

An Algebra for combining MPEG-4 compliant Facial Animations

Aldo Paradiso
Fraunhofer IPSI
D-64293 Darmstadt, Germany
paradiso@ipsi.fraunhofer.de

ABSTRACT

In this paper we discuss an extension of the Algebra of Expressions, an algebraic structure introduced in an earlier paper, consisting of a set of operators and related properties defined over a set that is a generalization of the MPEG-4's Facial Animation Parameters. In particular, we extend the definition of the operators acting on whole animations, i.e. sequences of frames representing facial dynamics, instead of still facial displays. A set of examples of algebraic expressions combining animations of a facial model is then presented, showing that the algebra of expressions may be used: a) to describe, manipulate, and generate in a compact way facial dynamics; b) as a tool to further study and better understand the role of emotions conveyed by facial expressions and their relationships; c) as a basis to define an animation algorithm for MPEG-4 compliant facial models.

Keywords: MPEG-4 FAP, facial expression manipulation, authoring animation of agents.

1 Introduction

Research on Facial Animation, developed since the 70s, has reached several results, including the definition of techniques for performing facial expressions in synthetic models. Of particular relevance are the FACS system, developed by Ekman, and its evolution, the MPEG-4 facial animation model.

The work described in this paper tries to give a contribution by introducing a set of operators acting on facial displays where the visual patterns are described and coded with MPEG-4 Facial Animation Parameters that in turn are applied onto a facial model. MPEG-4 is an object-based multimedia compression standard, which allows for encoding of different audio-visual objects of a scene independently. In particular, MPEG-4 enables the integration of facial animation with multimedia communications and presentations and allows facial animation over low-bandwidth communication channels. In MPEG-4, each facial expression is coded as an ordered sequence of 68 integer numbers, called Facial Animation Parameters (FAPs). Each of the parameters 3-68 acts on points defined on a synthetic face (feature points), and each point governs the deformation of its surrounding area, resembling muscle movements [8]. This approach is an evolution of the FACS coding system developed by Ekman and colleagues in 1972 [4]. Several works have been carried out where synthetic characters have been designed to simulate human senses. Among them it is worth noting the complete work of Cassell et al. Cassell's team developed REA [2], a system designed to sense users actions and behaviors during a face-to-face conversation, carrying on reactions and answers based not only on dialogical but even on non-verbal cues. Results achieved in the analysis developed by Cassell's team as well as their terminology has been adopted in this work [1].

Classifications of facial animation techniques and approaches can be found in [3], [14], and [7]. Other works can be found in [20]. Most of them address the possibility of generating facial expressions, and relate them to the ongoing dialogue. However, the adopted

solutions lack of generality. Most systems work optimally in narrow environments, where special facial models are needed, or a limited set of facial expressions has been defined, and tailored *ad-hoc* to the application. In other words, there is the need of an ideal facial animation system that gives the possibility to fully control a set of parameters for both the definition of the facial model and the definition of facial expressions and animations. Also, an ideal parameterization and interface should allow the animator to easily specify any individual face with any speech and expression sequence [15]. The FACS system has been the best current basis for low-level expression parameterization, but it was too abstract from the animator's viewpoint. The development of MPEG-4 facial animation model, with the introduction of the FAPs, has been a great development of the FACS. Still, MPEG-4 is too low-level, and it misses an upper layer that allows defining, mastering, and manipulating facial expressions as a whole. As a matter of facts, the human face can generate around 50.000 distinct facial expressions, which correspond to about 30 semantic distinctions [19]. Clearly the human face is extremely expressive. Is it possible to cover this amount with a minimal set of expressions and a set of transformations on such a set, without manipulating directly the set of facial animation parameters?

In this work an answer is suggested where operators have been defined in order to combine together existing facial expressions and derive new ones. In research on facial animation other authors have explored the possibility of combining facial expressions. Ken Perlin has developed a system – *Improv* – where he showed how to make an embodied agent react with responsive facial expression, without using repetitive pre-built animations, and how to mix those facial expressions to simulate shifting moods and attitudes [17]. A convincing demo has been given in [16] where Perlin's Noise function has been employed to provide the facial model with pseudo-natural movements. In MPEG-4 an algorithm to blend together two expressions is suggested; in this approach the concept of excitation (or intensity) of an expression is mixed with the one of combination of two expressions. On our side we extend

and distinguish these two concepts, by defining two different operators, introduced in the following section. Tao et al. [18] propose a method to encode human facial movement patterns. They compress FAP streams in a more compact format, and approximate the function that represents the FAP values over time (called by them projection function) with a series function being the sum of simpler functions (called humps). The composition of hump functions may therefore approximate with arbitrary precision the projection function by adding more terms.

