Affective Multimodal Control of Virtual Characters

Martin Klesen and Patrick Gebhard

***Abstract*—In this paper we report about the use of computer generated affect to control body and mind of cognitively modeled virtual characters. We use the computational model of affect ALMA that is able to simulate three different affect types in real-time. The computation of affect is based on a novel approach of an appraisal language. Both the use of elements of the appraisal language and the simulation of different affect types has been evaluated. Affect is used to control facial expressions, facial complexions, affective animations, posture, and idle behavior on the body layer and the selection of dialogue strategies on the mind layer. To enable a fine-grained control of these aspects a Player Markup Language (PML) has been developed. The PML is player-independent and allows a sophisticated control of character actions coordinated by high-level temporal constraints. An Action Encoder module maps the output of ALMA to PML actions using affect display rules. These actions drive the real-time rendering of affect, gesture and speech parameters of virtual characters, which we call Virtual Humans.**

***Index Terms*—Virtual characters, affect computation, multi-modal behavior markup language.**

I. INTRODUCTION

The VirtualHuman project aims at developing interactive virtual human-like characters that can be controlled and animated in real-time. One of the main goals was therefore the specification and implementation of a computational model that takes all relevant aspects of the human behavior into account. This includes speech, gestures, postures, lip-synchronous mouth movements, eye and head movements, and the display of emotions, e.g. through tears or through a change in complexion. All these different modalities have to be combined and properly synchronized to generate a consistent and life-like behavior.

In the VirtualHuman system the behavior control mechanism can be characterized by two major characteristics: (1) Using different modules to control different behavior aspects and (2) Using a stepwise refinement of action specifications to minimize dependencies between components. The communicative behavior of all characters in a scenario is controlled by the *Conversational Dialog Engines* (*CDEs*); see chapter “Multiparty Conversation for Mixed Reality” in this volume. Each CDE uses “A Layered Model of Affect” (ALMA) to compute the affective state of a virtual character based on the dialog contributions of the participants (see Fig. 1). Based on the specified personality profile and a set of appraisal rules, ALMA computes emotions and their intensity as well as a character’s current mood. This information is then used by a virtual character’s CDE to change the course and style of the conversation. In addition and parallel to the CDEs the affect output produced by ALMA is processed by the Action Encoder module to modify the nonverbal behavior associated with a character’s affective state. The Action Encoder serves as a post processing component for both the CDE and ALMA. It is responsible for the action encoding, i.e. the refinement of character and object actions and for the timing and synchronization of these actions. The CDEs specify the behavior of characters and objects in the scenario using the XML-based *Player Markup Language* (*PML*). The high-level behavior specifications in PML documents are processed by the Action Encoder. It selects an appropriate animation based on a gesture lexicon (gesticon) and uses a text-to-speech (TTS) system to generate the audio file and to obtain information about the phoneme types and their duration. The result is an enriched PML document in which all actions are properly synchronized. The document is send to the 3D player for execution. This processing path is depicted in the lower half of Fig. 1.

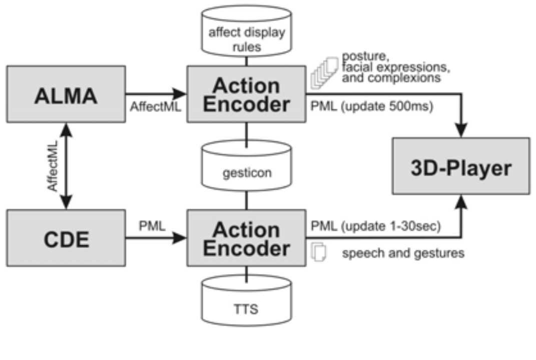


Fig. 1. Interplay between the VirtualHuman modules ALMA, CDE, and Action Encoder.

The PML actions generated by the CDEs typically comprise a sequence of verbal utterances, accompanying gestures, but might also contain action specifications for multimedia objects. The CDE waits until all actions have been performed before sending the next PML document. The update frequency lies therefore somewhere between 1 and 30 seconds and depends mainly on the length of the utterances. The affective state of characters however is periodically updated by ALMA every

Manuscript Received on March 16, 2007.

Martin Klesen works as a research scientist in the Intelligent User Interfaces department at the German Research Center for Artificial Intelligence (DFKI) in Saarbrücken. E-Mail: rumpler@dfki.de

Patrick Gebhard studied computer science and physics at the University of Saarbrücken and received his Diploma degree in 1999. His research interests are affective intelligent virtual agents and computational models of affect based on cognitive theories. E-Mail: Patrick.gebhard@dfki.de.

500 milliseconds to simulate emotion decay and mood changes over time. The Action Encoder generates with the same frequency PML actions that constantly change the facial expression, complexion, and posture (idle behavior) of the virtual characters. This ensures smooth transitions between facial animations and complexions. This processing path is depicted in the upper half of Fig. 1.

It's important to note that the PML actions produced by the two instances of the Action Encoder control different aspects of a character's behavior. The combination of the real-time computation of affect with the autonomous and plan-based generation of the communicative behavior enables the affective multimodal control of virtual characters in the VirtualHuman system. In the following sections the Affect module, the Player Markup Language, and the Action Encoder are described in detail.

II. AN AFFECT MODULE FOR VIRTUAL HUMANS

The employment of virtual humans as an interface in human-computer interaction scenarios makes high demands on their believability. This is mainly founded in the sophisticated embodiment of Virtual Humans that implies human-like conversational abilities and behavior aspects. As known from other projects employing embodied virtual characters, like MRE [1, 2], COSMO [3], Émile [4], Peedy [5], Greta agent [6] and in the SCREAM framework [7], affect successfully helps controlling behavior aspects. When analyzing them according to their temporal characteristics, there are short-term behavior aspects, like facial expressions, gestures, or the wording of verbal expressions. Also, there are medium-term and long-term aspects, like the process of making decisions, or the motivation of characters. The latter are traditionally implemented by dialog systems like the *Conversational Dialog Engine* (*CDE*) [8]. And, there are behavior aspects that consist of mixed-term aspects, like a character's idle behavior that includes for example eye blink (short-term) and medium term posture changes. Our approach to control such behavior aspects relies on a computational model of affect [9] that provides different affect types, which are described in the next section.

2.1. Affect Taxonomy

The affect module is designed to simulate affect types as they occur in human beings. As suggested by Krause [10] affect can be distinguished by the eliciting cause, the influence on behavior, and its temporal characteristics. Based on the temporal feature, we use the following taxonomy of affect:

- Emotions reflect short-term affect that decays after a short period of time. Emotions influence facial expressions, facial complexions (e.g. blush), and conversational gestures.
- *Moods* reflect medium-term affect, which is generally not related with a concrete event, action or object. Moods are longer lasting affective states, which have a great influence on humans' cognitive functions [11, 12].
- *Personality* reflects long-term affect and individual differences in mental characteristics [13].

As known by the research of psychologist those different types of affect naturally interact with each other. Personality usually has a strong impact on the emotions intensities [14, 15]. The same applies to moods [12]. With our computational model we want to simulate the interaction of the different affect types in order to achieve a more consistent simulation of affect.

2.2. ALMA

ALMA is a computational model for the real-time simulation of affect types that human beings can experience. Based on a real-time cognitive appraisal of the virtual environment different affect types are simulated in a hierarchical generation process. This inspired us to name the model *ALMA* (see Fig. 2), which stands for *A Layered Model of Affect*.

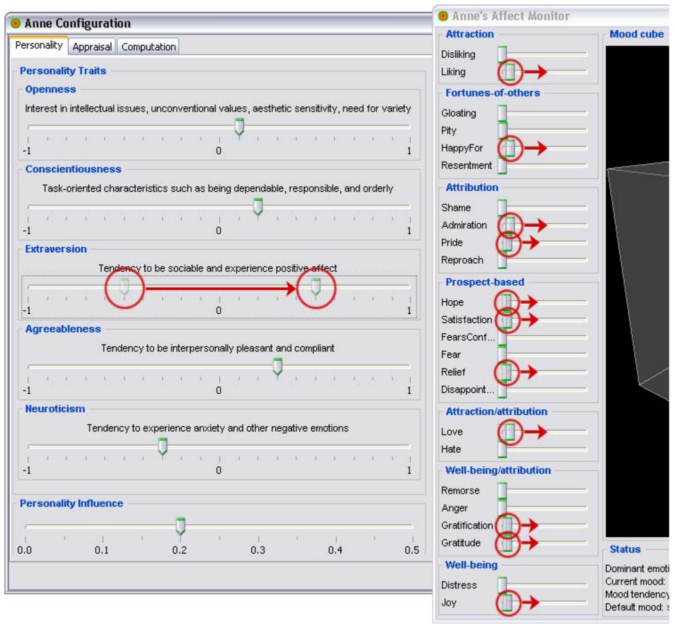


Fig. 2. Virtual Human ALMA personality configuration dialog and impact on emotion intensities.

