PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 15/05

An Expressive Talking Head Narrator for an Interactive Storytelling System

Paula S. L. Rodrigues César T. Pozzer
Bruno Feijó Luiz Velho
Angelo E. M. Ciarlini Antonio L. Furtado

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO – BRASIL

Monografias em Ciência da Computação, No. 15/05 ISSN 0103-9741
Editor: Prof. Carlos José Pereira de Lucena May 2005

An Expressive Talking Head Narrator for an Interactive Storytelling System*

Paula S. L. Rodrigues, César T. Pozzer, Bruno Feijó,
Luiz Velho[1], Angelo E. M. Ciarlini[2], Antonio L. Furtado

[1] Instituto de Matemática Pura e Aplicada

[2] UniRio – Departamento de Informática Aplicada

paula@icad.puc-rio.br, pozzer@icad.puc-rio.br, bruno@inf.puc-rio.br, lvelho@impa.br,
angelo.ciarlini@uniriotec.br, furtado@inf.puc-rio.br

Abstract: This paper addresses the problem of telling – or, in a broader sense, presenting and exhibiting – stories based on sequences of events previously computed in an interactive fashion. A scheme is proposed which, in addition to the participating 3D virtual actors, makes use of an expressive talking head system, represented by an avatar with facial expressions, who acts as an animated Narrator of the story to the attending spectators. The implemented system integrates modules for managing plot generation, user interaction, visualization and narration. Scene rendering as well as the Narrator's speech and facial expressions are real-time synchronized during plot dramatization. The resulting environment may be used in the development of applications involving childlike worlds, interactive TV, games, and distance learning.

Keywords: Real-time Expressive Facial Animation, Storytelling, Speech-and-scene Synchronization, Real-time Interaction and Rendering, Computer Graphics.

Resumo: Este artigo aborda o problema de narrar – ou, em sentido mais amplo, apresentar e exibir – estórias baseadas em seqüências de eventos previamente computados de forma interativa. Um esquema é proposto, no qual, além da participação de atores virtuais 3D, é utilizado um sistema de cabeça falante expressiva, representada por um avatar com expressões faciais, que age como narrador animado para os expectadores da estória sendo apresentada. O sistema implementado integra módulos para administrar a geração de enredos, interação com os usuários, visualização e narração. A renderização das cenas, bem como a fala e as expressões faciais do narrador, são sincronizadas em tempo real durante a dramatização do enredo. O ambiente resultante pode ser usado para o desenvolvimento de aplicações envolvendo mundos infantis, TV interativa, jogos e ensino à distância.

Palavras-chave: Animação Facial em Tempo Real, Narração de Estórias, Sincronização de Fala e Cena, Interação e Renderização em Tempo Real, Computação Gráfica.

[*] This work has been partly sponsored by the Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil

In charge for publications

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3114-1516 Fax: +55 21 3114-1530
E-mail: bib-di@inf.puc-rio.br

1. Introduction

Traditional storytelling is an interactive performance art form, wherein the teller adjusts the vocalization, wording, physical movements, gestures, and pace of the story to better meet the needs of the responsive audience. Storytelling in its new digital and interactive form combines participation, as occurs in computer games, with automatic story generation and narration. The user interaction is strongly related with the paradigm adopted to create the story. In a *character-based approach* [Cavazza02] [Mateas97] [Young00], also called emergent narrative, the storyline usually results from the real-time interaction between virtual autonomous agents and the user. In a *plot-based approach* [Spierling02] [Sgouros99] [Ciarlini05], plot generation (possibly with user participation) and visualization are treated separately, and well-defined stages of authoring, planning, and user interference are present. Plot dramatization (the "telling" component in storytelling) is also strongly related with the paradigm adopted for plot generation. Agents (virtual actors) perform events that may be defined in real time, as happens in character-based approaches, or pre-computed during plot simulation.

The presence of a synthetic *Narrator* is at the very heart of the storytelling experience. However, the existing literature lacks works on this subject. Research works on digital actors [Thalmann95], graphical multimodal user interface [Corradini05] [Cassell99] [Massaro03], and facial animation [Parke96] do not address the question of synthetic Narrators in interactive storytelling. This paper presents an innovative system that combines facial animation with plot generation and visualization of interactive stories. The virtual Narrator proposed here is capable of emotional expressions. Furthermore it can assist the author during plot generation.

Our implemented environment integrates a talking head system, called ETHs (Expressive talking Heads), with a plot-based storytelling system LOGTELL [Ciarlini05]. In the environment, the talking head featured by ETHs works as a *story Narrator*, receiving markup-texts containing story fragments and producing, on the fly, a facial animation that gives voice to this input text. The speech is automatically generated using text-to-speech (TtS) mechanisms. The Narrator facial animation controls, besides lip synchronization, the varying emotional expressions. These are obtained through the text markup parameters. Synchronized with the talking head narration output, a 3D module in the environment renders the story scenes.

The environment has the capability to build, present and narrate different stories from different *genres*. To provide an example, the present work is based on a Swords and Dragons context, where heroes, victims and villains interact in a 3D scenario occupied by castles and churches. The Narrator's primary duty is to tell, from a third person perspective and with the appropriate emotion, each scene of the story. However, the environment has the flexibility to allow the Narrator to assume the role of any character in the story (in first-person discourse); it has also the option to introduce more than one Narrator with a different physical appearance (man, woman, or child).

One of the main contributions of this article comes from the development of this environment for generating and visualizing stories involving virtual actors, as seen from the standpoint of an emotional Narrator. The benefits of including one such animated emotive Narrator in the interactive storytelling system seem evident. It should look particularly pleasant to a children's public, to which the presence of an expressive character is an important incentive in teaching and entertainment activities. Other fields of useful application are game development, interactive TV, and distance learning.