Other authors describe what they call “intensity of expressions” even if in most cases they do not clarify what “intensity” means: whether it refers to muscle tension or to human-perceived actions, or other. Won-Sook Lee and al. [21] describe dynamics of facial expressions in terms of three different types of envelopes. They also define an action blending technique in order to avoid discontinuities between facial movements when different expressions and visemes are dynamically concatenated in a unique animation flow. Neumann and al. [9] propose a process called *expression cloning* that provides a new alternative for creating facial animations for character models. Their method transfers each vertex motion vector from a source face model to a target model having possibly different geometric proportions and mesh structure.

As an alternative approach we introduce an algebraic structure (a preliminary version can be found in [12]) consisting of a set of operators and related properties defined over a set that is a generalization of the MPEG-4’s FAPs. In other words we pursue the possibility of creating new facial expressions by acting on instances of previous expressions, and we do that in a formal way, by first of all developing an algebra where a set of operators is defined on a domain modeled according to the definition of MPEG-4 FAP. Secondly, we discuss how to relate the formal construct to a world populated by “real” facial expressions.

We do not address directly body gestures, but the Algebra of Expressions that we introduce later may be employed to describe and generate such movements as well. In addition, we do not address speech. However, the representation model we introduce below allows speech synchronization, and we implemented a facial animation prototype where a TTS system has been integrated [6].

In the following section we briefly introduce an algebraic structure where an initial set (of facial displays) is defined, together with a set of operators. An extensive introduction of this structure together with a demonstration of relative set of properties can be found in [13]. Section 3 contains an extension of the algebra, by formally introducing operators for combining animations. The successive section contains some examples of operations, implemented in the Tinky system [10] and performed on a facial model. Finally, some conclusions are drawn in section 5.

2 Definition of the Algebra of Expressions

In this section we introduce an algebraic structure, called Algebra of Expressions (AoE), defined onto a set

modeled over the MPEG-4 set of Facial animation Parameters, and whose constructs will allow to manipulate facial expressions in order to generate new ones.

2.1 Facial states

A Facial state S is defined as a set of 66 integer numbers (s_1, \dots, s_{66}) where, each element s_i ($i=1, \dots, 66$) satisfies the constraints defined in the MPEG-4 Facial Animation Parameter definitions table (FAP 1 and 2 of the standard are not considered because they do not represent facial movements related to feature points. Thus, in this notation s_1 represents FAP 3, ..., s_{66} represents FAP 68). These constraints require some of the elements to be non-negative.

Domain: We denote the domain, i.e. the space of all the facial states, with the symbol \mathcal{S} . Such a domain is a subset of the Cartesian product of the integer set, \mathbb{Z}^{66} . That is, there exists a family of sets S_1, S_2, \dots, S_{66} such that

$$\mathcal{S} = S_1 \times S_2 \times \dots \times S_{66}$$

All the sets S_i ($i=1, \dots, 66$) are subject to the constraints defined in the MPEG-4 FAP definitions table (see table of FAPs in [8]). For example, the element $s_i \in S_i$, corresponding to FAP 3 (*open_jaw*) may only assume positive values.

2.2 Sum

If we have two facial states $S=(s_1, \dots, s_{66})$ and $T=(t_1, \dots, t_{66})$, with $s_i, t_i \in \mathbb{Z}$ (natural numbers), we define the sum operator $+_v$ as the following facial state:

$$+_v(S, T) = \lfloor s_i v \rfloor + \lfloor t_i (1-v) \rfloor \quad (1)$$

where $i=1, \dots, 66$ and $v \in [0,1]$

If $v=0.5$ this operator will produce a facial state being the mean of the first two ones. Since v is a real number, the products $s_i v$ and $t_i (1-v)$ are real numbers as well. Therefore we used the symbols $\lfloor \cdot \rfloor$ to denote the approximation to the closest integer number.

For simplicity we denote $+_v(S, T)$ with $S +_v T$.

Indeed, the parameter $v \in [0,1]$ introduces an infinite family of operators. However, for simplicity, we continue to denote $+_v$ as a single operator.