2.2.1 Emotions

Our work is based on the computational model of emotions (*EmotionEngine*) described in [16, 17]. It implements the model of emotions developed by the psychologists Ortony, Clores, and Collins (*OCC* model of emotions) [18] combined with the five factor model of personality [12] to bias the emotions' intensities. All five personality traits (openness, conscientiousness, extraversion, agreeableness, and neuroticism) influence the intensities of the different emotion types. We therefore adopted essential psychology research results on how personality influences emotions to achieve a more human-equivalent emotion simulation. Watson and Clark [15] and Becker [14] have empirically shown that personality, described through the big-five traits, impacts the intensity of emotions. They discovered, e.g. that extravert people experience positive emotions more intensely than negative emotions. In our computational model this is realized by the change of an emotion's basic intensity, the so-called *emotion intensity bias*. Note that, the intensity of elicited emotions cannot be lower than the emotion intensity bias. When the personality is defined by a graphical user interface one can directly observe the impact on the emotions intensity bias, see Fig. 2.

Fig. 2 consists of two screen shots showing the direct impact of the change of the extravert personality trait on emotions' intensity bias. In the example the extravert trait value is increased by moving the slider to the right side. As a consequence the basic emotion intensities of positive emotions increase. Note that not all emotions are biased in the same way. This depends on the fact that personality traits potentially bias emotion intensities at different strengths. Also the intensity biases are influenced by a Virtual Human's current mood, see next section.

The OCC cognitive model of emotions is based on the concepts of appraisal and intensity. The individual is said to make a cognitive appraisal of the current state of the world. Emotions are defined as valenced reactions to events of concern to an individual, action of those s/he considers responsible for such actions, and objects/persons. The EmotionEngine is able to compute all 24 emotions that are defined by the OCC theory.

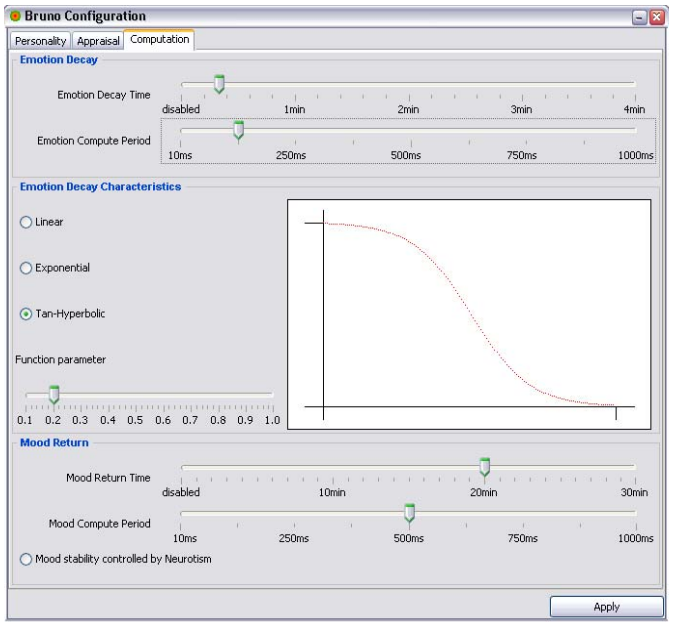


Fig. 3. Virtual Human ALMA configuration dialog showing emotion and mood simulation parameters.

The intensity of emotions underlies a natural decay, which can be configured by several decay functions (linear, hyperbolic, and exponential). This and other individual and static character information, like personality, can be defined for each Virtual Human by a graphical user interface that is shown in Fig. 3.

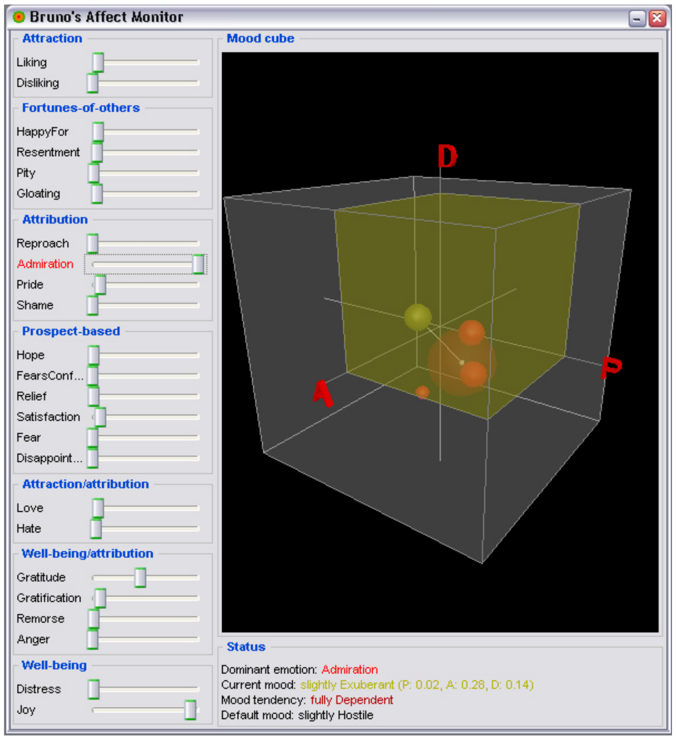
2.2.2 Moods

The employed computational model of moods is based on the psychological model of mood (or temperament) proposed by Mehrabian [19]. Mehrabian describes mood with the three traits *pleasure* (P), *arousal* (A), and *dominance* (D). Each trait represents a specific mood component. Pleasure describes how much an individual enjoys the actual situation. Arousal stands for the excitement of an individual in the actual situation. Dominance describes up to what extend an individual controls the actual situation. The three traits are nearly independent, and form a three dimensional mood space. A PAD mood can be located in one of eight mood octants. A mood octant stands for a discrete description for a mood: +P+A+D is exuberant, –P–A–D is bored, +P+A–D is dependent, –P–A+D is disdainful, +P–A+D is relaxed, –P+A–D is anxious, +P–A–D is docile, and –P+A+D is hostile. Generally, a mood is represented by a point in the PDA space.

For mood computation, it is essential to define a Virtual Human's *default mood*. A mapping, empirically derived by Mehrabian [20], defines a relationship between the big five personality traits and the PAD space. Using this mapping and Mehrabian's weighted coefficients, the computational model of affect, is thereby able to compute a default mood:

$$\begin{aligned}
\textit{Pleasure} &:= 0.21\bullet\text{Extraversion} + 0.59\bullet\text{Agreeableness} + 0.19\bullet\text{Neuroticism}\\
\textit{Arousal} &:= 0.15\bullet\text{Openness} + 0.30\bullet\text{Agreeableness} - 0.57\bullet\text{Neuroticism}\\
\textit{Dominance} &:= 0.25\bullet\text{Openness} + 0.17\bullet\text{Conscientiousness} + 0.60\bullet\text{Extraversion} - 0.32\bullet\text{Agreeableness}
\end{aligned}$$

We define the *mood strength* by its distance to the PAD zero point. The maximum distance is $\sqrt{3}$. This is divided into 3 equidistant sections that describe three discrete mood intensities: *slightly*, *moderate*, and *fully*.



Dominant emotion: Admiration
Current mood: slightly Exuberant (P: 0.02, A: 0.28, D: 0.14)
Mood tendency: fully Dependent
Default mood: slightly Hostile

Fig. 4. ALMA AffectMonitor visualizes ongoing mood changes and elicited active emotions.

