focuses on the problem of temporal synchronization between modalities, whereas the question of how to describe the surface form of a gesture is still open. In order to get a better understanding for how BML must be extended toward a complete specification of gestural form, we compare in this paper two existing formalisms for specifying human gestures. The first one, MURML, has been designed to specify coverbal gestures for an embodied conversational agent [3]. The second one has been designed to describe French sign language [4] and will be called LV in the further discourse. Both models have a similar theoretical background: sign language phonology. MURML bases some gesture description elements on HamNoSys [5]. LV is based on the Movement-Hold model by Liddel et Johnson [6]. The insights provided by this comparison will drive the design of a future BML extension. This extension does not yet exist, but its specification will be supported by EMBRScript, an intermediate animation language whose concepts can be used as building blocks to formally describe the future BML extension.

2 Related Work

Recent research has identified three fundamental layers of processing which facilitates the creation of generic software components [1, 2] for ECA applications: intent planner, behavior planner and surface realizer, as depicted in Fig. 1. In this framework called SAIBA[7], users work on the level of intent planning and behavior planning and then dispatch high-level behavior descriptions in the behavior markup language (BML) to the realizer which transforms it into an animation. Because the behavior description is abstract, many characteristics of the output animation are left for the realizer to decide.

As depicted in Fig. 1, the realizer itself can be decomposed into three components: the motion resolver, the motion planner and the animation engine. The role of the motion resolver is to select the motion segments that best convey the abstract behavior specified in the BML input. If needed, the user may override the motion resolver by embedding the description of a desired motion segment in the BML input using an higher *level of description* [2]. For instance, MURML can be directly integrated into BML on level 1. The motion segment description is then sent to the motion planner. Its role is to timestamp every motion segment in order to guarantee inter-channel synchronization (e.g. between prominent syllable in speech and a particular point in a motion segment) and ensure realistic velocity profiles for the character's limbs. This results in a time-stamped motion description which can be processed by the animation engine. Finally, the animation engine computes a geometrical description (angles, morph targets etc.) of the characters's animation that can be rendered by a 3D graphics engine.

Any sign language/gesture generation system based on a behavioral definition needs to map its input data to trajectories and postures. Researchers have developed form-based description languages allowing to specify a wide range of gestures by symbolically composing form and movement primitives in a structured way.

A language aimed at providing an intermediate representation of motion segments should fulfill the following requirements:

– for the user: expressive and easy to use,
– for the animation engine: complete, precise and convenient to interpret

Researchers have focused on the development of representation systems for multimodal utterances [8, 4, 9, 3]. Although such languages where initially developed independently from the behavioral layers, it is possible to integrate them as

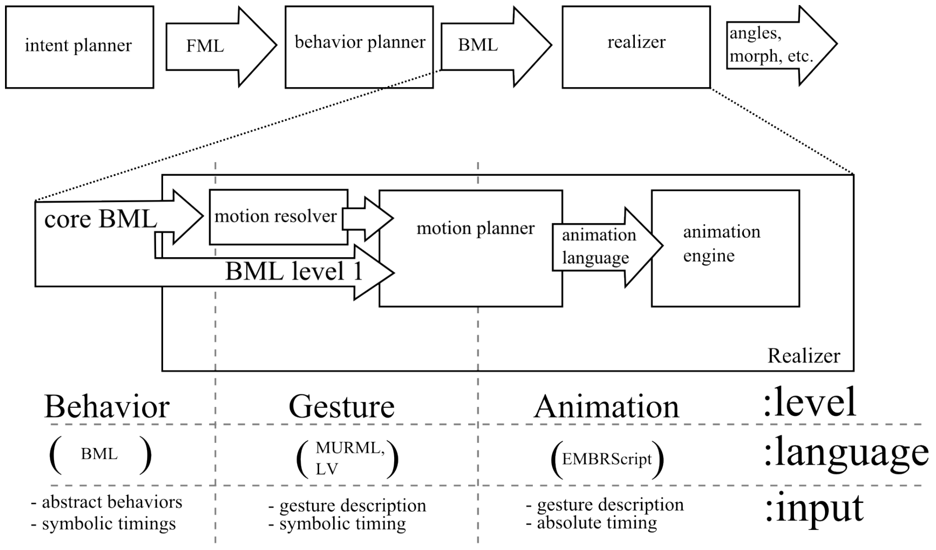

Fig. 1. Processing pipeline for producing human behaviors according to the SAIBA framework.

animation-centered description languages as extensions to existing higher-level behavior description languages like BML.