The set \mathcal{S} is closed under summation. In addition, the Sum operator is neither commutative nor associative in the strict sense, but in a slightly generalized sense both properties remain valid. Also, it does not have an identity. Demonstrations of these assertions can be found in [13].

2.3 Amplifier

An amplifier $w \in \mathfrak{R}$ is a scalar operator acting on a single facial state $S=(s_1, \dots, s_{66})$, such that:

$$S \cdot w = w \cdot S = (\lfloor w \cdot s_1 \rfloor, \lfloor w \cdot s_2 \rfloor, \dots, \lfloor w \cdot s_{66} \rfloor) \quad (2)$$

The symbols $\lfloor \cdot \rfloor$ denote the approximation to the closest integer number. Indeed, the amplifier is a particular case of sum, where the second element is the zero vector $N = (0_1, \dots, 0_{66})$. In fact, we have $S +_w N = S \cdot w$. However, for simplicity, we will consider the amplifier as a different operator, because it will be largely employed. The set \mathcal{S} is closed with respect to the amplifier operator. The identity value for the amplifier is 1 [13].

2.4 Overlapper

Let us introduce a mask $M=(\alpha_1, \dots, \alpha_{66})$ where $\alpha_i=0$ or 1 . Mask vectors will serve us to identify partial regions of the complete FAP vector without losing generality, i.e. we deal with vectors having the same size in all our computations. If we have a facial state $S=(s_1, \dots, s_{66})$, M identifies a partial region $S_M = S \cdot m_{ij}$ where m_{ij} is the diagonal matrix of M (i.e. $m_{ij}=0$ for $i \neq j$ and $m_{ij}=\alpha_i$ for $i=j$; $i, j=1, \dots, 66$). Then, if we consider two facial states S, T with masks respectively M_1 and M_2 and priorities p_1 and p_2 integers, $p_1 \neq p_2$, we introduce the overlapping operator Ω so defined:

$$\Omega(S_{M_1, p_1}, T_{M_2, p_2}) = (g_i) = \begin{cases} g_i = s_i + t_i & \text{if } s_i \text{ or } t_i = 0 \\ g_i = s_i & \text{if } p_1 > p_2; g_i = t_i \\ & \text{otherwise} \end{cases} \quad (3)$$

where $i=1, \dots, 66$.

For simplicity we denote

$\Omega(S_{M_1, p_1}, T_{M_2, p_2})$ with $S_{M_1, p_1} \Omega T_{M_2, p_2}$ or simply with $S \Omega T$ if this does not bring any ambiguity. If the priorities are the same, the operation remains undefined. As for the sum, we actually introduced a family of operators, defined both by the variability of mask vectors and the one of priorities. For simplicity we will consider a single operator, meaning the whole family.

The set \mathcal{S} is closed with respect to the overlapper. In addition, the set \mathcal{S} has an identity with respect to the overlapper, but it does not have an inverse element. The overlapper holds both the commutative and the associative properties, as demonstrated in [13]. In the same work it has shown that the amplifier distributes over the sum. That is to say, given two facial states $S, T \in \mathcal{S}$, we have: $w \cdot (S +_v T) = w \cdot S +_v w \cdot T$ with $v \in [0, 1]$.

2.5 Examples

The operators introduced above have been implemented as methods included in an Application Programming Interface developed in Java. The API has been incorporated as a part of the Tinky and Fanky systems, developed by the author in [10] and [11]. The facial model on which the operators have been applied has been employed in both Tinky and Fanky systems. It consists in a VRML-based 3-D cartoon face, with low level of details, but enough to represent facial expressions. Figure 1 shows the employment of the overlapper. In (a) a smiling face and in (b) an angry face are shown. The mask associated to (a) covers the mouth area, while the mask associated to (b) covers the eyes and eyebrows areas. Their overlap, according to these masks, is shown in (c). Note that the figure in (c) resembles a surprised expression. This shows how it is possible to derive new expressions from old ones that are semantically different. By using the same expression switching the role of the masks a different overlap is produced, as shown in (d): the mouth is smiling and the eyes are angry. Again, this expression holds a semantic different from the previous ones, signaling smartness, arrogance or sense of superiority. The employment of