Using the above mentioned mapping and the mood strength definition a person, whose personality is defined by the following big five personality traits: openness=0.4, conscientiousness=0.8, extraversion=0.6, agreeableness=0.3, and neuroticism=0.4 has the default mood *slightly relaxed* (pleasure=0.38, arousal=-0.08, dominance=0.50).

An AffectMonitor, shown in Fig. 4, is used to visualize a Virtual Human's current mood and mood changing emotions.

The left side of the AffectMonitor shows a Virtual Human's emotions and their intensities. Newly elicited emotions are marked dark gray (red). The right side shows a 3 dimensional PDA mood cube displaying the current mood (the highlighted octant stands for the discrete mood description, whereas the light gray (yellow) ball reflects the actual mood) and all active emotions (dark gray (red) balls). Below, the affective state, including the current dominant emotion, and the default as well as the current mood, is displayed.

A novelty of the actual version of ALMA is that the current mood influences the intensity of active emotions. The theory is that the current mood is related to personality values that interfere with a Virtual Human's actual personality values. Technically, this is realized by the reverse use of the (above shown) mapping of big-five personality values on PAD values. Based on the current mood, the most intense related personality trait is identified. The actual value of this trait blends over the Virtual Human's original personality trait value and is used to regulate the intensity of emotions. This increases, for example, the intensity bias of joy and decreases the intensity bias of distress, when a Virtual Human is in an exuberant mood.

2.3. Mood Changes

According to Morris [11] conditions for mood changes can be divided into (a) the onset of a mildly positive or negative event, (b) the offset of an emotion-inducing event, (c) the recollection or imagining of an emotional experience, and (d) the inhibition of an emotional responding in the presence of an emotion-inducing event. To keep the modeling of mood changes as lean as possible, we take elicited emotions as the mood changing factor. In order to realize this, emotions must be somehow related to a Virtual Human's mood. While using the PAD space for modeling mood, it is obvious to put emotions in relation to the PDA space too.

We rely on Mehrabian's mapping of emotions into the PAD space [19]. However, not all 24 emotion types provided by the EmotionEngine are covered by this mapping. For those that lack a mapping, we provide the missing pleasure, arousal, and dominance values by exploiting similarities to comparable emotion types [9].

Our approach to the human-like simulation of mood changes relies on a functional approach. We concentrate on how the intensity of emotions influences the change of the current mood and we consider the aspect that a person's mood gets the more intense the more experiences the person makes that support this mood. For example, if a person's mood can be described as slightly anxious and several events let the person experience the emotion fear, the person's mood might change to moderate or fully anxious.

The computation of mood changes is based on *active emotions* generated by the computational model of emotions. Each appraisal of an action, event or object elicits an active emotion that once generated, decays over a short amount of time (i.e. one minute). All active emotions are input to the mood change function. The function has two scopes. Based on all currently active emotions the function defines whether the current mood is intensified or changed. It will be intensified if all active emotions are mapped into the mood octant of the current mood. This is called the *mood push phase*. In the *mood pull phase*, a mood will be changed progressively if all active emotions are mapped into a different mood octant than the current mood. Based on all active emotions a *mood transition vector* (*MTV*) is computed. The MTV will in subject to the location of the actual mood be applied in different ways to change the mood position in the PAD space. Fig. 5 gives an overview about the four different scenarios. The light gray (yellow) ball represents a Virtual Humans current mood. The dark gray (red) ball stands for an active emotion or the center of all active emotions. Note that the mood change computation is a dynamic process. Ever since an active emotion's intensity decays, the MTV is recalculated.

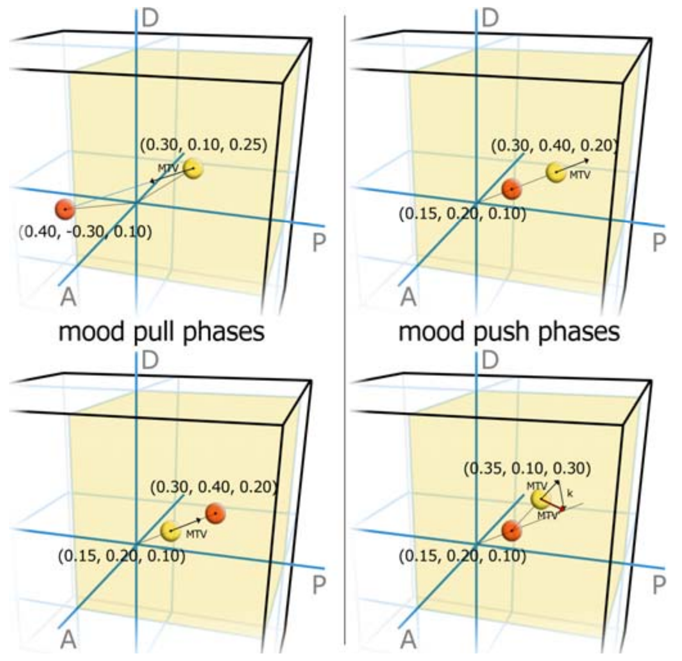


Fig. 5. Mood change scenarios. See Color Plate 9.

Another aspect of our mood simulation is that the current mood has a tendency to slowly move back to the default mood. Generally, the return time depends on how far the current mood is away from the default mood. We take the longest distance of a mood octant ($\sqrt{3}$) for defining the mood return time. Currently this is 20 minutes.

2.4. Appraisal based Affect Computation

In our cognitive inspired affect computation, the first step is to appraise relevant input by using a Virtual Human's own subjective appraisal rules, introduced in [17]. Input for affect processing has to be represented as structures of our XML based affect modeling language AffectML; introduced in [9]. Three types of affect input are distinguished: 1) basic appraisal tags, 2) act appraisal tags, and 3) affect display appraisal tags.

All appraisal tags are defined in a Virtual Human character's ontology. A Virtual Human's Conversational Dialog Engine (CDE) uses appraisal tags to appraise the current situation including its own actions and those of other Virtual Humans and users.

Basic appraisal tags express how a speaking character appraises the event, action or object about which it talks. There are 12 basic tags for appraising events, e.g. the tag GoodEvent, which marks an event to be good according to the subjective view of the one which does the appraisal. The other event tags are: BadEvent, GoodEventForBadOther, GoodEventForGood-Other, BadEventForGoodOther, BadEventForBadOther, GoodLikelyFutureEvent, GoodUnlikelyFutureEvent, Bad-LikelyFutureEvent, BadUnlikelyFutureEvent, Event-Confirmed, and EventDisconfirmed. For appraising actions, there are 4 basic appraisal tags: GoodActSelf, BadActSelf, GoodActOther, and BadActOther. And finally there are 2 basic tags for appraising objects: NiceThing, and NastyThing. All basic appraisal tags together are the basic set of a high-level appraisal language which can be used for a subjective appraisal of situations. These tags can be used to appraise dialog acts and other affective signals. For each of these types the appraisal language provides specific tags: act appraisal tags and affect display appraisal tags. Act appraisal tags represent the underlying communicative intent of an utterance, e.g. tease, or congratulate. Affect display appraisal tags stand for visual cues of an experienced emotion or mood, e.g. a blush of shame or a character that looks nervous. By rules, which are defined individually for each Virtual Human those tags will be mapped on basic appraisal tags that will be further processed to emotion eliciting conditions.

Generally, the output of the appraisal process is a set of emotion eliciting conditions. Based on them active emotions are generated that in turn influence a Virtual Human's mood. On the technical side, each Virtual Human's CDE has its own ALMA process, which processes affect input. The input consists of appraisal tags, dialog act input, emotion and mood input, information about who is speaker, addressee and listener. The computed affect (emotions and mood) is passed back to the working memory of the CDE and influences its cognitive dialog simulation. Also, the affect output is passed through the Action Encoder module to the player component which is responsible for rendering the Virtual Human's visual body appearance and its speech output (see Fig. 1). All affect output is represented in structures of AffectML

The evaluation of this computational model of affect shows that nearly all generated affect types are plausible to humans (see next section). Based on these results we are confident that the affect visualization through facial expressions and complexions, gestures, posture changes, and through different dialog behavior realized by each Virtual Human's CDE is plausible too.

