A Fuzzy Rule-Based Interactive Methodology for Training Multimedia Actors

Savant Karunaratne and Hong Yan

School of Electrical and Information Engineering, University of Sydney,
J03 Maze Crescent, Sydney, NSW 2006

Savant@ee.usyd.edu.au and yan@ee.usyd.edu.au

Abstract

Computer animation has come a long way during the last decade and is now capable of producing near-realistic rendered 3D computer graphics models of expressive, talking, acting humanoids and other characters inhabiting virtual worlds. However, the component of work that needs to be done by animators and artists in producing these synthetic character performances is quite significant. In this paper, we present an expert system based on fuzzy knowledge bases that helps in moving towards automating the task of animating virtual human heads and faces. Our Virtual Actor (Vactor) framework is based on several subsystems that use mainly fuzzy and a minor degree of non-fuzzy linguistic rules to teach virtual actors to know the emotions and gestures to use in different situations. Theories of emotion, personality, dialogue, and acting, as well as empirical evidence are incorporated into our framework and knowledge bases to produce convincing results.

Introduction

The face is one of the most vital components in the vast ability of humans to communicate information as well as emotional and motivational cues. Thus, we have concentrated almost entirely on the modelling and animation of virtual human heads. In our previous work we have developed an abstract muscle model to generate facial expressions of a synthetic talking head (Karunaratne and Yan 2001). We have also developed mathematical models to handle the combinations of emotional, gestural and visual-speech-related components into synthesizing these expressions (Karunaratne and Yan 2000). The synthetic heads can be either generic or conformed based on the geometry of an individual (Karunaratne and Yan 1999). This work accomplished the automation of the geometric modelling component of virtual actor heads and faces. Several other systems, e.g. Eisert and Girod (1998), Thorisson (1997) perform similar functions to ours, concentrating on virtual faces, and still others, (Densley and Willis 1997, Maiocchi and Pernici 1990, Sato and Miyasato 1997) work on complete human bodies. In the present paper, our focus is on adding a certain amount of knowledge to these 3D computer graphics actors so that the task of animators will be more like the task of directors, giving guidance and extra knowledge to virtual actors, who perform a script in a movie. This involves the design of a framework for synthetic actors to select the appropriate gestures and emotions when "speaking out" a script, based on the situation of the movie, the emotional state of the characters and the text being uttered. The emotional model used is that formulated by Ortony, Clore and Collins, which is called the OCC model or OCC theory for convenience (Ortony, Clore and Collins 1988). We call our virtual actors 'vactors' and our framework, the `vactor framework'. The rest of the paper is organized as follows. Section 2 discusses some related research, which uses concepts similar to ours; section 3 discusses our entire vactor animation framework; section 4 describes the user interface of our system; section 5 introduces the fuzzy, linguistic, rule-based expert system; section 6 discusses the teaching methodology; section 7 gives examples of how synthetic actors would use the expert rules in "acting out" a script; and section 8 maps out some future work and draws conclusions.

Related Research

Kurlander, Skelly and Salesin (1996) have developed a system for online communication using comics. Different aspects of comic generation are automated. They have observed that interesting comics can be generated automatically using simple rules and semantics and without the need for natural language processing. According to their definitions gestures are body poses and expressions are facial poses. There is no animation, only static poses as in comics. Manske and Muhlhauser (1997) present a system for comic actor animation that can be a representation of different types of agents, and also for rapid authoring of animations to augment multimedia presentations. Reuse of artwork is possible using this modular, building-block approach. For this, a set of basic animation sequences is created by professional graphic artists, once per character. These animations can be repeatedly combined in custom animations by commands issued at runtime.

Thorisson (1997) describes an action composition and selection framework for synthetic characters capable of full-duplex, real-time, face-to-face interaction with a human. Some of the issues addressed are exchanging glances, turn-taking during speech, and making facial and manual gestures. Dialogue between human and synthetic character is multi-modal but task specific, relating to

"commanding a solar-system", for example. The main focus of the paper is on action-selection and low-level motor command composition. Perception mechanisms for real-time control and knowledge-bases for dialogue are required. The animation of the system is simple and caricature-like with the face being represented by 2D polygons with only 21 degrees of freedom.