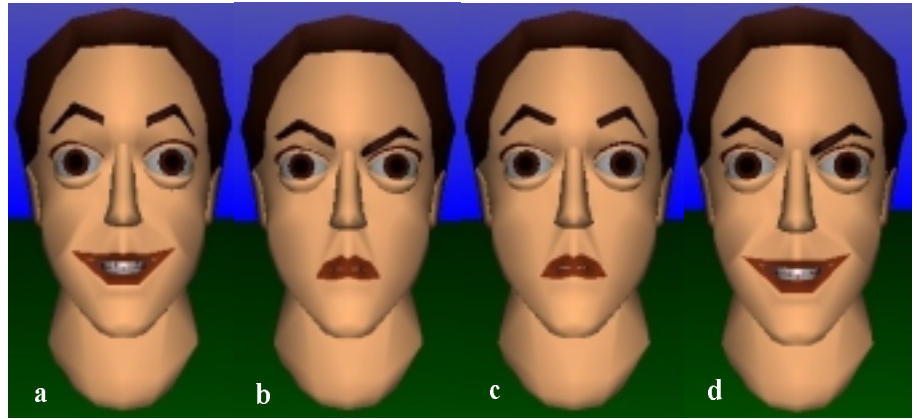


Figure 1 - Overlapping expressions. In order we have: smile (a), angry (b), one overlap (c), and a second different overlap (d).

the overlapper is somehow complicated by the necessity of defining masks. In practice, such masks allow defining patches to be glued together, like in a patchwork. However, it is not straightforward to define masks. In order to make such a task easier, a GUI containing a window with colored labels have been defined in the Tinky system, that allows to easily define masks by clicking onto check buttons identifying facial areas. Other examples with operators can be found and are discussed in [13].

3 Towards an algebra of animations

Facial displays, as introduced and described so far, are only snapshots taken, or produced, at fixed instances of time. The algebra of expressions, as developed so far, cannot be employed in dynamics of facial displays. One way to properly exploit the algebra is to employ it in animation systems where animations are not pre-defined but produced combining sets of elementary animations, each reproducing a unique emotion or a single comment (eye blinking, eyebrow squeezing, etc.). In our approach, an animation is a sequence of expressions, together with a set of time steps occurring between them. We exploit

the animation technique called *keyframe animation*, where only a few keyframes are defined and an animation engine generates in-between frames according to an interpolation algorithm. In this section an extension of the algebra of expressions is proposed, by defining operators able to manipulate animations, intended as sequences of frames played by an animation engine.

Let us introduce a generic animation, called A , acting over a set of keyframes D_i , starting at a time T and lasting for an interval of duration d , in symbols: Animation = $A_{T, d}(D_i)$; in short $A()$.

In this context $A()$ is an animation that refers to same facial regions, that is, a sequence of frames that simulate performing some facial display starting from a neutral expression, evolving across the sequence of displays D_i and returning back to the neutral expression, according to an envelope function. In other terms we have:

$A_{T, d}(D_i) = A_{T, d}(D_0, D_1, \dots, D_n)$ where $D_0 = D_n = N$, the neutral expression (i.e. the facial state having all the values equal to zero) The parameter n , that we indicate with $|A()$ is the *cardinality* of an animation, i.e., the number of keyframes of which it is composed.

An animation is characterized not only by the value and number of its components, i.e. the display vectors, but also by the parameters T and d , the starting time, and the duration. In order to give sense to these parameters we hypothesize some *Animation Machine AM* that is able to interpret the animation sequences and perform the animation on a MPEG-4 compliant *facial model FM*.

Interpreting an animation means that the animation machine is able to produce facial displays (keyframes) onto the facial model FM , starting at time T , and ending at time $T+d$. In particular, the set of frames D_0, D_1, \dots, D_n is reproduced in ordered sequence onto the facial model. This means that there exist a sequence of time

intervals I_1, I_2, \dots, I_{n-1} such that $\sum_{i=1}^{n-1} I_i = d$ and that D_0 is

rendered at time T , D_1 at time $T+I_1$, D_2 at time $T+I_1+I_2, \dots, D_n$ at time $T+d$. The time intervals may have the same length or not. For simplicity we will consider only animation machines that are able to produce time intervals of equal length. In other terms, if d is the duration of the animation, and n the number of keyframes, we have $I_i = d/(n-1)$ for any $i = 1, \dots, n$.

We define two animations $A_{T,d}()$ and $B_{T_1,d_1}()$ to be *homogeneous* if the time intervals occurring between the keyframes of both animations coincide. In other words, they are homogeneous if, given $n=|A|$ and $m=|B|$ we have $d/(n-1) = d_1/(m-1)$.