The presentation covers the methodology used for the environment, describes how the two independent systems were joined, and mentions relevant aspects of the implementation. The paper is organized as follows. Section 2 shows the LOGTELL architecture and mechanisms for building plots with user interaction. Section 3 briefly describes the ETHs system, and overviews the main modules of the facial animation tool. Section 4 initially points out aspects that could be improved in the story generation and dramatization through the use of a talking head Narrator; next, it gives details of the integration of the two systems, including synchronization and data exchange schemes. Section 5 discusses related work, and section 6 contains the conclusions.

2. StoryTelling Module: The Interactive LOGTELL System

LOGTELL [Ciarlini05] is an integrated system, articulating a number of diverse modules to provide support for the generation and tridimensional visualization of interactive stories. The general architecture (Figure 1) can be seen as a pipeline, along which data is transformed from morphological functions into real-time 3D animations, to be dramatized by virtual actors handled by a graphical engine. The control of the dataflow is under the responsibility of the **Plot Manager**. This module encapsulates all user interfaces through which the user conducts all the processes, including both plot generation and dramatization. The **IPG** (**Interactive Plot Generator**) module [Ciarlini02], in turn, is specifically responsible for plot generation. Each character in the simulated story is represented by a virtual actor. The **Drama Manager** controls all virtual actors, according to the events supplied by the Plot Manager. The dramatization comprises selecting and sending to the Drama Manager sets of ordered events, each one containing parameters referring to characters, actions and places.

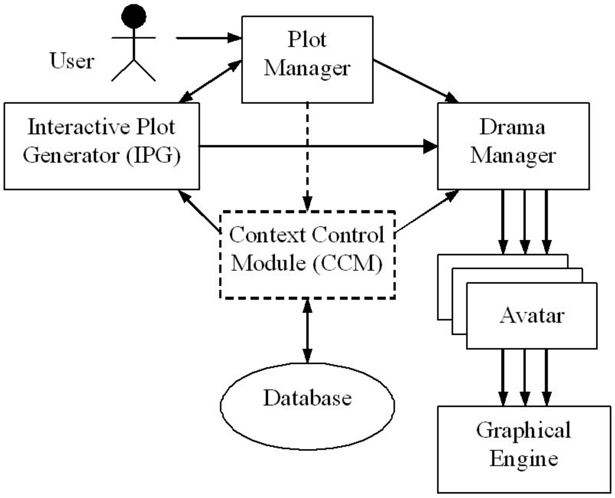

User
Plot Manager
Interactive Plot Generator (IPG)
Drama Manager
Context Control Module (CCM)
Avatar
Database
Graphical Engine

Figure 1: LOGTELL's architecture.

The IPG module adopts a plot-based paradigm for the generation of the stories. The initial context is specified in an authoring phase in a Prolog Database, and includes the specification of the characters, scenarios, and logical rules used to infer goals pursued by each character. It also provides a repertoire of *operations*, designed to bring about the events that are expected in plots conforming to the intended genre. In our work, we took as an example a simplified Swords and Dragons genre. As a consequence of this choice, characters are heroes, villains and victims, and common events in the plots include, for example, the abduction of the victim, fights between heroes and villains, the rescue of the victim, and marriage.

Representing, as said, typical events of the genre, these operations change the state of the world. Through their simulated execution, with appropriate parameters, the story moves forward. Executing an operation may, for example, affect characters' attributes like strength, location, and level of affection for some other character, among others. IPG uses a scheme that alternates between goal inference, planning, and simulation phases.

The use of parameterized operations, defined by their *pre-conditions* and *post-conditions* (i.e. effects), allows the use of planning algorithms in the simulation process, whereby new event-producing operations, with satisfied pre-conditions, are inserted into the plot. Any generated plot is thus made up of a partially ordered sequence of logically chained events (executed operations). As a consequence, events do not occur randomly, but in response to *goals* that the characters aim to achieve.

Plot generation and dramatization are two separate processes, in contrast to pure character-based approaches, where user interaction affects plot structuring at real-time. The way the stories are structured also dictates how the stories are being told. We use a third-person viewpoint, which means that the user acts as spectator of a pre-computed story, told by means of animated virtual actors, and further described by subtitles.

During the dramatization, each event is mapped, by means of the Drama Manager, onto a set of predefined behaviors assigned tho each virtual actor. In our implementation, we employ *reactive* agents to represent the characters of the plot. It is not expected that agents behave deliberately (as in a character-based approach), since the general structure of the plot has already been defined during the simulation process. Further details on the dramatization are presented in section 4, which deals with strategies for better transmitting the essence of the scenes to the user.

The coordination of the plot generation and visualization is in charge of the Plot Manager, which is the core of the system (Figure 2). The simulated plot of the story is displayed by a set of colored icons representing events and goals. User interaction is always indirect, at the level of the events or characters' goals depicted by these icons. By moving the cursor over the respective icons, events and goals can be inserted, removed, and ordered by the user, within the limits of the logical temporal constraints imposed during the simulation process. Thus, in the course of the simulation, the user can intervene either passively, just letting the partially-generated plots that seem interesting to be continued, sometimes asking for alternatives also generated automatically, or, in a more active way, trying to force the occurrence of events and situations. These are checked by the system: if they violate any constraint, it tries to introduce further changes to accommodate the inclusion, which is rejected if no such adaptation is possible.