Early formal attempts to encode multimodal utterances for animated virtual characters were motivated by the need of automatic sign language production like the GESSYCA system [8]. In GESSYCA, simple gestures involving the arms and the hands can be described as an arrangement of formational parameters (hand shape, hand position and hand orientation) inspired by early studies dedicated to sign language phonology [10]. This pioneering work has been followed by the Signing Gesture Markup Language (SigML [9]), which itself stems from the HamNoSys linguistic notation system [5]. SigML can be described as mediating between a behavioral and a physical/anatomical sign description for the automatic production of sign language sequences. In 2001, Losson and Vannobel proposed a formal description language which we will call LV [4] in the further discourse. This language takes into account gestures and facial expression as well as the different gesture phases (preparation, stroke, hold and retraction) as described by Kendon [11]. More recently, Huenerfauth introduced a multichannel coding system for sign language called the Partition/Constitute (PC) formalism [12]. The system uses a two-dimensional grammar (one dimension representing time, the other representing channels) and introduces an explicit representation of coordination and non-coordination for a multichannel animation stream. However, the described concepts have not been formalized into a syntactic description to the best of our knowledge.

In parallel, many interactive applications featured virtual characters as a new human-machine interaction metaphor. Early attempts did not need elaborate multimodal specification models, as they mainly relied on pre-recorded or pre-specified animations, either obtained using motion capture or key-framed by a skilled animator [13–16]. These approaches used speech as the dominant modality to which all other modalities (gesture, facial expression etc.) would refer for temporal synchronization. Synchrony was achieved by starting the animation pieces simultaneously with the correlated verbal phrase. The nonverbal behaviors are referred to by unique identifiers and are drawn from a behavior database. In these systems, it is impossible to create nonverbal behaviors from atomic elements and to adapt their structure in the synchronization process.

The MURML gesture description language [3] starts from straightforward descriptions of the multimodal utterance in a XML-based specification language. Such a description contains the verbal utterance augmented by nonverbal behaviors including gestures. In MURML, the desired gesture can be described explicitly in terms of spatiotemporal features. In addition to gestures, further behaviors can be incorporated such as arbitrary body movement and facial animations given as sequences of face muscle values.

To sum up, it seems that so far both LV and MURML are the most elaborate description languages providing an implementable grammar. However, even if they share similar goals, there are significant differences in both their structure and expressiveness. We thus present in the following a comparison between the two languages.

3 Comparison of MURML and LV

We base our comparative study on a sample iconic gesture. This gesture, depicted in Fig. 2 may be used to describe a square-shaped structure like a fireplace frame or the structure of a bridge and we will call it **BRIDGE** in the following. The **BRIDGE** gesture conveys two changes in wrist position (one where the wrists follow a horizontal straight path and a one where they follow a vertical straight path) and one change in hand orientation (the back of the hand changes form pointing upward to pointing toward the sides) which occur before the second wrist position's change.

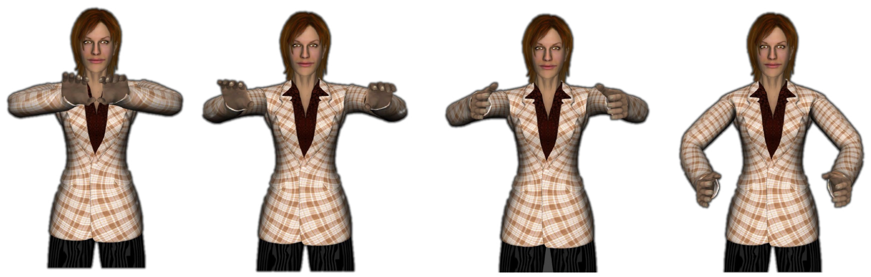

Fig. 2. Iconic **BRIDGE** gesture.

Both LV and MURML can describe multimodal gestures including hand-arms configurations and trajectories as well as facial expressions. In both systems, gesture timing is expressed symbolically and are resolved later on by the animation engine. Both systems limit their descriptions to the meaningful phases of the described gestures (stroke or independent hold). In an architecture like the one outlined in Fig. 1, the timing of non-meaningful surrounding phases such as retraction, preparation and dependent hold are left to the motion planner engine while the final realization is left to the animation engine.