III. AFFECT EVALUATION

We ask people how plausible they perceive the generated emotions and moods in order to prove that ALMA's computational model of affect is able to produce coherent affect that is comparable to human affect. To eliminate most of the side-effects that might blur the results, we decided to evaluate the plausibility through textual dialog descriptions. If we could show at this level that the generated affect is plausible, the visualization of them – if done correctly – will be plausible as well.

We check the plausibility of affect with an offline textual questionnaire by 33 participants. They judge the plausibility of 24 emotion types and 8 different moods that can be generated by our computational model of affect. The materials we use for the evaluation consist of single *dialog contributions*, and *dialog scenes* that can be defined as a set of dialog contributions. An example of those is given by Fig. 6 and Fig. 7.
The basic assumption we made is that emotions will be elicited by dialog contributions. For example, the dialog contribution of Bob "Anne, it's cool that you're helping grand-mother in cleaning up the garden!" elicits the emotion pride on the side of Anne, the addressee. On the side of the speaker (Bob) the emotion pride is elicited. Therefore, they have to be enriched by appraisal tags, which stand for the intentional content (see section Appraisal based Affect Computation). These appraisal tags are used by ALMA for generating emotions. They are not shown in the evaluation questionnaire. Taking the example above, in which Bob encourages Anne for her exam, the enriched version of the dialog contribution looks like:

Bob: "Anne, it's cool that you're helping grand-mother in cleaning up the garden!" [PraiseAction Anne].

Bob: Anne, it's cool that you're helping grand-mother in cleaning up the garden!

Anne's emotion: *pride* **Bob's emotion**: *admiration*

Fig. 6. Dialog contributions for emotions.

Situation: Mark is reorganizing his computer hard drive by letting Microsoft Windows removing unneeded files. Tanja just shows up.

Mark: Crap, Windows has killed all pictures of our last summer holiday at Mallorca.
Tanja: Don't panic, you'll find them surely in the waste bin.
Mark: Are you sure? But what if not, what I'm doing then – they will be lost forever!
Tanja: Well, I've no clue, I'm not the computer expert.
(Mark tries to recover the files by restoring them of the waste bin)
Mark: No, damn it! All the pictures gone – and there's no way to get them back!
Tanja: Oh no, All our pictures are lost! You are a clean up maniac. I always told you that this will led some days to something bad. Well, and that's just happened. Wonderful!
Mark: Get of my back!

Marks mood **after**: hostile

Fig. 7. Dialog scenes for moods.

Appraisal tags, like this act tag are used as input for ALMA. As described above, each character has a set of appraisal rules (about 30-50), which appraise the act tag by taking into account the role of the individual. The act tag [PraiseAction Anne] is appraised by Bob as GoodActOther, a praiseworthy action of one other, whereas Anne appraises the act tag as GoodActSelf, a praiseworthy action of herself that Bob has put into her mind by saying the above line. Following the OCC emotion theory a praiseworthy action of one other will elicit the emotion admiration (Bob) and a praiseworthy action of oneself will elicit the emotion pride (Anne).

According to Morris' theory (see section Mood Change), which is implemented by ALMA, emotions influence the current mood. The emotions that are elicited by a set of dialog contributions in a specific time interval can change the current mood of an individual to another mood. For the questionnaire, we use short (mostly singular) dialog contributions for the elicitation of emotions and dialog scenes for the change of moods. For the plausibility check of the 24 emotion types, we rely on 24 short dialog contributions that influence, the speaker's and the addressee's emotions, see Fig. 6 and Fig. 7.

Therefore, on average, each emotion is rated 2 times. For the plausibility check of the 8 mood types, we rely on 24 dialog scenes. Thus, every mood type is rated 3 times. In a pre evaluation, experts (a computer linguist, a dialog expert, and a psychologist) have reviewed the dialog contributions and the dialog situations for being realistic. All problematic formulations, unrealistic contributions, and unclear situations have been rewritten and modified. In a next step, the annotated appraisal tags that represent the intentional content are reviewed for being appropriate. All inappropriate tags have been identified and changed.

Participants are asked to evaluate in about half an hour how plausible emotions and moods are through a discrete 5 point ranking scale. 1 denotes the "lowest plausibility", 5 stands for the "highest plausibility", and 3 marks the "neutral plausibility".

Since rating scales can be treated as interval scales [21], we used parametrical tests for the statistical analysis. The t-test for one sample is a statistical significance test that proves whether a measured mean value of an observed group differs from an expected value. In our study, ratings were proven to be "positive" if the mean score significantly exceeded the moderate plausible value of 3.

To test the effect of a factor with multiple values (e.g. emotion type) or interactive effects of several factors (e.g. affect type and gender) we calculated an analysis of variance (ANOVA).

The overall result of the evaluation is that emotions and moods generated by ALMA are plausible. 22 out of 24 emotions and of 7 out of 8 moods are rated positive plausible. Considering all participants, the results are independent from age or gender. The full information about the evaluation can be found in [22].

Based on these results, we were more confident about that the embodiment of ALMA generated emotions and moods through Virtual Humans will be plausible as well. This, however, has to be evaluated separately.

The next section explains by example how emotions and mood are computed and displayed in behavior during a dialog between a user and three Virtual Humans.

IV. DISPLAY OF AFFECT

A dialog transcript between a user and three Virtual Humans is used to illustrate the generation of emotions and their surface rendering in behavior. The dialog is about that a user should anticipate if a soccer player scores a goal or not while viewing a soccer video. The video stops at a dramatic scene, see Fig. 9. A user can ask virtual experts what they think about the situation. Basically it contains of the same dialog lines which can be found in chapter "Multiparty Conversation for Mixed Reality" on pp. 3 but features relevant input for the affect generation and shows affective behavior examples.

A major goal of the VirtualHuman project is to simulate realistic affective behavior of Virtual Human's. At first, affect influence the mind of the characters, which is realized by CDE dialog strategies. There emotions and moods bias the selection of dialog strategies, turn taking behavior and wording of utterances. See chapter "Multiparty Conversation for Mixed Reality" for more details. For the body layer emotions are used to control a) *facial expressions*, b) *facial complexions*, like red cheeks, c) affective animations, like weeping (see fig. 8). Moods are used to control a) *posture* and b) *idle behavior* of Virtual Human's, such like the eye blink rate, the simulation of breath, and specific idle gestures (e.g. look at watch). A character's mood in VirtualHuman is mainly reflected by postures. According to the 8 different moods which can be simulated by ALMA, each Virtual Human has 8 different mood-related posture animations. For example, exuberant Virtual Humans show more body and head movements than bored ones.

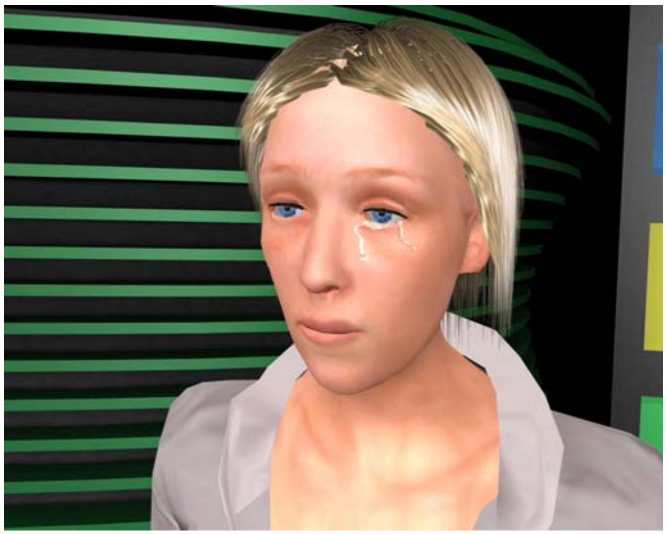


Fig. 8. Example of affective body behavior by e.g. real-time affective weeping animation (caused by emotion pity). See Color Plate 10.

The following example dialog highlights the synchronisation between ALMA and CDE processes of each Virtual Human. During the interaction between users and Virtual Humans, emotions are elicited by dialog moves (of one of the users or other Virtual Humans). The correlated appraisal process is described verbally and annotated in a bracketed section (*[…]*) below a dialog turn. Emotions and mood display in behaviour are shown in various screenshots.

