Defining the Gesticon: Language and Gesture Coordination for Interacting Embodied Agents

Brigitte Krenn, Hannes Pirker
*Austrian Research Institute for Artificial Intelligence (ÖFAI)
Freyung 6, A-1010 Vienna, Austria
{brigitte,hannes}@oefai.at

Abstract

In this paper we address problems of the automatic assignment of speech accompanying gestures and present solutions we have developed and still develop in the IST-project NECA. Special emphasis is put on the presentation of the central repository of information necessary for this assignment: the so called *gesticon*.

1 Introduction

The task of automatic gesture assignment discussed in this paper, can be described as follows: Given a dialogue between two or more embodied agents, specify their non-verbal behavior by automatically selecting ”appropriate” gestures and facial expressions from a given set. Take care of the temporal alignment of gestures with the spoken utterance and provide the information in a way that it subsequently can be used as input for an animation engine. The crucial task here is to design a gesture repository with representations general enough to be reusable in different multimodal generation systems and to be applicable in combination with different animation engines. What is required from the generation system is the availability of information on the dialogue structure, the dialogue related emotion, and the prosody and timing of speech.

Our discussion will be centered around

a) the design and representation structure of a central gesture repository, which we call *gesticon* in analogy to lexicon,[1]

b) the methods and strategies employed in gesture generation and the alignment of gestures with speech.

As regards a), we discuss what information shall be represented in the gesticon, and how this information shall be structured and represented. As regards b), we make proposals for gesticon-based gesture generation and gesture timing in collaboration with multimodal natural language and speech generation. In order to do so, we introduce a general purpose multimodal representation language, the RRL (Rich Representation Language, see `http://www.oefai.at/NECA/RRL`), which is the back-bone of the whole gesture-assignment process and functions as an interface to the individual system components. Both gesticon and RRL are represented in XML format, thus ensuring compatibility with a variety of existing representation and standardisation efforts of multimodal information. For an overview see (Pirker and Krenn, 2002). Coupling gesticon and RRL also allows us to design a component which makes the gesture representations and gesture generation strategies and methods independent from implementation details of individual system modules.

Thus, even though we describe gesture representation and gesture assignment in the context of the NECA system[2], our proposals are general in nature and not restricted to NECA.

The paper is organized as follows: To set the context, we briefly introduce the NECA project (section 2.1) and the architecture of the NECA system (section 2.2). In section 2.3 we give an outline of gesture encoding in the RRL (Rich Representation Language), a general purpose multimodal representation and scripting language which has been developed in the NECA project. In sections 3.1 to 3.6 the overall gesticon structure is defined and the organization of gesture relevant information is discussed. Gesticon entries are exemplified in section 3.7.

2 The NECA System

2.1 Outline

NECA (”Net Environment for Embodied Emotional Conversational Agents”) aims at the development of a toolkit that allows for time- and cost efficient implementation and adaption of Web-applications for the following scenario: Animated scenes are generated where two or more

[1] Other terms in use for a repository of gesture definitions are *gestuary*, a term coined by (deRuiter, 1998) and subsequently employed by (Kopp and Wachsmuth, 2000), and *gestionary*, a term used by Isabella Poggi (Poggi, 2002a) to refer to a dictionary of symbolic gestures, or *dictionary of gestures* such as ’The Berlin dictionary of Everyday Gestures’ (Posner et al., 2002) or ’The Nonverbal Dictionary of Gestures, Signs & Body Language Cues’ (Givens, 2002).

[2] `http://www.oefai.at/NECA/`

virtual human-like characters communicate with each other using expressive (emotionally rich) speech, gesture and facial expression.

Due to bandwidth restrictions, the use of lean player technologies is necessary.

For various reasons it is important that the system can easily be adapted to different player technologies. For instance, in different applications varied animation styles are preferred, the state-of-the-art in player technology is rapidly changing, improvements in bandwidth capacities increase the choice of web compatible player technology. Thus special emphasis needs to be put on keeping the influence of player-specific aspects as small as possible. In the two NECA demonstrators we currently work with two fairly different animation/player technologies, namely Charamel (`http://www.charamel.de`) and Macromedia Flash (`http://www.macromedia.com`).