3.1 MURML

MURML describes gestures in a tree structure of constraints and features, each of which specifies either a sub-gesture (atomic gesture) or a set of feature conveying the configuration of one or more modalities (facial animation parameter, hand shape, hand position, hand orientation). In MURML, two different types of movement constraints are provided in order to specify a feature over a certain period of time: A static constraint, which defines a postural feature that is to be held for a certain period of time and a dynamic constraint which specifies a significant submovement within a feature that is fluently connected with adjacent movement phases. The relationships between the feature constraints can be denoted by specific MURML elements like *parallel, sequence, repetition and symmetry.* Such constraints can be arranged in a flexible fashion.

The lower part of Fig. 3 shows a MURML description of the **BRIDGE** gesture presented in Fig. 2. In our example gesture, the hands are kept in the same configuration through the entire gesture, hand configuration is thus expressed in a separate branch under a *parallel* element. The remaining constraints describing the gesture are assembled in a *sequence* element. Features are located under a *static* or *dynamic* constraint depending on whether they are kept static or change during each sequential sub-motion. At the upper end of the tree, a *symmetry* constraint expresses the fact that the gesture is symmetrical, i.e. the description applies to both hands.

The structure of MURML encourages feature factorization through hierarchical arrangement of constraints and features. On the one hand, by allowing arrangements of arbitrary complexity between constraints and features, MURML can be viewed as a concise language which prevents duplication, but, on the other hand, because no restriction is imposed on how constraints can be arranged, MURML allows multiple (syntactically) valid descriptions for a single gesture. For instance, the tree depicted on the upper side of Fig. 3 shows a MURML description of the **BRIDGE** gesture which is valid, but against MURML's philosophy (the handshape feature is, for instance, redundant between the three atomic gestures).

We believe that the non-uniqueness of MURML descriptions for certain gestures and the a-priori arbitrary complexity of a gesture description is a challenging aspect both for an author and for implementing a realizer based on MURML. Furthermore, MURML does not introduce a *contact* constraint (see Sec. 3.2).