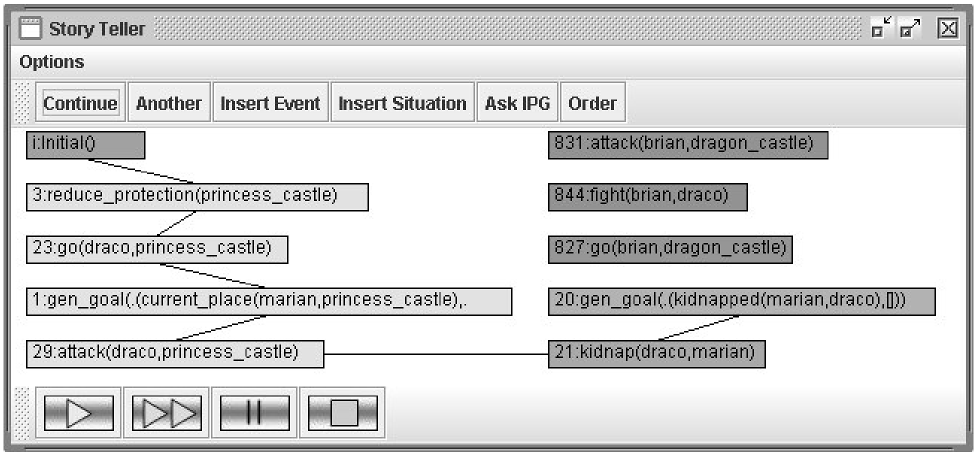

Story Teller
Options
Continue
Another
Insert Event
Insert Situation
Ask IPG
Order
i:Initial()
3:reduce_protection(princess_castle)
23:go(draco,princess_castle)
1:gen_goal(.(current_place(marian,princess_castle),.
29:attack(draco,princess_castle)
831:attack(brian,dragon_castle)
844:fight(brian,draco)
827:go(brian,dragon_castle)
20:gen_goal(.(kidnapped(marian,draco),[]))
21:kidnap(draco,marian)

Figure 2: Interface of the Plot Manager.

3. Facial Animation Module: The Emotional ETHs System

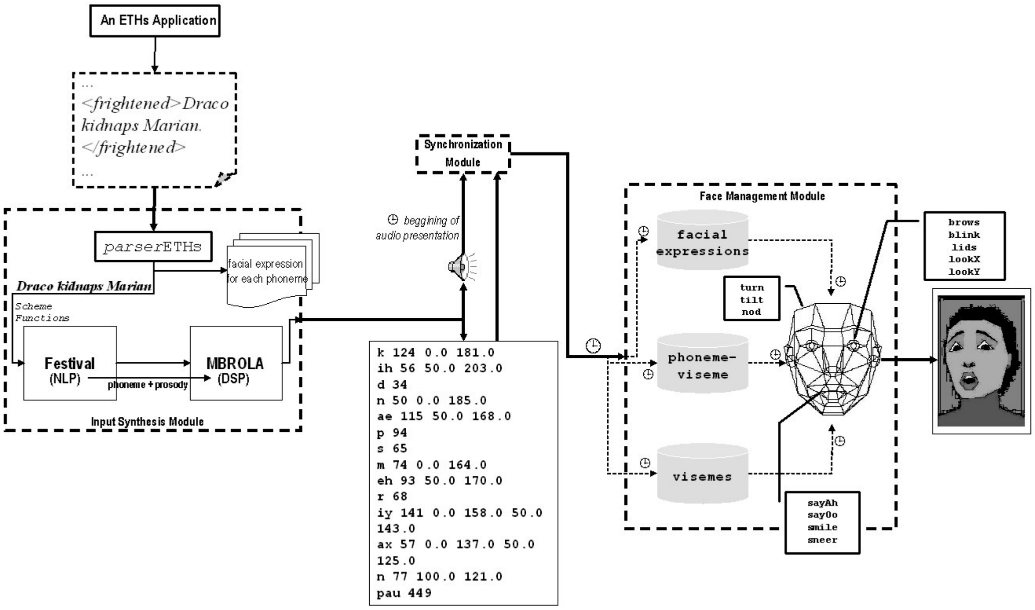
The **Expressive Talking Heads** (**ETHs**) is a facial animation system. When it receives an input text with some special markups, it generates a real time character facial animation, prepared to speak the input text with synchronization of facial and lip movements, and expressing the appropriate emotion. The text markups can set, for example, speech idiom accents (such as American or British English), voice gender (male or female), character emotion (e.g. natural, frightened, annoyed, happy), eyes and head positioning, and text anchors.

Initially, ETHs was developed as a stand-alone facial animation system, but it was soon modified to provide a framework for any application in which a talking head unit might be desirable. To facilitate the use of ETHs, its services were made available through a single façade that hides the internal organization.

Some applications have already been developed through the ETHs framework, like a 3D chat where the sending user types text messages, and the receiving user listens to them from the facial animation system. Another important application is the integration with a hypermedia presentation system [Rodrigues04]. In this application, virtual characters can be associated with other media objects, such as slide presentations, videos, subtitles, and audio tracks.

ETHs[1] has three major modules: the Input Synthesis, the Face Management and the Synchronization modules. Figure 3 gives an overview of the ETHs' modular structure.

[1] The ETHs system was developed using the Java Programming Language *http://java.sun.com*.



```
k 124 0.0 181.0
ih 56 50.0 203.0
d 34
n 50 0.0 185.0
ae 115 50.0 168.0
p 94
s 65
m 74 0.0 164.0
eh 93 50.0 170.0
r 68
iy 141 0.0 158.0 50.0
143.0
ax 57 0.0 137.0 50.0
125.0
n 77 100.0 121.0
pau 449
```

Figure 3: An overview of the ETHs architecture and modules. This picture shows the Input Synthesis module, the Synchronization module and the Face Management module.

The **Input Synthesis** module is responsible for (a) capturing and treating the input text, which can have some markup elements conveying information about the character's emotion, the voice's gender (currently, male and female adults) and the speech's language (currently, American English and British English) provided by the user, and (b) for generating as output a data structure containing the fundamental units (phonemes, duration, emotion, etc.) needed to generate the facial animation corresponding to the input text. The module interprets the markup text by means of a parser, separating speech and control information.