2.2 Architecture

Because in NECA whole dialogues are planned in advance in the way a playwright designs a scene, a strict pipeline-architecture as depicted in Figure 1 can be employed. The information between modules is passed on using NECA's XML-compliant Rich Representation Language (cf. `http://www.oefai.at/NECA/RRL`, (Piwek et al., 2002)). First, the scene generation and an affective reasoning component (Gebhard et al., 2003) specify the dialogue acts to be produced and feed into the multi-modal natural language generator (M-NLG) (Piwek, 2003). M-NLG is responsible for the generation of the textual representation of the agent's utterances as well as the selection of semantically motivated gestures and emotion-driven facial expressions. Relevant information is encoded in the <function>-element of a gesticon entry.

The following concept-to-speech synthesis module (Schröder and Trouvain, 2003) does not only produce speech files containing emotional speech, but also provides full timing information, i.e., the exact position and duration of all phonemes, syllables, words, phrase boundaries and tonal accents.[3] This information is crucial for the Gesture Assignment module (GA). Here the final selection of gestures takes place and the animation is timed. Phonemes are mapped to visemes, tone accents are aligned with eyebrow raises, and selected parts of an intonation phrase are aligned with specific components of a gesture. The GA module makes use of information encoded in the <form>-element of a gesticon entry. Both M-NLG and GA make use of constraints encoded in the <restrictions>-element. After GA, a further component, the Animation Generator, produces the player-specific animation instructions. Player-specific information can be accessed via the <playercode>-element of a gesticon entry. While the input to this component is an RRL document, the output is code which can be directly rendered

[3]The speech synthesis system MARY can be tested online at `http://mary.dfki.de`

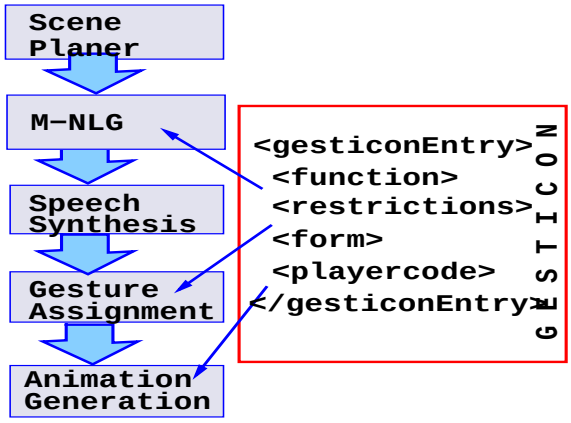

Figure 1: *Schematic diagram of NECA-architecture and how different modules make use of specific information provided in the gesticon.*

by the player employed.

2.3 Gesture Encoding in the RRL

Generally speaking, dialogue accompanying gesture generation is a two-step process.

1. During multimodal language generation, gestures are selected on the basis of the semantic and pragmatic content of the utterances, and are symbolically linked to whatever entity is appropriate, e.g. a word or a sentence.

2. Based on the prosodic and temporal information produced by a speech synthesis component a fine-grained alignment between the verbal and nonverbal communication systems is performed.

The relevant information is encoded by means of the RRL. The interplay of the different aspects of multimodal information is exemplified in the following. The RRL-snippet below illustrates the result of step 1) multimodal generation.

```
<gesture identifier="hipshift"
         id="g001"
         aligntype="seq_before"
         alignto="s001"/>
<gesture identifier="wave"
         id="g002"
         aligntype="par_end"
         alignto="s001"/>
<sentence id="s001">
  Hello, how are you?
</sentence>
```

Two classes of gestures – identifier="hipshift" and identifier="wave" – have been selected to accompany the sentence "Hello, how are you?". This is specified via the value of the *alignto* attribute, i.e., the unique "id" of the

sentence. The *aligntype* attribute designates the temporal relationship between gesture and anchor element. In this case the gesture ”hipshift” would be realised before sentence ”s001” starts and the gesture ”wave” should end when the sentence stops.

The speech synthesis system produces the according soundfile for the sentence, and also provides information on its internal structure (syllables and phonemes) as well as information on the location and type of tonal accents and prosodic phrase boundaries, represented in ToBI format (Baumann et al., 2001). See the RRL representation below.

```
<sentence id="s001" src="s001.mp3">
  <word id="w_1" accent="H*" pos="UH"
        sampa="h@l-'@U">
     Hello
     <syllable id="syl_1" sampa="h@l">
        <ph dur="75"  p="h"/>
        <ph dur="48"  p="@"/>
        <ph dur="100" p="l"/>
     </syllable>
     <syllable id="syl_2" sampa="'@U"
               stress="1" accent="H*">
        <ph dur="230"  p="@U"/>
     </syllable>
   </word>
  <prosBoundary breakindex="4"
                dur="200"
                p="_"
                tone="H-L%"/>
  <word id="w_2" ... />
  ...
</sentence>
```

With the availablility of exact phoneme durations the alignment-specifications produced by multimodal generation can now – in step 2) of the gesture assignment process – be transformed into concrete time-measures. More sophisticated alignto-types can be processed such as the alignment of a certain gesture component to the syllable which bears the nuclear accent of a phrase, information not available at step 1) of gesture processing.