Two homogeneous animations, then, run according to the same clock frequency. In addition, we say that two animations are *synchronized* if their frequencies are synchronized. The synchronization of two animations, however, does not provide information on how much time occurs between the two of them. Therefore we say that two animation functions are *adjacent* if they are homogeneous and synchronized and there exist a unique time T where a facial display of both of them is reproduced onto a facial model FM . In other words, if we have two animations $A_{T,d}(D_1, \dots, D_n)$ and $B_{T_1,d_1}(E_1, \dots, E_m)$ there exist an absolute time t_i where the animation machine interprets a certain D_k of $A()$ and contemporary a certain E_l of $B()$.

At this point we are able to extend the definition of the algebra operators to the animations.

Definition of overlapper for animations

Given two *adjacent* animations $A(D_1, \dots, D_n)$ and $B(E_1, \dots, E_m)$ acting on the two sets of displays D_i ($i=1, \dots, n$) and E_j ($j=1, \dots, m$) we define:

$A(D_i) \Omega B(E_j) = C(F_1, \dots, F_p)$ where

$$F_k = \begin{cases} D_i & \text{if at the } i^{\text{th}} \text{ interval only } D_i \text{ occurs} \\ E_i & \text{if at the } i^{\text{th}} \text{ interval only } E_i \text{ occurs} \\ D_i \Omega E_i & \text{if at the } i^{\text{th}} \text{ interval both } D_i \text{ and } E_i \\ & \text{occur} \end{cases} \quad (4)$$

Of course, we need to define masks and priorities for all overlapped facial displays, and in general, for the complete sequences D_i and E_j . Overlapping two animations means, in general, obtaining a longer animation, whose starting time is $T_{\min} = \min(T_i, T_j)$, the smallest starting time of the two operands, and its duration d lays in the interval $[d_{\min}, d_i+d_j]$ where $d_{\min} = \min(d_i, d_j)$.

In our framework, our purpose is to generate sequences of facial displays to seamlessly reproduce facial animation. Therefore we still need an operator to concatenate together animations, without necessarily overlapping them. For this purpose, we extend the definition (4), as follows:

If we have two *synchronized* animations $A(D_1, \dots, D_n)$ and $B(E_1, \dots, E_m)$ acting on the two sets of displays D_i ($i=1, \dots, n$) and E_j ($j=1, \dots, m$) we define:

$A(D_i) \Omega B(E_j) = C(F_1, \dots, F_p)$ where

$$F_k = \begin{cases} \text{The same values in the cases seen in} \\ \text{definition (4)} \\ N \text{ (neutral display) if none of } D_i \text{ and } E_i \text{ occur} \end{cases} \quad (5)$$

The only difference between definition (4) and (5) is that in the last case also not adjacent (disjoint) animations are allowed. In such a case the overlapper is simply a concatenation of the given animations, where the in-between empty slots (if any) are filled with sequences of neutral displays. With definition (5) we are able to formally produce concatenations of animations that last at desire, without interruptions. This allows the definition of an animation engine that guarantees a continuous animation (even if filled with trivial displays, i.e. neutral expressions). Of course, a real implementation of the animation engine, here only formally introduced, is necessary to guarantee the production and concatenation of the stream of displays that realize the animation flow.

Definition of sum for animations

Given two *adjacent* animations $A(D_1, \dots, D_n)$ and $B(E_1, \dots, E_m)$ acting on the two sets of displays D_i ($i=1, \dots, n$) and E_j ($j=1, \dots, m$) we define:

$A(D_i) +_v B(E_j) = C(F_1, \dots, F_p)$ where

$$F_k = \begin{cases} D_i & \text{if at the } i^{\text{th}} \text{ interval only } D_i \text{ occurs} \\ E_i & \text{if at the } i^{\text{th}} \text{ interval only } E_i \text{ occurs} \\ D_i +_v E_i & \text{if at the } i^{\text{th}} \text{ interval both } D_i \text{ and } E_i \\ & \text{occur} \end{cases} \quad (6)$$

As in the definition of the sum for still displays, $v \in [0,1]$.

Definition of amplifier for animations

Given an animation $A(D_1, \dots, D_n)$, we define the amplifier for animations a scalar operator $w \in \mathfrak{R}$ so that $A \cdot w = w \cdot A = A(w \cdot D_1, \dots, w \cdot D_n)$, where $w \cdot D_i$ is the amplifier operator applied to the facial display D_i . The amplifier may be used to enhance or inhibit the effect of facial displays involved in an animation (for example, make a grimace stronger or a smile weaker, etc.).