Sato and Miyasato (1997) develop a method where virtual actors' behaviours are determined in response to their interactions with each other. The model presented is called the Autonomous Interactive Reaction (AIR) model. The actor behaviours depend on emotion and personality parameters. Densley and Willis (1997) describe a method for putting emotion into figure animation. An emotional model based on psychological theory is proposed and used for posturing figures. General posturing functions are interpreted based on emotions. Joint angles and body stance is affected by these emotional parameters. The emphasis of this work is on joint angles and body stance as opposed to facial animation. We have managed to fuzzify some of the aspects of this model to incorporate into ours, which handles more complex movie scene situations in facial animation.

El-Nasr, Ioerger and Yen (1999) introduce an emotional model called PETEEI (a PET with Evolving Emotional Intelligence) which is based on a fuzzy-logic model to simulate emotions in agents, with emphasis on learning where the agent can adapt its emotions according to its experience. PETEEI can also recognize and deal with moods and emotional responses of its owner. The actions, however, are the simple ones of a virtual pet dog rendered in a 2D world. Rousseau and Hayes-Roth (1998) provide synthetic actors (autonomous or avatars) that portray fictive characters by improvising their behaviour in a multimedia environment. Improvisation is based on directions received and context. Direction can be high-level actions, user commands or personality changes. A social-psychological model is used whereby personality traits can be defined based on moods and attitudes. However, direction is menu-based and can only be one of several options automatically suggested by the system. Our virtual actor facial animation framework is discussed next.

Entire Vactor Framework

Figure 1 shows a schematic of the entire Vactor Framework. The input to the system is basically a script, which is divisible into a textual script and a situational script. The input format of the script specifies a certain syntax for the script which enables the easy sub-division of the script as well as facilitating the extraction of vital parameters. The extraction of phonemes as well as gestural and emotional parameters from the speech script is done automatically based on a set of crisp rules. The situational script also has a specific format. The extraction of OCC emotions as well as other situational parameters from this script is done automatically, or by means of interactive dialogue with a human director. The main emphasis of this paper is on the automatic fuzzy expert system that uses OCC emotions, personality parameters, and situational parameters to automatically generate expressive emotional and gestural parameters for animating an expressive synthetic talking head of a virtual actor. This is elaborated in the remaining sections. Other inputs to the system include attitudes towards others as well as vactor personalities. The attitudes characterize interpersonal relationships and are similar to those modelled in Rousseau and Hayes-Roth (1998). Some attitudes include status, degree of sympathy, trust, and friendliness.

The conversational module extracts, the emotion and gesture specifications of the textual script. The situational script also extracts information on which characters are present in a situation, the size and nature of the audience, whom an actor speaks to etc. Variables that specify these parameters are: whomActorTalksTo, audiencePresent, audienceSize, and audienceActive. These are either boolean or crisp variables which are fuzzified. whomActorTalksTo has the fuzzified set values of SELF, ONE, GROUP. audienceSize has fuzzified set values of SMALL, MEDIUM, and LARGE.

The emotional-gestural-viseme-compiler, and the abstract muscle model for the expressive talking head have already been developed by us in previous research (see Karunaratne and Yan 2001, 2000) for details). We define expressions as aspects displayed via the actor's face and head. We divide them into emotional expressions and gestural expressions or simply gestures. For the head, we define a posture as a static snapshot of emotional and/or gestural expression. Thus, while an expression is dynamic, its snapshot is defined as an expressive pose. In key-frame animation systems, each key-frame can thus be considered an expressive pose. The expressed emotions comprise the six primary emotions of anger, disgust, fear, happiness, sadness, and surprise as mentioned by Ekman and Friesen (1975). Facially expressed emotion is not necessarily the felt or internal emotion. This is due to inhibitions, deceit, play-acting, and pretending by a vactor. The output emotion, besides being a composition of a subset of the six primaries can also be a specified secondary emotion. A list of secondary emotions is stored in a secondary emotion database. Combinations of these with the primaries are handled by the emotional, gestural, viseme compiler described in Karunaratne and Yan (2000).