(1) MODERATOR: ... Now look closely [shows video on screen]. What will happen next? The alternatives are One -- Ballack scores the goal, Two -- the keeper does a parade, Three -- Ballack kicks the ball into the sky.
(2) MODERATOR: What do you think, Mister Kaiser?

(3) Mr. KAISER: I think Ballack scores the goal.
[His CDE appraises the moderator's choice to consult him and his competent answer as a good action of himself (appraisal tag GoodActSelf), because he's pretty sure what is going to happen in this situation. This elicits the emotion pride, which is not visualized on the body layer]
(4) MODERATOR: Spoken like a real football trainer.
(5) MODERATOR: Now, player one, what is your guess?
(6) USER: Mrs. Herzog, what do you think?
(7) Mrs. HERZOG: I think the keeper does a parade.
[She is doubtful about her estimation. Her CDE appraises this as a bad action of herself (appraisal tag BadActSelf). This elicits the emotion shame, which lets her blush, see Fig. 9]

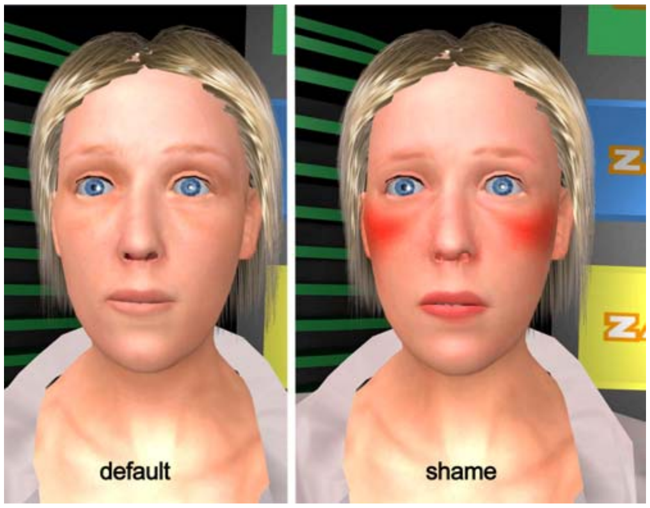


Fig. 9. Example of an affective facial complexions: red cheeks as a result of the emotion shame. See Color Plate 11.

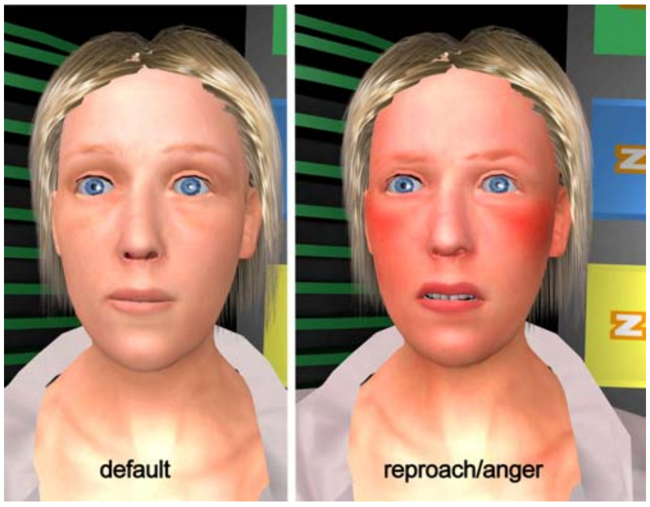


Fig. 10. Example of an affective facial expression combined with complexion: distorted lips and red head triggered by the emotion reproach/anger. See Color Plate 12.

(8) MODERATOR: An interesting opinion.
(9) MODERATOR: Now it's your decision, player one.
(10) USER: I think Mr. Kaiser is right.

[The CDE of Mr. Kaiser appraises this dialog move as a GoodActSelf, because the user trusts him in his opinion. This elicits again the emotion pride, which is not visualized on the body layer]
(11) Mrs. HERZOG: How can you believe this amateur!
[Her CDE appraises this as a BadActOther and a few seconds later as a BadEvent. The latter, because she realizes that her opinion about the outcome has been mistrusted in front of all! This elicits the emotion reproach and later the emotion distress, which results in the complex emotion anger, see Fig. 10]
(12) Mr. KAISER: [smiles]
[He appraises Mrs. Herzog's display of reproach/anger emotion as a GoodEvent (emotion joy), which lets him smile. Due to the fact that Mr. Kaiser has experienced many positive emotions his mood has changed from relaxed to exuberant. When being exuberant, he shows more active idle behavior, compared to, when he's being in a relaxed, see Fig. 11]

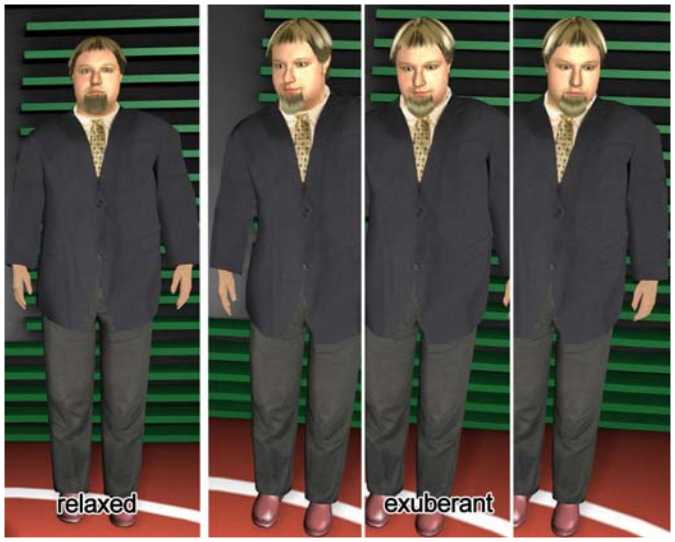


Fig. 11. Example of mood display in posture

(13) MODERATOR: Alright, answer one.

V. PLAYER MARKUP LANGUAGE

The Player Markup Language (PML) serves as an interface language between the CDEs, ALMA, the Action Encoder and the 3D player. It was designed to meet the following requirements:

- Support combination of both high-level abstract concepts and detailed, application-specific information.
- Stepwise refinement of character and object actions to minimize dependencies between components.
- Timing and synchronization support for multimedia content.

As a representation and interface language PML must be able to specify the properties and the behaviors of characters and objects in a 3D virtual environment independently from their realization in a concrete setting. Gestures, for example, are selected and parameterized by the CDEs based on their communicative function discounting at first their realization by the 3D player (e.g. using either key-frame animations, inverse kinematics or some other animation technique). At a later stage however this detailed player- and character-specific information has to be provided, e.g. by specifying the animation type along with the required animation parameters and exact timing information. The stepwise refinement of character and object actions is therefore another important requirement. PML supports the incremental specification of synchronized multimodal output (e.g. postures, gestures, facial animations, speech) using both qualitative and quantitative temporal constraints. In a first step actions are synchronized by specifying temporal relations (e.g. before, overlaps, during). In a second step these qualitative constraints are resolved by the Action Encoder which computes the start time and active duration for each action (see section Ⅵ).

PML focuses on the specification of verbal and non-verbal behaviors of virtual characters in multi-party dialogs but it also contains elements to specify and coordinate the presentation of other *scene elements* over time. In VirtualHuman the term scene element covers a broad range, including discrete media types such as still images, graphical user interface elements (e.g. menus and sliders), as well as continuous media types that are intrinsically time-based, such as video, audio and object animations.

```
<definitions id="cde::000">
 <repository id="rep1">
   <path src="file:///local/vh/characters/Herzog"/>
 </repository>
 <character id="Herzog" src="rep://rep1/main.wrl">
   <voice id="voiceHerzog" refId="femaleVoice1" pitch="high"
          range="default" rate="x-slow" volume="loud"/>
 <viseme>
   <phoneme id="aa" refId="viseme:A" intensity="0.8"/>
   <phoneme id="ao" refId="viseme:O" intensity="1.0"/>
   ...
 </viseme>
 <complexion id="redCheeks" refId="face"
                     src="rep://rep1/textures/redCheeks.jpg"/>
 <complexion id="tears" refId="face"
                     src="rep://rep1/textures/tearsmap.mtd"/>
 <multiPoses id="fold" src="rep://rep1/fold.wrl" dur="6166"/>
 <multiPoses id="nod" src="rep://rep1/nod.wrl" dur="933"/>
 ...
 <singlePose id="joy" src="rep://rep1/joy.wrl" dur="2000"/>
 <singlePose id=" viseme:A" src="rep://rep1/visA.wrl" dur="2000"/>
 ...
 <idlePoses id="idle1" random="true">
   <multiPoses refId="idle_var1" dur="7333"/>
   <multiPoses refId="idle_var2" dur="13100"/>
 </idlePoses>
</character>
```

Fig. 12. PML character definition.