The ETHs parser is responsible for separating the speech content itself (text without markups) from the speech and animation markups. The parser interacts with the tool synthesizer to build the facial animation and lip-sync data structures. The parser sends each fragment of marked text to a double synthesizer (Festival [Black04] and MBROLA [Dutoit98] as shown in Figure 3), which first creates the speech phonetic description (list of phoneme entries, each one containing the phoneme label, duration and pitch). In this blend of two synthesizers, Festival works as the Natural Language Processing unit (NLP) and MBROLA works as the Digital Signal unit (DSP). The advantage of putting them together is the acquisition of a TtS synthesizer that offers a multilingual platform (MBROLA's contribution). This flexibility is important to Expressive Talking Heads, because it allows the user to select language and gender as system parameters, thereby enhancing the system's expressiveness. Finally, the Festival-MBROLA subsystem is used in server mode, in which the ETHs establishes a TCP connection with the TtS synthesizer.

Working on the phonetic structure, the parser can identify the phonemes corresponding to the beginning and end of a fragment, and can then assign the beginning and end phonemes for each emotion, eyes positioning and head positioning. All these results are stored in the animation data structure. After handling all fragments of marked text, the parser concatenates the several phonetic structures and, together with the speech metadata (idiom, gender and emotion), sends the information to the synthesizer to have the digitized speech audio generated.

The second ETHs module, the **Face Management** module, links it to another external subsystem, named Responsive Face [Perlin97]. Actually, the ETHs face was inherited from this subsystem, which defines a three-dimensional polygonal mesh, as illustrated in Figure 4. The ETHs character adopts a simple model, with minimal controls, but supports nevertheless a fair degree of expressiveness.

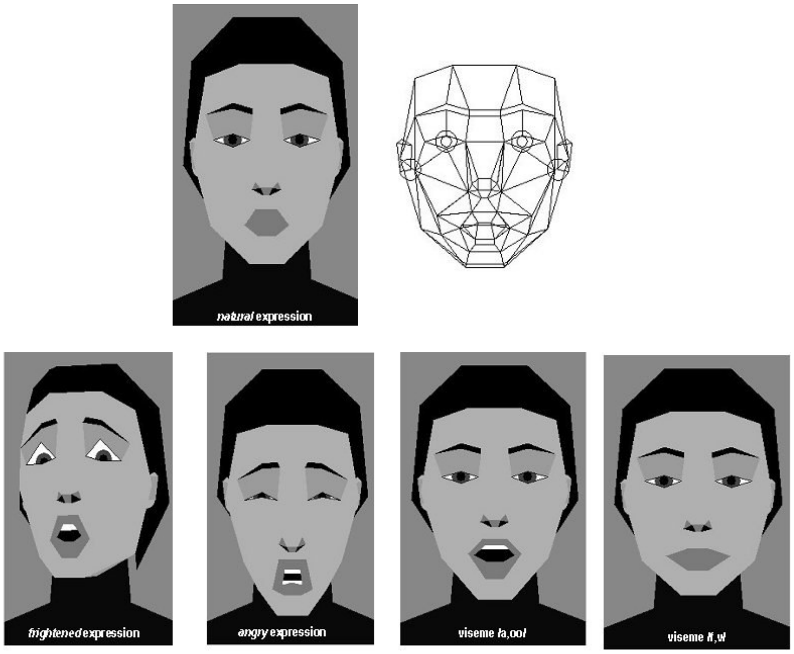


Figure 4: The ETHs neutral emotion and its mesh, some facial expressions and visemes.

The face is animated by the application of relax and contract commands over the mesh edges (face muscles). ETHs improves the Responsive Face features, adding speech and the concept of visemes. *Viseme* is the name given to a mouth configuration for a specific phoneme. In the system's initialization, the module provides a database of 16 visemes and 8 facial expressions, as well as a special database containing the phoneme-viseme mapping (see examples in Figure 4). Each database entry stores the values for contracting/relaxing the face corresponding muscles commanding the Responsive Face. During the animation process, the module receives requests from the Synchronization module (see below) to supply information from such databases. Requests to activate facial muscles are also received.

The third and last ETHs module is the **Synchronization** module, which is responsible for the fine synchronization between speech and facial muscle movements.

Parallel to the audio file reproduction, the synchronizer polls the audio controller to check the effective playing instant. Using the phoneme durations, the Synchronizer discovers the current phoneme and the current character emotion. Then it gets the associated viseme and facial expression muscle contracting/relaxing values, and asks the Face Manager to apply these values on the ETHs face. In reality, instead of working just with the current phoneme, the Synchronizer uses *diphones* (two consecutive phonemes interpolated by their durations), since the lip positions for the same phoneme generally changes according to the phoneme context (speech co-articulation aspect). Finally, the Synchronization module also includes components to control the movements of the head and of the eyes, in order to produce a more natural output.

4. The Virtual Narrator in Action

Thanks to the integration of ETHs to LOGTELL, some steps in the plot generation and visualization processes are made more intuitive and accessible. The new interface should look friendlier to users guiding the compostion of a story plot, or merely attending as spectators, and even to authors working on the specification and revision of the story genre in use.

The ETHs system, by means of live audio synchronized with a 3D emotive virtual Narrator, provides an additional medium to communicate information. During plot generation, it complements what is presented, perhaps too concisely, in dialog text boxes. During visualization, the virtual Narrator not only reads aloud the subtitles narrating the current action, but also explains what is happening and reveals what is “behind the scene”. This is possible because all metadata, i.e. the internal definition of the genre, especially the pre-conditions and post-conditions of operations and the goal-inference rules, stay available at running time. The result is a more complete comprehension of the story by the spectators.

Both for plot generation and for visualization, dialogs and speeches can be either produced in real-time, or pre-synthesized and later inserted in the appropriate context. The real-time strategy favors the richness of details, obtainable, for example, when parameter variables are instantiated (e.g. in response to queries posed by the user), and at the time of occurrence of all kinds of unanticipated situations. Pre-synthesized speech can be prepared at leisure, with more attention to information contents, communicative efficacy and stylistic quality, and has the advantage to reduce CPU processing time.