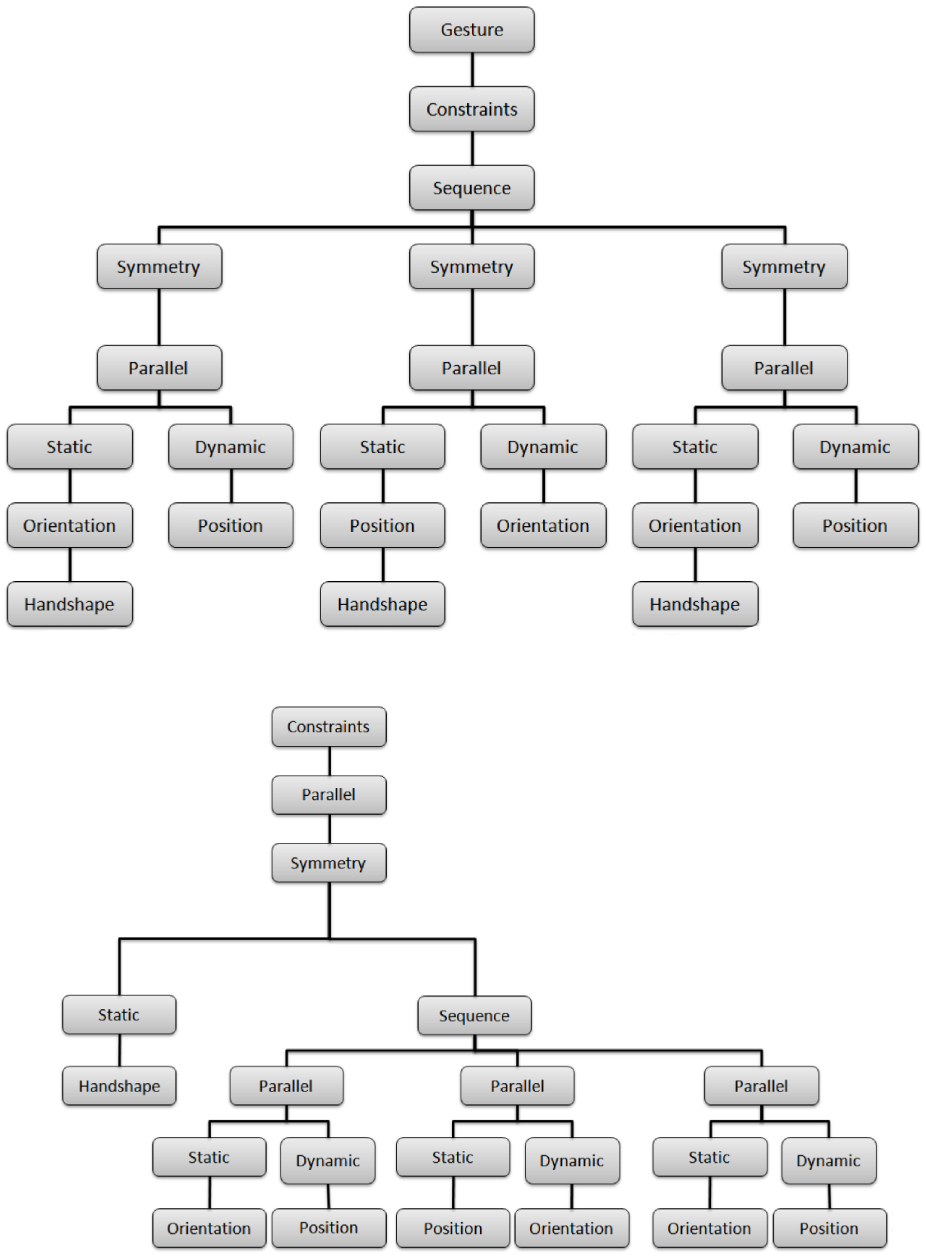

Fig. 3. MURML representation.

Finally, we believe that although very expressive, the complex tree structures required to describe the gestures may be challenging for users.

3.2 LV

LV follows the *movement-hold theory* which describes a gesture in terms of postures and transitions. The illustration in Fig. 4 shows the LV description of **BRIDGE**.

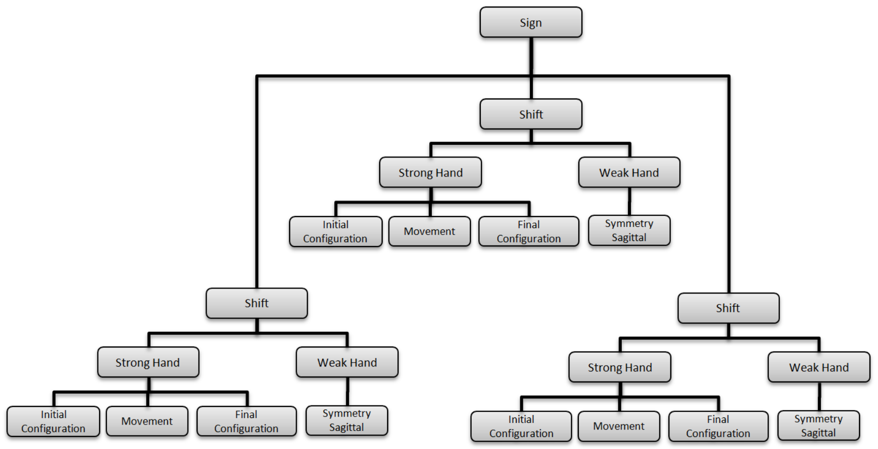

Fig. 4. LV Representation.

Like MURML, LV describes a gesture as a combination of features organized in a tree. However, as opposed to MURML, the feature arrangement is fixed and follows the following organization: a gesture (*sign*) is described as a succession of *shifts*. The *shifts* are bundles of hand features at the beginning and end of the sign and of the movement itself. Depending on its complexity, a *shift* can be described as one of the following three options:

– single hand shift primitive
– dominant hand shift primitive plus a specification for the weak hand and a spatial relationship (as it is the case for the description in Fig. 4)
– two shift primitives, one for each hand

The drawback of specifying hand configuration at the beginning and end of the *shift* is the redundancy between the end of one *shift* and the beginning of the following *shift* when a gesture is described as a sequence. The tree depicted on the right side of Fig. 4 highlights this redundancy problem.