The output then is an unambiguous specification of the animation stream, which is expressed by means of a subset of W3C's Synchronized Multimedia Integration Language (SMIL 2.0, `http://www.w3.org/TR/smil20/`), i.e., via a collection of <seq> and <par> elements. At this step all linguistic information is discarded and replaced by an <audio>-element which holds the name and duration of the speech soundfile. The symbolic alignment between gestures and language-related entities (e.g. sentences, words, syllables) is replaced by the specification of the exact temporal alignment between this <audio>-element and the according <gesture>-objects.

The example from above would render to:

```
<animationSpec>
  <seq>
     <gesture key="g023"
              identifier="hipshift"
              id="g001"
              dur="1650"/>
     <par>
        <audio src="s001.mp3"
               dur="1459"/>
        <seq>
          <!-- visemes -->
          <viseme identifier="v_h"
                  dur="75"/>
          <viseme identifier="v_@"
                  dur="48"/>
          <viseme identifier="v_l"
                  dur="100"/>
          <viseme identifier="v_@U"
                  dur="230"/>
          ...
        </seq>
        <gesture key="g012"
                 identifier="wave"
                 id="g002"
                 begin="259"
                 dur="1200"/>
     </par>
 <seq>
</animationSpec>
```

It can be seen, that the <sentence>-element of the input is now replaced by an <audio>-element, which refers to the soundfile to be played. The sequence of visemes is of course parallel to the audio-element, and the aligntype “par-end” for the “wave”-gesture is reflected by the temporal offset specified in its *begin*-attribute. The *id* attributes used as unique identifiers throughout the processing are redundant at this stage, and are kept for debugging purposes only.

3 The Gesticon

As already indicated, the gesticon is designed as a general repository of meaningful bits and pieces of animation descriptions which are relevant for the generation of dialogue accompanying nonverbal behaviour. In other words, the gesticon is the direct equivalent to the lexicon in language-processing systems. As the latter is a mapping from phonetic form to the meaning of words, the gesticon represents the mapping between the form and the semantics of a gesture. In analogy to words in a dictionary, gesticon entries store information about the form (phonology), the meaning (semantics), the combinatory properties (syntax) and the pragmatics of gestures. Thus our conception of gesticon corresponds to Poggi's notion of (gesture) 'lexicon'. In (Poggi, 2002b) it reads

> In a “codified” communication system, the

> signal-meaning link is shared and coded in the memory of both a Sender and an Addressee (as it is the case, for example, with words or symbolic gestures) and a whole set of these links makes a "lexicon".

Note though, that Poggi's work focuses mainly on a verbal description of symbolic (emblematic) gestures, i.e., gestures with a conventionalized meaning within a certain community such as 'thumbs up' meaning 'o.k.' in many western countries. In contrast we aim at a machine readable gesture repository, which functions as the basic resource for the automatic generation of all different types of gestures. With the gesticon we propose the foundations for a framework for the uniform symbolic representation of different nonverbal communication systems such as gesture and facial expression. Without doubt, descriptive work such as the one by Poggi or the descriptions available in the Berlin dictionary of Everyday Gestures (Posner et al., 2002) will be valuable resources to instantiate the gesticon structure. As a precondition, however, these works need to be made machine-readable. Another open question is how effectively the textual descriptions can be transformed into appropriate entries for automatic gesture generation.

In the following we present the general structure of a gesticon entry and discuss the representational details of entries for facial expression and gesture. An illustrative example is provided in section 3.7. The gesticon is represented in XML format. Each entry comprises a form, a function and a restriction element, and pointers to player-specific representations. The fact that currently only information on facial expressions and hand-arm gestures is represented in the gesticon results from the NECA context where animated characters do not move within the scene.