4.1 Assistance during Plot Generation

As mentioned before, the IPG module of LOGTELL offers to users/authors both weak and strong ways to interact with the planning algorithms, so as to guide plot generation at the broad level of characters' goals, and even gives the option, at a lower level of detail, to include specific events. To help a beginner to become familiar with the genre and with the use of the system, the Plot Manager module offers a facility for querying IPG about the state of any element of the narrative at an indicated time. This feature helps the user to find out, for instance, why an operation or goal is not being allowed,

and is most useful for author's intent on revising and tuning the story requirements.

While the system is operating in this query mode, the virtual Narrator is called to act as a live assistant, to express vocally the answer that is also being displayed in compact and precise – but not so intuitive – clausal format. Currently, since the queries themselves and other forms of user input are composed manually, with the help of buttons and menus, it must be noted that the vocal communication is unidirectional. The virtual Narrator is also in charge of voicing pre-defined speech lines, signaling system-events such as "nothing to do" (no goals to be pursued), or "no solution" (user-supplied situation or event rejected).

4.2 Graphical and Narrator Output

The graphical engine supports real-time rendering of the 3D elements. Characters in a generated plot are regarded as actors for the dramatization. Each actor is implemented as a materialized reactive 3D agent. The graphical engine does not have to perform any intelligent processing. It is merely responsible for rendering, at each frame, from a third-person perspective, the scene and the current actors' aspect and movements, resulting from real-time interactions with the scene and, occasionally, with other actors. In doing that, it follows the ordered sequence of events generated at preceding stages of simulation. When accompanying dramatization, the virtual Narrator is responsible for synchronously narrating the ongoing actions being performed by the actors.

Since the rendering duration of each action is well known, and is not less than 10 seconds, the virtual Narrator has enough time to describe the events being dramatized, and to add relevant contextual information. This extra material can be readily extracted by IPG, which has access to properties of characters and places at each state reached by the plot simulation, as also to the clausal definition of the genre (which stays available online, as said before).

As an example, consider the event that portrays the abduction of the victim by the villain. In our story model, the victim is called Marian, the villain is a dragon called Draco, and the heroes are the knights Brian and Hoel. A pre-condition for this event is the fragility of the victim and, as post-condition, the kidnapped princess is confined to the villain's castle. What ultimately motivates the event is the villain's goal is to kidnap unprotected victims. Since one of the heroes' goals is to free damsels in distress, usually in order to marry them as a reward, the kidnapped victim's situation arouses in the hero the desire (goal) of rescuing her. The simulated execution of a plan to achieve this goal leads, in turn, to a new state wherein other goals are inferred, thus causing the story to move forward.

As a consequence of its knowledge, through IPG, of this logical chaining of events, determined by specified causes, effects and goals, the virtual Narrator finds how to interpret the various events in detail. Currently, we make use of simple templates (Prolog lists intercalating variables and fixed character strings) to translate the formal terms into the natural language sentences used for narration.

In our example, these are some of the subtitles automatically generated by the system, to be recited as the respective scene is being visualized, by the virtual Narrator:

- The protection of the Princess's castle is reduced.
- Draco kidnaps Marian.
- Brian kills Draco.
- Brian frees Marian.
- Brian and Marian get married.

A side-effect of the event “Brian frees Marian” is that the level of affection of the princess for the hero is raised to 100. Showing a clause like `affection('Marian', 'Brian',100)` to the user, no matter if visually or vocally, would be equally tasteless. The application of a simple conditional template, in Prolog notation, such as:

```
template(affection(CH1, CH2, Level),[CH1, feels now', Aff, ' for ' ,CH2]) :-
    (Level =    0, !, Aff = 'absolutely nothing';
     Level =<  50, !, Aff = 'a moderate liking';
     Level =<  99, !, Aff = 'some tenderness';
     Level =  100,    Aff = 'a perfect love').
```

induces the virtual Narrator to comment, with a happy smile: *Marian feels now a perfect love for Brian.*

ETHs is responsible for dialog synthesis, in real time, and also for handing over the speech audio and the phoneme sequences to be spoken in a synchronized way. For each phoneme, there is an associated viseme, and to visualize the viseme the Narrator facial muscles are moved, as mentioned in Section 3.

Each user's operation to build the story is hitched with an emotion. This emotional information is used by the virtual Narrator in the exact instant when it tells the story. On the other hand, it knows that, for each word, sentence or paragraph, there is a facial expression. Internally, operations must somehow be mapped into emotions, for example as indicated in Table 1[2].

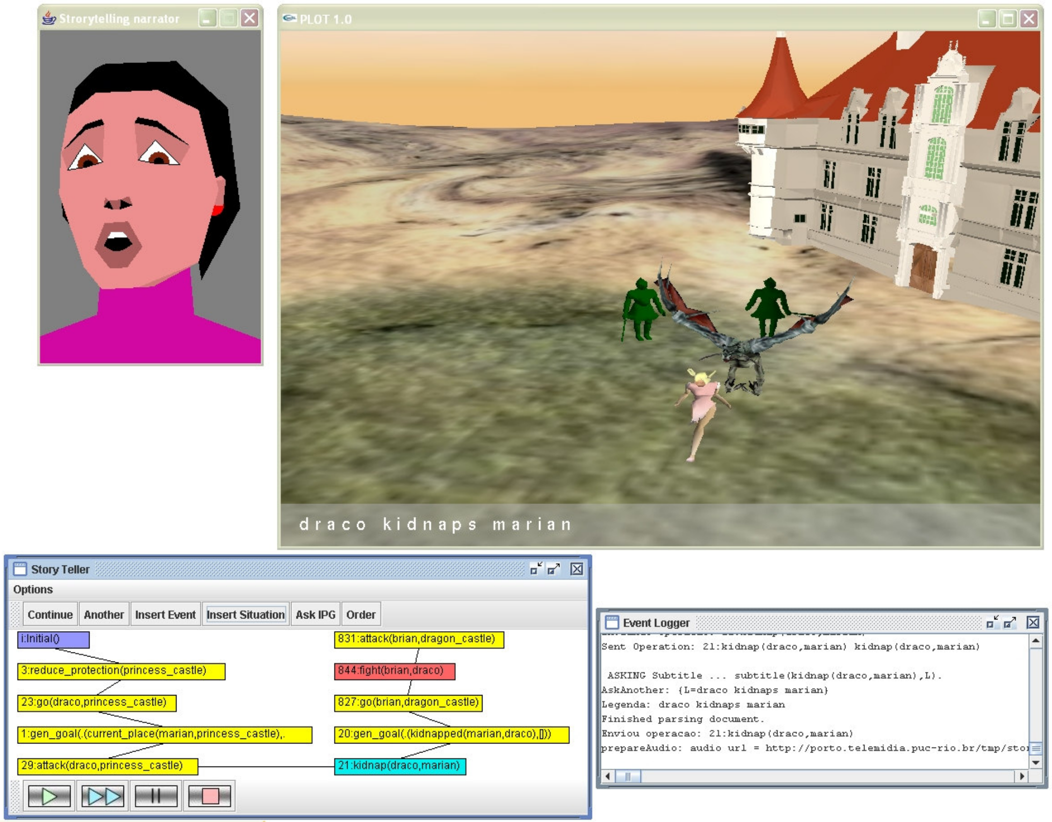
Table 1: Operation-and-Emotion Mapping.