In LV, the constraints conveying relationships between features are considered as attributes belonging to specific nodes. For instance, the repetition constraint is a feature of the *shift* element and all features under a shift are realized in parallel. For instance, shift elements belonging to the tree depicted in Fig. 4 aggregate the simultaneous movement of both weak hand and strong hand. Finally, LV offers a feature to represent *contact gestures*, i.e. gestures that touch a part of the body (expressed using landmarks) somewhere along their trajectory.

First, LV leads to unique description of gestures. Second, because LV uses assumptions about human modalities (e.g. symmetry is only related to arm/hands, the default hand is strong hand), it appears that its gesture description scripts have a comparatively flat tree structure. The following script presents the textual version of our example gesture.

```
Sign %BRIDGE.
 Manual:
  (Shift from: (HandSpec config: #C point: #ThumbRoot
           at: [Shoulders FrontProx Sagittal] ori: [f f & u])
           to: [Shoulders FrontProx IpsiSide]
         move: #Linear
     weakMove: #Symmetrical ),
  (Shift ...),
  (Shift ...).
```

We believe that a flat description is a desirable feature for a gesture description language, because it provides the user with a more intuitive way to specify the character's configuration using sets of constraints. However, the variety of modalities LV can address are rather limited (hand and basic face). Furthermore, LV uses a terminology and contains some elements that are too dependent on its underlying theory (e.g. sign language modifiers, grammatical modifier element).

3.3 Discussion

Based on the gesture description we presented in the sections above, we now draw a comparison between MURML and LV based on the complexity of their structure, the way dynamic elements are described, the presence of convenience features like arm-swivel or contact specification and the ability to specify co-occurring movements in the main animation. Our comparison is summarized in Table 1.

Although both LV and MURML use tree structures, we found descriptions provided by LV to be flatter. We believe that flat structures are more desirable for a high-level language dedicated to the specification of human gestures for two reasons: first, we believe that a flat structure is easier to author and leads to unique less ambiguous descriptions, second, although one could argue a flat structure narrows the range of expressiveness of the language, we believe that this limitation can be overcome using certain assumptions on human modalities (two hands, symmetry only applies on arms/hands, etc.).

Another issue is the redundancy introduced when describing dynamic features that are occurring sequentially in both LV and MURML. As described

Aspect	LV	MURML	Conclusion
Complexity of representation	Relatively flat structure	Deep, nested structure	Use flat structure
Redundancy	Start/end poses	None	Move to a pose-based representation
Arm swivel	---	Featured	Add feature for hand-arm configuration
Contact	Featured for start and end poses	---	Add static contact constraints
Secondary movement	Predefined set of secondary motions	Generic way to define e.g. hand shape changes	Allow both

Table 1. Summary of limitations of MURML and LV.

in paragraph 3.2 LV implies a duplication between the end pose of an atomic gesture and the start pose of the subsequent atomic gesture. in MURML, this problem can be avoided if the user carefully factorizes static and dynamic constraints. We believe high-level features conveying contact and arm swivel are particularly convenient for authors. LV provides a feature to describe contacts, MURML does not. MURML takes arm swivel into account, LV does not. Finally, both LV and MURML permit the specification of movement co-occurring with a main gesture (e.g. fingers wriggling while tracing the **BRIDGE** gesture). LV uses a set of pre-defined secondary movements as a parameter of a shift (e.g. wriggling, rubbing, counting). MURML can define such movements in a generic fashion using a subtree that is a sibling of the tree depicting the main gesture, both trees grouped under a *parallel* constraint.

4 The Animation Layer and EMBRScript

Having compared MURML and LV, we found that MURML was very expressive but probably too complicated for easy authoring and that LV is an elegant formalism for specifying gestures but may have limitations when it comes to multi-channel coordination. But how can we devise a powerful language without overloading the syntax in terms of structure and lexicon? We decided to start by first creating another layer of abstraction *underneath* the BML layer. We call this layer the *animation layer* and argue that this layer should have maximum expressivity without regard to the question whether it is easy to author and easy to read by humans. It should provide an abstract interface to character animation engines and form a basis for the future development of the BML layer which can then be built on top of it. This strategy allows us to more systematically define the BML layer using clearly defined building blocks. In this section we will outline the animation layer, using a concrete language we call EMBRScript.