To specify the properties and the behavior of scene elements, PML distinguishes between three types of documents: PML definitions, PML actions, and PML messages. The focus in the following subsections is on the language specification. The way how these elements are processed by the Action Encoder is discussed in section 6. PML is an XML-based language. The XML schema can be downloaded from the VirtualHuman website[2].

[2] http://www.virtual-human.org/xsd/PML.xsd

5.1 PML Definitions

PML definitions are used to specify the properties of objects and characters in a 3D virtual environment. We will use the term scene element introduced in the previous paragraph to refer to any element that can be defined in such a document. There are three types of definitions: repository definitions, character definitions, and object definitions. Repository definitions are comparable to classpaths definitions in a programming language like Java. They tell the 3D player where the resources for the various scene elements (images, audio and video files, etc.) are to be found on the local platform. Character definitions specify the acoustic parameters of the synthetic voice (pitch baseline, pitch range, speech rate, and volume), the available animations, their default durations, and the phoneme-viseme mapping to be used. They also specify the set of available complexions (skin textures) and targets in the 3D environment for the procedural animations (e.g. deictic gestures, eye and head movements).

Fig. 12 shows an example of a character definition. Each scene element has a unique identifier (‘id’) by which it can be referenced (via the ‘refId’ attribute) in other elements. The voice definition is followed by the phoneme-viseme mapping and a list of complexions. What follows is the set of available animations. PML distinguishes between animations that specify a movement through a sequence of multiple poses (e.g. using key frames) and animations that define a single pose (e.g. using morph targets). This distinction is reflected in the <multiPoses> and <singlePose> element definitions. A third type (<implicitPose>) is used to specify procedural animations. These are the basic building blocks for a character’s behavior. New behaviors (e.g. facial expressions and idle behaviors) can be defined as a combination of these action primitives. In the PML definitions example the idle behavior ‘idle1’ is defined using two animations. If this idle behavior is started the player will repeatedly and randomly choose one of the animations. It is also possible to define new facial expressions as a combination of single poses as shown in Fig. 13.

```
<createSinglePose id="angry">
  <singlePose refId="L_Eyebrow_Down" intensity="0.7"/>
  <singlePose refId="R_Eyebrow_Down" intensity="0.7"/>
  <singlePose refId="L_Mouthcorner_Down" intensity="0.4"/>
  <singlePose refId="R_Mouthcorner_Down" intensity="0.4"/>
  <singlePose refId="Neutral" intensity="-1.2"/>
</createSinglePose>
```

Fig. 13. PML facial expression definition.

Object definitions are used to specify graphical user interface elements (e.g. on-screen menus), virtual cameras, and to define the various media types that will be used in the scenario. For images and videos it must be specified where in the 3D scene they should be displayed. This is done via a reference to an object that plays the role of the canvas. Similarly for audio files an object must be specified that acts as the sound source.

All scene elements used in PML actions must first be defined in a PML definitions document. Scene elements that are no longer required (e.g. videos and audio files that have been played) can be deleted from the list of definitions using the <undefine> element that releases the system resources that have been allocated by the 3D player.

5.2 PML Actions

PML actions are used to specify the appearance and behavior of all characters and objects in a 3D environment. Some actions can be applied to both characters and objects while others are only available for specific scene elements. 'Show' and 'hide' are universal actions as well as 'transform' which is used to change the location and orientation of an object or character. Idle lists for both element types (for an object this might be some background animations) can be started and stopped or replaced by other idle lists, e.g. to display a character's mood change as described in section "Affect Rendering". Actions that are only applicable to virtual characters are 'speak' for verbal output, facial animations, gestures, postures, and complexions. Examples for PML character actions are given the following sections. Object actions comprise the starting and stopping of audio and video, the parameterization and manipulation of graphical user interface elements, and the control of the virtual cameras. Fig. 14 gives an example of a PML object action that starts a video and an audio comment.

```
<actions id="cde::122" start="true">
  <object refId="Studio">
    <startVideo id="a1" refId="soccer-em04" alignTo="null"
                alignType="null"/>
    <pause  id="a2" dur="3000" alignTo="a1"
            alignType="finishes"/>
   <startAudio id="a3" refId="comment-goal"
                alignTo="a2" alignType="meets"/>
  </object>
</actions>
```

Fig. 14. PML object action.

PML uses an object-oriented approach, i.e. each action is associated with a single character or object. Actions are synchronized by defining a temporal alignment with another action using qualitative temporal constrains as described in section "Timing and Synchronization".

5.3 PML Messages

PML messages are used to control the execution of actions, and to exchange information between the 3D player and other system modules. There are three different types of messages: commands, states, and facts. Commands can be used to start and stop the execution of the set of actions in a PML actions document. States are used by the 3D player to inform other modules about the execution state (e.g. started, failed, finished) of PML actions. This information is crucial to synchronize the behavior of characters and objects *across* different sets of actions. The exact timing and synchronization of actions is defined within a PML actions document that contains, for example, a character's verbal and nonverbal behavior for a single utterance. When all actions have been performed by the character, the player sends a 'finished' message to signal that it is ready to execute the next set of actions. Facts are used to inform the CDEs about user actions (e.g. the user has selected a menu entry) as shown in the following example (see Fig. 15).

Facts are represented by attribute-value pairs. In this example 'playerList' refers to a multiple-choice menu that has been previously defined and displayed in the 3D virtual environment using PML definitions and actions. The item selected by the user has the value 'Klose' associated with it. This information is used by the CDEs in their dialog planning process.

```
<message id="player::321">
  <fact refId="playerList" value="Klose"/>
</message>
```

Fig. 15. PML message document.

5.4 PML Processing

The structure of PML definitions, actions, and messages is defined in a XML schema. In addition, a protocol is established that specifies how these three document types are processed within the system. The protocol consists of a number rules such as: If the 3D player receives a definitions or actions document, it registers all scene elements, allocates the required resources and sends a PML message with the state 'fetched'. Only PML actions need to be started explicitly. This can be done via the 'start' attribute in the document itself (see Fig. 15) or by sending a PML message with the 'start' command. The 3D player uses the states 'started', 'failed' and 'finished' to signal that it has started executing the set of actions, that some error occurred while trying to execute them, and that all actions have been successfully terminated. PML definitions and messages on the other hand do not have to be started explicitly but are executed immediately.

5.5 Other Multimodal Markup Languages

In the last two decades a number of multimodal markup languages have been developed to specify the behavior of virtual characters and multimedia objects in 2D or 3D environments. Some of them have been designed with a human author in mind, while others are capable of representing expert knowledge by providing deep information structures created by dedicated modules (e.g. a natural language generator) at runtime. The Virtual Human Markup Language[3] (VHML) and the Multimodal Presentation Markup Language[4] (MPML) have been designed to specify the behavior of virtual characters in multimedia applications. While these two markup languages support a rather broad range of concepts, other languages address more specific issues. In [23] several scripting languages for life-like characters are described, like, for example, the Affective Presentation Markup Language (APML) that focuses on the affective aspects of the communication.

PML differs from these languages mainly in the strict separation of object and character definitions and actions, and in the way PML messages are used to synchronize modules and to inform them about system and user actions. PML definitions play an important role, since they encapsulate the knowledge about scene elements and how they can be manipulated by the

[3] http://www.vhml.org/
[4] http://www.miv.t.u-tokyo.ac.jp/MPML/en/

behavior generation components. Besides they are used by the 3D player to locate and allocate the resources associated with a character or object action. PML is based on the Rich Representation Language (RRL) developed in the NECA project [25]. Both languages focus on the specification of verbal and non-verbal behaviors of characters in multi-party dialogs and on a system-internal use rather than providing a human editable form. The RRL however does not support other media types (images, videos, graphical user interface elements, etc.) which made it unsuitable for the VirtualHuman scenarios.