Operation	Emotion
GO(CH, PL)	Natural
Reduce_protection(VIC,PL)	Annoyed
Kidnap(VIL,VIC)	Frightned
Attack(CH,PL)	Surprised
Fight(CH1,CH2)	Angry
Kill(CH1,CH2)	Angry
Free(HERO,VIC)	Happy
Marry(CH1,CH2)	Happy

[2] In Table 1, CH is a character, PL is a place, VIC is a victim, VIL is a villain, and HERO is a hero.

In addition to speech, it is also possible to incorporate a background music line. The music can play throughout different narration phases, reflecting the varying emotions associated with the events.

Figure 5 illustrates the visual aspect of the environment.


Strorytelling narrator
PLOT 1.0
d r a c o   k i d n a p s   m a r i a n
Story Teller
Options
Continue
Another
Insert Event
Insert Situation
Ask IPG
Order
i:Initial()
3:reduce_protection(princess_castle)
23:go(draco,princess_castle)
1:gen_goal(.(current_place(marian,princess_castle),.
29:attack(draco,princess_castle)
831:attack(brian,dragon_castle)
844:fight(brian,draco)
827:go(brian,dragon_castle)
20:gen_goal(.(kidnapped(marian,draco),[]))
21:kidnap(draco,marian)
Event Logger

```
Sent Operation: 21:kidnap(draco,marian) kidnap(draco,marian)

 ASKING Subtitle ... subtitle(kidnap(draco,marian),L).
AskAnother: {L=draco kidnaps marian}
Legenda: draco kidnaps marian
Finished parsing document.
Enviou operacao: 21:kidnap(draco,marian)
prepareAudio: audio url = http://porto.telemidia.puc-rio.br/tmp/sto
```

Figure 5: A snapshot of the environment.

4.3 Implementation Issues

When a user requests plot visualization, each event is processed separately from the others, according to the connected sequence drawn by the user in the Plot Manager interface. The dramatization process involves the delivery of all specific data associated with the current event to both the Drama Manager and the Narrator, as illustrated in Figure 6. For doing that, for each individual event, the Drama Manager initially consults the IPG module to obtain the information required for describing the event, including subtitles and dialogs. The Drama Manager determines when an event dramatization has been finalized, and, in this case, requests a new one from the Plot Manager. All modules are implemented in Java[3], except the Drama Manager, which is implemented in C++/OpenGL [Woo00].

[3] Java language- URL: *http://java.sun.org*.

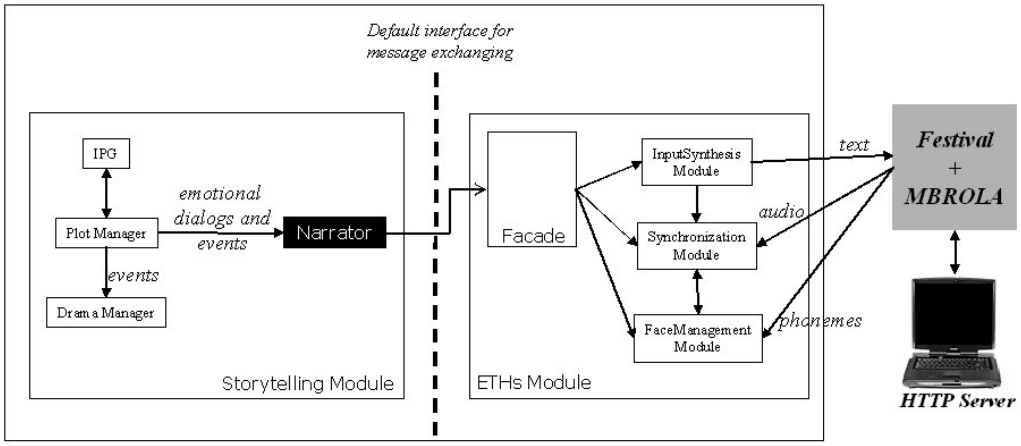


Figure 6: Overview of the whole environment developed to generate and rendering interactive stories using a Narrator.

The Drama Manager receives the event itself, and its corresponding subtitle. Its job is to transform the event into graphical animation. It coordinates the virtual actors in order to ensure accurate representation. The Narrator receives the event and dialogs describing the scene. One must recall that, before starting the telling activity, it is necessary to create the corresponding speech, sending the text to the synthesizer. The synthesized text may be a simple subtitle, a text with emotion, or a concatenation of such texts corresponding to a sequence of events (the associated emotions being obtained by consulting the operation-emotion mapping table).