3.1 Overall Structure of a Gesticon Entry

We propose the following overall structure for a gesticon entry.

```
<gesticonEntry>
      <verbatim/>
      <function/>
      <form/>
      <restrictions/>
      <playercode/>
</gesticonEntry>
```

The attributes *key* and *identifier* in the gesticonEntry are both used for naming the entry. The first is the entry's unique key, while the identifier is used as common name for gestures that share the same meaning, i.e., there can be numerous gestures with the identifier "greeting".

Gesticon entries are classified according to the main modality expressed. This information is specified via the *modality* attribute. In our examples the value is either "arms" which means the entry is a representation of a gesture or "face" which indicates that the entry is a representation of a facial expression. In the context of NECA, a further modality is "body", which stands for posture such as relaxed versus upright, etc. In the long run, however, the modality "body" needs to be further subclassified, for example, into posture, movement, and spatial location.

3.2 The <verbatim>-element

In the verbatim element, a verbal description of the gesticon entry is stored. This is information for the human reader.

3.3 The <function>-element

The function element contains information about the meaning and type of an entry, where the entry is attached to, and which type of temporal alignment is to be used (before, after, parallel, etc.)

The *type* attribute is not defined for facial expressions. As regards gestures, we distinguish between the following types:

- deictic (indicative or pointing gesture)
- beat (repetitive or rhythmic movement mainly coordinated with speech prosody)
- iconic (a gesture which "bears a close formal relationship to the semantic content of speech" (McNeill, 1992) quoted after (Serenari, 2002), p. 57, e.g. the hands forming a box in order to depict a container)
- emblematic ("gestures that have a specific social code of their own" (McNeill, 1985) quoted after (Serenari, 2002), p. 57, e.g. a nod meaning 'yes')
- illustrator (e.g. a wave accompanying or substituting a greeting act; in our use illustrators are similar to emblems, but are less strict as regards their social or cultural norms than emblems)
- metaphoric ("similar to iconics in that they present imagery, but present an image of an abstract concept" (McNeill, 1992) quoted after (Serenari, 2002), p. 57)
- adaptor ("part of adaptive efforts to satisfy self or bodily needs, or to perform bodily actions, or to manage emotions, or to develop or maintain prototypic interpersonal contacts, or to learn instrumental activities" (Ekman and Friesen, 1968) quoted after (Serenari, 2002), p. 59)
- idle (we have introduced a number of idle gestures which are selected when the animated characters "do nothing", i.e., they are not engaged in a dialogue, they are waiting till data transmission is completed)

Summing up, we have drawn the values for our *type* attribute mainly from work by Ekmann/Friesen and McNeill, cf. (Ekman and Friesen, 1968), (McNeill, 1985), (McNeill, 1992). The selection was guided by practical decisions, i.e., which classification is useful in the context of the NECA demonstrators. In general the classification of gestures is somewhat controversial in the literature, see for instance (Krauss et al., 2000) or (Serenari, 2002) for an overview of gesture classifications.

At the current stage of development, the values for the *meaning* attribute are simple atomic labels. Of course this is a shortcoming and reflects a rudimentary semantic classification of gestures and facial expression. This approach, however, is sufficient for the current stage of development of the NECA system. Especially for the generation of metaphoric gestures, however, the encoding of meaning via a symbolic label is inappropriate. Instead a more complex representation structure of the meaning and the pragmatics of gestures needs to be developed. Currently this is approached from different angles such as descriptive work as represented in (Poggi, 2002a) or work on coupling gesture recognition and gesture generation such as (Kopp et al., 2004).

Meaning in gesticon entries for facial expression refers to the six basic emotions (happy, sad, anger, fear, disgust, surprise) known from (Ekman, 1993) and a few other labels which are appropriate in the context of the demonstrators such as 'neutral', 'false laugh', 'melancholy', 'reproach' etc. which are inspired by (Faigin, 1990).

The *alignto* attribute is mainly used for gestures and specifies the type of entity the particular gesture shall be aligned to. This can be a sentence, a word, an accented syllable, etc. At the current stage of development of the NECA system, facial expressions are per default aligned at sentence level.