VI. ACTION ENCODER

The Action Encoder decouples the action planning on an abstract symbolic level from the character- and player-specific rendering of these actions. It serves as post processing component for the CDEs and ALMA. It is responsible for the action encoding, i.e. the refinement of character and object actions, for the timing and synchronization of these actions, and for generating the nonverbal behavior associated with a character's affective state.

6.1 Action Encoding

```
<actions id="cde::117" start="true">
  <character refId="Herzog">
    <speak id="s1" alignTo="null" alignType="null">
      <text>Hallo.</text>
    </speak>
    <animate id="a1" alignTo="s1" alignType="finishes">
      <gesture refId="gazeAtModerator"/>
    </animate>
  </character>
</actions>
```

Fig. 16. PML actions before Action Encoder processing.

The CDEs specify the behavior of characters and objects in the scenario by generating PML actions for verbal utterances, accompanying gestures, multimedia objects and graphical user interface elements. Available gestures are defined in a so-called *gesticon*. We use this term analogous to lexicon for a repository of gesture specifications. Gesticon entries describe gestures in terms of their physical form, meaning, and communicative function. In addition, information about character- and player-specific animations associated with this gesture is provided. Our gesticon has 158 entries (102 multiPoses, 45 singlePose, and 11 implicitPose elements). Each character has about 90 animations and 20 facial expressions (including visemes). PML actions generated by the CDEs comprise the symbolic name of the gesture (e.g. "finger ring") and possibly additional parameters (e.g. speed and hand(s) to be used). For each gesture specification an appropriate animation is selected and added to the <animate> element based on the information provided in the gesticon and the PML character and object definitions. The textual output specified by the CDEs is processed as follows: A pronunciation mapping is applied to each word in order to deal with unknown or difficult to pronounce words. Then a text-to-speech (TTS) system is used to generate the audio files and to obtain information about the phoneme types and their duration. This information is inserted in the PML actions document and later used by the player to select character-specific animations (visemes) for the lip-synchronous mouth movements. Fig. 16 gives an example of a PML actions document before it has been processed by the Action Encoder.

The character should say 'Hallo' accompanied by an eye gaze which finishes with the end of the utterance. Fig. 17 shows the same document after is has been processed by the Action Encoder. The <speak> element contains now the URL of the generated audio file and the list of phonemes and their duration obtained from the TTS. The eye gaze is realized by a procedural animation with the target 'Moderator'. The timing and synchronization of the two actions is discussed in the next section.

```
<actions id="cde::117" start="true">
  <character refId="Herzog">
    <speak id="s1" alignTo="null" alignType="null">
      <text>Hallo.</text>
      <audio src="http://vh-demo/tts_files/s1.wav">
        <phoneme refId="h" dur="74"/>
        <phoneme refId="aa" dur="46"/>
        <phoneme refId="l" dur="47"/>
        <phoneme refId="ow" dur="195"/>
      </audio>
    </speak>
    <animate id="a1" alignTo="s1" alignType="finished-by">
      <gesture refId="gazeAtModerator"/>
      <implicitPose refId="lookAtHold" target="Moderator"/>
    </animate>
  </character>
  <schedule>
    <par>
      <action refId="s1" begin="638" dur="362"/>
      <action refId="a1" begin="0" dur="1000"/>
    </par>
  </schedule>
</actions>
```

Fig. 17. PML actions after Action Encoder processing.

6.2 Timing and Synchronization

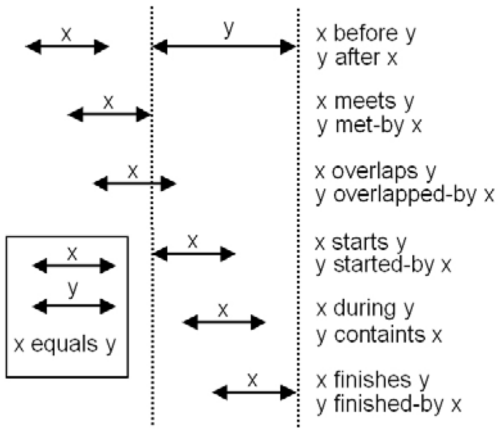


Fig. 18. Temporal constraints between two actions.

The Conversational Dialog Engines have no information about the exact duration of the specified actions since the corresponding animations and audio files have not yet been selected or generated. Therefore, only qualitative constraints have been used to synchronize these actions.

The simple duration defines the default duration of an action. For some actions the simple duration can be modified by specifying the speed with which it should be performed. The simple duration and the speed are combined to define the active duration. The start time and the active duration define the time interval during which the action will be executed in the 3D player. The set of temporal constraints depicted in Fig. 18 covers all possible relations between two time intervals.

After all character and object actions have been processed by the Action Encoder, the active duration of these actions has been determined. The temporal constraints between the actions in a PML document are then mapped to a set of linear inequalities. This allows us to use a standard constraint solver to find a solution to this constraint problem. If no solution is found, an error is generated that forces the CDEs to change the temporal alignment of the actions or to omit one or more actions. Otherwise the start times for all actions are computed and the exact timing and synchronization is specified in the <schedule> element of the PML actions document using a SMIL (Synchronized Multimedia Integration Language) compliant syntax (see http://www.w3.org/AudioVideo/).

6.3 *Affect Rendering*

The Action Encoder is also responsible for the nonverbal behavior associated with a character's affective state. It receives the affect output produced by the affect module and produces PML character actions that control a character's facial expression, complexion, and idle behavior. A character's dominant emotion and its intensity are used to select an appropriate facial animation and to instruct the player to change a character's complexion by smoothly interpolating between different skin textures. The current mood is expressed through the character's idle behavior. The idle behavior for each mood is a set of animations that are performed in between and sometimes in addition to the gestures specified by the CDEs. The mapping between emotions and moods on the one hand and a character's behavior on the other hand is defined by affect display rules as illustrated by the following examples:

R1: gratification → face=joy*0.8 AND complexion=redCheeks
R2: bored → bored, boredPreparation, boredRetract
…

If the left hand side is an emotion type, the right hand side contains a list of facial expressions (optionally with a factor that is combined with the emotion intensity to obtain the intensity of the resulting expression) and a complexion. If the left hand side is a mood type, then the right hand side contains the name of the new idle behavior and optionally the corresponding animations that initiate or reverse the posture shift with respect to the character's default posture. If a character's dominant emotion is 'gratification' with intensity 0.4 and the mood changes from relaxed to bored, the Action Encoder generates the respective PML actions (see Fig. 19).

The information about the dominant emotion and the current mood can also be used to modify the animation parameters, e.g., to increase or decrease the speed of conversational gestures and the frequency of the eye blinking idle behavior.

VII. SUMMARY

In this paper we presented an overview to the affective behavior modeling of VirtualHuman characters. Based on an empirically evaluated model of affect various behavior modalities of VirtualHuman characters are controlled. These can be divided in affective and conversational behaviors. Affective non-verbal behavior is realized by facial expressions, and complexions, as well as affective animations, and mood dependent idle-behaviors reflecting mood specific posture changes and idle gestures. The interaction style of each VirtualHuman character is also influenced by affect. According to the affective state, which is represented by active emotions and mood, dialog strategies are selected. On the surface level, the wording and the phrasing also reflects the current affective state of a VirtualHuman character.

```
<actions id="ae::234" start="true">
  <character refId="Herzog">
    <animate id="a1" alignTo="null" alignType="null">
      <face refId="Gratification" intensity="0.32"/>
      <singlePose refId="joy" intensity="0.32"/>
    </animate>
    <complexion id="a2" alignTo="null" alignType="null"
                    refId="redCheeks" intensity="0.4" dur="10000"/>
    <stopIdleList id="a3" refId="Relaxed" alignTo="null"
                    alignType="null"/>
    <animate id="a4" alignTo="a3" alignType="after">
      <posture refId="RelaxedRetract"/>
      <multiPoses refId="idle_relaxed_stop"/>
    </animate>
    <animate id="a5" alignTo="a4" alignType="after">
      <posture refId="BoredPreparation"/>
      <multiPoses refId="idle_bored_start"/>
    </animate>
    <startIdleList id="a6" refId="Bored" alignTo="a5"
                    alignType="after"/>
  </character>
  <schedule>
    <par>
      <action refId="a3" begin="0" dur="0"/>
      <action refId="a2" begin="0" dur="10000"/>
      <action refId="a1" begin="0" dur="2000"/>
      <action refId="a4" begin="1" dur="1900"/>
      <action refId="a5" begin="1902" dur="2600"/>
      <action refId="a6" begin="4503" dur="0"/>
    </par>
  </schedule>
</actions>
```

Fig. 19. PML actions for affect display.