4 Examples

The operators introduced in the previous section have been implemented in the Tinky system, developed by the author in [10]. Let us show some examples of animations built by using the above-defined operators combining previously defined animations. For simplicity, we have considered animations starting all at the same time, having the same duration, and synchronized; the animation machine is the one embedded in the Tinky system, and all the animations have the neutral expression as the initial keyframe. In the following pictures only keyframes have been shown, though the

animation engine generates in real time linearly interpolated intermediate frames.

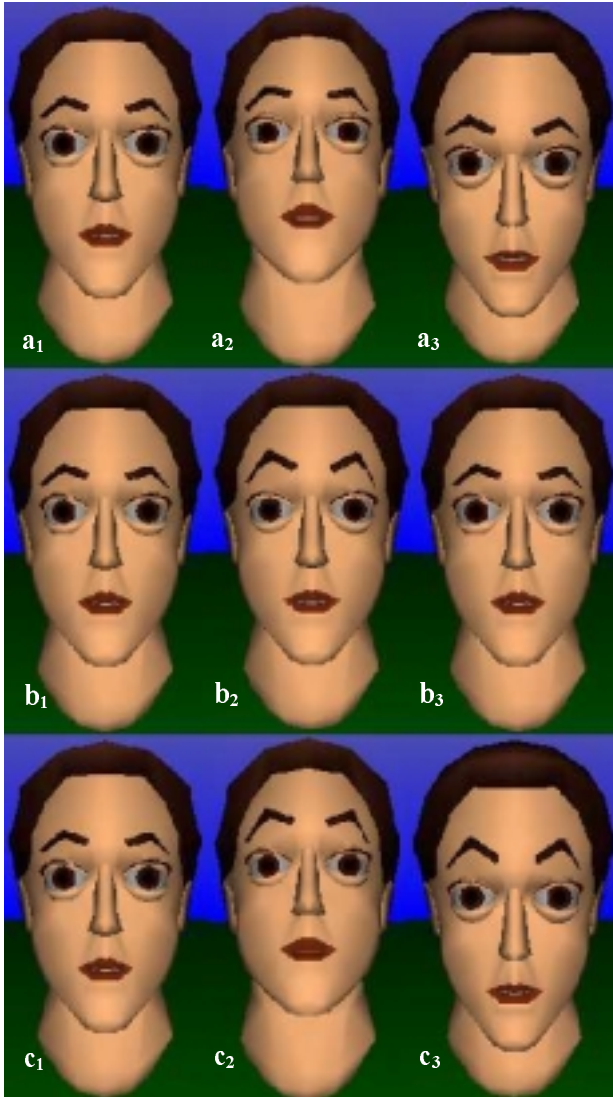


Figure 2 - Overlap of two animations

A nodding expression includes one’s head up and down performed once or several times (not more that three-four, otherwise it may bring a completely different meaning, like sarcasm, or even a pathological behavior). Such a behavior may be implemented by generating at least two frames, containing the FAP number 48 (*head_pitch*) with values, for example, 2000 and -2000 , or similar values. Let us call such a display N (nod) and the relative animation $A_{T,1}(N)$ (starting at time T , and lasting 1 second). The progression is shown in figure 2, sequence (a1), (a2), and (a3). However, nods may be accompanied with other contemporary displays that emphasize the whole meaning or indicate some acknowledgment of some other’s words, such as raising eyebrows. Let us define a facial display with raised eyebrows, and call it E (eyebrows). We can define an animation $B_{T,1}(E)$ (starting, for simplicity, at time T and lasting 1 second). The sequence is (b1), (b2), and (b3) in figure 2. In our example we want that nodding and raising eyebrows must occur at the same time, and integrally, i.e. their respective intensity is not changed. Since these two displays must be shown contemporary, they concur to form a unique facial display R (result)

being the overlap of the two ones. We have $R = E \Omega N$ where the mask of E covers (i.e. turns to zeroes) all the FAPs that do not involve head movement, and the mask of N covers everything except the eyebrows area. If we overlap the two animations we have $C_{T,1}(R) = A_{T,1}(N) \Omega B_{T,1}(E)$. The resulting animation starts, for simplicity, at the same time, and lasts 1 second. The progression is shown in figure 2, sequence (c1), (c2), and (c3).