4.1 Overview

In the processing pipeline depicted in Fig. 1 the animation engine is the last element which creates the animations sent to the graphics renderer. This animation engine can involve multiple motion generation mechanisms acting on different parts (channels) of the character to be animated. Because of the heterogeneity of the different motion generation mechanisms, we argue that a new layer of control, the *animation layer*, is necessary to keep the higher-level control layers (behavioral/functional) consistent and slim, while allowing a unified and abstract access to the animation engine, e.g. for the procedural animation of nonverbal behavior. EMBRScript is the language that provides this interface.

EMBRscript is a channel-oriented, pose-based gesture description language. Every animation is described as a succession of *key poses* timestamped with absolute times and spanning multiple channels. Contrary to LV and MURML which only depict meaningful phases of the gestures, EMBRScript does not make distinguish between gestures phases. Fig. 5 depicts a EMBRScript description of the **BRIDGE** gesture.

4.2 Description

A *key pose* describes the state of part of an animated character at a specific point in time using a set of constraints. A *key pose* spans over at least one channel, a channel represents one of the character's modalities. When spanned over multiple channels, the *key pose* expresses synchronization between them. A pose can be specified using a combination of four principal methods, each acting on a separate channel: skeletal configuration (e.g. reaching for a point in space, bending forward), morph targets (e.g. smiling and blinking with one eye), shaders (e.g. blushing or paling) or autonomous behaviors (e.g. breathing). In the example we show in Fig. 5, only the skeletal configuration is specified using spatial constraints (wrist position and wrist orientation). The skeletal configurations satisfying the constraints are computed in the animation engine using inverse kinematics.

For instance, in the EMBRScript description of the **BRIDGE** gesture in Fig. 3, two channels are depicted: The *wrist position* channel and the *wrist orientation* channel. The two first motion segments in both channels represent the preparation phase. They are both bounded by a different *key pose* which means that there are independent in time, in this case, the preparation phase corresponding to the wrist orientation channel is achieved before the preparation phase corresponding to the wrist position channel.

The subsequent key poses are bundling both the wrist position and the wrist orientation. The first key pose specifies the wrist's position change while its orientation is kept still, the second key pose expresses the wrist's orientation change while the wrist's position is kept still, the last key pose specifies the last wrist's position change while the wrist orientation is kept still.

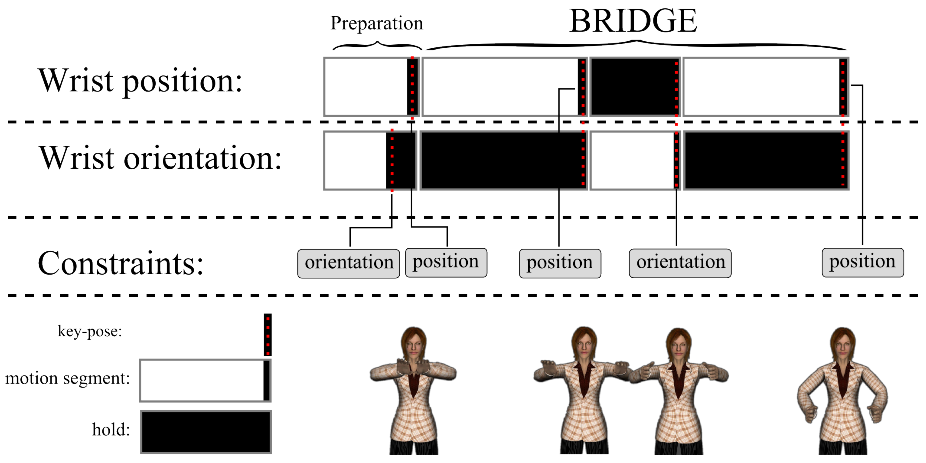

Fig. 5. This illustration displays an EMBRScript based description of the **BRIDGE** gesture. The description is spread over two channels, synchronization is done using key poses which can span several channels.