On the technical side this is realized by a new approach that allows direct affective behavior commands for the rendering of virtual characters. We presented the Player Markup Language that supports the incremental specification of the affective multimodal behavior of the virtual characters using both qualitative and quantitative temporal constraints. We developed an Action Encoder module that decouples the action planning on an abstract symbolic level from the character- and player-specific rendering of these actions. The Action Encoder is also responsible for generating the nonverbal behavior associated with a character's affective state by mapping the Affect Module's output to PML actions using a set of affect display rules. The parallel processing of conversational and affective behaviors in our modular architecture using a broad set of different modalities enhances the expressivity and produces a believable human-like interaction behavior of our virtual characters.

ACKNOWLEDGMENT

We gladly acknowledge the valuable contributions to the PML specifications of our colleagues Yvonne Jung and Alexander Rettig at the Fraunhofer IGD in Darmstadt and our colleague Gernot Gebhard at DFKI.

The present work of the VirtualHuman project is funded by the German Ministry for Education and Research under grants 01 IMB 01A.

REFERENCES

[1] J. Gratch and S. Marsella. Modeling Emotions in the Mission Rehearsal Exercise, in *Proceedings of the 10th Conference on Computer Generated Forces and Behavioral Representation*, pp. 457-466, 2001.

[2] D. R. Traum and J. Rickel. Embodied Agents for Multi-party Dialogue in Immersive Virtual Worlds, in: *Proc. of the First International Joint conference on Autonomous Agents and Multiagent systems*, pp. 766-773, 2002.

[3] J. Lester, J. L. Voerman, S. G. Towns and C. B. Callaway. Cosmo: A Life-like Animated Pedagogical Agent with Deictic Believability, in: *Proc. of the IJCAI97 Workshop on Animated Interface Agents: Making them Intelligent*, Nagoya, 1997.

[4] J. Gratch. Émile: Marshalling Passions in Training and Education, in: *Proc. of Autonomous Agents '00*, pp. 325-332, 2000.

[5] G. Ball and J. Breese. Emotion and Personality in a Conversational Agent, in [24], pp. 189-219, 2001.

[6] B. de Carolis, C. Pelachaud, I. Poggi and M. Steedman. APML, a Markup Language for Believable Behavior Generation, in [23], pp. 65-85.

[7] H. Prendinger, S. Saeyor and M. Ishizuka. *MPML and SCREAM: Scripting the Bodies and Minds of Life-Like Characters*, in [23], pp. 213–242.

[8] M. Löckelt. Action Planning for Virtual Human Performances, in: *Proceedings of the International Conference on Virtual Storytelling 2005*, Strasbourg, France, 2005.

[9] P. Gebhard. ALMA-A Layered Model of Affect, in: *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 29-36, 2005.

[10] R. Krause , Affekt, Emotion, Gefühl, in: *W. Merten and B. Wandvogel Handbuch psychoanalytischer Grundbegriffe*, Kohlhammer, pp. 73-80, 2000.

[11] W. N. Morris. Mood: The Frame of Mind, New York: Springer-Verlag, 1989.

[12] R. J. Davidson. On Emotion, Mood and Related Affective Constructs, in: P. Ekman & R. J. Davidson (Eds.) *the Nature of Emotion: Fundamental Questions*, New York: Oxford University Press, pp. 51-55, 1994.

[13] R. R. McCrae and O. P. John. An Introduction to the Five-factor Model and Its Implications, in *Journal of Personality*, vol. 60, pp. 171-215, 1992.

[14] P. Becker. Structural and Relational Analyses of Emotion and Personality Traits, in *Zeitschrift für Differentielle und Diagnostische Psychologie*, vol. 22, no. 3, pp. 155-172, 2001.

[15] D. Watson and L. A. Clark. On Traits and Temperament. General and Specific Factors of Emotional Experience and Their Relations to the Five-factor Model, in *Journal of Personality*, pp. 441-476, 1992.

[16] P. Gebhard, M. Kipp, M. Klesen and T. Rist. Adding the Emotional Dimension to Scripting Character Dialogues, in: Proc. of the *4th International Working Conference on Intelligent Virtual Agents*, pp. 48-56, 2003.

[17] P. Gebhard, M. Klesen and T. Rist. Coloring Multi-Character Conversations through the Expression of Emotions, in: Proc. of the *Tutorial and Research Workshop on Affective Dialogue Systems*, pp. 128-141, 2004.

[18] A. Ortony, G. L. Clore and A. Collins. The Cognitive Structure of Emotions, *Cambridge University Press, Cambridge*, MA, 1988.

[19] A. Mehrabian. Pleasure-arousal-dominance: A General Framework for Describing and Measuring Individual Differences in Temperament, in *Current Psychology*, vol. 14, pp. 261-292, 1996.

[20] A. Mehrabian. Analysis of the Big-five Personality Factors in Terms of the PAD Temperament Model, in *Australian Journal of Psychology*, vol. 48, no. 2, pp. 86-92, 1996.

[21] R. Westermann. Empirical Tests of Scale Type for Individual Ratings, in *Applied Psychological Measurement*, 9, pp. 265-274, 1985.

[22] P. Gebhard and K. H. Kipp. Are Computer-Generated Emotions and Mood Plausible to Humans? in: Proceedings of *the 6th international Conference on Intelligent Virtual Agents (IVA 2006)*, pp. 343-356, 2006.

[23] H. Prendinger and M. Ishizuka. Life-Like Characters-Tools, *Affective Functions and Applications*, Springer, 2004.

[24] J. Cassell, J. Sullivan, S. Prevost and E. Churchill. Embodied Conversational Agents, The MIT Press, Cambridge, Massachusetts, 2000.

[25] P. Piwek, B. Krenn, M. Schröder, M. Grice, S. Baumann and H. Pirker. RRL: A Rich Representation Language for the Description of Agent Behaviour in NECA, in Proceedings of the *Workshop on Embodied Conversational Agents-Let's specify and evaluate them*! 2002.

Martin Klesen was born in St. Wendel, Germany. He studied computer science, mathematics, and cognitive psychology at the University of Saarbrücken and the University of Edinburgh and received his Diploma degree in 1997.

He works as a research scientist in the Intelligent User Interfaces department at the German Research Center for Artificial Intelligence (DFKI) in Saarbrücken. His research interests are intelligent virtual agents, in particular, the role of status in social interaction, authoring tools for high-level behavior control, and XML-based multimodal markup languages. He participated in several in-house, national and international projects, including PUPPET, NECA, CROSSTALK, COHIBIT, and VIRTUALHUMAN.

Martin Klesen served as a member of the program committee of the Intelligent Virtual Agents (IVA) conferences and acted as a reviewer for several major conferences in the field of intelligent user interfaces and embodied conversational agents.

Patrick Gebhard was born in Heilbronn-Sontheim, Germany. He studied computer science and physics at the University of Saarbrücken and received his Diploma degree in 1999.

He is working as a research scientist at the Intelligent User Interfaces department at the German Research Center for Artificial Intelligence (DFKI) in Saarbrücken. His research interests are affective intelligent virtual agents and computational models of affect based on cognitive theories. Furthermore he is engaged in the investigation of authoring tools for interactive story telling that use multiple virtual characters. He participated in several in-house, national and international projects, including PRESENCE, SAFIRA, CROSSTALK, COHIBIT, and VIRTUALHUMAN.

Patrick Gebhard served as a member of the program committee of the Intelligent Virtual Agents (IVA) conferences and acted as a reviewer for several major conferences in the field of intelligent user interfaces and embodied conversational agents.