In the *aligntype* attribute it is specified how a gesture G and an entity X from the verbal communication system are coupled together.

par	G starts exactly when X starts
par_end	G stops exactly when X stops
par_adjust_to_fit	G's duration is forced to be the same as X's, i.e., they start and stop at the same time
atstress	G is aligned to the STRESSED position of X
seq_before	G is performed before X, i.e., G preceedes X
seq_after	G is performed after X, i.e., G succeeds X

3.4 The <form>-element

In the form element, information on the basic physical properties of a gesture or facial expression is specified. The form element comprises two sub-elements: the <position>-element providing information on static (spatial) aspects of a gesture or facial expression, and the <components>-element encoding information about the dynamics (the sub-parts and temporal properties) of a gesture or facial expression.

As we treat facial entries as snapshots of facial expressions, the components element is reduced to the specification of a duration range and a default duration. The position element in facial entries specifies eyebrows (up, relaxed, center down, ...), eyes (relaxed, open wide, open narrow, ...) and mouth shapes (open smile, closed relaxed, pursed, ...). These values are inspired by (Faigin, 1990). An alternative, more fine grained representation of form information of the face are the Face Animation Parameters (FAPs) used in MPEG4, see for instance (Tekalp and Ostermann, 2000). This information can be used as extra filter for selecting appropriate facial expressions during multimodal generation. In the NECA system, however, facial expressions are currently selected according to the emotion specified for the individual dialogue acts by the affective reasoning component.

Regarding gestures, the availability of information on the basic physical properties as encoded in the position element is a prerequisite for performing basic reasoning on the well-formedness of combinations of gesture. Minimal positional information is required to decide whether two gestures can be directly concatenated or whether the combination of two gestures requires an intermediate gesture for the sequence to look natural.

Information on gesture dynamics as encoded in the components element is required for the calculation of the temporal alignment of gestures to speech as well as for modulation of the expressivity of a gesture.

In the position element spatial information of gestures is encoded very coarsely specifying the position of the left and right wrists at the very beginning and end of a gesture. This is encoded by a two-dimensional grid (top, mid, down) $\times$ (center, outwards) distinguishing 6 possible positions per wrist. This information is required for reasoning on the necessary time of moving from the end-position of one gesture to the start-position of its successor. Depending on the available time and on the interpolation capabilities of the animation technology used, the information in the position element is employed to decide on either ruling out a particular gesture, directly interpolating between two gestures or inserting movements to neutral (idle) positions in between the gestures to be concatenated.

The mechanism can also be extended in order to cope with gestures that rely on the existence of specific predecessors, e.g. return-movements from special gestures. For these an attribute *special* is added to the <start> or <end>-element, and it is enforced, that only gestures which share the same *special*-value can be combined.

As already mentioned, our proposal to positional encoding of gesture information is a minimal approach. An example for a much more detailed encoding is MURML (Kransted et al., 2002). As both our gesticon structure and MURML are XML compliant, an enhancement of the

proposed gesticon entries by MURML representations is straight forward.

For the components element of gestures the following sub-elements are defined: prepare, stroke, hold, retract (cf. (McNeill, 1992)). Each of these elements has its duration element <dur> where an appropriate range and a default for the duration of the respective phase is specified in milliseconds.

Note, that a majority of our gesticon entries are gesture fragments which only comprise stroke and hold phases, whereas the prepare and retract phases result from player-specific interpolation between adjacent gestures. In general, stroke and hold are the most important phases for aligning gesture and speech. The stroke phase for instance is employed to fine-tune the timing of gesture and speech. The stroke phase is typically aligned with a particular (accented) syllable. In cases where a gesture needs to be elongated, the hold phase is of importance, as it will be unproportionally more affected than any other phase of a gesture.

3.5 The <restrictions>-element

While in the function and form elements semantic and structural aspects of a gesture or facial expression are described, the restrictions element serves as a repository for all kinds of additional constraints that specify the applicability of a particular gesticon entry in the context of a specific system. For instance, in the NECA system for each dialogue act an emotion category is calculated by an affective reasoning component (Gebhard et al., 2003) implementing the OCC model (Ortony et al., 1988). These emotion categories need to be related to emotion expressing entries in the gesticon such as facial expressions and adaptor gestures, so that appropriate nonverbal behaviours can be selected from the gesticon. This is reflected in the constraint element <constraint name="occ_emotion" val="..."/>.[4] Another example is the activation constraint <constraint name="activation" val="..."/> by means of which we specify for which affective activation level or range a particular gesticon entry is applicable.

The structure of the restrictions element is defined as follows: It holds a set of <constraint>-elements, which can be logically combined by bracketing <and>, <or> and <not>-elements (i.e. conjunction, disjunction and negation).