5 Conclusion

In order to identify and highlight the requirements for a high-level gesture specification language that could be integrated as a BML level of description, we conducted a comparative study over two gesture specification languages: MURML, designed to specify coverbal gestures and LV, designed to describe French Sign Language utterances. The study gave us insights regarding the desired structure and the level of expressivity of this new language. We found that it should have a flat structure and take advantage of the assumptions that can be made regarding the modalities of a human being. The language should take into account the multichannel nature of gestures while being able to express both synchronization and temporal independence between channels. The design of such a language is a work in progress which will build on the EMBRScript animation language. EMBRScript provides an interface to the animation layer which gives to users the possibility to describe fine-grained output animations without requiring a deep understanding of computer animation techniques [17]. In future work, the concepts of EMBRScript will be used as building blocks to formally describe fine-grained behaviors on the BML layer.

References

1. Kopp, S., Krenn, B., Marsella, S., Marshall, A.N., Pelachaud, C., Pirker, H., Thórisson, K.R., Vilhjálmsson, H.: Towards a common framework for multimodal generation: The behavior markup language. In: Proc. of IVA-06. (2006)
2. Vilhjalmsson, H., Cantelmo, N., Cassell, J., Chafai, N.E., Kipp, M., Kopp, S., Mancini, M., Marsella, S., Marshall, A.N., Pelachaud, C., Ruttkay, Z., Thórisson,

K.R., van Welbergen, H., van der Werf, R.J.: The behavior markup language: Recent developments and challenges. In: Proc. of IVA-07. (2007)

3. Kranstedt, A., Kopp, S., Wachsmuth, I.: Murml: A multimodal utterance representation markup language for conversational agents. Proceedings of the Workshop Embodied Conversational Agents, Autonomous Agents and Multi-Agent Systems (AAMAS02), Bologna, Italy (2002)
4. Losson, O., Vannobel, J.M.: Sign specification and synthesis. In: Proc. of Gesture Workshop. (1999)
5. Prillwitz, S.L., Leven, R., Zienert, H., Zienert, R., T.Hanke, Henning, J.: HamNoSys. Version 2.0. International Studies on Sign Language and Communication of the Deaf (1989)
6. Liddell, S.: American sign language : The phonological base. Sign language studies **64** (1989) 195–278
7. Vilhjálmsson, H., Thórisson, K.R.: A brief history of function representation from gandalf to saiba. In: Proceedings of the 1st Function Markup Language Workshop at AAMAS. (2008)
8. Lebourque, T., Gibet, S.: High level specification and control of communicative gestures: the gessyca system. In: Computer Animation, 1999. Proceedings. (26-29 May 1999) 24–35
9. Kennaway, R.: Synthetic animation of deaf signing gestures. In: Gesture Workshop. (2001) 146–157
10. Stokoe, W.: Sign language structure: An outline of the visual communication systems of the american deaf. In: Studies in Linguistic, Occasional Papers. Volume 8. (1960)
11. Kendon, A.: Some relationships between body motion and speech: an analysis of an example. In Siegman, A., Pope, B., eds.: Studies in Dyadic Communication. Pergamon Press, New York (1972) 177–210
12. Huenerfauth, M.: Representing coordination and non-coordination in an american sign language animation. In: Assets '05: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility, New York, NY, USA, ACM (2005) 44–51
13. Cassell, J., Vilhjalmsson, H., Bickmore, T.: BEAT: The behavior expression animation toolkit. In Fiume, E., ed.: SIGGRAPH 2001, Computer Graphics Proceedings. (2001) 477–486
14. Stone, M., DeCarlo, D., Oh, I., Rodriguez, C., Stere, A., Less, A., Bregler, C.: Speaking with hands: Creating animated conversational characters from recordings of human performance. In: Proc. ACM/EUROGRAPHICS Symposium on Computer Animation. (2004)
15. Kipp, M., Neff, M., Kipp, K.H., Albrecht, I.: Toward Natural Gesture Synthesis: Evaluating gesture units in a data-driven approach. In: Proc. of the 7th International Conference on Intelligent Virtual Agents (IVA-07), Springer (2007) 15–28
16. Hartmann, B., Mancini, M., Pelachaud, C.: Implementing Expressive Gesture Synthesis for Embodied Conversational Agents. In: Gesture in Human-Computer Interaction and Simulation. (2006)
17. Heloir, A., Kipp, M.: Embr: A realtime animation engine for interactive embodied agents. In et al., R., ed.: Proc. of the 9th Int. Conf. on Intelligent Virtual Agents, Springer (2009) 393–404