User Interface

An actor in real life learns the skills of acting from acting or drama school, real-life experiences of oneself and others, as well as watching other performances. Secondly, when an actor performs at rehearsal, his/her performance can be refined or improved by a director. This also leads to learning by an actor. In this paper, it is the first aspect of teaching an actor that we wish to duplicate for synthetic actors. The human teacher in this scenario, who basically "teaches" a virtual actor how to perform in diverse theatrical situations by providing the expert rules is termed the 'drama school teacher' or 'experience provider'.

The term 'user' in this paper is used to refer to both the designer of the system as well as the 'drama school

teacher'. The user interface has two main components: 1. the variable definition and 2. the rule base. With the variable definition section, the designer is able to specify the input as well as output variables relevant to the facial movements of the vactor. These variables can be modified at any time during the design of the expert system. They keep evolving with each new experience or rule taught by the teacher. The user interface is an ASCII script.

1.1 Variable Definitions

In the variable definition file, the fuzzy variable names are followed by the fuzzy set information and membership function definitions. Triangular membership functions are used. However, the system can be extended to use any shape of membership function including trapezoidal, Gaussian, etc. For the triangular membership functions, their end base points as well as peak points are specified. Usually normalized membership functions (maximum membership = 1.0 and minimum membership = 0.0) are used. An abridged form of the ASCII script version of the user interface is given below for reference:

Fuzzy_Inputs: occEmotions, moods, textEmotions // these are described in another database
……..
Fuzzy_outputs = 23
outputAnger #fuzzy_sets = 5 {VLOW LOW MEDIUM HIGH VHIGH}
numerical_range [0.0 1.0]
membership_shape triangular
peak_point_list {0.0 0.25 0.5 0.75 1.0}
left_base_list {0.0 0.0 0.25 0.5 0.75}
right_base_list {0.25 0.5 0.75 1.0 1.0}
……..

As the example shows, most of the input variables are conversational characteristics, OCC emotions, moods, or gestures and emotions extracted from the text. The outputs are output primary emotions, the presence or absence of different gestures, and their parameters. When an input fuzzy variable is specified as a mood or an OCC emotion, the system consults its database of these emotions specified elsewhere to obtain their fuzzy membership function definitions.

1.2 Rule Specification

This section of the user interface is more relevant to the 'drama school teacher' than to the designer. The rules are specified in an ASCII database file. Further rules can be added to the rule-base at any time. Some examples of the 1000 or so initial rules we have provided look as follows in the rule-base.

occLiking = HIGH $\Rightarrow$ outputHappiness = MEDIUM ;

occLiking = MEDIUM $\Rightarrow$ outputHappiness = LOW ;

occHappyFor = LOW & textSadness = LOW $\Rightarrow$ outputSadness = LOW & outputHappiness = LOW ;

Fuzzy System – Motivation and Structure

Our decision to use a fuzzy approach is as follows. One advantage of fuzzy systems or function estimators is the fact that they are model-free estimators. Hence a complete mathematical model of a virtual actor or its interactions with its environment is not required in order to display its actions. Fuzzy systems, much like neural function estimators, learn from samples. Fuzzy systems also require that we partially fill in a linguistic rule matrix. This is simpler than designing and training a neural network. The availability of structured, rather than purely numerical data biased us in the use of a fuzzy rather than a neural system. In addition, neural networks require hundreds of sample data to encode a rule and may require the recycling of these data tens of thousands of times during learning (see Kosko 1992). Also, with fuzzy systems, we can combine the purely numerical approaches of neural networks and mathematical modelling with symbolic, structure-rich AI methods.