In the current form each constraint element just contains an attribute *name* which holds the name of a constraint and an attribute *value* or *range* that is to be used as argument of that test.

In order to facilitate the processing of the different constraints used under <restrictions> and to ensure consistency, maintainability and readability of the gesticon, a macro-mechanism is offered in the gesticon:

For the most common type of constraints, namely the lookup of a certain value already stored in the RRL, the semantic of that constraint can be specified within the gesticon itself, using a separate <constraintCode>-section.

The example in section 3.7 shows such a <constraintCode>-entry for the constraint "occ_emotion". It defines, what a program really has to do in order to test whether <constraint name="occ_emotion" val="anger"> is fulfilled: Under the current dialogueAct (this is the scope) look for the element <emotionExpressed> and test whether the value of its *type*-attribute equals "anger".

For the constraint with the name "gender" it states, that the information on the speaker has to be dereferenced and that the gender value is to be found under the element <gender>, more precisely in the attribute *type*.

This should facilitate the authoring of individual gesticon-entries and helps to keep constraint-entries consistent. The inclusion of novel constraints or changes in the structure of the RRL thus do not necessarily require changes in the code of the interpreting programs.

3.6 The <playercode>-element

Finally, the necessary mapping to player-specific gesture-code is defined in the playercode element. For the players currently used in NECA, this element is very simple. For Charamel the playercode directly points at a animation-file, for Flash it contains the key to entries in an external gesture-repository. This playercode information is embedded in the SMIL-based timing specification and forms the output to the player-specific Animation Generator.

3.7 Example Gesticon Entries

Gesture Entry

```
<gesticonEntry key ="g001"
               identifier="Thinking"
               modality="arms">
  <verbatim>
    Thinking: adaptor:  Tina: adaptor:
    moves right hand to chin but in
    addition left hand moves to
    shoulder-hight +
    palm up
  </verbatim>
  <function type="adaptor"
            alignto="sentence"
            aligntype="par" start="-200"
            meaning="think"/>
  </function>
  <form>
    <position>
      <!-- starts with D(own) O(ut) -->
      <start left="DO" right="DO"/>
      <!-- ends with T(op) C(enter) -->
```

[4] Note, that mapping between emotion categories resulting from an OCC-based approach and the basic emotion categories for facial expressions a la Ekman is in general problematic. A principled way still needs to be developed.

```
      <end   left="TO" right="TC"/>
    </position>
    <components>
      <stroke>
        <dur min="1000" default="1300"
             max="2000"/>
      </stroke>
      <hold>
        <dur min="500" default="1000"
             max="50000"/>
    </components>
  </form>
  <restrictions>
    <and>
      <constraint name="gender"
                  val="female"/>
      <constraint name="speaker"
                  val="tina"/>
      <constraint name="occ_emotion"
                  val="anger"/>
    </and>
  </restrictions>
  <playercode type="charactor"
     id="tina/char/motions/gs_thinking"/>
 </gesticonEntry>
```

Facial Expression Entry

```
<gesticonEntry identifier="happy"
               key="18"
               modality="face"  >
 <verbatim>
    flash
    eager_smile
    applicable to John and Vanessa
 </verbatim>
 <function>
    attach_to="sentence"
    aligntype="unknown"
    meaning="happy"
 </function>
 <form>
    <position>
       <eyebrows>
       <eyes>
       <mouth type="smile_open"/>
    </position>
    <components>
       <hold>
          <dur min="50"
           default="400"
           max="5000"/>
       </hold>
    </components>
 </form>
 <restrictions>
   <and>
     <or>
       <constraint typ="occ_emotion"
                   val="joy"/>
       <constraint typ="occ_emotion"
                   val="liking"/>
     </or>
     <constraint typ="activation"
                 range="1.0:0.2"/>
   </and>
 </restrictions>
 <playercode>
    type="flash"
    length="400"
    id="f_eagersmile"/>
 </gesticonEntry>
```

Constraint Code

```
<constraintCodes mapgoal="neca_rrl.0.4">
    <constraintCode name="gender"
                    typ="attributeEquals"
                    scope="speakerInfo"
                    element="gender"
                    attribute="type">
      <verbatim>
        this specifies that the gender of
        the SPEAKER has to have a certain
        value
      </verbatim>
    </constraintCode>

    <constraintCode name="occ_emotion"
                 typ="attributeEquals"
                 scope="dialogueAct"
                 element="emotionExpressed"
                 attribute="type">
      <verbatim>
        for constraint "occ_emotion":
        look under emotionExpressed
      </verbatim>
    </constraintCode>
  ...
</constraintCodes>
```