The Drama Manager and the Narrator must be initialized before plot visualization. The initialization of the Drama Manager comprises the tasks of constructing the scenario and loading the virtual actors. These remain in a standing state until some specific action is delegated. The initialization of the Narrator includes the initialization of the façade, which manages the other main modules of ETHs (Input Synthesis, Face Management and Synchronization modules). During the Narrator initialization it is possible to define certain features of the virtual agent, such as the server synthesized machine, the face gender (masculine or feminine), the output voice quality and the initial facial expression.

Since LOGTELL and ETHs are independent modules, the user may select whether he would like to see the story with 3D scenes and a narration, or only with 3D scenes, or only with the Narrator. The default option is the first one, with both visual and speech narrations. With the purely visual option, the Narrator is simply not created. The last choice, only speech narration, is a little more complex. In this case, the Narrator must assume the Drama Manager role. After that, the environment functions normally.

5. Related Work

We were unable to find, in academic literature, any work encompassing the entire combination of features and aims presented in this paper. Therefore the comparison with related works had to be separated into two halves.

Our first survey considered *storytelling systems*. The analysis compares how the stories are built, the degree of interaction, plot limitations, answer time, as well as the style of presentation. A second series of comparisons was done for *talking head facial animation systems*, broaching interactive aspects, real time, lip synchronization and synchronization between speech and facial expressions for given characters.

5.1 Work related with LOGTELL

The work of Spierling [Spierling02] adopts a modular system approach for interactive storytelling. In that work, a prototype of a virtual character was developed, which stays in a digital newsstand, interacting with the user through dialogs. This system appears to have originated from a fairy-tales background, but its context was adapted for business dialog, giving information about products.

It models an agent's conversational habits and social behavior. An author is supposed to specify rules to direct the dialog, which should derive from the character's personality. The system's modularity allows it to manage complexity at selected levels for authoring and user interaction. Finally, the characters were modeled using the Java3D language[4], and morphing techniques were utilized to animate the face and the hands.

The scope of the work looks somewhat restricted when compared with the environment proposed in the present article. It does not offer a visualization module for 3D scenes, the avatar acting is tailored to the business world and product selling, and, more importantly, the user interaction is limited and coordinated by the Narrator. With that, the user has no freedom to modify the story direction. This kind of limitation is also a consequence of the morphing techniques used, since possible Narrator behavior representations have to be previously defined and stored. On the other hand, its modular design opens the possibility to integrate a graphical view module, which would allow the system to become more complete and better adapted to the proposed environment.

5.2 Work related with ETHs

Video Rewrite [Bregler97] is a system that uses an existing footage sequence to create a new video automatically with the same content but with a new sound track. Direct applications for systems of this kind are film dubbing, teleconferencing, and special effects. An example of the latter can be found in the Forrest Gump movie.

Basically, a new sequence video creation comes in two stages: an analysis stage to build a training database, and a synthesis stage to synthesize the new video sequence. The analysis stage is responsible for constructing, from the original footage, a database with examples of video frames (also called training database). In this stage, the system can learn how a specific person's face changes during the speech. Still in the analysis stage, Video Rewrite segments the original soundtrack into phonemes and uses these

[4] Java3D language: *http://java.sun.com/products/java-media/3D/*.

phonemes to label all training database images. After the analysis, starts the synthesis stage, wherein the new audio is segmented and the obtained phonemes are used to select a video sequence with triphones (three sequential phonemes), which are nearer to the new soundtrack. With the labels defined at the analysis stage, the new mouth images are deformable into the background face using morphing techniques.

The Video Rewrite system builds a new sequence video to be a talking head Narrator. A real-time Narrator is not feasible in Video Rewrite, because the video sequence and the sound track are preprocessed (see the discussion about real-time versus pre-synthesized approaches at the head of Section 4). However, the offline approach is possible, since the video could be prepared beforehand and played synchronized with the 3D scenes.

A second related work is a group of research projects, in the facial animation area, using the *MPEG-4* pattern [Pandzic02]. The MPEG-4 pattern offers a framework with a number of facilities for facial animation implementation, particularly strong in allowing talking head applications development. The pattern makes possible the use of synthesized and recorded audio, as well as recorded and synthesized video. The reference [Pandzic02] also describes the development of a talking head web tool, using the MPEG-4 as the facial animation basis.

The MPEG-4 facial animation pattern allows the development of a real-time and an offline talking head Narrator. With the pattern it is possible to work with pre-synthesized audio and video, and also to synthesize both speech and facial animation in real-time, synchronized with the 3D scenes. Nevertheless, the MPEG-4 facial animation framework suffers from a limitation. The system loses portability and platform independence, because the framework requires an encoder and a decoder to propagate (send and receive) MPEG-4 facial animation streams.

6. Conclusions

We described in this paper the development of an interactive environment for story generation and visualization, using an expressive avatar to augment user immersion and output reality. Our work aims at contributing both to the storytelling and the facial animation areas. Besides its usage in entertainment, the developed system can be adapted and applied in areas such as training, news presentation, distance learning and e-commerce. The system's flexibility for incorporating different kinds of modules increases its ability to cope with an ample variety of applications.

The environment presented in this article has two main components: a plot-based storytelling system, called LOGTELL, and an expressive talking head system, called ETHs. In the environment, the facial avatar is used as a story Narrator, integrated with a 3D rendering module, with voice output generated on the fly. Moreover, the avatar exhibits emotional facial expressions in order to enhance the user perception during storytelling.

Besides working in this capacity, expressive avatars could be explored in other ways in the same environment. For example, they might be assigned roles as characters of the stories. And multiple instances could be generated, so as to have a number of expressive avatars interacting with each other, with an automatically generated playing performance, which is one line under development. Another future research work is the use of emotional and reacting agents to model the characters' behaviors. An intriguing possibility is to have the avatars (working as Narrators or characters) interacting vocally with user audiences, allowing the users to conduct the story through their intermediacy. As an even more ambitious objective for future work, we intend to investigate how this kind of machinery could be upgraded to the point of providing useful help in the story authoring task.