Let us now imagine that the character is talking, or more generally it’s performing some mouth movement, like visemes or opening for surprised expression (figure 3, sequence (d1), (d2), and (d3)).

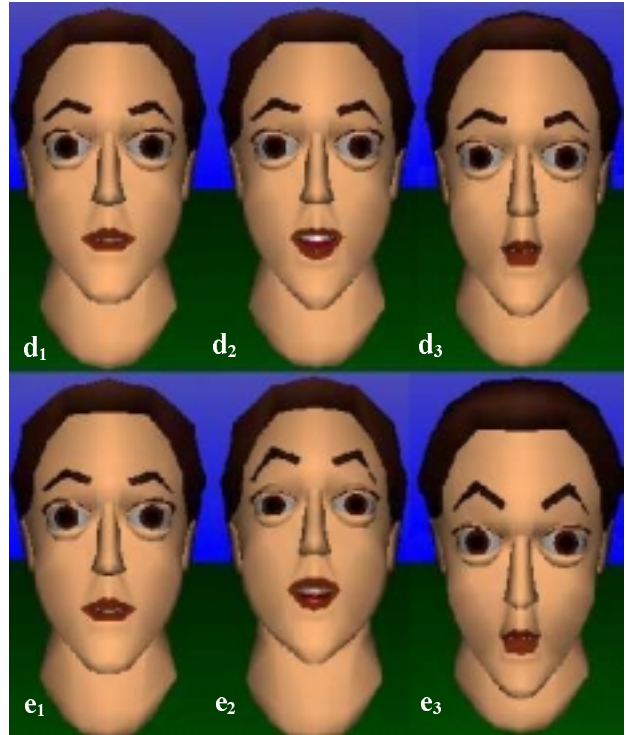


Figure 3 - Overlap of three animations

In such a case a third area is involved in the facial display that corresponds to a third, separate and independent communication channel like the mouth. The independency is also revealed by the fact that the FAPs involved in mouth opening are different from the others so far considered. If we name M the facial display showing a moving mouth we have $R = E \Omega N \Omega M$ (the

Table 1 - Mapping of behavior to facial displays

Keyword	Agent Behavior	Facial Displays
“Really”	Showing interest	1 - Raise eyebrows 2 - Rolling the head slightly
“Yeah”	Being bored	3 - Looking ahead 4 - Closing eyelids longer
“What?”	I do not understand	5 - Inner eyebrows pulled downward and together 6 - turning the head to show one ear meaning “What?”
“Hummm”	I’m thinking	5 - Inner eyebrows pulled downward and together, and 3 - looking ahead, or 7 -downwards.
</emotion/> tag	joy, sadness, anger, fear, disgust, surprise	8) smiling, 9) sad, 10) angry, 11) frightened, 12) disgusted, 13) surprised

brackets are not necessary because the overlapper is associative). The resulting progression, showing a nodding face with raising eyebrows and a moving mouth, is in figure 3, sequence (e1), (e2), and (e3).

A system where the algebra has been partially employed is the chatterbot included in COGITO where facial expressions are produced according to keywords and sets of rules defined in an embedded dialogue system called eBrain [12, 22]. Table 1 shows a mapping between some keywords, a related behavior, and facial displays showing it. Each single display has been numbered, and some displays are employed in different behaviors. In the example, display 3 is used in “being bored” and “I’m thinking” and display 5 in “I do not understand” and “I’m thinking”. In the same way, if each display is realized with a single animation, the agent behavior is implemented by combining together simple animations, obtaining new more complex animations representing whole behaviors. Thus, the behavior “showing interest” is obtained combining the animations “raise eyebrows” and “rolling the head slightly”. Of course, it is necessary to define a criterion on how to combine the animations. However, we believe that any combination may be obtained by using one or more operators above defined.