We assume our training system to behave as fuzzy associative memories (FAMs). In the case of quantised continuous or discrete domains, we can consider a fuzzy system, S to be a mapping of the form: $S: I^{n1} \text{ x } \ldots\ldots \text{ x } I^{nr} \longrightarrow I^{p1} \text{ x } \ldots\ldots \text{ x } I^{ps}$, where r = maximum number of antecedents in a fuzzy rule, s = maximum number of consequents in a fuzzy rule, n1 to nr are the number of quantisations in each of the antecedent domains, p1 to ps are the number of quantisations in each of the consequent domains.

If a fuzzy system has continuous mappings, we can consider it to map balls of fuzzy sets in $I^{n1 + n2 + \ldots.. + nr}$ to balls of fuzzy sets in $I^{p1 + \ldots.. ps}$, where I^n represents the n-dimensional unit cube [0,1] x [0,1] x ….. [0,1]. This idea is based on the "fuzzy sets as points" concept (see Kosko 1992). Such a FAM has the following structure.

$A_{11}\ A_{21}$ ….. A_{r1} ; $B_{11}\ B_{21}$ …… B_{s1}

$A_{12}\ A_{22}$ …… A_{r2} ; $B_{12}\ B_{22}$ ….. B_{s2}

……

……

$A_{1m}\ A_{2m}$ …… A_{rm} ; $B_{1m}\ B_{2m}$ … B_{sm}

where m = number of rules, r = maximum number of antecedents, s = maximum number of consequents

It is interesting to consider how many rules are needed to completely specify such a system. We assume that the number of input variables to such a system is sufficient to provide the diversity of actor emotions, gestures, poses, moods, attitudes, and situations. If each antecedent variable has g fuzzy sets associated with it the maximum value for $m = g^r$. If r = 10 and g = 5, this amounts to 9, 765, 625 rules. If g = 3 and r = 30, it would be of the order 10^{14}, which is not practical. The need for such a large number of rules to specify the system, is, however, overcome by several factors.

1. The use of generalized rules: Most rules only cover a few antecedent/input dimensions (2 to 4 dimensions)

2. Certain fuzzy sets (e.g. VLOW, NEUTRAL) of antecedents do not appear in rules, because their effects are either insignificant or handled by general rules.

3. The system is an evolving one, so that any unspecified rule can at any stage during vactor training can be added to the system with suggestions from the director. (This is analogous to a novice actor learning more about acting from an expert).

As a more realistic estimate on the upper bound of m (the number of rules specified, after which the vactor is a veteran), consider each rule to contain, on average 3 antecedent variables. There will be ${}^{r}C_{3}$ ways of choosing such a set of antecedents. For the type of fuzzy rule-based system being considered, there can be about 40 antecedent variables in total (of which, on average 3 are used in a rule). With r = 40, ${}^{r}C_{3}$ = 9880 (number of rules with 3 antecedents). For each combination of such antecedents, it is reasonable to assume two fuzzy sets being specified (e.g. MEDIUM, HIGH or HIGH, VERY_HIGH) for each of the 3 antecedents. So, there will be 2^3 = 8 cases for each rule made of such a 3-antecedent combination, giving a total of 8 x 9880 = 79040 rules. However, the 2^3 = 8 combinations of antecedents are variants of the same rule and often designed together, and we can consider the system to be designed with approximately 10,000 (9880) expert rules. Each rule can be assumed to have up to 15 fuzzy outputs. Now since each antecedent variable index can be stored in 6 bits and each antecedent fuzzy set index in 3 bits, 2 bytes (9 bits rounded to nearest byte) can be used to hold each antecedent in a fuzzy rule. Each consequent can also, thus, be represented by 2 bytes. With 3 antecedents on average and up to 15 consequents, the fuzzy system will require (3 + 15) x 2 x 8 x 9880 = 2.8 MB of storage, which is feasible bearing in mind that this will be a near-completely (usefully) specified system.

Although a reasonable upper bound for a “veteran vactor” system is close to 10,000 rules, a developing vactor with much fewer rules (for example 1000) is more common and according to our experiments can give good results. Refinements can always be provided by human directors by adding more specific rules. The advantage is that transferring this acquired knowledge from one actor to another is a simple matter of copying the rule-base which is much simpler than transferring knowledge from a veteran human actor to a novice.