4 Conclusion

Summing up, we have outlined an overall structure for a *gesticon*, a reusable, system independent repository of gesture snippets and facial expressions relevant for the generation of dialogue accompanying nonverbal behavior. To achieve a seamless integration of gesture and language we rely on XML-based gesture representations (the gesticon) that closely interact with the RRL, a multi-modal representation structure/language used as interface to the individual system components of a multimodal generation system for spoken dialogue. Both RRL and gesticon have been developed in the context of the NECA project, but are designed to be system independent.

As regards the representation of the physical properties of gestures, our work draws upon MURML, but, for practical reasons, does not implement a similar level of detail. The general approach taken, however, allows for an extension to MURML. In contrast to MURML which concentrates on the representation of gestures, we aim at

defining a uniform representation for gestures as well as facial expressions. Moreover, due to interlinking the gesticon and the RRL, we have defined a clearcut interface to individual processing components. The linking between gesture descriptions and an XML-compliant multimodal representation language relates our work to the work described in (Ruttkay et al., 2003). Here the scripting language STEP is used to define and process gestures for h-anim[5] agents. While our aim is to separate the representation structure of a gesture repository from the processing and animation components, STEP representations are a genuine part of the STEP animation engine. Nevertheless it would be a beneficial exercise to separate out the STEP representations for gestures and incorporate the knowledge into the gesticon. On the one hand this would enhance the gesticon entries by the joint information available in h-anim and the dynamism of gestures encoded in STEP. On the other hand it would foster the understanding of which information shall be represented in a gesticon and which information belongs to a rule system for gesture generation.

As regards language and gesture coordination, the approach presented in this paper is comparable to the one pursued in the BEAT system (Cassell et al., 2001). However, other than in BEAT where thematic structure is widely used for fine-tuning of gesture assignment, we strongly rely on the prosodic information (intonation phrases, accents) directly available from speech synthesis. Another recent system for dialogue related gesture animation utilizing an XML-based framework is presented in (Hartmann et al., 2002). This work also comes with its own gesture repository.

All in all, a number of gesture repositories exist, typically being closely tied to specific gesture animation systems. Partially these repositories encode similar information, partially the information differs regarding the dimensions and the granularity of the representations. In the current situation, it would be an advantage for the work on ECAs if the community could agree on common representation structures for gesticons to decouple the gesture repositories from the individual gesture generation systems, and thus to enable the exchange of data sets. We hope that with the presented work we have made a small contribution to a common structure for gesticons which comprise definitions of elements of nonverbal communication systems (gestures, facial expressions etc.), rather than encode concrete body-specific or animation system-specific instances of such communication elements.

Acknowledgments

The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science and Culture and the Federal Ministry for Transport, Innovation and Technology. The work reported in this paper is supported by the EC Project NECA IST-2000-28580. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

[5] See `www.hanim.org`. H-anim agents are built in VRML (`www.web3d.org/vrml/vrml.htm`).

References

Stefan Baumann, Martine Grice, and Ralf Benzmüller. GToBI - a phonological system for the transcription of German intonation. In *Proceedings of Prosody 2000. Speech Recognition and Synthesis*, pages 21–28, Poznan: Adam Mickiewicz University, Faculty of Modern Languages and Literature, 2001.

Justine Cassell, Hannes Vilhjálmsson, and Timothy Bickmore. BEAT: The Behaviour Expression Animation Toolkit. In *Proceedings of SIGGRAPH '01*, pages 477–486, 2001.

Jan-Peter deRuiter. Gesture and Speech Production. MPI Series in Psycholinguistics. Technical report, Ph.D. dissertation, University of Nijmegen, 1998.

Paul Ekman. Facial expression of emotion. *American Psychologist*, 48:384–392, 1993.

Paul Ekman and Wallace V. Friesen. Nonverbal behavior in psychotherapy research. In John M . Shlien, editor, *Research in Psychotherapy: Vol. 3*, pages 179–216. American Psychological Association, 1968.

Gary Faigin. *The artist's complete guide to facial expression*. Watson-Guptill Publications, 1990.

Patrik Gebhard, Michael Kipp, Martin Klesen, and Thomas Rist. Adding the emotional dimension to scripting character dialogues. In *Proceedings of IVA'03*, Kloster Irsee, Germany, 2003.