Finally, the general goal of our project is the integration of the storytelling environment with an interactive multimedia authoring and presentation system. The idea is to model the generated stories as multimedia/hypermedia documents, where choice points in the story are represented as hypermedia links. This correspondence should permit the inheritance of hypermedia conceptual models and languages to easily incorporate other resources for improving plot visualization (e.g. sound tracks, flashback videos, animations and subtitles), and to formally describe the synchronization among the various story resources. To control the synchronous presentation, existing hypermedia formatting tools [Soares00][Rodrigues04] can then be used.

Acknowledgements

The authors thank CNPq and Finep for financial support under the form of PhD scholarships and research grants, in special through the contracts CNPq PQ Proc. 305982/2003-6, SEPIN-CNPQ-FINEP No. 01.02.0235.00 (Ref. 2425/02) and FINEP No. 01.04.0945.00 (Ref. 3110/04). Thanks are also due to Ken Perlin for his valuable collaboration on "Responsive Face".

References

[Black04] Black, A.; Taylor, P., **Festival Speech Synthesis**, CSTR – The Centre for Speech Technology Research, University of Edinburgh, Software Package, version 2.0, United Kingdom, 2004.

[Bregler97] Bregler, C.; Covell, M.; Slaney, M., **Video Rewrite: Driving Visual Speech with Audio**, ACM Computer Graphics Proceedings, SIGGRAPH 97, Los Angeles, CA, 353-360, 1997.

[Cassell99] Cassell, J.; Bickmore, T. W.; Billinghurst, M.; Campbell, L.; Chang, K.; Vilhjalmsson, V. V.; Yan, H.. **Embodiment in Conversational Interfaces: Rea**, Proceedings of CHI, 520-527, 1999.

[Cavazza02] Cavazza, M.; Charles, F.; Mead, S., **Character-based interactive storing**, IEEE Inteligent Journal Systems, special issue on AI in Interactive Entertainment, vol. 17(4), pg.17-24, 2002.

[Ciarlini05] Ciarlini, A.; Pozzer, C. T.; Furtado, A.; Feijó, B., **A Logic-Based Tool for Interactive Generation and Dramatization of Stories**, ACM

SIGCHI International Conference on Advances in Computer Entertainment Technology - ACE 2005, Valencia, Spain, 2005,

[Ciarlini02] Ciarlini, A.; Furtado, A., **Understanding and Simulating Narratives in the Context of Information Systems**, Proceedings of the 21st International Conference on Conceptual Modeling - ER'2002, pg. 291-306, Tampere, Finland, 2002.

[Corradini05] Corradini, A.; Mehta, M., Bernsen, N.; Charfuelan, M.. **Animating an interactive conversational character for an educational game system**, In Proceedings of. 10th Int. Conf. on Intelligent User Interfaces, San Diego, California, p. 183-190, 2005.

[Dutoit98] Dutoit, T. and et al., **The MBROLA Project**, TCTS Lab – Théorie des Circuits et Traitement du Signal, Faculté Polytechinique de Mons, Software Package, Belgium, 1998.

URL: *http://tcts.fpms.ac.be/synthesis/introtts.html* (last accessed at April, 16, 2005).

[Massaro03] Massaro, D. W. **A computer-animated tutor for spoken and written language learning**, In Proceedings of the 5th Int. Conf. on Multimodal Interfaces, ICMI 2003, Vancouver, British Columbia, Canada, Nov. 5-7, 2003. p. 172-175, 2003.

[Mateas97] Mateas, M., **An Oz-Centric Review of Interactive Drama and Believable Agents**, Technical Report , School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1997.

[Pandzic02] Pandzic, I.S.; Forchheimer, R., **MPEG-4 Facial Animation: The Standard, Implementation and Applications**, John Wiley & Sons Ltd., ISBN 0-470-84465-5, 2002.

[Parke96] Parke, F. I.; Waters, K. **Computer Facial Animation**, AK Peters Ltd., 1996.

[Perlin97] Perlin, K., **Responsive Face**, Media Research Lab, New York University, USA, 1997.

URL: *http://mrl.nyu.edu/~perlin/demox/Face.html* (last accessed at April, 15, 2005).

[Rodrigues04] Rodrigues, R.; Lucena-Rodrigues, P.; Feijó, B.; Velho, L.; Soares, L.F.G., **Cross-Media and Elastic Time Adaptive Presentations: the Integration of a Talking Head Tool into a Hypermedia Formatter**,

Proceedings of Adaptive Hypermedia and Adaptive Web-Based Systems, Eindhoven. Lecture Notes in Computer Science (LNCS 3137), pg. 215-234, 2004.

[Sgouros99] Sgouros, N. M., **Dynamic generation, managing and resolution of interactive plots**, Artificial Intelligence Journal, vol. 107, pg. 29-62, 1999.

[Soares00] Soares, L.F.G.; Rodrigues, R.F.; Muchaluat-Saade, D.C., **Modeling, Authoring and Formatting Hypermedia Documents in the**

HyperProp System, ACM Multimedia Systems Journal, Springer-Verlag, vol. 8(2), pg. 118-134, 2000.

[Spierling02] Spierling, U.; Braun, N.; Iurgel, I.; Grasbon, D., **Setting the scene: playing digital director in interactive storytelling and creation**, Computer and Graphics, vol. 26, pg. 31-44, 2002.

[Thalmann95] Thalmann, N. M.; Thalmann, D., **Digital actors for interactive television**, Proceedings of the IEEE (Special Issue on Digital Television, Part 2), Vol. 83, No.7, July 1995, p. 1022-1031.

[Woo00] Woo, M.; Neider, J.; Davis, T.; Shreiner, D., "OpenGL: Programming Guide -- Version 1.2", Third Edition, Addison Wesley, 2000.

[Young00] R.M. Young, **Creating Interactive Narrative Structures: The Potential for AI Approaches**, Proceedings of AAAI Spring Symposium in Artificial Intelligence and Interactive Entertainment, AAAI Press , Palo Alto, California, 2000.