Vactor Training Theory

A vactor with a rule-base of approximately 1000 well chosen rules is a good starting point. Default emotions of a person under a given circumstance can be specified in the knowledge-base by very general (e.g. with a single antecedent) rules.

e.g.: ortonyGuilt = HIGH $\Rightarrow$ outputFear = MEDIUM & outputAnger = MEDIUM,

ortonyGuilt = MEDIUM $\Rightarrow$ outputFear = LOW & outputAnger = LOW and so on.

More specific rules like how a person with MEDIUM guilt with a nasty personality should “shake his head” to denote a negative response, can be specified by more specific rules. Rules with greater than 10 antecedents are uncommon. Even in the case of a human actor, the way to act in very specific situations can only be guessed roughly, even by a good actor. Refinements have to be either specified by a director or obtained through research of real people in similar situations (e.g. How would an autistic person lash out in anger when threatened to be blackmailed in a guilty situation). An expert system is unable to do such “research” for the vactor and so it too, like a human actor “guesses” or uses the best approximation it can to be further refined by a human director.

Consider a situation characterized by the fuzzy variables a, b, c, d, e, f, g and their corresponding numerical integer rate values, which map best to the fuzzy sets A, B, C, D, E, F, G. A rule in the knowledge-base that would completely model this scenario would have the following antecedents.

a = A & b = B & c = C & d = D & e = E & f = F & g = G $\Rightarrow$ …….

However, as the number of situational parameters increases, there is a reduced probability of finding this exactly matching rule. It is more likely to find partially matching rules with antecedents of the form: a = A & b = B $\Rightarrow$ ….. , f = F & g = G $\Rightarrow$ ….. , e = E & f = F $\Rightarrow$ ….. , a = A & f = F & g = G $\Rightarrow$ ….. In this set, for example, variables c and d do not appear at all and some are represented in more than one rule. We propose a fuzzy inference method that uses the “best guess” solution for the output variable with suggestions of refinement to the human director.

The steps of this algorithm are listed below.

1. Find if exact rule match is available. If so, use it in performance.

2. If exact rule match is not available, extract all rules, which include matches with at least one input.

3. From these extracted rules, eliminate all rules that are consumed/further specialized by others in the extracted set.

4. From the resulting set, display the variable names of those variables, which appear in input situation but don't appear in any of the rules in the set. These are used as suggestions to the tutor/director for further refinement.

5. List all output variables which result from all the rules in the new set.

6. For each output, find the numerical value using fuzzy set specifications, fit values and weights, the weights being higher for a higher number of matched rules. (1.0 for n-1 matches, 0.5 for 1 match e.g.).

7. Don't use but display any rules which have a close match of at most (THRESHMISS = 2 e.g.) missing variables and at most (THRESHADD = 2 e.g.) additional variables. This display suggests refinements by trainer/director.

Using the above algorithm and the given example, the following results are obtained. The rule with antecedents: a = A & f = F & g = G specializes the rule with antecedents f = F & g = G , and so the resulting rule set has antecedents: a = A & b = B , e = E & f = F , a = A & f = F & g = G . Now, suppose that these resulting rules (termed r1, r2, and r3), including their outputs/consequents were:

Rule r1: a = A & b = B $\Rightarrow$ $o_1 = O_{12}$ & $o_3 = O_{33}$ & $o_7 = O_{71}$

Rule r2: e = E & f = F $\Rightarrow$ $o_2 = O_{22}$ & $o_7 = O_{72}$

Rule r3: a = A & f = F & g = G $\Rightarrow$ $o_3 = O_{31}$ & $o_5 = O_{51}$ & $o_6 = O_{63}$