5 Conclusions

In this paper an algebra for combining facial expressions and animations has been proposed, based on the formalization and manipulation of MPEG-4 FAP-like families of vectors that allow an algebraic representation of facial displays. The introduction of operators that allow combining not only facial expressions but also their dynamics, i.e. animations of faces, opens new interesting opportunity to investigate how the facial clues, distinguished both anatomically (different facial areas) and semantically (visemes, emotions, comments, biological needs), may coexist and interact each other. In addition, it may be possible that a limited set of predefined animations is enough as a basis to produce a wide set of (different) animations suitable to be employed in several application scenarios, as shown in table 1. Future work includes a more thoroughly investigation on the possibility opened by the operators introduced: how effective they are in producing new animations? What is the relationship between the semantics carried out by facial expressions and the operations of this algebra? An evaluation session is planned to assess the effectiveness of the approach. In addition, we concentrate on the development of an animation algorithm that exploits the constructs of the algebra to dynamically and possibly in real-time obtain derivative animations starting from a possibly initial small set.

6 References

[1] Cassell, J.; Sullivan, J.; Prevost, S.; Churchill E. Embodied Conversational Agents. The MIT Press, Cambridge, Massachusetts, 2000. Pp. 17-21.
 [2] Cassell, J.; Sullivan, J.; Prevost, S.; Churchill E. Embodied Conversational Agents. The MIT Press, Cambridge, Massachusetts, 2000. Pp. 29-63.
 [3] Dendi, Vikram R. A face for a robot: The path to creating a face for a socially interactive robot.

Applications to Human computer Interaction. Available from the Internet at the URL: www.its.caltech.edu/~vikram/cs286/report
 [4] Ekman, P., Friesen, W. Facial Action Coding System. Consulting psychologists Press, Inc., Palo Alto, CA, 1978.
 [5] Elkman, P. The argument and evidence about universals in facial expressions of emotion. In H. Wagner and A. Monstead, editors, Handbook of Social Psychophysiology, pages 143-146. John Wiley, Chichester, 1989.
 [6] 3rd INVITE Status Report released by the FhG IPSI to the German Federal Ministry of Education and Research, February 2001.
 [7] Jun-Yong Noh: A Survey of Facial Modeling and Animation Techniques.
 [8] ISO/IEC IS 14496-2 Visual, 1999 and ISO/IEC IS 14496-1 Systems, 1999.
 [9] Neumann et al. Human Face Modeling and Animation. Integrated Media Systems Center University of Southern California 2001 NSF Report.
 [10] Paradiso, A.; Nack, F.; Fries G.; Schuhmacher, K. The Design of Expressive Cartoons for the Web – Tinky. Proceedings of ICMCS Conference, June 7-11 1999, Florence (Italy).
 [11] Paradiso, A., Zambetta, F., Abbattista, F. Fanky: A Tool for Animating Faces of 3D Agents in: Intelligent Virtual Agents - Third International Workshop, IVA2001, Madrid, Spain, September 2001 Proceedings. Springer-Verlag - Lecture notes in computer sciences, vol. 2190: Lecture notes in artificial intelligence, pp. 242-243.
 [12] Paradiso, A., L'Abbate, M. A Model for the Generation and Combination of Emotional Expressions. Workshop on Representing, Annotating, and Evaluating Non-Verbal and Verbal Communicative Acts to Achieve Contextual Embodied Agents, Fifth International Conference on Autonomous Agents, May 29, 2001, Montreal, Canada.
 [13] Paradiso, A. An Algebra of Facial Expressions. Workshop on Embodied Conversational Agents, First International Joint conference on Autonomous Agents & Multi-Agent Systems, 16 July 2002, Bologna, Italy.
 [14] Parke, F.I., Waters, K. Computer Facial Animation. Edited by A K Peters, 1996.
 [15] Parke, F.I., Waters, K. Computer Facial Animation. Edited by A K Peters, 1996, Pp. 145-146.
 [16] <http://mrl.nyu.edu/~perlin/facedemo/>
 [17] Perlin, K, Goldberg, A. Improv: A System for Scripting Interactive Actors in Virtual Worlds. Computer Graphics; Vol. 29 No. 3., 1996.
 [18] Tao, H., and S. Huang, Motion Patterns in Face Animation, IJCAI Conference, Workshop in Animated Interface Agents: making them intelligent, Nagoya, Japan, August 25, 1997.
 [19] Terzopoulos D and Waters K: ‘Analysis and synthesis of facial image sequences using physical and anatomical models’, IEEE, PAMI, 15, No 6 (06/93).
 [20] <http://mambo.ucsc.edu/psl/fan.html>
 [21] Won-Sook Lee, Escher, M., Sannier, G., Magnenat-Thalmann N., MPEG-4 Compatible Faces from Orthogonal Photos.
 [22] www.darmstadt.gmd.de/delite/Projects/COGITO/