David B. Givens. *The Nonverbal Dictionary of Gestures, Signs & Body Language Cues*. Center for Nonverbal Studies Press, Spokane, Washington, 2002.

Björn Hartmann, Maurizio Mancini, and Catherine Pelachaud. Formational parameters and adaptive prototype instantiation for mpeg-4 compliant gesture synthesis. In *Proceedings of Computer Animation*, pages 111–119, 2002.

Stefan Kopp, Timo Sowa, and Ipke Wachsmuth. Imitation games with an artificial agents: From mimicking to understanding shape-related iconic gestures. In Antonio Camurri and Gualtiero Volpe, editors, *Gesture-Based Communication in Human-Computer Interaction, 5th International Gesture Workshop, Genova, Italy, April 15-17, 2003, Selected Revised Papers*, volume 2915 of *Lecture Notes in Computer Science*, pages 436–447. Springer, 2004.

Stefan Kopp and Ipke Wachsmuth. A knowledge-based approach for lifelike gesture animation. In Werner Horn, editor, *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 663–667, Berlin, Germany, 2000.

Alfred Kransted, Stefan Kopp, and Ipke Wachsmuth. MURML: A Multimodal Utterance Representation Markup Language for Conversational Agents. In Andrew Marriott et al., editor, *Embodied Conversational Agents: Let's Specify and Compare Them!, Workshop Notes AAMAS*, Bologna,Italy, 2002.

Robert M. Krauss, Yihsiu Chen, and Rebecca F. Gottesman. Lexical gestures and lexical access: A process model. In David McNeill, editor, *Language and gesture: Window into thought and action*, pages 261–283. Cambridge University Press, 2000.

David McNeill. So you think gestures are nonverbal? *Psychological Review*, 92:350–371, 1985.

David McNeill. *Hand and mind: What gestures reveal about thought*. University of Chicago Press, 1992.

Andrew Ortony, Gerald L. Clore, and Allan Collins. *The Structure of Emotions*. Cambridge University Press, 1988.

Hannes Pirker and Brigitte Krenn. Assessment of markup languages for avatars, multimedia and multimodal systems. Technical report, Austrian Research Institute for Artificial Intelligence, Vienna, 2002.

Paul Piwek. A flexible pragmatics-driven language generator for animated agents. In *Proceedings of ACL-2003*, pages 151–154, East Stroudsburg, PA, 2003.

Paul Piwek, Brigitte Krenn, Marc Schröder, Martine Grice, Stefan Baumann, and Hannes Pirker. RRL: A rich representation language for the description of agent behaviour in NECA. In Andrew Marriott et al., editor, *Embodied Conversational Agents: Let's Specify and Compare Them!, Workshop Notes AAMAS*, Bologna,Italy, 2002.

Isabella Poggi. Symbolic gestures: The case of the Italian gestionary. *Gesture*, 2(1):71–98, 2002a.

Isabella Poggi. Towards the alphabet and the lexicon of gestures, gaze and touch. In *Multimodality of Human Communication. Theories, problems and applications. Virtual Symposium edited by P.Bouissac (http://www.semioticon.com/virtuals/index.html)*, University of Toronto,Victoria College, 2002b.

Roland Posner, Reinhard Krüger, Thomas Noll, and Massimo Serenari. The Berlin Dictionary of Everyday Gestures. Version 9 2002. Technical report, Research Center for Semiotics, TU Berlin, 2002.

Szofia Ruttkay, Zhisheng Huang, and Anton Eliens. Reusable gestures for interactive web agents. In Thomas Rist, Ruth Aylett, Daniel Ballin, and Jeff Rickel, editors, *Intelligent Virtual Agents, Proceedings of IVA 2003. LNAI 2792*, pages 80–87, Springer, 2003.

Marc Schröder and Jürgen Trouvain. The German Text-to-Speech Synthesis System MARY: A Tool for Research, Development and Teaching. *International Journal of Speech Technology*, 6:365–377, 2003.

Massimo Serenari. Survey of existing gesture, facial expression, and cross-modality coding schemes. Technical Report of the NITE project IST-2000-26095. Technical report, TU Berlin, 2002.

Murat Tekalp and Joern Ostermann. Face and 2-D Mesh Animation in MPEG-4. *Signal Processing: Image Communication*, 15(4-5):387–421, 2000.