Then the output would consist of the variables: o_1, o_2, o_3, o_5, o_6, and o_7. o_1 and o_2 result only from one rule each, and will acquire the numerical value corresponding to the centroid of the fuzzy sets O_{12}, O_{22}, O_{51}, and O_{61} respectively. o_3 and o_7 will have centroid defuzzified values resulting from the fuzzy sets O_{33} and O_{31} with weights w(2) x f_{r1} and w(3) x f_{r3} respectively for o_3, and O_{71} and O_{72} with weights w(2) x f_{r1} and w(2) x f_{r2} respectively for o_7. w(k) is a weight attached to a rule with k antecedents. Since this procedure is resorted to, when the exact rule match is absent, w(k) is designed to give its maximum value of 1.0 for a rule that matches p - 1 or more of the antecedents where p = the total number of parameters in the situation. w(1) is assigned a value 0.5. w increases linearly from k = 1 to k = p - 1. f_{ri} is the antecedent fit value of rule i, resulting from the product of membership values of each of the input parameters in the corresponding antecedent fuzzy sets.

Training Examples

Our training system is quite elaborate and there is only sufficient space to discuss a simple training example. Suppose a movie scene involves a character, who has just lost a competition with his brother with whom his relationship is mildly positive. The situational processor generates the 'Sad' and 'HappyFor' emotions (of the OCC model) as having value 7 (in scale of 1-10) each; i.e. ortonySad = 7, ortonyHappyFor = 4 (happy for his brother). Suppose the textual script requires this vactor to utter the sentence "I am sad that I lost today", what would be the facial expressions and gestures it would use? The textual emotion and gesture extractor specifies that the vactor hang his head down and shake his head somewhat and also specifies a textual emotion of 'sadness'. The rule-base is then searched for the rule with antecedents matching these specific parameters. If an exact rule match is found, it is used in the performance. Otherwise, the closest match is used and performed for further refinement by the director. Some non-contiguous but sequential animated frames of a virtual actor uttering the phrase "I'm sad that I lost today", automatically generated according to the above input situation specifications, is given in figure 2.

Conclusions and Future Work

Our results show that an interactive fuzzy-knowledge-based approach can be used to give multimedia actors emotional and gestural knowledge in virtual human presentations. Applications of our work include movies with synthetic actors, video games, web-based information bots, and virtual newscasters. The fuzzy expert system discussed, essentially comprises a knowledge-base which allows for a higher, user-friendly level of control. Any output performance can be further refined by a human director, by observing the output performance produced, and by considering the input parameters and the desired performance. In fact, due to our system being modular, an animator can refine an output performance at multiple levels, including the OCC emotion and personality level, expressed emotion, viseme and gesture level, abstract muscle level, or even the 3D facial vertex level. As future work, we propose to automate this process one step further and incorporate neural learning, which would tune the fuzzy system based on minimal feedback provided by the human director. Although, this would be a supervised learning system, the feedback would be high-level and the learning automatic (despite being supervised).

References

[1] BADLER, N.I., PALMER, M.S. and BINDIGANAVALE, R. (1999): Animation control for real-time virtual humans, Communications of the ACM, **42** (8), 65 - 73.

[2] DENSLEY, D.J and WILLIS, P.J. (1997): Emotional posturing: a method towards achieving emotional figure animation. Proc. Computer Animation '97, 8 - 14.

[3] EISERT, P. and GIROD, B. (1998): Analyzing facial expressions for virtual conferencing, IEEE Computer Graphics and Applications, 70 - 78.

[4] EKMAN, P. and FRIESEN, W. (1975): Unmasking the face: a guide to recognizing emotions from facial cues, Prentice Hall.

[5] EL-NASR, M.S., IOERGER, T.R., and YEN, J. (1999): PETEEI: A PET with evolving emotional intelligence, Proc. Autonomous Agents '99, 9 - 15.

[6] KALRA, P., MAGNENAT-THALMANN, N., MOCCOZET, L., SANNIER, G., AUBEL, A. and THALMANN, D. (1998): Real-time animation of realistic virtual humans, IEEE Computer Graphics and Applications, 42 - 56.

[7] KARUNARATNE, S.K., and YAN, H. (2001): A new efficient expression generation and automatic cloning method for multimedia actors, Circuits, Systems and Signal Processing: Special Issue on Multimedia Communication Services.

[8] KARUNARATNE, S.K., and YAN, H. (1999): Generating individualized, texture mapped 3D head models, VIP '99: Proc. Pan-Sydney Area Workshop on Visual Information Processing, 16 - 19.

[9] KARUNARATNE, S.K., and YAN, H. (2000): Techniques for combining emotions, visual speech and gestures of multimedia talking faces, Proc. First IEEE Pacific-Rim Conference on Multimedia, 398 – 401.

[10] KOSKO, B. (1992): Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence, Prentice Hall.

[11] KURLANDER, D., SKELLY, T., and SALESIN, D. (1996): Comic chat, Proc. SIGGRAPH ’96, 225 – 236.

[12] LOEHLIN, J.C. (1968): Computer models of personality, Random House.

[13] MAIOCCHI, R., and PERNICI, B. (1990): Directing an animated scene with autonomous actors, The Visual Computer, Vol. 6, 359 - 371.

[14] MANSKE, K., and MUHLHAUSER, M. (1997): An open architecture for comic actor animation, Proc. ACM Multimedia '97, 251 - 259.

[15] ORTONY, A., CLORE, G.L., and COLLINS, A. (1988): The cognitive structure of emotions. Cambridge University Press.

[16] ROUSSEAU D. and HAYES-ROTH, B. (1998): A social-psychological model for synthetic actors, Proc. Autonomous Agents '98, 165 - 172.

[17] SATO, J. and MIYASATO, T. (1997): Autonomous behaviour control of virtual actors based on the AIR model, Proc. Computer Animation '97, 113 - 118.

[18] STURMAN, D.J. (1998): Computer puppetry, IEEE Computer Graphics and Applications, 38 - 45.

[19] TERZOPOULOS, D. (1999): Artificial life for computer graphics, Communications of the ACM, **42** (8), 33 - 42.

[20] THORISSON, K.R. (1997): Layered modular action control for communicative humanoids, Proc. Animation '97, 134 - 143.

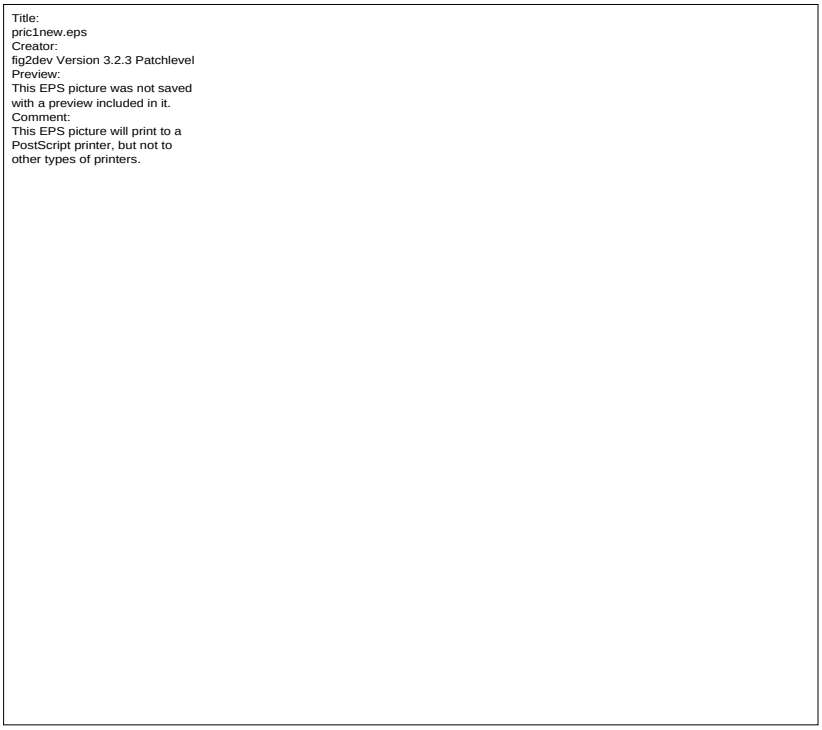

Figure 1: Complete Vactor framework

Figure 2: Animation frames from expressive utterance sequence based on input script